DTIC FILE COPY

AGARD-CP-473

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE · 92200 NEUILLY SUR SEINE FRANCE

AD-A228 155

AGARD CONFERENCE PROCEEDINGS No.473

Computer Aided System Design and Simulation

(Système de Conception Aidé par Ordinateur et Simulation)

DTIC
ELECTE
NOV 0 1 1990
S D
B

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

NORTH ATLANTIC TREATY ORGANIZATION

DISTRIBUTION AND AVAILABILITY
ON BACK COVER

90 10 25 034

NORTH ATLANTIC TREATY ORGANIZATION

ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT

(ORGANISATION DU TRAITE DE L'ATLANTIQUE NORD)

AGARD Conference Proceedings No.473

Computer Aided System Design and Simulation

(Système de Conception Aidé par Ordinateur et Simulation)

Papers presented at the Guidance and Control Panel 50th Symposium held at the Altın Yunus Hotel, Çeşme/Izmir, Turkey, from 22nd to 25th May 1990.

The Mission of AGARD

According to its Charter, the mission of AGARD is to bring together the leading personalities of the NATO nations in the fields of science and technology relating to aerospace for the following purposes:

- Recommending effective ways for the member nations to use their research and development capabilities for the common benefit of the NATO community;
- Providing scientific and technical advice and assistance to the Military Committee in the field of aerospace research and development (with particular regard to its military application);
- Continuously stimulating advances in the aerospace sciences relevant to strengthening the common defence posture;
- Improving the co-operation among member nations in aerospace research and development;
- Exchange of scientific and technical information;
- Providing assistance to member nations for the purpose of increasing their scientific and technical potential;
- Rendering scientific and technical assistance, as requested, to other NATO bodies and to member nations in connection with research and development problems in the aerospace field.

The highest authority within AGARD is the National Delegates Board consisting of officially appointed senior representatives from each member nation. The mission of AGARD is carried out through the Panels which are composed of experts appointed by the National Delegates, the Consultant and Exchange Programme and the Aerospace Applications Studies Programme The results of AGARD work are reported to the member nations and the NATO Authorities through the AGARD series of publications of which this is one.

Participation in AGARD activities is by invitation only and is normally limited to citizens of the NATO nations

The content of this publication has been reproduced
directly from material supplied by AGARD or the authors.

Published August 1990

Copyright © AGARD 1990
All Rights Reserved

ISBN 92-835-0578-6

Printed by Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ

Theme

As guidance and control systems have become more complex, the role of computers in their design and development has become increasingly important. The results of simulation have been presented regularly in guidance and control symposia and, to a lesser extent, the use of computer design aids. However, it is considered that a symposium dedicated to computer aided design and simulation will provide a valuable opportunity to highlight the possibilities, the problems and solutions in this important field.

Computer aided design and simulation find applications at all stages of a project's life, starting with the conceptual design phase in which a basic system is defined and its performance evaluated using standard or special purpose design aid tools and simulation software. In the subsequent development of the system the effects of individual components or subsystems such as filters, limiters and other non-linearities, sensor and actuator dynamics, and embedded computer algorithms, are progressively quantified. In the later stages of development and evaluation, the complete system is simulated in sufficient detail to verify system performance against specifications.

As system development proceeds, increased emphasis is placed on real time computer simulation, with some or all of the hardware included in the simulation, depending on the phase of the project. Hardware-in-the-loop simulation includes the special case in which a human operator is included.

The aim of the symposium was to cover all stages of the development process.

Keywords: French language
NATO format
missile applications
aircraft applications
pilot in the loop Simulations, (ICP)

DTIC
COPY
INSPECTED
6

Accession For
NTIS GRA&I ☑
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A-1

Thème

Au fur et à mesure que les systèmes de contrôle et guidage deviennent de plus en plus complexes, le rôle des calculateurs dans leurs phases de conception et de guidage est devenu de plus en plus important. Les résultats de simulation ont été régulièrement présentés dans les symposia de guidage et pilotage, et à un degré moindre, l'utilisation d'aide à la conception des calculateurs. On espère cependant qu'un symposium dédié à la conception assistée par ordinateur et à sa simulation offrira une opportunité appréciable de mettre en lumière les possibilités, les problèmes et les solutions dans cet important domaine.

La conception aidée de calculateurs et sa simulation trouve des applications à tous les stades du projet, en commençant par la phase de conception dans laquelle un système de base est défini et ses performances évaluées en utilisant un outil d'aide standard ou spécifique, ainsi que la simulation du logiciel. Lors des développements ultérieurs du système, les interactions de chaque composant ou sous-système, tels que filtres limiteurs et autres non-linéarites, dynamique des capteurs et actuateurs, et logiciels câblés sont progressivement quantifiées.

Dans les développements et phases d'évaluation ultérieurs, le système complet est simulé avec suffisamment de précision pour confronter les performances du système avec les spécifications.

Lors du développement du système, une attention plus grande est portée à la simulation en temps réel, en incluant partie ou totalité du matériel, suivant l'avancement du projet. La simulation qui inclut du matériel dans la boucle comprend le cas particulier où l'homme est dans la boucle.

Le symposium avait pour but de couvrir tous les stades du développement.

Guidance and Control Panel Officers

Chairman: Ir P.Ph.van den Broek
Department of Aerospace Engineering
Delft University of Technology
Kluyverweg 1
2629 HS Delft
The Netherlands

Deputy Chairman: Professor E.B.Stear
Director, Washington Technology Center
University of Washington
376 Loew Hall — FH10
1013 NE 40th Street
Seattle, WA 98195
United States

TECHNICAL PROGRAMME COMMITTEE

Chairman:	Mr Stanley Leek	UK
Members:	Dr André Benoît	BE
	Dr Marc J.Pelegrin	FR
	Pr Dr Ing. Reiner C.Onken	GE
	Ir Pieter Ph.van den Broek	NE
	Pr Dr Ozay Oral	TU
	Mr Sterling Haaland	US
	Pr Edwin B.Stear	US

PANEL EXECUTIVE

From Europe:
Commandant M.Mouhamad, FAF
Executive, GCP
AGARD-OTAN
7 rue Ancelle
F-92200 Neuilly-sur-Seine
France
Telephone: 33 (1) 4738 5780
Telex: 610 176F
Fax: 33 (1) 4738 5799

From USA and Canada only:
AGARD-NATO
Attention: GCP Executive
APO New York 09777

HOST PANEL COORDINATOR

Professor Dr Ozay Oral
Dean, Faculty of Engineering and Science
Vice President, Bilkent University
PK 8 Maltepe, Ankara
Turkey

ACKNOWLEDGEMENTS/REMERCIEMENTS

The Panel wishes to express its thanks to the Turkish National Delegates to AGARD for the invitation to hold this meeting in their country and for the facilities and personnel which make the meeting possible.

Le Panel tient à remercier les Délégués Nationaux de la Turquie près l'AGARD de leur invitation à tenir cette reunion dans leur pays et de la mise à disposition de personnel et des installations nécessaires.

Contents

	Page
Theme/Thème	iii
Panel Officers and Programme Committee	iv
Keynote Address by Mr David Humphries, Assistant Chief Scientific Adviser (Projects & Research) MOD, Whitehall, London, UK	K
	Reference
SESSION I – COMPUTER AIDED SYSTEM DESIGN Chairman: Dr A.Benoît (BE)	
Méthodologie de Développement d'Algorithmes de Commande pour l'Optimisation des Performances Rotor par Commande Multicyclique sur Hélicoptère (OCAPI) (Algorithms Development Methodology for Performance-Optimized Multicyclic Rotor Commands) par S.Germanetti et B.Gimonet	11
A Decentralized Controller for Highly Augmented Aircraft by K.A.Ünyelioğlu and A.B.Özgüler	12
Parameter Space Design of Robust Flight Control Systems by A.Cavallo, G.de Maria and L.Verde	13
Computer Aided Design and Simulation of the Automatic Approach and Landing Phase of a Combat Aircraft by F.D.Langer	14
SESSION II – SIMULATION TECHNOLOGY FOR MISSILE APPLICATIONS Chairman: Dr E.Zimet (US)	
Conception d'Autodirecteurs Rigides (Conceptual Design of Strapdown Seekers) par F.Bertrand et M.Breuzet	21
Computer Simulations for the Development of Missile Navigation Systems by D.Berton, D.Duhamel and G.Cuvelier	22
Paper 23 withdrawn	
Paper 24 withdrawn	
Guided Weapon Simulation – The "SBGL" Development Experience by J.L.Quesada Rodriguez, R.Minguez and P.Segurola	25
GWSIM – A Computer Based Design and Simulation Package for Land Based and Air Weapon Systems by P.M.G.Silson and B.A.White	26
The Microcomputer as a Tool for Guidance and Control Visualization by P.Zarchan	27

SESSION III – SIMULATION TECHNOLOGY FOR AIRCRAFT APPLICATIONS
Chairman: Captain A.Sezgin (TU)

A Unified Approach to Simulation Software and Operational Software 31
by A.Mattissek

A Simulation Study for Analysing Pilot's Rule-Based Behavior 32
by B.Döring

Symbolic Generation of Aircraft Simulation Programmes 33
by P.Maes and P.Y.Willems

Outils Formels et Outils de Simulation: Un Atelier Cohérent 34
(Formal Tools and Simulation Tools: A Coherent Workshop)
par P.Schiele

Aircraft Control System Design. Synthesis, Analysis and Simulation Tools at Aermacchi 35
by L.Mangiacasale, L.V.Cioffi and C.A.Bonatti

Cockpit Mock Up CMU, a Design and Development Tool 36
by C.Weber

Computer-Aided Control Law Research – from Concept to Flight Test 37
by B.N.Tomlinson, G.D.Padfield and P.R.Smith

SESSION IV – HARDWARE-IN-THE-LOOP SIMULATION
Chairman: Mr S.Haaland (US)

Real-Time Hardware-in-the-Loop Simulation for "ATTAS" and "ATTHES" Advanced Technology Flight Test Vehicles 41
by P.Saager

Hardware-in-the-Loop Simulation at the Naval Weapons Center 42
by R.A.Licklider, A.B.Galloway, J.Schiavone, E.J.Bevan and W.Williams

Simulation of Multipath for Semiactive Missiles 43
by R.M.Smith, J.Y.Yee, C.S.Ang and A.I.Haun

A New Approach to Hardware in-the-Loop Simulation (FALKE Shuttle) 44
by C.-H.Oertel, K.Alvermann, R.Gandert and B.Gelhaar

Computer Graphics in Hardware-in-the-Loop Missile Simulation 45
by B.J.Holden

SESSION V – SYSTEMS APPLICATIONS
Chairman: Dr M.J.Pelegrin (FR)

Modelling of Land Based Air Defence Systems in Research and Procurement Support 51
by K.A.Hurst and S.Flynn

Integration of a Realistic Airline/Aircrew/Aircraft Component in ATC Simulations 52
by A.Benoît and S.Swierstra

NAVPACK – Simulation Tools for Design of High Performance Integrated Navigation Systems 53
by J.Z.Zywiel, J.S.A.Hepburn and B.M.Scherzinger

Application of Modelling and Signal Processing in Air Defence 54
by N.M.Vepa and W.Kreuzer

The Use of System Simulation During the Definition Phase of the Passenger Transport Aircraft MPC75 55
by D.Dey and A.Kröger

Reference

SESSION VI – PILOT-IN-THE-LOOP SIMULATIONS
Chairman: Pr Dr Ing. R.C.Onken (GE)

COMTESS – Combat Mission Training Evaluation and Simulation System 61
by W.Kraft, U.Krogmann, H.P.Müller and E.Platt

Integrated Technology Development Laboratories 62
by D.E.Dewey

Simulation of Nap-of-Earth Flight in Helicopters 63
by G.W.Condon

Results of Man in the Loop Simulator Experiments Using Air-to-Air Missile Models 64
by N.Seavers

The Development of Avionics-Intensive, Multi-Sensor Cockpits: 65
Simulation Doesn't Always Equal Success
by C.G.Killberg

KEYNOTE ADDRESS - AGARD GUIDANCE AND CONTROL PANEL MEETING. COMPUTER AIDED DESIGN AND SIMULATION.

Author: David Humphries, Assistant Chief Scientific Adviser (Projects & Research), Ministry of Defence, Room 2273, Main Building, Whitehall, London, SW1A 2HB

Ladies and Gentleman

I am most grateful for your very kind invitation to give the Keynote Address at this very timely and important conference, not least of course because it is being held in such a beautiful place, and enjoying such fine weather. There is however, another reason why I am pleased to be here, and that is because my association with AGARD, and with the Guidance and Control Panel in particular goes back 23 years. Before preparing this talk, I looked in my diary for 1967, and found that I attended a two day meeting of the G & C panel in Oxford on the 21/22 September of that year, on the subject of inertial navigation and Kalman Filtering. So began association with AGARD and the very important work that it does in enabling aerospace specialists of the NATO nations to get together to exchange ideas, give and receive lectures, and generally work together for the mutual benefit of all participants. Later, I went on to deliver papers at AGARD meetings, mainly on the subject of digital computers as applied to Navigation and Weapon Aiming, and Kalman Filtering in particular, with AGARD providing me with the main opportunity for discussing the work that I was doing with other workers in the same field.

In the light of those memories, therefore, I was particularly pleased to learn that you had decided to hold this meeting at this time on the subject o. Computer Aided System Design and Simulation. The first airborne computer that I was closely involved with was the one which went into the Navigation and Attack System of the SEPECAT Jaguar Aircraft, it had a memory capacity of 8,000 18 bit words. We found it quite a struggle, even with that size of programme, to keep a track of everything that was going on within the programme since it covered in the one computer Navigation, course computation and steering, driving the moving map display, weapon aiming and a certain amount of air data computing for good measure.

Since then things have changed markedly, as a result of the vast growth in computer speed and capacity. To take one example, from the aerospace field, and using as a yardstick the Jaguar system which I mentioned earlier, the number of words of code which comprise the airborne software for a succession of combat aircraft, has shown a growth of 1,000 times over a period of about 40 years. It would appear however, that in certain areas of the defence field, our programming techniques, have not kept up with the growth in capacity. We in the department of the Chief Scientific Adviser of the U.K. Ministry of Defence, have a particular interest in improving the quality of management of software intensive projects. That is to say the development of equipments which either have a very large software content, or where the software is absolutely critical to their successful operation. I am firmly of the view that simulation, emulation, fast prototyping, and languages which allow one to develop the programme for the target machine, directly from the software developed as part of the earlier simulations are the key steps on the path to the low risk development of software.

What then does the use of computers for the simulation and exploration of future system concepts do for us. I would say that:

1. It enables alternative system concepts to be explored and emulated with a human operator in the loop, well in advance of system and concept requirement freeze.

2. It provides both a guard against unnecessary or expensive changes to the requirement during development, and a framework for controlled change when this is seen to be essential as a result of changes to the threat over which there is no control.

3. It allows the specification of the chosen system, which has been fully evaluated and found to be acceptable in the simulator, to be defined and specified unambiguously prior to its development "for real", by using the same language which programmed the simulation to define the requirement.

4. It allows us to keep track of the complexity of modern integrated and highly interactive systems, despite the total intellectual content of the system being way beyond the amount which a single individual can comprehend.

5. It allows a top-down design approach. That is to say, you start with the concept, from that you develop the optimum man-machine interface, next the overall system design, then the software, and only finally specifying the computing power and the hardware required to do the job.

It is important to ensure that all involved, especially those who hold the purse strings and those who specify the military requirement, appreciate that these benefits are not luxuries, that enable us to develop the system in a more comfortable environment, but are essential if we are to manage effectively the enormous complexity which has stemmed from the explosive growth in computing power which I referred to earlier.

One very important benefit which comes from this new found ability to try it before you buy it, or in other words to evaluate a novel concept in a realistic environment before you commit yourself to the expense of its full development, is that you can perhaps be more adventurous in the concepts which you explore. One can be more prepared to question the received wisdom, or the views of the old hands, and try more radical alternative approaches. After all, the next generation of pilots, those who are, I suppose, just about now leaving school and starting on their flying training, have grown up in the era of computer games. They are used to manipulating symbols, and performing tasks requiring manual dexterity and mental agility against a computer synthesised scene. For this reason I am going to devote a little of my talk to the exploration of one or two such novel and some might say even heretical ideas.

THE VIRTUAL COCKPIT.

Recent advances in sensors, computing and displays, of the kind which provide the backbone for this conference, have, in the past decade, changed the cockpits of both military and civil aircraft almost beyond recognition. I say almost beyond recognition, because many of the electronic displays that are in today's cockpit, are in fact designed to emulate the mechanical instruments which they replace. - An important point to think about when we are considering the electronic cockpit of the future. As a result, most of the decisions, and actions which a pilot takes are already based on information which he gains from looking at some form of electronic display rather than actually looking outside the aircraft. Outside the aircraft, such things as sunlight, mist, cloud and fog present an enormous variation in both the quality and the quantity of the information received from the outside world. Inside the aircraft, symbols and displays are more predictable, the availability of information may vary but the way it is presented to the operator never changes, and it is exactly in the format which he trained with in the simulator. But there is another, man made reason, for not looking out of the cockpit, and that is the possible threat to the pilot from laser weapons. Laser engineering is now a well established discipline, the theory is widely understood and they can be built relatively cheaply so that presumably they are easily available to anyone of evil intent anywhere in the world. A sensor whose detectors or optics are damaged by a laser, can be replaced after the aircraft returns to base, this is not the case for the pilot's eyes, however, where one mistake may result in permanent damage to a pilot's eyesight to the extent that he may be unable to fly again.

This leads me to my suggestion that we should give serious consideration to the idea of the "virtual" cockpit, where all the information available to the pilot is presented to him by electro-optical or opto-electronic means. I would point out that all the technology which is needed is already available to us, the helmet mounted display, the head position sensors, realistic 3 Dimensional portrayal of the outside world on head up and head down displays, and high definition colour available head-down. I don't of course envisage the pilot using this system from take off to landing, since he would have to have a "get-you-home" capability in the event of system failure, but I do suggest that he might well operate "closed down" during his stay in a hostile area.

Another development which is closely related to the concept of the virtual cockpit, and the technologies needed to realise it, is the likelihood that much greater use will be made of simulators in training over the next 20 years. Simulators are getting increasingly sophisticated, and displays get more realistic with each new generation of digital scene generator. With the virtual cockpit, of course, there will be hardly any difference at all between the what the pilot sees in the simulator and what he sees in real flight. The one remaining degree of realism which is required is that of providing realistic acceleration cues, and even that is likely to be available in future to an increasing degree, and I draw your attention to the major full motion simulator facility which is just now undergoing commissioning at RAE Bedford and which has a motion system unique in Europe. It also has the flexibility to link with and assess customers' hardware. Flying aeroplanes and driving tanks close to heavily populated areas is increasingly becoming regarded by the public as environmentally unfriendly, greater use of simulators is not just an option for the future, I believe that it is likely to be an obligation.

THE UNIVERSAL SIMULATOR.

This brings me very neatly to the one other idea that I would like to lay before you this afternoon and that is the idea of the universal simulator. With the increasing power of the underpinning digital computing, universal displays under software control, and the ability to replace elements of the simulation with the real hardware for "in-loop" evaluation. I have been increasingly struck as I have visited various parts of the aerospace and military hardware industry how remarkably similar in design simulators for system design, hardware evaluation, man in the loop evaluation and training are becoming. Visiting a factory the other day where a complex system was under development, I was told, somewhat plaintively, by the development engineers that the customers were casting covetous eyes on their system development simulator, saying that it was just what they wanted for training the future users of the system. If you look at the block diagram of a simulator, it can be seen that the same basic design can be used for widely different purposes, depending which of the blocks you hold constant and which you vary. For example:

i) If you hold the scenario, the operator, the displays and the weapons constant, but vary the way in which data is manipulated and presented, then you have system development and requirements capture simulation.

ii) If you vary the displays, and present alternatives to a small sample of operators you have a man machine interface development tool.

iii) If you replace the computing block which represents a missile say, and replace it with the actual missile, you have a hardware test bed for performance testing and acceptance.

iv) Finally if you hold everything constant but the scenario and the operator, you have a weapon training simulator for use alongside the actual equipment in service.

There are, of course, other combinations which I am sure that you can think of for yourself, but I hope that with these examples, I have illustrated my main message. I am not postulating of course that all these applications should be executed on the same simulator, but that a universal design, first conceived at the very start of development, can by means of suitable variants, provide you with all these facilities; and you no longer have to design each device separately for its specific application.

MISSILE CONTROL SYSTEM DEVELOPMENT.

Most of my remarks this afternoon can be applied to the development of weapon systems generally, and can apply equally to platforms or to their weapons. But I would just like to touch briefly on certain aspects which in my view apply particularly to missile system development.

The cost of delivering a missile up to the point at which it arrives in the vicinity of the target is the same whether it hits or misses. There is clearly then a substantial benefit to be gained from improving kill probability. If overall numbers of weapons and their delivery systems are limited, either as a result of financial pressures or treaty obligations, then reliability and accuracy are at a premium. We need to ensure that every weapon, launched correctly within its theoretical performance envelope will kill its target. It is likely to become increasingly expensive and difficult to develop such a capability primarily through in-flight testing. As designs become more sophisticated, and exploit the limits of performance in every aspect, the interaction between the physical limits of performance and the digital processing gets ever more subtle. The depth of the interaction which it is necessary to explore between the key elements of weapon aerodynamics, engine thrust profile, limiting sensor performance and the guidance loop , has reached the point, I suggest, where optimisation and demonstration becomes impossible without a fully representative mathematical model, linked to various levels of hardware in the loop simulations as the development proceeds.

As an important means of avoiding costly failures and timescale overruns on future programmes, we in the U.K. Ministry of Defence are looking to early demonstration of critical system parameters, well in advance of commitment to full system development. Simulation and performance assessment, using "real" data gathered independently, and using various combinations of mathematical model simulation and 'breadboard" prototyping of key circuits is to my mind a vital step in proving the viability of a novel concept, to the extent needed to justify the further big expenditure which has to go into a complete missile system.

CONCLUSION

You can tell by now, I hope, that I am a great enthusiast for the use of simulation. I believe that it has a key role to play right the way through from formulating the initial concept, developing the system design, capturing the requirement, proving the hardware, running acceptance tests on the delivered product and finally training the operator. I am an enthusiast, not least because I believe that this approach can make a major contribution to solving a number of thorny problems which we will have to face in the defence field over the next few years. Amongst these are:-

i) Saving money; in the defence world money is always in short supply since, not surprisingly, the tax-payer, while recognising that he has to pay for defence, likes to know that he has got the necessary level of security for the minimum expenditure of his money. One of the best ways of saving on development expenditure is, of course, to get it right first time. Analysis of some past projects which ran into serious time and cost overruns and failed to meet their specification indicated a number of common features. A high software content, and a failure to define the requirement with sufficient precision frequently appeared in the list.

ii) Saving Manpower; in the U.K. as in the rest of NATO we face a falling birth-rate, we need to make not only the operation of military equipment less manpower intensive, but also its development and its maintenance when in service.

iii) Saving Time; with the threat which we face becoming less well defined as a result of the momentous changes in Eastern Europe and the Soviet Union, we need to have greater flexibility, flexibility to develop systems to match any new threat which might arise, in a timescale which matches the speed with which that new threat might develop.

iv) Saving scarce skill resources; the skills which are needed to develop closely integrated digital/mechanical systems to match a complex scenario, are in short supply and likely to become ever more scarce. We need to use the minimum amount of that scarce resource on defence, so that the remainder can be used to develop ideas which will enable us to earn our living in highly competitive civil markets.

Once again, thank you for inviting me to your conference, as I think you will have gathered from my remarks, I believe that the subject is very timely, it certainly is of considerable interest to me, so that I look forward very much to hearing the various papers, and joining in the discussion both inside and outside the meeting hall.

METHODOLOGIE DE DEVELOPPEMENT D'ALGORITHMES DE COMMANDE POUR L'OPTIMISATION DES PERFORMANCES ROTOR PAR COMMANDE MULTICYCLIQUE SUR HELICOPTERE (OCAPI)

par

S.Germanetti
Aerospatiale Division Helicopteres
Direction Systemes/PT
13725 Marignane Cedex
France

B.Gimonet
ONERA CERT/DERA
2 Ave Edouard Belin
BP 4025
31055 Toulouse Cedex
France

1. INTRODUCTION

La simulation est devenue un point de passage obligé entre l'idée et la réalisation d'une loi de commande.La démarche part classiquement de modèles simples destinés à focaliser l'attention sur des aspects limités mais importants du problème posé, tout en minimisant le volume de calculs nécessaires. La simulation s'étoffe ensuite, augmentant ainsi son réalisme et sa crédibilité mais perdant un peu de sa souplesse, en intégrant de plus en plus d'éléments réels, théoriques ou même matériels. L'essai en soufflerie, l'essai en vol ne sont après tout que les phases ultimes de la simulation.

La simulation numérique permet de choisir la complexité du phénomène abordé. L'informatique permet alors de transformer cet outil d'analyse et de vérification en outil de synthèse grâce à la puissance des moyens interactifs disponibles.

L'objet de cette présentation est de préciser l'apport des différents outils de simulation au cours de l'avancement de l'étude de lois de commande dans le cadre de l'optimisation du fonctionnement des rotors.

2. LA COMMANDE MULTICYCLIQUE SUR HELICOPTERE

2.1. Historique

De nouveaux types de commande au niveau du rotor sont testées depuis quelques années dans le but d'améliorer le comportement vibratoire de l'hélicoptère. Ces commandes dites 'muticycliques' viennent se superposer aux commandes de pilotage classiques en générant des harmoniques $(b-1)\Omega, b\Omega, (b+1)\Omega$ au niveau

des pales. Le pilotage classique conserve la gestion du collectif et du cyclique qui constituent la composante continue et le premier harmonique 1Ω; Ω étant la vitesse de rotation du rotor.Le système multicyclique a pour tâche d'identifier le transfert entre commandes multicycliques et vibrations (variable en fonction du cas de vol et de la configuration appareil), de façon à calculer le module et la phase de chacune des trois commandes optimales à appliquer aux vérins , afin de réduire les vibrations dans la cellule.

Le développement de ces commandes à conduit à des essais en vol couronnés de succès en 1987 pour la réduction des vibrations (références 6 et 7).Il a été montré de plus que ces commandes multicycliques avaient un effet sur les performances rotor et qu'elles permettaient notamment de minimiser la puissance consommée ou d'augmenter les efforts de traction ou de portance développés par le rotor.

Afin de quantifier les effets de ce type de commande au sens des performances, un modèle de connaissance à été élaboré à partir du modèle rotor déjà existant à l'Aérospatiale ce qui à permis de reconstituer en simulation la réponse d'un rotor au sollicitations multicycliques. Des essais en soufflerie corrélant ces résultats ont permis une validation de ce modèle vis à vis des effets sur les efforts et la puissance. Dés lors il a été permis de montrer que parmi l'ensemble des commandes multicycliques certaines avaient un effet favorable sur le comportement du rotor. Une étude plus poussée a alors vu le jour afin de mesurer en vol les effets du multicyclique au sens de la répartition du flux aérodynamique du rotor (référence 9).

2.2. *Evolutions technologiques*

La technologie classique du rotor basée sur l'utilisation d'un plateau cyclique a permis de réaliser aisément les commandes antivibratoires en repère fixe (génération de commande en repère fixe à la fréquence principale des vibrations en cabine bΩ se traduisant en repère tournant par une commande en (b–1)Ω,bΩ,(b+1)Ω homogène c'est à dire commande identique pour chaque pale à azimut donné).

Les études précédentes ayant montré l'intérêt de la fréquence 2Ω dans le cadre de l'optimisation des performances, des études technologiques ont été conduites dans le but de se doter d'un système capable de générer des commandes homogènes à toutes les fréquences.(la fréquence 2Ω n'est pas commandable en repère fixe sur un rotor quadripales) (figure 1)

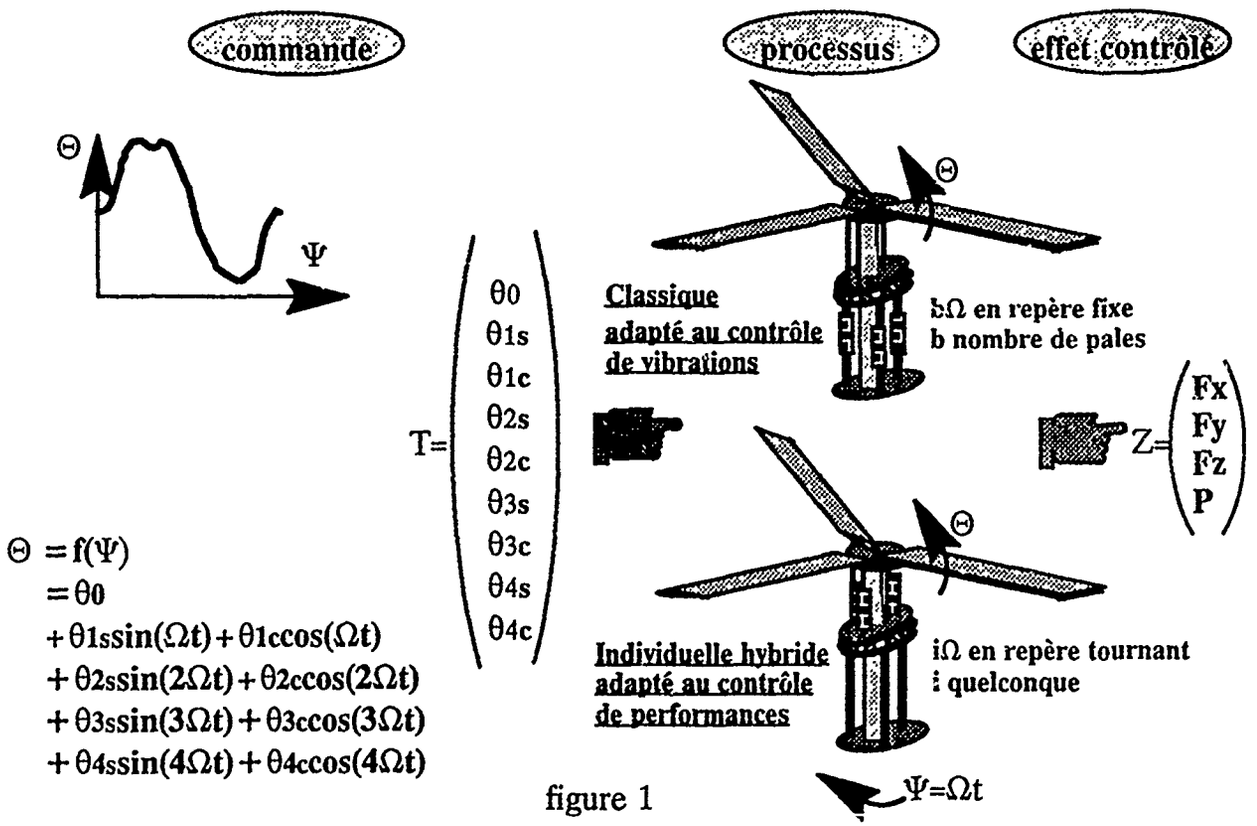


figure 1

Deux solutions sont aujourd'hui envisagées :

-La commande intégrale en repère tournant (référence 1)

-La commande hybride comportant un plateau fixe générant les commandes de pilotage et une activation des biellettes de pales permettant d'ajouter en repère tournant les fréquences désirées.

La comparaison de ces deux solutions sur les aspects intérêt, simplicité, bilan énergétique, sécurité ne fait pas l'objet de ce document mais il faut insister sur la possibilité de générer dans les deux cas des commandes multicycliques à tous harmoniques du rotor.

3. DEVELOPPEMENT D'UNE COMMANDE MULTICYCLIQUE D'OPTIMISATION DES PERFORMANCES

3.1. Objectifs

L'ensemble de l'approche ayant montré que des gains pouvaient être attendus au sens des performances l'objectif était alors de développer un algorithme de commande prenant en compte les contraintes liées à l'utilisation en vol de telles commandes :

-prudence des commandes car il n'est pas suffisant de limiter l'autorité des commandes multicycliques, il faut également éviter de trop grandes variations entre deux ordres consécutifs délivrés par le calculateur de bord.

-non perturbation du pilotage liée à la capacité donnée à l'algorithme de gérer partiellement les ordres cycliques qui lui permettent d'équilibrer le cas de vol en coordination avec les actions de l'équipage.

-calcul en ligne permettant la programmation sur calculateur embarqué.

Pour cela un programme de développement a été mis en place au département système de l'AEROSPATIALE (Marignane) en s'appuyant sur le CERT/ONERA(Toulouse) en ce qui concerne les méthodes d'optimisation. L'objectif étant de se doter d'un algorithme 'embarquable 'déterminant la commande optimale du rotor soit vis à vis de la puissance consommée soit vis à vis de la portance.

Ce programme a donné lieu à un soutien des organismes de recherche représentés par la DRET qui a conduit à la définition de l'algorithme "OCAPI"(Optimal Controler Adaptativ with Process Identification.

3.2. Etapes du développement

Les étapes mises en place ont été dictées par la volonté d'appréhender progressivement les différents problèmes. (figure 2)

-Phase 1 : Définition de la structure de l'algorithme de commande à partir d'une étude théorique de la réponse du rotor et simulation sur modèle d'action. Ce modèle permettant d'accélérer la démarche de mise au point de la loi de commande, même si l'aspect interactif test/évolution est minimisé par les apports de la théorie.

-Phase 2 : Tests et réglages de l'algorithme sur modèle plus complexe intégrant des phénomènes non pris en compte dans la première phase et relatifs au comportement aérodynamique du rotor.(simulation avec modèle de connaissance)

-Phase 3 : Prise en compte des capteurs et des actionneurs en simulation.

-Phase 4 : Intégration du modèle de connaissance rotor et d'un modèle simplifié d'hélicoptère afin de prendre en compte l'équilibre global cellule rotor ainsi que les contraintes de pilotage.

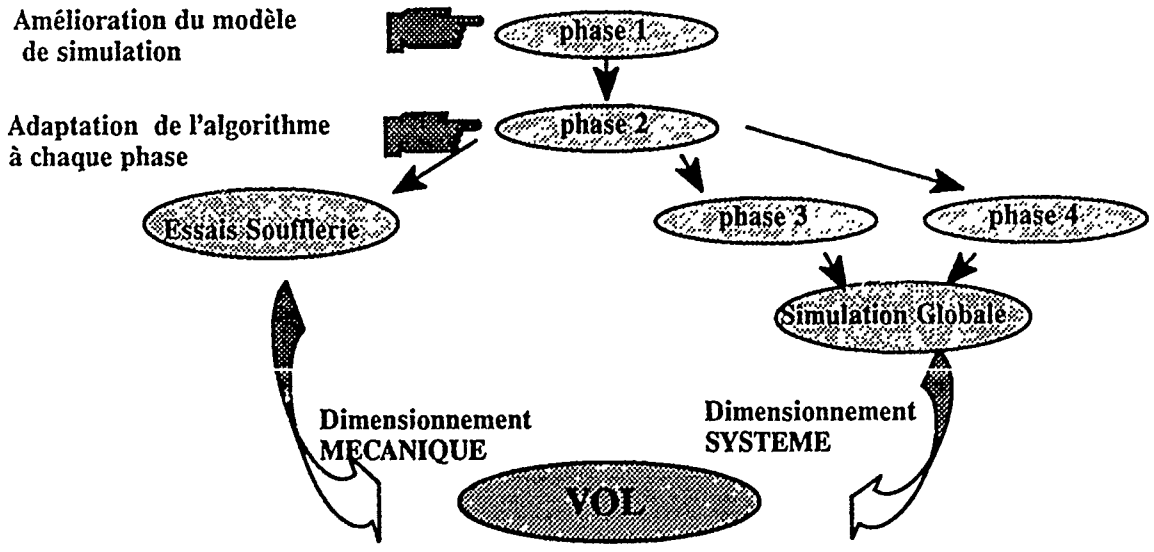


figure 2

–***Simulation globale :*** Deux points seront testés dans cette phase

–Il faudra quantifier les effets secondaires de la commande optimale des rotors (moments parasites , effets de la commande sur les 1eres harmoniques d'efforts) en introduisant un modèle plus représentatif à ce niveau et en analysant l'effet de ces variations au niveau de la cellule appareil. Pour cela des réactualisations des modèles existant pourront être obtenues à partir des essais en soufflerie.

–D'autre part la prise en compte de ces effets sur un modèle complet d'hélicoptère permettra de mettre au point les stratégies opérationnelles d'utilisation de cette nouvelle commande et de quantifier les améliorations du comportement dynamique d'un appareil muni d'un tel système au sens de la maniabilité.

–***Essais soufflerie :*** Ils peuvent être de deux types

–Hors des aspects temps réel tester les effets des phénomènes non modélisés sur le comportement de l'algorithme.

–Qualifier les performances potentielles de l'algorithme dans un cadre technologique proche de l'utilisation en vol.

–***Vol :*** Dans un premier temps, au niveau d'un démonstrateur, il peut s'avérer prometteur d'utiliser l'algorithme développé dans le contexte d'une commande en repère fixe ne disposant donc pas de tous les degrés de liberté.

4. FORMULATION DU PROBLEME

4.1. Les modèles utilisés

L'impossibilité d'appréhender le fonctionnement du rotor vis à vis de la commande multicyclique, en dehors du modèle de simulation mathématique du rotor à nécessité l'identification d'un modèle d'action. Le but de ce modèle était de remplacer un modèle de connaissance par plusieurs modèles de type 'entrée/sortie' adaptés essentiellement aux paramètres que devait controler l'algorithme.(figure 3)

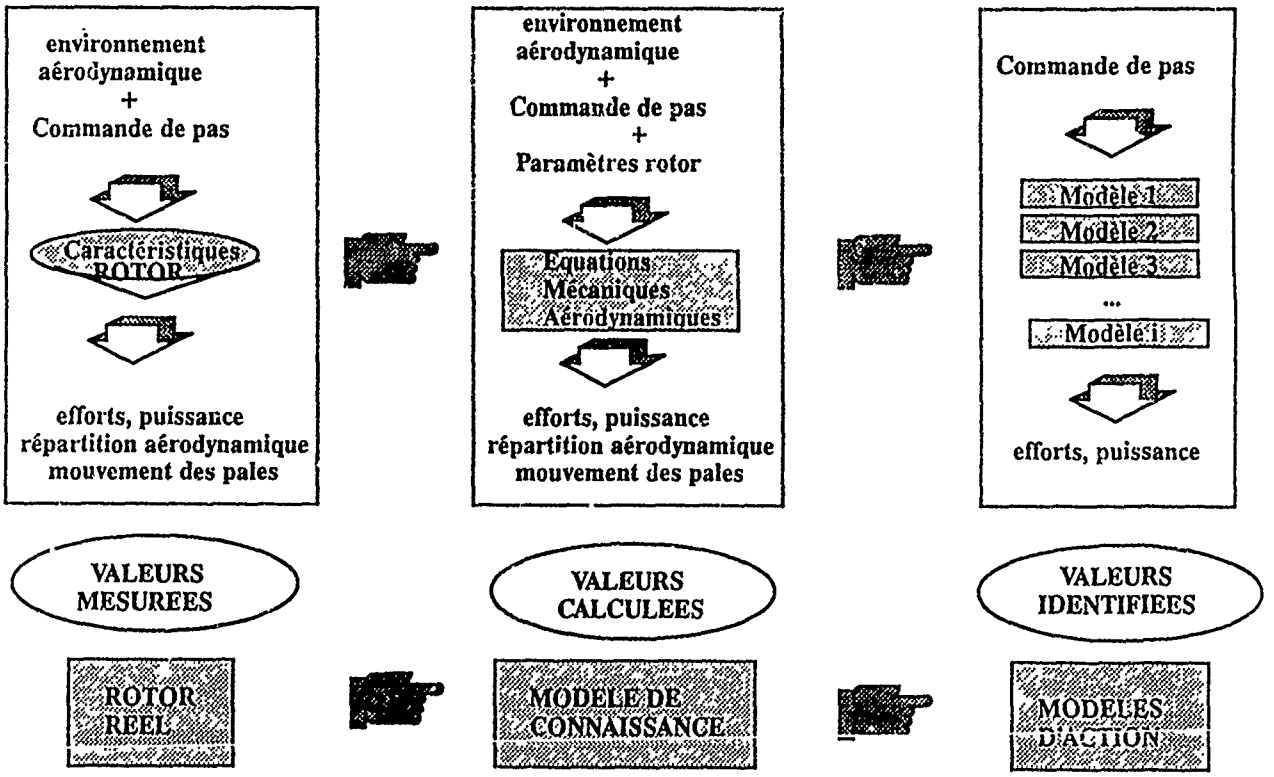


figure 3: ***PROCESSUS DE DEFINITION DU MODELE D'ACTION***

4.1.1. Modèle de connaissance

Ce modèle a été développé à partir de la simulation rotor existant à l'Aérospatiale. Les commandes multicycliques ont été introduites en les superposant aux commandes de pilotage. La modélisation s'effectue en quatre étapes

- **Modélisation des coefficients de portance et traînée Cz,Cx** uniquement fonction du profil de la pale et calcul des forces élémentaires en un point de la pale
- Calcul des efforts sur une pale par **intégration sur le rayon**
- **Intégration sur un tour rotor** et sommation des différentes pales
- Afin de compléter ces équations il faut ajouter les **deux équations d'équilibre de la pale en traînée et en battement**. Ces équations de type dynamiques introduisent des couplages efforts/battements/traînée/flux aerodynamique qui rendent impossible la résolution directe du problème (résolution numérique sous forme du modèle de connaissance R85)

4.1.2. Modèle d'action

La structure du modèle a été définie à partir des équations issues du modèle de connaissances dans des hypothèses de fonctionnement simplifié du rotor.

La détermination du modèle d'action a donc consisté à définir les variations d'efforts en fonction d'une variation de commande $\theta=f(\Psi)$. En décomposant la commande en série de Fourier, en faisant l'hypothèse d'une faible variation des angles de battement et de traînée due à cette commande, en admettant que le champ aérodynamique reste globalement peu changé et que les courbes de portance et traînée de profil sont au plus représentées par un deuxième ordre, on montre que la relation effort/commande ou puissance/commande peut s'exprimer par un polynôme de degrés 2 dont les variables sont les coefficients de Fourier de la commande.

Le processus qui a conduit à la forme finale des modèles d'action est présenté figure 4

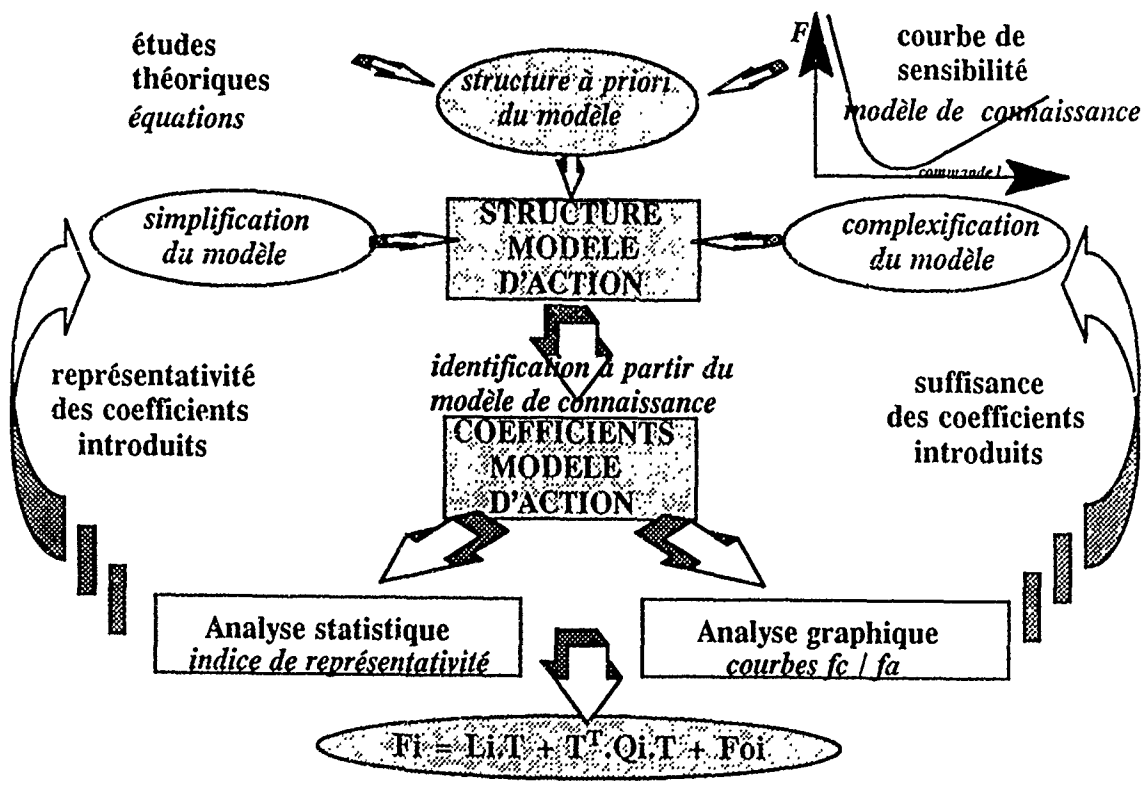


figure 4 : ***DETERMINATION ET VALIDATION DES MODELES D'ACTION***

Outils de validation du modèle

Les coefficients du modèle ont été identifiés à partir d'une méthode des moindres carrés. L'outil d'identification qui a été adapté à nos besoin dispose d'un certain nombre de traitements statistiques qui nous ont permis de statuer sur les degrés de confiance que l'on pouvait attribuer aux différentes modélisations. Le choix des bases de données constitue l'une des parties les plus délicates du processus de définition des modèles d'action Pour chaque cas de vol, elles sont constituées:

–d'un ensemble de commandes èquirèparties sur l'ensemble du domaine correspondant aux saturations envisagées
–des valeurs des efforts ou de la puissance correspondants à ces commandes
Une fois le modèle entièrement défini le tracé des *Courbes fa/fc* a permis d'analyser visuellement la répartition du nuage de points constituant la base de données. Cette analyse associée aux traitements statistiques qualifie l'apport de nouveaux termes dans le modèle.(figure 5)

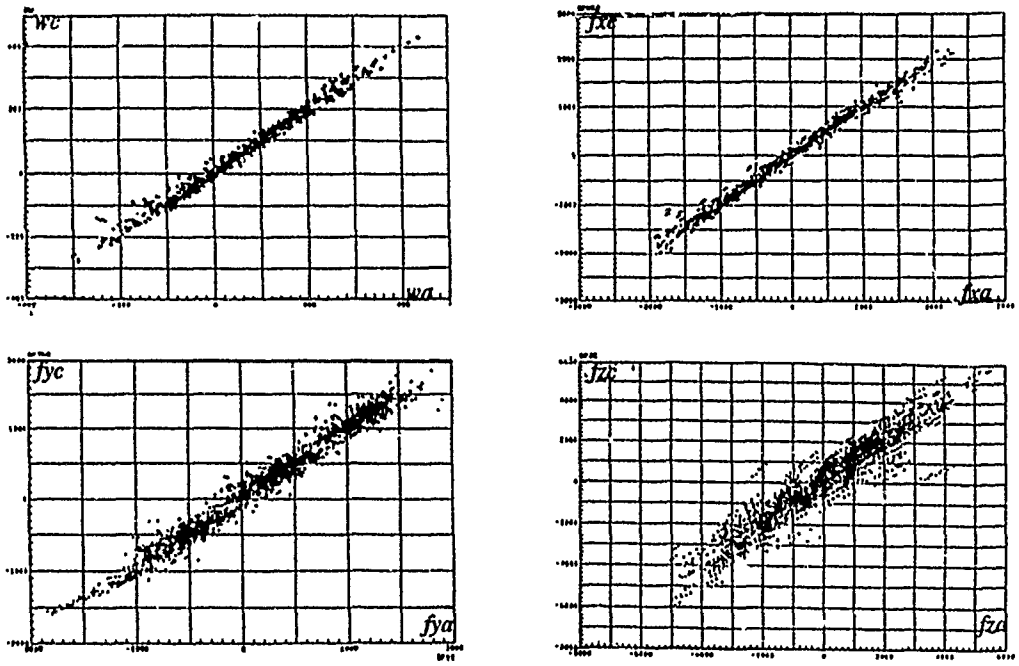


figure 5 : *COMPARAISON MODELE DE CONNAISSANCE / MODELE D'ACTION*

Les différentes étapes qui ont marqué notre étude sont essentiellement:
–l'introduction d'une constante dans le modèle (biais)
–la nécessité de termes de degrés deux
–l'apport des termes croisés cyclique/multicyclique
La structure obtenue est du type polynomial de degré deux :

$$F = L.T + T^T.Q.T + Fo$$

Une équation de ce type est affectée à chaque variable à commander (c.a.d 4)
F est un scalaire constitué soit d'une force soit de la puissance (Fo est la valeur de F à commande multicyclique nulle)
T est le vecteur de commande correspondant aux coefficients de Fourier des premières harmoniques
L et Q sont les matrices de sensibilité linéaires et quadratiques affectées à la variable F

4.2. L'algorithme de commande

L'algorithme mis en place est de type séquentiel. Après les mesures effectuées à l'instant n l'algorithme estime la sensibilité du phénomène à contrôler puis délivre la commande qui minimise le critère à l'instant n + 1. L'algorithme dispose pour cela d'un modèle de commande.

4.2.1. Modèle de commande et identification

L'identification en vol de l'ensemble des paramètres composant les matrices Li et Qi s'est avéré rédhibitoire (360 paramètres dépendants du cas de vol). La solution retenue a été de reporter sur la qualité de l'adaptabilité de l'identification une partie des erreurs commises par la prise en compte d'un modèle de commande différent du modèle d'action.
Un modèle linéaire a été retenu avec identification autour du point de fonctionnement ce qui impose à la commande d'avoir une dynamique plus lente que la dynamique de l'identification.
A chaque instant on dispose des trois mesures d'effort (Fx,Fy,Fz) et de la mesure de puissance P qui sont regroupées dans le vecteur Z. Le vecteur de commande T regroupe quant à lui les 9 premières composantes de Fourier de la fonction définissant le pas de pale sur un tour rotor.

Le modèle de commande s'écrit donc sous la forme suivante:

$$Z_n = S.T_n + Z_{0,n} + w_n$$

où w_n est le bruit de mesure et $Z_{0,n}$ est le vecteur de forces et puissance à commande multicyclique nulle. A chaque composante $Z^{(i)}$ de Z on associe un filtre de Kalman qui estime l'état X :

$$X^T = (S^{(i)}, Z_0^{(i)})$$

$S^{(i)}$ représente la $i^{ème}$ ligne de la matrice de sensibilité S

Les quatre filtres travaillent sur des équations d'état qui traduisent la constance des paramètres entre deux pas de calcul successifs :

$$X_{n+1} = X_n + v_n$$

v_n est le bruit d'état dont la variance représente la variation possible des paramètres à estimer dans X. L'équation de mesure du $i^{ème}$ filtre s'exprime donc:

$$Z_n^{(i)} = (T_n^T, 1).X_n + w_n^{(i)}$$

Les équations du filtre obtenu sont donc dans ces conditions :

$$\hat{X}_{n+1} = \hat{X}_n + K_{n+1}.[Z_{n+1} - (T_n^T, 1).\hat{X}_n]$$

K_{n+1} est calculé en fonction de la covariance d'erreur d'estimation de X fournie elle même par le filtre et la covariance du bruit w_n. Le suivi des paramètres de X est amélioré grâce au facteur d'oubli qui pondère les observations passées. Les quatre filtres de Kalman ne diffèrent que par leurs initialisations et ils constituent ligne par ligne une estimation de la matrice S du modèle de commande.

4.2.2. Critère et commande

Quatre types de contraintes sont à prendre en compte au niveau de l'algorithme (figure 6):
-Tenue du cas de vol
-Prudence des commandes
-Optimisation du critère (minimisation de puissance ou maximisation de portance)
-Saturation des commandes

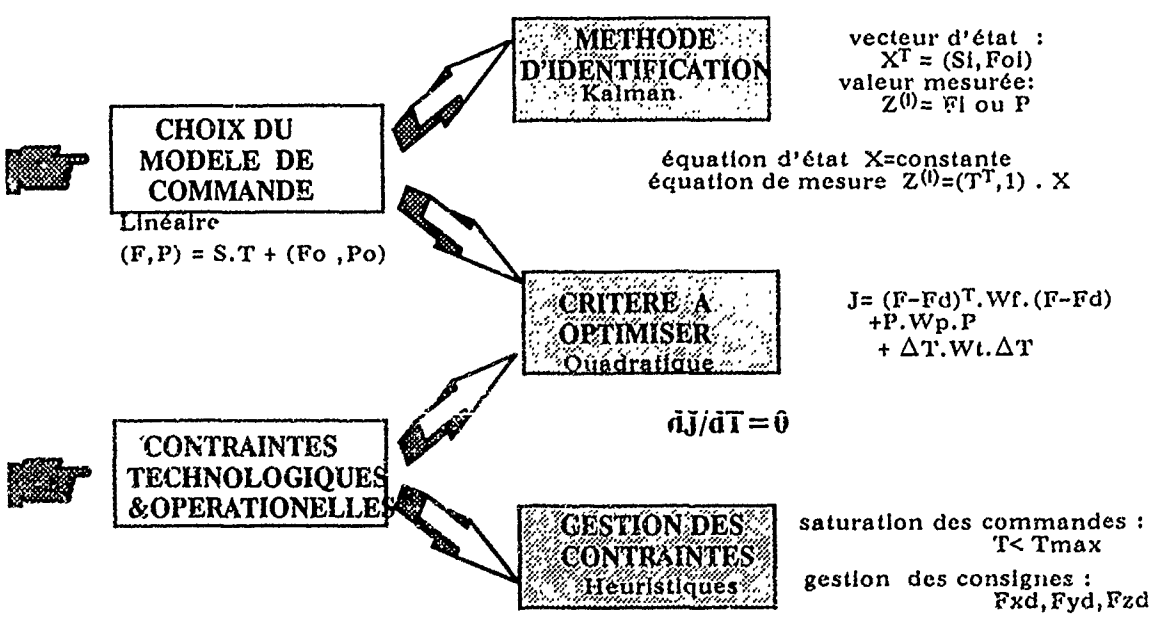


figure 6: ***PRINCIPES DE DETERMINATION DE L'ALGORITHME***

La tenue du cas de vol peut être gérée de deux façons; soit en utilisant l'identification linéaire des sensibilités comme autant d'équations à résoudre soit en introduisant les contraintes d'efforts dans le critère et en utilisant le principe des pénalités. Suite à la mise en oeuvre des deux méthodes la deuxième solution à montré une bien meilleure robustesse à toutes les erreurs d'identification et permet de traiter avec le même formalisme l'optimisation de puissance ou de portance. (ceci constitue le 1er terme du critère)

La prudence correspond à limiter la dynamique d'évolution des commandes en ajoutant dans le critère une pondération sur l'écart de commande (2em terme du critère) ; par ce même terme on dispose d'un moyen de gestion du compromis entre le suivi des évolutions des sensibilités, la précision, la rapidité de convergence et répond donc au besoin de l'identification.

Le critère à minimiser est:

$$J_{n+1} = E[(Z_d - Z_{n+1})^T W_z (Z_d - Z_{n+1}) + \Delta T_n{}^T W_t \Delta T_n]$$

formule dans laquelle E représente l'espérance mathématique et ΔT_n la croissance de la commande

$$\Delta T_n = T_{n+1} - T_n$$

Z_d est la valeur désirée du vecteur de mesure et définit donc les objectifs de la commande en termes de cas de vol (F_{xd},F_{yd},F_{zd}) et de puissance $P_d = 0$

Les matrices diagonales W_z et W_t permettent de pondérer les écarts sur les objectifs de commande et sur les variations de commande.Le critère J_{n+1} est alors minimisé pour $dJ_{n+1}/T_{n+1} = 0$:

$$\Delta T_n = [S_n{}^T W_z S_n + W_t]^{-1} S_n{}^T W_z (Z_d - Z_n)$$

Ce calcul ne nous protège pas contre les dépassements des saturations physiques imposées par l'autorité des vérins multicycliques. Si une commande dépasse les contraintes elle est saturée pour l'harmonique concernée et le calcul d'optimisation est réactualisé sur les autres harmoniques.

4.2.3. Mise en oeuvre en ligne

La volonté de développer un algorithme dont l'application en vol constituait l'une des finalités a conduit à faire certains choix au niveau de sa structure. En particulier l'organisation séquentielle des taches fait partie des contraintes imposées. Ainsi l'application d'une commande au niveau du rotor conduit à prendre en compte trois phases:

-Stabilisation de la commande appliquée et obtention de l'équilibre aérodynamique du rotor
-Mesure des effets moyennés sur un ou plusieurs tours
-Calcul de la nouvelle commande

Etant donné la périodicité des effets (période égale à un tour rotor) la synchronisation de l'ensemble s'effectue grâce à un top de référence fourni par le rotor. Dans le cadre de nos simulations un tour rotor est attribué à chaque phase ce qui correspond à réactualiser la commande deux fois par seconde pour un rotor dont la fréquence de rotation est de 6Hz.(figure 7)

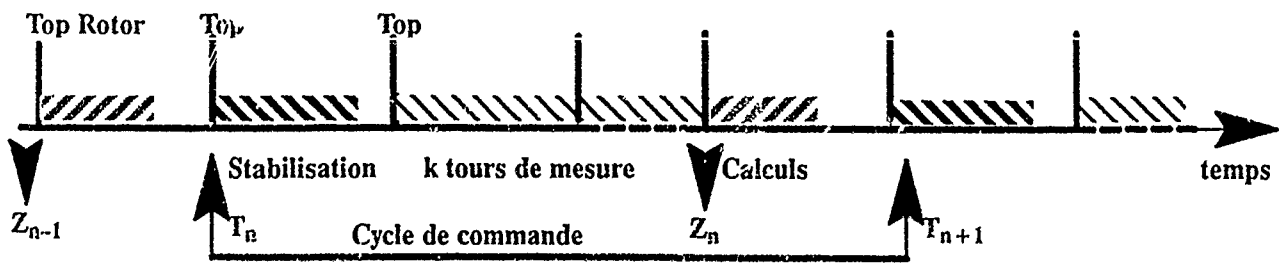


figure 7

5. PHASES DE L'ETUDE

5.1. Résultats de simulation sur modèle d'action

Les simulations ont été effectuées tout d'abord en cas de vol fixe afin de tester la compatibilité du modèle de commande choisi par rapport au modèle d'action. Ceci a permis de montrer que l'algorithme était capable de s'adapter aux non linéarité de type polynomial. Différentes stratégies d'écrétage de commande et de réglage du filtre d'identification ont été envisagées à ce niveau. L'ensemble des simulations réalisées a permis de déterminer l'effet réel de chacun des coefficients du critère et de définir les principes généraux de réglage suivant l'objectif à atteindre. Afin de tester l'adaptativité de l'algorithme des évolutions de cas de vol ont ensuite été envisagées(figure 8).Pour simuler cette évolution un glissement linéaire des coefficients de sensibilité a été programmé. Afin que l'adaptabilité générale du système soit bonne la mise en oeuvre d'une identification à mémoire glissante s'est avérée nécessaire.

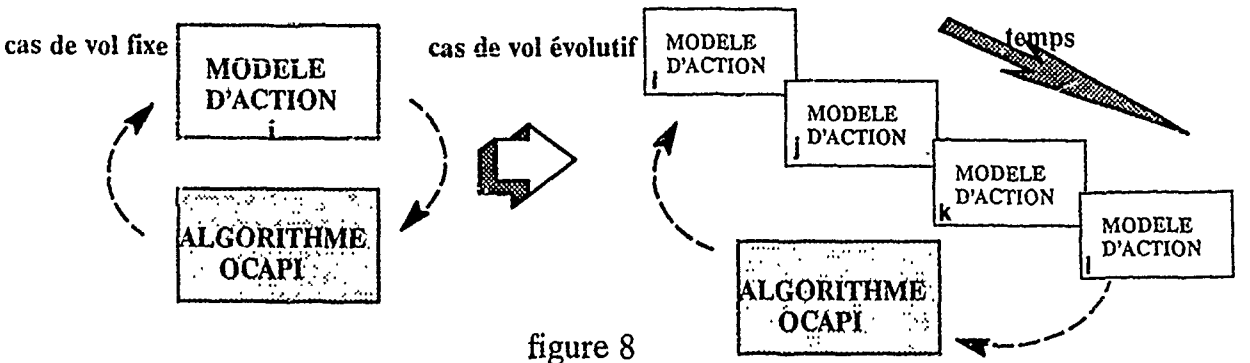
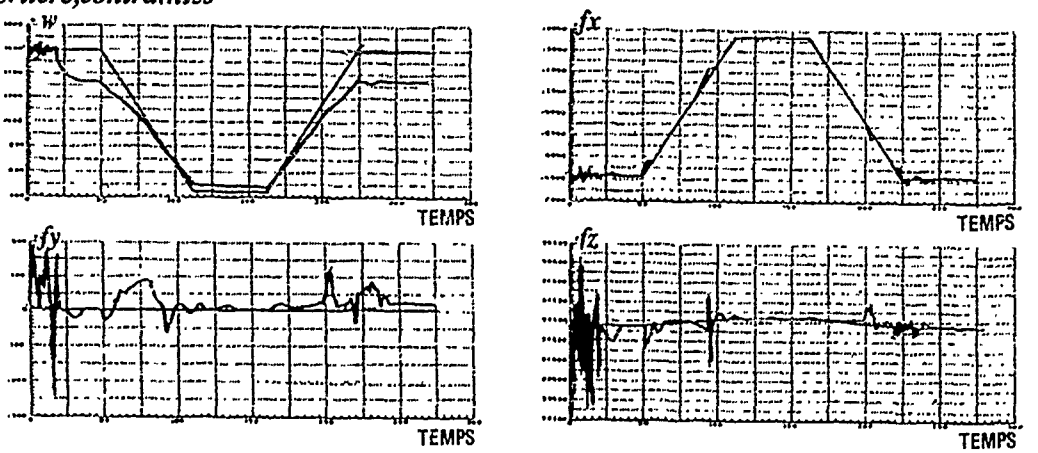


figure 8

Comme on peut le voir sur la figure 9 l'adaptabilité de l'algorithme lui permet d'ajuster ses commandes grâce à la remise en cause de l'identification qui se traduit par l'évolution des commandes. On notera la faible perturbation des efforts alors que la puissance reste optimisée.

Critère,contraintes



Paramètres de commande

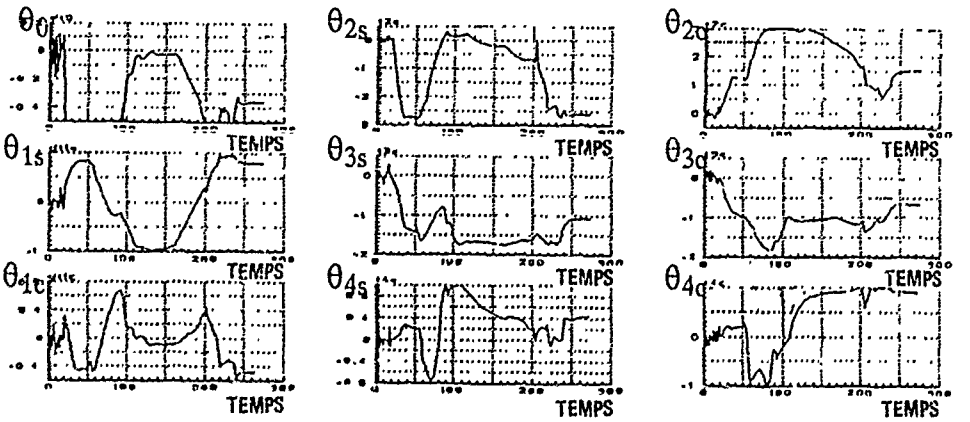


figure 9: ***CHANGEMENT DE CAS DE VOL, OPTIMISATION DE PUISSANCE CHANGEMENT DE MODELE D'ACTION***

5.2. Résultats de simulation sur modèle de connaissance

Cette phase dont le but était de tester la robustesse de l'algorithme en utilisant le modèle aérodynamique de départ à permis d'adapter l'algorithme aux effets non modélisés dans le modèle simplifié. Plus que la simple prise en compte des non linéarités polynomiales d'ordre 3 cela permettait d'introduire les effets se seuil de décrochage au cas où la commande conduirait le rotor dans un tel état. Ces effets ont conduit à fortement augmenter les valeurs des covariances des états jusqu'alors retenues.

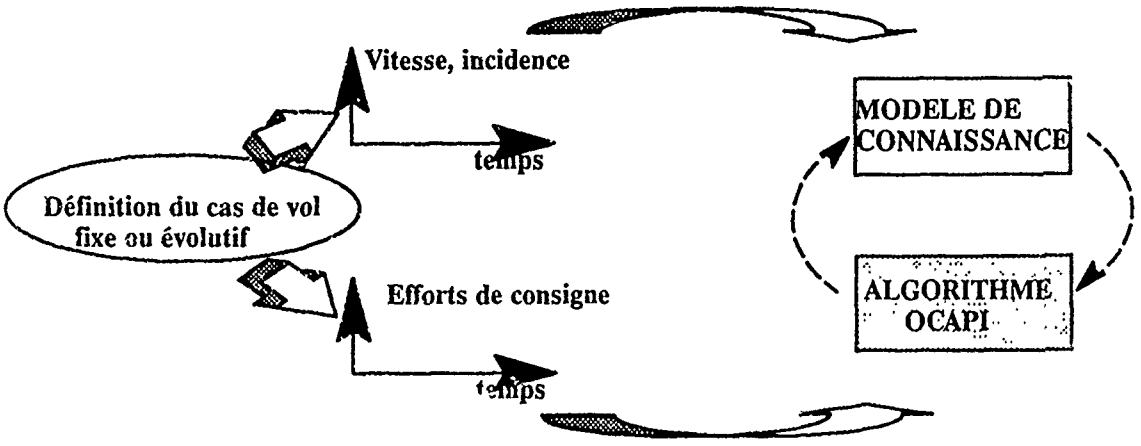


figure 10

Les évolutions de cas de vol ont été modélisées par l'évolution des paramètres d'environnement aérodynamique (vitesse air V et angle d'attaque αq) et par la préprogrammation des consignes d'efforts correspondantes (figure 10). Les courbes présentées sur la figure 11 montrent le découplage temporel que l'on a obtenu grâce aux réglages du filtre ce qui permet de piloter en priorité le cas de vol en augmentant les efforts de portance avant de diminuer la puissance correspondant au nouveau cas de vol.

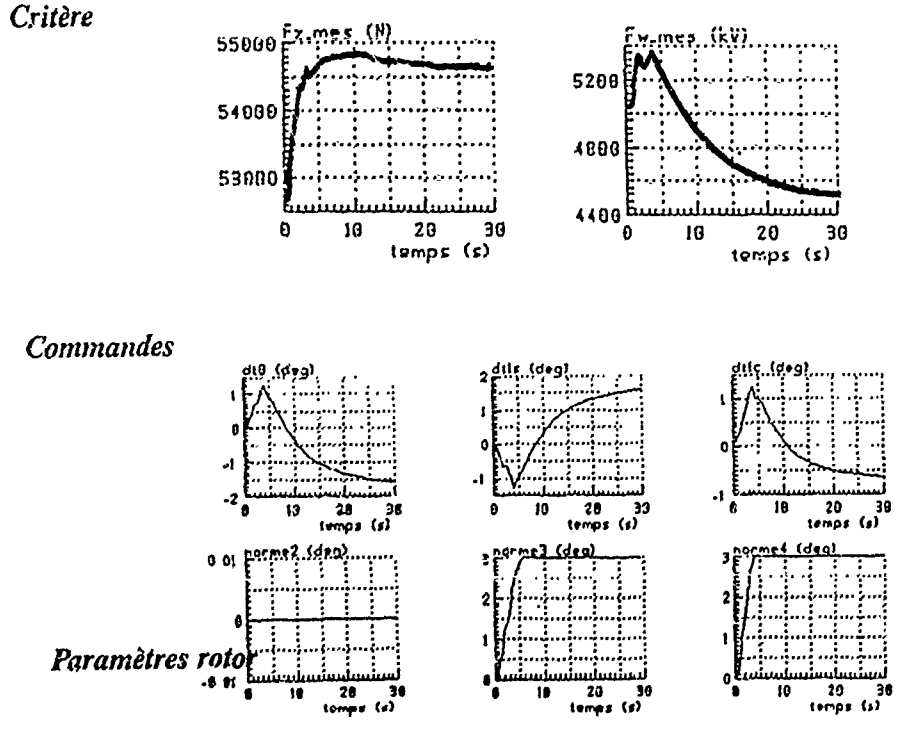


figure 11 : ***OPTIMISATION DE PORTANCE EN VOL STABILISE AVEC CONTRAINTE DE PUISSANCE MODELE R85***

5.3. Prise en compte des capteurs et actionneurs

5.3.1. De la linéarité des actionneurs

Dans le but de définir l'architecture du système de commande la qualité des capteurs et des actionneurs a du être prise en compte.En ce qui concerne les actionneurs une précédente étude liée au multicyclique antivibratoire a montré que toutes les dégradations de type linéaire apportées par ceux ci étaient transparentes vis à vis de l'algorithme de commande qui identifie globalement la sensibilité du système. Déphasages et erreurs de gain sont donc sans effet; seuls les de jeux nécessitent de prendre des précautions. L'expérience passée ainsi que les études actuellement conduites dans le domaine des actionneurs montre que la technologie actuelle est capable de fournir des vérins dont les caractéristiques satisferont les besoins dans ce domaine.

5.3.2. Observateur hybride

Aucun capteur d'effort n'est directement implantable sur l'appareil. Ce problème s'il n'avait pas trouvé de solution aurait été susceptible de remettre en cause le principe même de la tenue du cas de vol ou de l'optimisation de portance qui n'aurait pas été détectable. Pour résoudre ce problème un observateur à été défini afin de recréer à partir de paramètres mesurables les effets des commandes multicycliques (figure 12).Cet observateur basé sur l'hybridation entre informations locales issues des phénomènes générés au niveau du rotor et informations globales représentatives du cas de vol a ensuite été implanté dans la simulation.

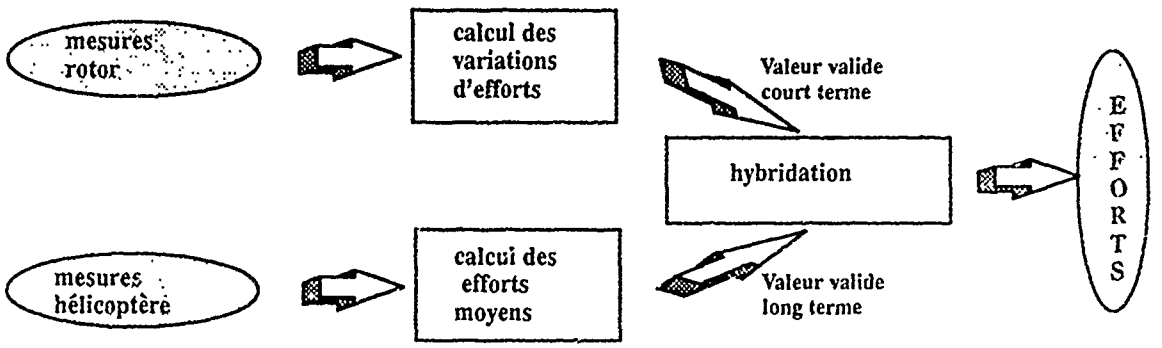


figure 12

5.4. Boucle fermée hélicoptère

Afin d'approcher de façon plus réaliste les effets de la commande optimale des rotors telle que sa mise en oeuvre a été définie au travers de notre algorithme il s'est avéré nécessaire d'envisager une simulation globale permettant de reboucler les variations d'efforts sur les changements de cas de vol ainsi générés (figure 13). Ainsi un modèle hélicoptère a remplacé en simulation les trajectoires prédéterminées du paragraphe 5.2
Ceci a permis de remarquer que la seule tenue d'efforts n'était pas suffisante pour garantir la tenue d'un cas de vol. Un correcteur élaborant des consignes d'effort à partir des écarts de vitesse à été introduit et mis au point.

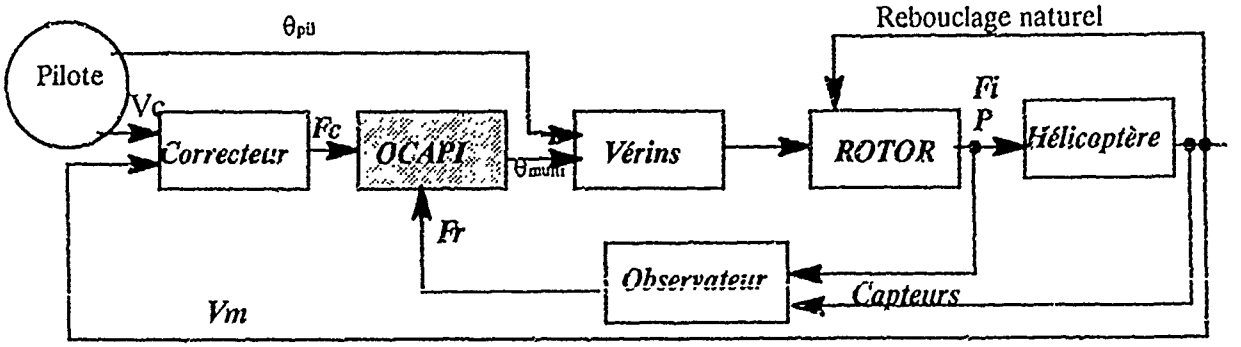


figure 13

Les simulations effectuées sur ce principe ont donné des résultats satisfaisants qui sont présentés dans l'exemple figure 14. On notera que l'algorithme suit la consigne tant que son autorité au niveau des commandes est suffisante. Dans le cas contraire l'algorithme fournit la commande minimisant cet écart. De la même façon que cela avait été remarqué dans le réglage de la saturation d'écart pratiquée pour l'optimisation de portance on notera qu'il est bon de limiter l'écart consigne/mesure si l'on ne veut pas perturber la tenue des consignes sur les autres axes. En effet un écart trop important sur un axe conduit à rendre négligeable les autres termes dans le calcul de la commande optimale.

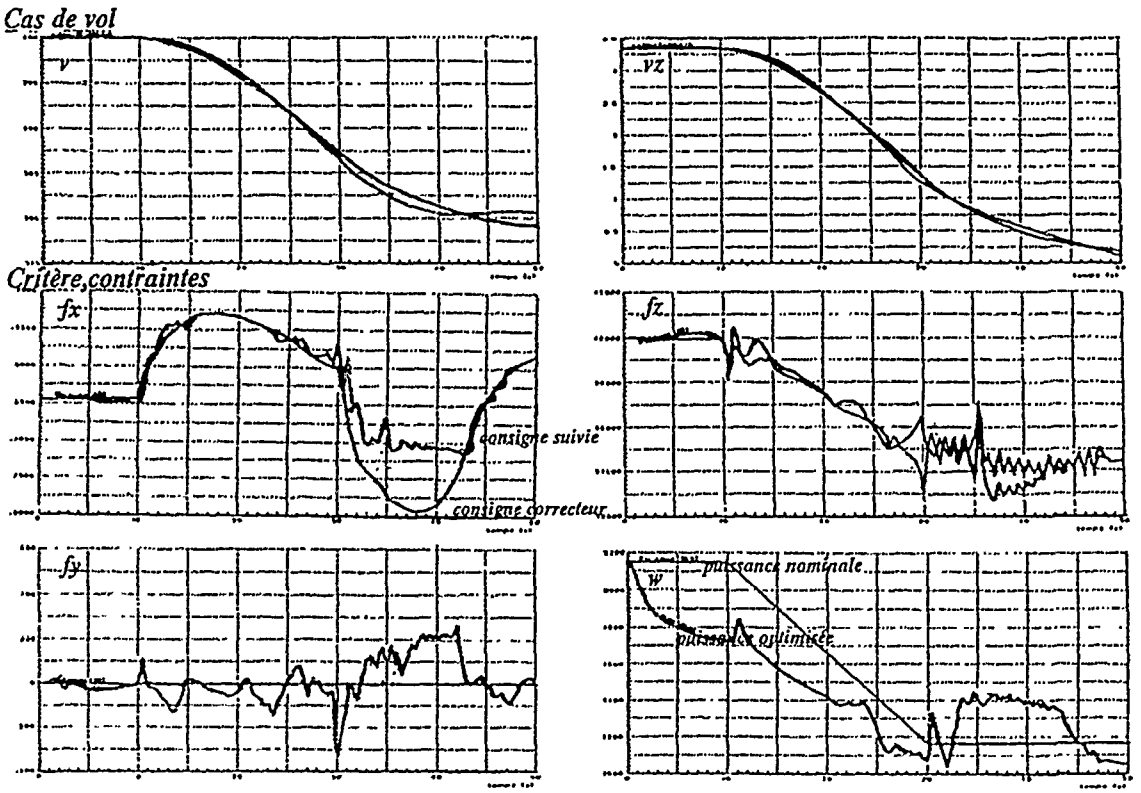


figure 14 : *CHANGEMENT DE CAS DE VOL OPTIMISATION DE PUISSANCE MODELE R85 COUPLE A UN MODELE HELICOPTERE*

5.5. Analyse des résultats

L'objectif de ce chapitre est d'analyser les résultats obtenus au sens du phénomène étudié et des limitations physiques imposées à l'algorithme. Ces analyses basées sur le rotor simulé par le programme R85 sont des analyses entièrement tributaires du modèle et ne sont donc pas significatives dans l'absolu.

5.5.1. Comportement aérodynamique du rotor

L'objet de ce paragraphe n'est pas d'analyser l'ensemble des phénomènes aérodynamiques mais de montrer au travers de la modification du régime aérodynamique du rotor la complexité des phénomènes mis en jeu. Deux termes sont représentatifs de la portance locale ainsi que de la puissance consommée. Il s'agit respectivement du produit Cz Mach au carré et de Cx Mach au cube.

Ayant effectué une simulation dont le critère était l'optimisation de portance (§5.2) à court terme combinée à une limitation de la puissance consommée nous avons tracé les courbes de répartition de portance et de puissance à différents instants de la simulation (figure 15)

Ceci montre une modification conséquente et quasi instantanée de la portance en secteur pale reculante combinée à une perte de puissance equirépartie sur 3 secteurs du rotor. Ceci est lié à la forte part de la fréquence 3Ω dans la commande. Dans un deuxième temps la prise en compte de la minimisation de puissance conduit à une commande différente utilisant une autre répartition 3Ω,4Ω. Il apparait donc que si l'on donne suffisamment de degrés de liberté à l'algorithme celui ci est capable d'atteindre des objectifs multicritères. On constate donc que le choix qui a consisté à utiliser des valeurs moyennées sur un tour rotor (effort statique et puissance moyenne) permet de gérer les problèmes globaux de tenue de cas de vol tout en fournissant une commande qui résout les problèmes locaux liés au décrochage.

5.5.2. Bilan des performances

Disposant de l'algorithme OCAPI nous nous sommes efforcés de le faire fonctionner dans les domaines pour lesquels il conduit à des commandes susceptibles de placer l'appareil en vol dans des domaines inaccessibles à la simple commande cyclique. En effet, si la minimisation de puissance qui avait été notre premier objectif présente un intérêt certain en cas de vol stabilisé pour minimiser la consommation en croisière, elle ne constitue qu'une part des applications du multicyclique (figure 16).

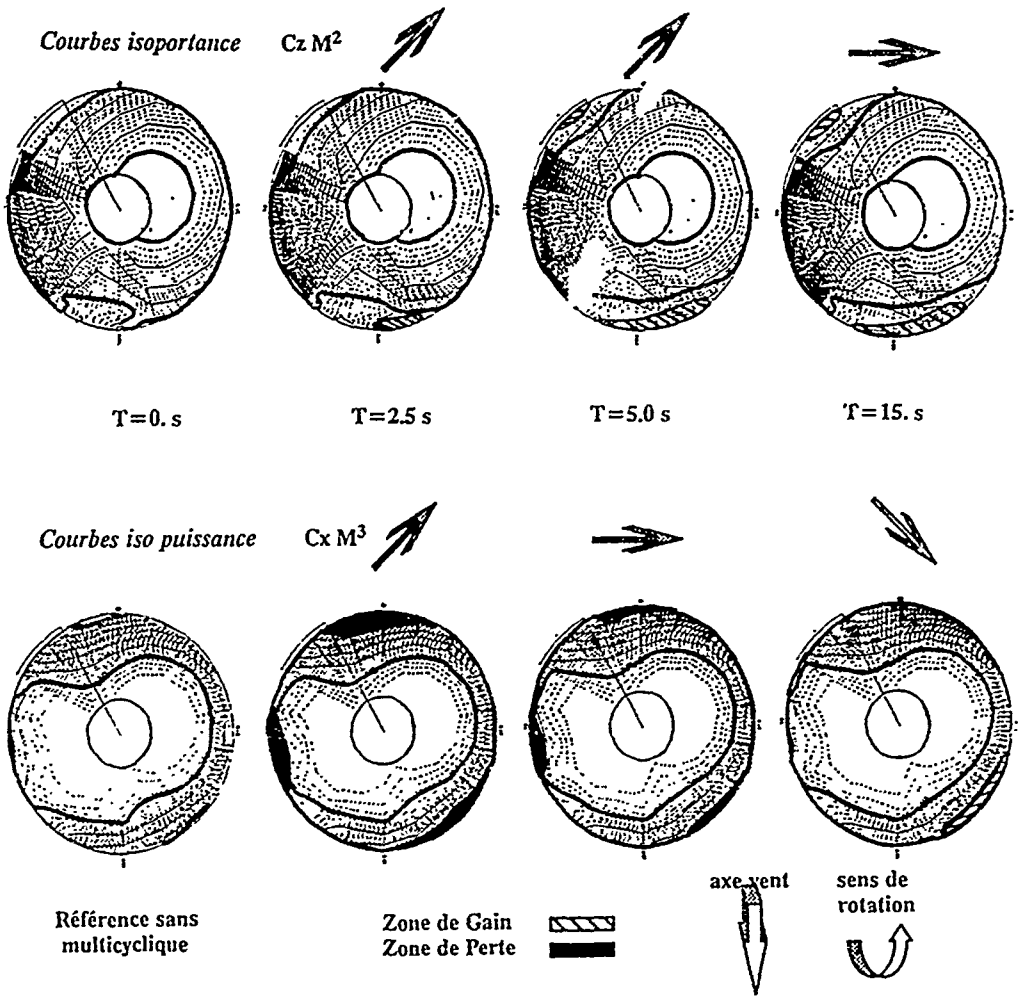


figure 15 : *CARTOGRAPHIE ROTOR*
EVOLUTION AERODYNAMIQUE AU COURS DE L'OPTIMISATION

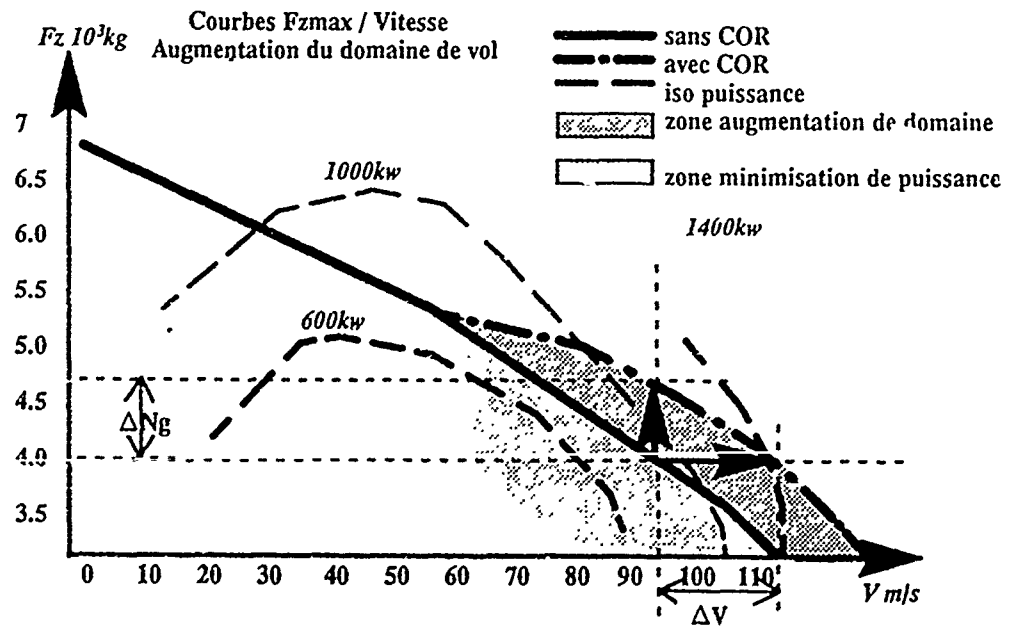


figure 16 : *GAIN POTENTIEL EN DOMAINE DE VOL*

La possibilité d'optimiser traction ou portance se traduisent par des gains en facteur de charge ou des vitesses plus importantes. Disposant d'un point d'équilibre en limite du domaine de vol fourni par le modèle de connaissance, deux types d'optimisation ont été alors effectués avec l'algorithme OCAPI: l'une en commande monocyclique pure, l'autre avec monocyclique et multicyclique. L'écart entre les deux optimisations est donc directement imputable à l'effet multicyclique à la confiance près que l'on a dans la représentativité du modèle de connaissance R85.

Le même type d'exploitation a été réalisé en ce qui concerne l'optimisation de traction. Les résultats obtenus se traduisent par une définition de domaine de vol accessible avec ou sans multicyclique.

6. CONCLUSION

La loi de commande en boucle fermée auto–adaptative obtenue constitue la synthèse d'une méthode d'identification et d'une commande à variance minimale.Ce principe déjà employé pour définir la commande multicyclique anti–vibratoire a été enrichie de la gestion des contraintes de type égalité par l'intermédiaire de l'introduction de pénalités dans le critère de commande.

Tout au long de cette étude d'optimisation des performances rotor la simulation est apparue comme un outil privilégié de synthèse des lois de commande. Son intérêt va bien au delà de la vérification de la validité de solutions scrutées de manière systématique.

La simulation est d'abord présente au coeur même de l'algorithme de contrôle par l'intermédiaire du modèle de commande.

Elle joue de plus un rôle capital dans la méthodologie en introduisant une graduation dans les difficultés abordées. Cette graduation est sensible par la complexité et le réalisme croissants des différents modèles du système à commander (modèle d'action, de connaissance, avec contraintes physiques sur les capteurs ou les vérins ou même avec éléments réels.

L'introduction d'éléments réels qui débouche sur l'essai en soufflerie avec maquette de rotor marque les limites de la simulation numérique pure incapable d'introduire les instationnarités, les cycles limites et autres phénomènes stochastiques liés au décrochage. L'existence toujours possible de non répétitivités, de seuils de sensibilité rend nécessaire le passage en soufflerie qui est prévu en deux temps dans notre programme.

Tout d'abord l'objectif est de valider les aspects algorithmiques à la soufflerie de Chalais sur un rotor tripale dont toutes les harmoniques sont commandables en repère fixe. Une deuxième campagne à la soufflerie Modane avec un rotor quadripale abordera les aspects technologiques de la mise en oeuvre d'une commande en repère tournant. Cette campagne permettra en outre de définir le potentiel de la commande pour les forts facteurs d'avancement.

La simulation ne disparait pas de notre programme puisque, enrichie des résultats de ces campagnes, elle permettra en outre d'approfondir les problèmes liés à l'interaction entre cet algorithme de commande et le pilotage de l'hélicoptère.

La mise au point d'OCAPI (Optimal Controler Adaptativ with Process Identification) constitue donc une étape importante conduisant à la future mise en oeuvre en vol d'une commande multicyclique optimale au sens des performances.

REFERENCES

1–Individual Blade Control Independant of a Swashplate	K.Guinn (Bell)
2–Free Vibration Characteristics of Multiple Load Path Blades by the Transfert Matrix Method	V.R Murthy (Syracuse University)
3–A Unified Formulation of Rotor Load Prediction Methods	R.E Hansford (Westland)
4–Higher Harmonic Control for Helicopters with two–bladed and four bladed rotors	J.G Yen (Bell)
5–Helicopter individual blade control and its applications	N.D Ham (Massachusetts University)
6–Higher Harmonic Control: flight tests of an experimental system on SA 349 research Gazelle	M.Polychroniadis M.Achache (AHS forum 1986)
7–Development of an experimental system for active control of vibration on helicopter	M.Achache M.Polychroniadis (12th European Rotor Aircraft Forum)
8–NASA rotorcraft technology for the 21th Century	J.ALBERS (AIAA/AHS/ASE Seattle 1989)
9–Hub loads analysis of the S.A 349/2 Helicopter	R.M.Heffernan G.K.Yamauchi M.Gaubert W.Johnson (AHS 01/1990)

A DECENTRALIZED CONTROLLER FOR HIGHLY AUGMENTED AIRCRAFT

by

Konur Alp Ünyelioğlu and A. Bülent Özgüler

Bilkent University, PK 8, 06572, Maltepe, Ankara, Turkey

SUMMARY

In this paper we consider the design of a decentralized controller for the yaw pointing/lateral translation control of the FPCC aircraft, to increase the reliability of the closed loop system with respect to absolute sensor failures. We show that better robustness results concerning absolute sensor failures with fixed zero output can be achieved by using decentralized dynamic compensator with high gain in the canard loop, at the expense of reduced phase and gain margins.

1. INTRODUCTION

Advanced fighter aircraft with unconventional control surfaces, such as canards, provide the capability of implementing specialized modes for air-to-air combat, bombing and strafing [1]. The corresponding modes for the lateral dynamics of an aircraft are the direct sideforce control, yaw pointing and lateral translation. One of the main control objectives for these modes is to command a chosen variable without a significant change in the other variables. In [1], yaw pointing/lateral dynamics control of the FPCC aircraft is established by using eigenstructure assignment technique. In [2], an extension of these results is given where the pseudocontrol strategy is employed to improve the stability robustness of the closed loop system. The results show that the design with pseudocontrol strategy provides improved gain and phase margins compared to the design in [1].

An interesting consequence of [1] is that, even when some of the entries in the feedback gain matrix is set to zero, the controller still shows acceptable performance. Motivated by this fact, our objective here is to design a decentralized controller for the yaw pointing/lateral translation control of the FPCC aircraft of [1].

Any controller designed to control the motion of an aircraft is desired to satisfy tolerable performance when the sensors that measure the output variables of the motion have failures, or when the feedback channels are disconnected giving imprecise information to the controller. In this paper we consider the absolute sensor failures causing fixed zero output to the controller. We expect that a decentralized controller, i.e. a controller processing constrained feedback information, can achieve a much more reliable performance compared to the centralized control case when such failures occur. This is because, decentralized controller utilizes less information compared to a centralized controller. Decentralized control, however, has several disadvantages. The order of a decentralized compensator might be greater than that of a centralized controller which achieves the same control objectives. In addition, some of the desired closed loop characteristics may not be achieved by a decentralized controller due to lack of information available ' controller. In this study we allow feedback from the bank angle and roll rate only to aileron, and the heading angle, yaw rate, and the sideslip angle only to the rudder and the vertical canard. A controller under these feedback constraints is designed and tested when one or more rows of the output matrix is set to zero, to model the absolute sensor failures corresponding to the related variables of motion. It is shown that the closed loop system attains the desired control specifications and remains stable for many of the sensor failure combinations. We examine the effect of the same sensor failures in the design of [2] for the purpose of comparison. It is observed that the reliability of the design in [2] is not as tolerable as the decentralized control design. It must be noted however that, the gain and phase margins of [2] is better than the design here, which may be assumed as a drawback of the constrained feedback information that we use.

The design method in [1] is distinguished by its eigenstructure assignment technique. This has the advantage of being suitable for easily reflecting various decoupling requirements on eigenvectors. In cases where exact decoupling is not possible, the best possible eigenvector assignment can be shown to correspond to an approximate decoupling among the modes. In our design, a more direct approach for decoupling is taken and the decoupling among certain modes are achieved by assigning certain entries of the closed loop state matrix to zero.

The following section is mainly concerned with some mathematical theory about the relations between decoupling by eigenstructure assignment and our direct method. In that section, we will also encounter the concept of absolute sensor failures with fixed zero output, in the multivariable system design. The structure of the decentralized controller that we propose is also introduced. Section 3 summarizes our design procedure. The last section contains the design results, and comparisons with some other designs resulting from different methodologies.

2. DIRECT ZERO ASSIGNMENT AND ABSOLUTE SENSOR FAILURES

We begin with the following theorem.

Theorem *Let A be an $r \times r$ real matrix with distinct eigenvalues. Assume V is an $r \times r$ matrix such that its columns are the eigenvectors of A, each corresponding to a distinct eigenvalue. Then, V has the following block triangular structure*

$$V = \begin{bmatrix} V_{11} & \cdots & V_{1N} \\ 0 & \ddots & \vdots \\ 0 & 0 & V_{NN} \end{bmatrix},$$

if and only if A has the following block triangular structure

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ 0 & \ddots & \vdots \\ 0 & 0 & A_{NN} \end{bmatrix}.$$

The proof of the theorem utilizes the linear dependence relations on the matrices $\lambda_i I - A$, where $\lambda_1, \lambda_2, ..., \lambda_r$ are the eigenvalues of A. We omit this straightforward proof, and just notice that the statement of the theorem still holds, when the term "triangular" is replaced by "diagonal".

Now consider the linear, time-invariant system represented by the equation

$$\dot{x} = Ax, \tag{1}$$

in which we neglected the effect of the input on the behavior of the system, so as to investigate the zero input transient behavior. As an example, assume the following eigenvectors correspond to the distinct eigenvalues of the A matrix

$$\begin{bmatrix} \times \\ \times \\ 0 \end{bmatrix}, \quad \begin{bmatrix} \times \\ \times \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 0 \\ \times \end{bmatrix}.$$

It can be shown that the above structure indicates a mode decoupling between the states of the system in Eq.(1). That is, when the first and second states have nonzero, appropriate initial values, and the third state has zero initial value, the zero input response of the system contains only the dynamics of the first two states. Conversely, when the first two states have zero initial value, and the last state has an appropriate initial condition, the zero input response of the system includes only the dynamics of the last state, while the first two states remain zero.

Since it is an important design objective to obtain mode decoupling between several dynamics of an airplane, the eigenstructure assignment is a useful design tool in the flight control problems. This method is also suitable when the desired eigenvector assignment is not possible by the available input/output structure. In this case, the set of best possible eigenvectors are assigned, by projecting onto some achievability subspace. The details of the eigenstructure assignment design technique can be found in [4]. In this study, to obtain mode decoupling, we approach the problem in a more direct way, by considering the structure of the closed loop state matrix. The relation between the structures of the eigenvectors and the closed loop state matrix, then can be obtained from the previous theorem. The desired specifications on the closed loop state matrix will be further explained in the following section.

The reliability of flight control systems is a vital problem. During flight conditions a sensor, or a combination of sensors might undergo a failure, causing the loss of sensor signals that are processed by the controller for feedback purposes. As a result of this, the closed loop system may become unstable, and the desired control objectives may not be obtained appropriately. In our design, by employing a decentralized feedback structure, we hope to increase the reliability of the closed loop system with respect to absolute sensor failures. There are several ways to model such failures, such as in [3] (see also the references therein). Here, we consider the absolute sensor failures with fixed zero output to the controller. This can be modeled as setting one or more rows of the output matrix in the state space description of the aircraft model, to zero. After completing the overall design, we simulate the closed loop system to observe several combinations of absolute sensor failures with fixed zero output.

We now introduce the decentralized feedback structure utilized in the design process. The allowed feedback structure can be expressed as follows:

$$\begin{array}{c|ccccc} & \beta & \psi & r & \phi & \rho \\ \delta_r & \times & \times & \times & 0 & 0 \\ \delta_e & \times & \times & \times & 0 & 0 \\ \delta_a & 0 & 0 & 0 & \times & \times \end{array},$$

where $\times$'s denote the allowable feedback entries and 0's denote the feedback entries that are set to zero. The output variables are the sideslip angle β, the heading angle ψ, the yaw rate r, the bank angle ϕ, and the roll rate ρ. The input variables are the deflections in the rudder δ_r, in the aileron δ_a, and in the vertical canard δ_e.

3. DESIGN METHODOLOGY

Consider the aircraft model represented by the following state space equations:

$$
\mathcal{S}:\quad
\frac{d}{dt}\begin{bmatrix}\beta\\ r\\ \gamma\\ \phi\\ \rho\end{bmatrix}
=\begin{bmatrix}-0.34 & -0.997 & 0 & 0.0517 & 0.001\\ 5.91 & -0.506 & 0 & 0 & 0.138\\ -0.34 & 0.0031 & 0 & 0.0517 & 0.001\\ 0 & 0 & 0 & 0 & 1\\ -2.69 & 0.738 & 0 & 0 & -1.15\end{bmatrix}\begin{bmatrix}\beta\\ r\\ \gamma\\ \phi\\ \rho\end{bmatrix}
+\begin{bmatrix}0.0755 & 0.0246 & 0\\ -5.03 & 0.984 & 0.0998\\ 0.0755 & 0.0246 & 0\\ 0 & 0 & 0\\ 4.48 & -0.742 & 5.22\end{bmatrix}\begin{bmatrix}\delta_r\\ \delta_c\\ \delta_a\end{bmatrix}, \tag{2}
$$

$$
y=\begin{bmatrix}1 & 0 & 0 & 0 & 0\\ -1 & 0 & 1 & 0 & 0\\ 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\beta\\ r\\ \gamma\\ \phi\\ \rho\end{bmatrix},
$$

where the state matrix is A, the input matrix is B, and the output matrix is C. The state variables are the sideslip angle β, the yaw rate r, the lateral/directional flight path angle $\gamma(=\psi+\beta)$, the bank angle ϕ, and the roll rate ρ.

A more detailed linearized model of the plant is given in [1]. The above model is a reduced order description, where the actuator dynamics are ignored; it is identical to the model employed in [2]. We are now concerned with the following control objectives:

i. Pole placement: The closed loop system is required to have the following eigenvalues

$$
\begin{array}{ll}-3\pm 2i & : \text{ roll mode}\\ -2\pm 2i & : \text{ dutch roll mode}\\ -0.5 & : \text{ flight path mode}\end{array}
$$

ii. Mode decoupling: It is desired that the flight path mode is decoupled from the dutch roll mode. It is further aimed that these two modes are decoupled from the roll mode. These requirements can be explained more precisely by imposing them on the structure of the closed loop state matrix, as below:

$$
A_c=\begin{bmatrix}a_{11} & a_{12} & a_{13} & 0 & 0\\ a_{21} & a_{22} & a_{23} & 0 & 0\\ 0 & 0 & a_{33} & 0 & 0\\ 0 & 0 & 0 & a_{44} & a_{45}\\ 0 & 0 & 0 & a_{54} & a_{55}\end{bmatrix},
$$

where

$$
\begin{bmatrix}a_{33}-a_{11}\\ -a_{21}\end{bmatrix}=k\begin{bmatrix}a_{13}\\ a_{23}\end{bmatrix} \tag{3}
$$

for some real number k.

The theorem in the previous section, and some further manipulations utilizing the linear dependence relation represented by Eq.(3) yield that the eigenvector structure of A_c is as follows:

$$
\begin{bmatrix}\times\\ \times\\ 0\\ 0\\ 0\end{bmatrix},\ \begin{bmatrix}\times\\ \times\\ 0\\ 0\\ 0\end{bmatrix}\ :\ \text{dutch roll mode}
$$

$$
\begin{bmatrix}\times\\ 0\\ \times\\ 0\\ 0\end{bmatrix}\ :\ \text{flight path mode}
$$

$$
\begin{bmatrix}0\\ 0\\ 0\\ \times\\ \times\end{bmatrix},\ \begin{bmatrix}0\\ 0\\ 0\\ \times\\ \times\end{bmatrix}\ :\ \text{roll mode.}
$$

In this case, the roll mode eigenvalues correspond to the eigenvalues of the lower diagonal submatrix, and the dutch roll and flight path mode eigenvalues correspond to the upper diagonal submatrix of A_c.

iii. Tracking: A feedforward controller that yields zero steady state error due to step inputs in either heading angle or lateral flight path angle is to be designed.

The controller utilizes a constrained feedback information as described in the previous section. It can easily be shown that the system has no fixed modes with respect to this feedback structure [5], that is, the system can be stabilized by applying (possibly dynamic) feedback coinciding the above structure.

We start with an initial constant feedback

$$K = \begin{bmatrix} k_{11} & k_{12} & k_{13} & 0 & 0 \\ k_{21} & k_{22} & k_{23} & 0 & 0 \\ 0 & 0 & 0 & k_{34} & k_{35} \end{bmatrix}. \tag{4}$$

Define $A_c := A - BKC$. We would like to obtain,

$$A_c = \begin{bmatrix} A_{c11} & A_{c12} \\ 0 & \begin{matrix} 0 & 1 \\ -13 & -6 \end{matrix} \end{bmatrix}, \tag{5}$$

where A_{c11} and A_{c12} are 3×3 and 3×2 matrices, which will provide that the flight path mode and the dutch roll mode are decoupled from the roll mode. In addition, the roll mode eigenvalues $-3\pm 2i$ are attained. This is because, the lower left submatrix of Eq.(4) is zero, and the lower right submatrix of Eq.(4) is the submatrix corresponding to the roll mode.

The vertical canard is the least effective of the control surfaces. Using high gain in the canard loop might cause increased sensitivity with respect to plant perturbations [2]. So, to keep the feedback gains corresponding to the vertical canard as low as possible, we let $k_{21} = k_{22} = 0.3$, and $k_{23} = 0.6$. Then, by choosing

$$[k_{34}\ \ k_{35}] = [2.4904\ \ 0.9291],$$

and

$$\begin{bmatrix} k_{11} \\ k_{12} \\ k_{13} \end{bmatrix} = \begin{bmatrix} -4.48 & 4.48 & 0 \\ 0 & 0 & -4.48 \\ 0 & -4.48 & 0 \end{bmatrix}^{-1} \cdot \left(\begin{bmatrix} 2.69 \\ -0.738 \\ 0 \end{bmatrix} - \begin{bmatrix} 0.742 & -0.742 & 0 \\ 0 & 0 & 0.742 \\ 0 & 0.742 & 0 \end{bmatrix} \begin{bmatrix} k_{21} \\ k_{22} \\ k_{23} \end{bmatrix} \right),$$

Eq.(4) becomes

$$A_c = \begin{bmatrix} -0.2927 & -1.0317 & -0.0111 & 0.0517 & 0.001 \\ 2.8898 & 0.2321 & -0.0453 & -0.2485 & 0.0453 \\ -0.2927 & -0.0316 & -0.0111 & 0.0517 & 0.0010 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -13 & -6 \end{bmatrix}. \tag{6}$$

The upperleft 3×3 submatrix of Eq.(6) corresponds to the interaction between the dutch roll and flight path modes. We now want to locate the eigenvalues of these modes to $-2\pm 2i$, and -0.5. We, therefore, consider the corresponding subsystem,

$$\begin{aligned} \mathcal{S}_2 : \\ \dot{\tilde{x}} &= \tilde{A}\tilde{x} + \tilde{B}(\tilde{u} + \tilde{v}) \\ \tilde{y} &= \tilde{C}\tilde{x} \end{aligned} \tag{7}$$

where

$$\tilde{A} = \begin{bmatrix} -0.2927 & -1.0317 & -0.0111 \\ 2.8898 & 0.2321 & -0.0453 \\ -0.2947 & -0.0316 & -0.0111 \end{bmatrix}$$

$$\tilde{B} = \begin{bmatrix} 0.0755 & 0.0246 \\ -5.03 & 0.984 \\ 0.0755 & 0.0246 \end{bmatrix}, \quad \tilde{u} = \begin{bmatrix} \delta_r \\ \delta_c \end{bmatrix},$$

and

$$\tilde{C} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

The following dynamic output feedback law

$$\begin{aligned} \dot{\tilde{z}} &= \tilde{a}_z\tilde{z} + \tilde{F}\tilde{y} \\ \tilde{v} &= \tilde{H}\tilde{z}, \end{aligned}$$

will be used to locate the desired eigenvalues in the resulting closed loop system below,

$$\tilde{\mathcal{S}}_2 : \quad \begin{bmatrix} \dot{\tilde{x}} \\ \dot{\tilde{z}} \end{bmatrix} = \tilde{A}_e \begin{bmatrix} \tilde{x} \\ \tilde{z} \end{bmatrix} + \begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} \tilde{u},$$

where

$$\tilde{A}_e := \begin{bmatrix} \tilde{A} & \tilde{B}\tilde{H} \\ \tilde{F}\tilde{C} & \tilde{a}_z \end{bmatrix}. \tag{8}$$

In this case, the overall closed loop system becomes

$$\tilde{S}: \quad \begin{bmatrix} \dot{x} \\ \dot{\tilde{z}} \end{bmatrix} = \begin{bmatrix} A - BKC & B\tilde{H} \\ FC & \tilde{a}_z \end{bmatrix} \begin{bmatrix} x \\ \tilde{z} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \begin{bmatrix} \delta_r \\ \delta_c \\ \delta_a \end{bmatrix}, \tag{9}$$

where

$$F := [\tilde{F} \ 0 \ 0], \text{ and } H := \begin{bmatrix} \tilde{H} \\ 0 \end{bmatrix}.$$

Let

$$\begin{bmatrix} \tilde{h}_{11} \\ \tilde{h}_{21} \end{bmatrix} := \tilde{H}.$$

To maintain the decoupled structure of the roll mode and the other two modes, we need to set $\tilde{h}_{21} = 6.0377\tilde{h}_{11}$. This is because of the fact that the last two rows of BH , that correspond to the effect of the lateral/directional flight path response and yaw rate response on the roll rate and bank angle responses, equal

$$\begin{bmatrix} 0 \\ 4.48\tilde{h}_{11} - 0.742\tilde{h}_{21} \end{bmatrix}.$$

Furthermore, we require the eigenvalue corresponding to the compensator to be -1. So, solving a set of equations for $\tilde{h}_{11}$, $\tilde{A}_2$, and $\tilde{F}$ we obtain,

$$\tilde{h}_{11} = 1/6.0377, \ \tilde{a}_z = -5.4263,$$

and

$$\tilde{F} = [-83.4096 \ \ -24.7418 \ \ -53.6523],$$

which gives us that the eigenvalues of Eq.(8) are $-2 \pm 2i$ and -0.5.

A final refinement is made on the feedback channel between the sideslip angle and the aileron, by perturbing the corresponding entry in the initial constant feedback matrix from 0 to 0.1. The purpose of this further adjustment is the following. If the sensor measuring the bank angle undergoes a failure, information about this state variable is available to the part of the controller that actuates the aileron. Since ϕ influences the behavior of β, allowing feedback from β to the aileron controller provides better robustness with respect to a sensor failure in the bank angle. On the other hand, this modification causes very slight changes in the eigenvalues of the closed loop matrix obtained so far. In this case, the closed loop eigenvalues, i.e. the eigenvalues of A_e in Eq.(9) becomes,

$$\begin{array}{ll} -2.99 \pm 2.02i & \text{roll mode} \\ -2.01 \pm 1.97i & \text{dutch roll mode} \\ -0.50 & \text{flight path mode} \\ -1.02 & \text{compensator,} \end{array}$$

whereas A_e is given by

$$\begin{bmatrix} -0.2947 & -1.0317 & -0.0111 & 0.0370 & 0.0517 & 0.0010 \\ 2.8798 & 0.2321 & -0.0453 & 0.1594 & -0.2485 & 0.0453 \\ -0.2947 & -0.0316 & -0.0111 & 0.0370 & 0.0517 & 0.0010 \\ -54.7665 & -51.8319 & -24.4336 & -5.4263 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -0.5220 & 0 & 0 & -0.0076 & -12.9999 & -5.9999 \end{bmatrix}.$$

Note that in the above matrix the row and column corresponding to a_z are permuted into the upper subblock for convenience. With this new notation, the corresponding states in the above matrix are in the following order: β, r, γ, a_z, ϕ, ρ. It is seen that the decoupling objective of the roll mode from the dutch roll and flight path modes are satisfactory, while the dutch roll and flight path modes show some undesirable coupling.

To accomplish the tracking objectives we now design a feedforward controller. As in [1], the controlled variables are ψ, γ, and ϕ. The heading command sets ψ to 1, the lateral flight path command sets γ to 1, and the bank angle ϕ is set to 0 in both cases.

Let

$$C_e = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \ B_e = \begin{bmatrix} B \\ 0 \ 0 \ 0 \end{bmatrix}.$$

Then $G_e(s) := C_e(sI - A_e)^{-1}B_e$ is the transfer matrix from $[\delta_r \ \ \delta_c \ \ \delta_a]'$ to $[\psi \ \ \gamma \ \ \phi]'$. The steady state error of the system combined with a feedforward controller E, due to an input $v(s)$ is given by

$$\lim_{s \to 0} s(I - G_e(s)E)v(s). \tag{10}$$

The signal to be tracked can be expressed by

$$v(s) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

Then, Eq.(10) becomes

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} = -C_e A_e^{-1} B_e E \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

This last equation yields that the submatrix of E, consisting of its first two columns can be expressed by

$$\begin{bmatrix} -3.6058 & 4.4107 \\ -17.2192 & 22.0788 \\ 0.5470 & -0.5470 \end{bmatrix}.$$

Note that, since $C_e A_e^{-1} B_e$ is nonsingular, the first two columns of E are unique. The last column, however, is arbitrary, because the bank angle Φ is set to zero in both of the heading and the lateral flight path commands.

4. CONCLUSIONS

The performance of the overall system, compared with two other systems, which are designed according to different methodologies is shown in Table 1. One of these systems under comparison is designed according to [2]. The design method of the other system is the constrained feedback eigenstructure design method proposed in [4]. It is seen that when s_β and s_r or s_β, s_r, and s_ρ are defective, then some of the closed loop eigenvalues of the centralized design have positive real parts, which implies that the closed loop is unstable. On the other hand, the decentralized closed loop system still remains stable under the same sensor failures. It is observed that the constraint feedback eigenstructure design also shows a reliable performance in all of the sensor failures considered in the table. Another important result is shown in Figure 1. When s_ϕ, s_ρ, and s_r have failure, the centralized design is still stable, but it shows an oscillatory motion. In this case, the decentralized design achieves a better performance and reaches to its steady state value without oscillation. The behavior of the control surfaces in this example shows that the canard variation has a significant influence on the compensation of the sensor failures. In the decentralized case, the canard movement is greater than the movement in the centralized case. This, however, is a disadvantage, since using high gain in the canard loop causes higher sensitivity in the closed loop. As a result, we have reduced gain and phase margins in the decentralized design. This can be understood more clearly by investigating $inf\ \underline{\sigma}(I + Z_c Z)$. In our case, this number equals approximately 0.035, whereas the centralized design achieves 0.95. These results can be related to the gain and phase margins directly, as in [2]. We just note that the lower $inf\ \underline{\sigma}(I + Z_c Z)$ implies lower gain and phase margins and vice versa.

The characteristics of the decentralized design under normal conditions are seen in Figure 2. It is observed that the response for the lateral flight path command is satisfactory, whereas the heading angle response shows a coupling between the lateral flight path angle and the heading command. However, in the case of constrained feedback eigenstructure assignment design this decoupling is eliminated. Unfortunately, in this case the coupling between the bank angle and the heading angle command has a significant increase.

Since, theoretically, it is quite difficult to solve a noninteracting control problem with pole placement using a decentralized controller, a suitable design algorithm is not yet available which satisfies these two uncompromising constraints. It is hoped however that our initial results in this paper stimulate research in this direction.

5. REFERENCES

[1] Sobel, K. M. and Shapiro, E. Y., "Application of eigensystem assignment to lateral translation and yaw pointing flight control," *Proceedings of the 23'rd IEEE Conference on Decision and Control*, Las Vegas, NV, 1984, pp. 1423-1428.

[2] Sobel, K. M. and Lallman, F. J., "Eigenstructure assignment for the control of highly augmented aircraft," *Journal of Guidance, Control and Dynamics*, Vol. 12, No. 3, 1985, pp. 318-324.

[3] Mc Lean, D. and Alkhatib, K., "An analytical redundancy scheme for flight control systems," *Aeronautical Journal*, November 1985, pp. 353-361.

[4] Andry, A. N., Shapiro, E. Y. and Chung, J. C., "Eigenstructure assignment for linear systems," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-19, September 1983, pp.711-729.

[5] Wang S. and Davison E. J., "On the stabilization of decentralized control systems," *IEEE Transactions on Automatic Control*, Vol. AC-18, No. 5, 1973, pp. 473-478.

Table 1. FPCC sensor failure characteristics

Failed sensors	Decentralized design e.values	Centralized design e.values	Constrained f/b design e.values
s_r, s_ρ	-4.2	$-0.58 \pm 3.50i$	$-0.68 \pm 3.46i$
	$-0.57 \pm 3.54i$	$-0.11 \pm 3.02i$	$-0.14 \pm 2.83i$
	$-0.52 \pm 0.83i$	-0.45	-0.5
	-1		
s_ϕ, s_ρ	$-2.00 \pm 2.01i$	$-2.00 \pm 2.00i$	$-2.00 \pm 1.89i$
	$-0.59 \pm 0.10i$	-0.42	$-0.18 \pm 0.23i$
	-1.44	-1.17	-1.18
	-0.02	-0.04	
s_ϕ, s_ρ, s_r	-4.20	$-0.1 \pm 2.96i$	$-0.30 \pm 2.75i$
	$-0.47 \pm 0.75i$	-1.15	$-0.18 \pm 0.23i$
	$-1.12 \pm 0.16i$	-0.04	-1.18
	-0.0259	-0.42	
s_β, s_r	$-0.40 \pm 2.42i$	$0.01 \pm 2.46i^*$	$-0.20 \pm 3.70i$
	$-3.00 \pm 1.98i$	$-2.99 \pm 1.98i$	$-3.01 \pm 1.90i$
	-0.23	-0.68	-0.41
	-5.23		
s_β, s_r, s_ρ	$-0.40 \pm 2.44i$	$0.04 \pm 2.49i^*$	$-0.68 \pm 3.61i$
	$-0.58 \pm 3.52i$	$-0.60 \pm 3.52i$	$-0.11 \pm 3.60i$
	-5.23	-0.67	-0.41
	-0.23		

* unstable eigenvalues.

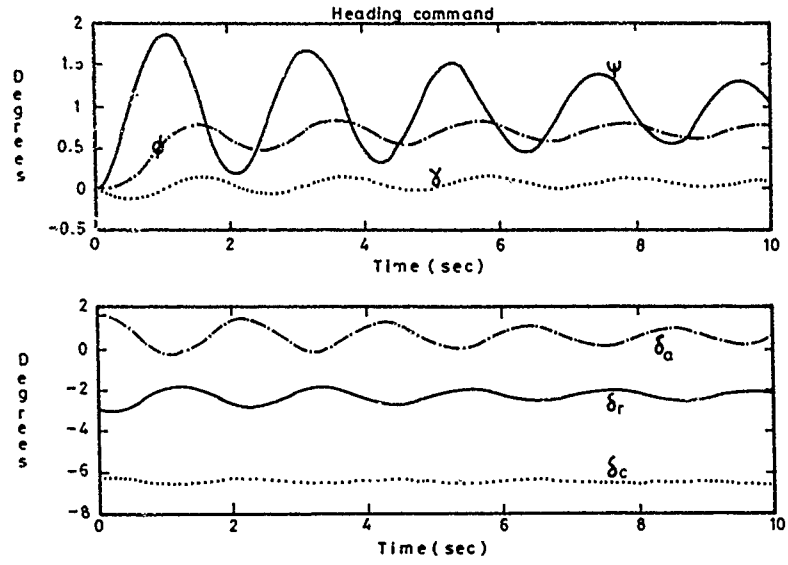


Figure 1.a: Heading command response of the centralized design when s_ϕ, s_ρ and s_r have failure.

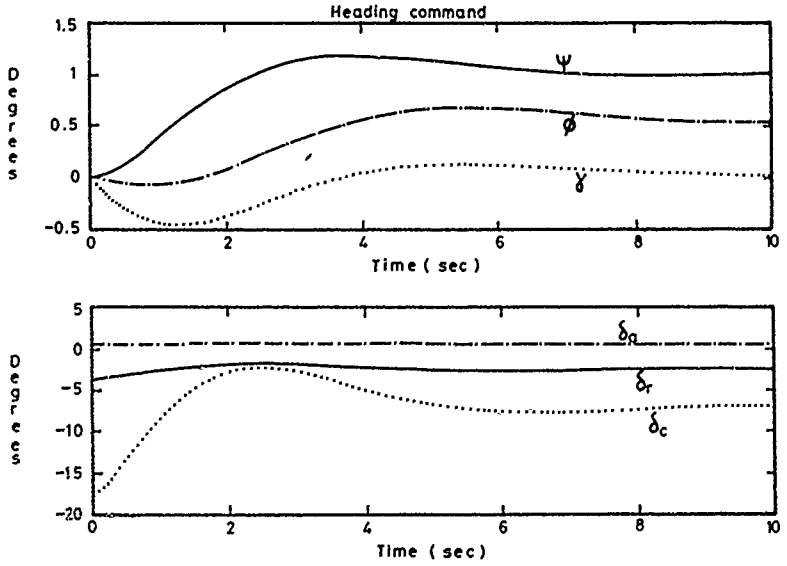


Figure 1.b: Heading command response of the decentralized design when s_ϕ, s_ρ and s_r have failure.

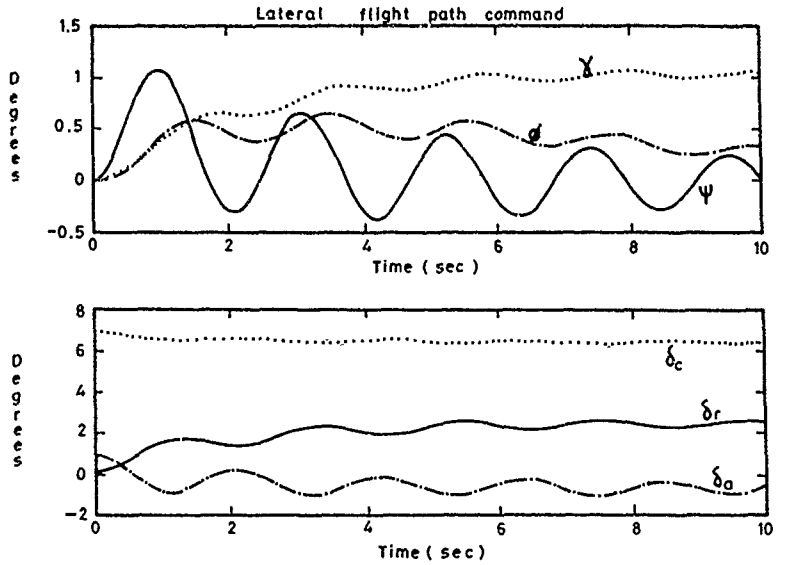


Figure 1.c: Lateral flight path command response of the centralized design when s_ϕ, s_p and s_r have failure.

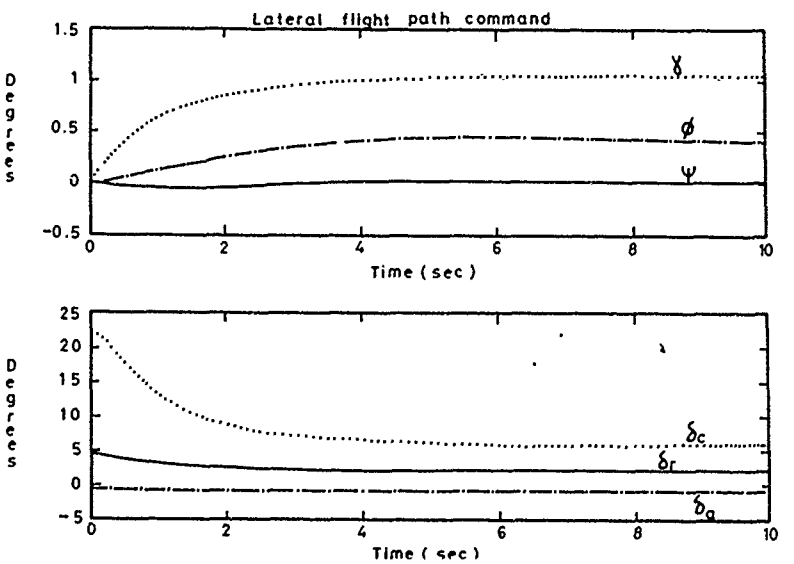


Figure 1.d: Lateral flight path command response of the decentralized design when s_ϕ, s_p and s_r have failure.

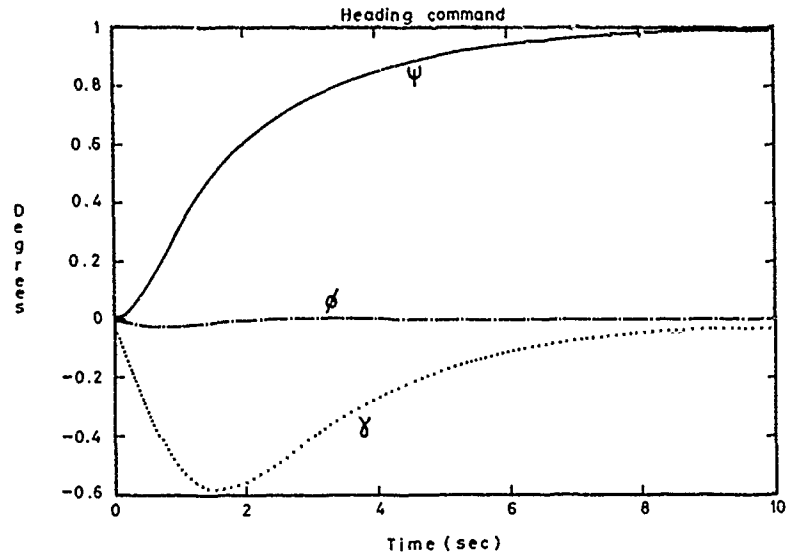


Figure 2.a: Heading command response of the decentralized design under normal conditions.

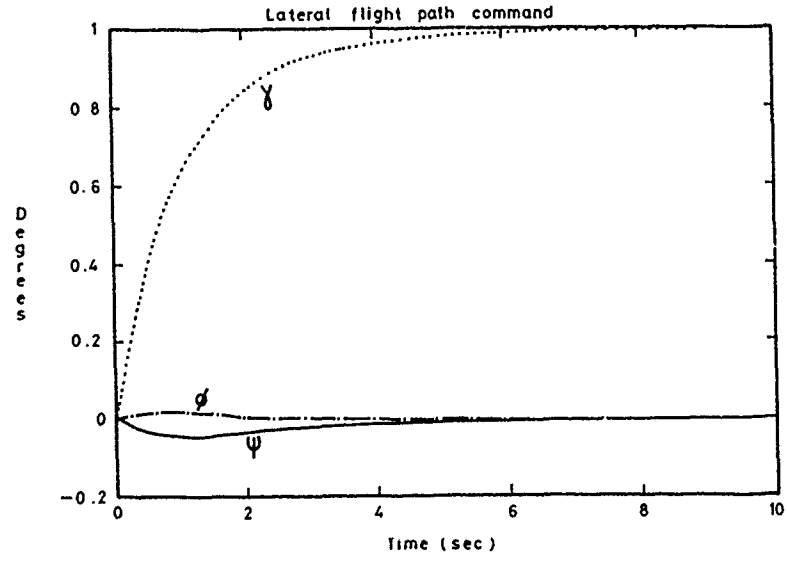


Figure 2.b: Lateral flight path command response of the decentralized design under normal conditions.

PARAMETER SPACE DESIGN OF ROBUST FLIGHT CONTROL SYSTEMS

A. Cavallo G. De Maria
Dipartimento di Informatica & Sistemistica
Universitá degli Studi di Napoli
via Claudio 21
80125 Napoli
ITALY

L. Verde
Centro Italiano Ricerche Aerospaziali (C.I.R.A)
via Maiorise
81043 Capua (CE)
ITALY

SUMMARY

Generally, high performance requirements in terms of better efficiency (reduction in fuel comsuption) and manoeuvrability, impose intrinsic instability on the aircraft. Then a Stability Augmentation System is required for its stabilization. Moreover, the aircraft must be safely controllable without any exceptional piloting skill. The requirements of stability and control are referred in literature as handling qualities. According to handling quality specifications, a feedback controller must be designed with robustness criteria with respect to flight conditions and sensor failure. In this paper a new design procedure of feedback controllers which allow to achieve simultaneous stabilization, and provide some kind of fault tolerance with respect to sensor failure, will be proposed. An application to the F4-E military aircraft will be also presented.

INTRODUCTION

The modern Control Configurated Vehicle (C.C.V) design technologies aims at improving the handling qualities (stability and control) or at conferring them artificially when the modern aerodynamic configurations, directed to obtain a better efficiency (reduction in fuel consumption), better performances and manovrability, impose intrinsic instability on the aircraft.
In both cases it is necessary to ensure that the above performances are guaranteed on the whole flight envelope of the aircraft and over a wide range of center-of-gravity locations.
Moreover if the aircraft is unstable, the Stability Augmentation System (S.A.S.) is vital for its stabilization. This means that the added S.A.S. must be designed with robustness criteria also with respect to actuators effectiveness, sensor failure and software reliability.
By using linearized model of longitudinal and lateral-directional closed-loop dynamics, with reference to an equilibrium flight path and to fixed values of aerodynamic, propulsive, inertial, structural and controller parameters, the handling qualities are expressed in terms of pole location with an assigned tolerance in the complex plane (MIL-F-8785). The above linearized model can be fully parameterized with respect to the most important parameters such as

a) flight condition parameters (altitude h, mach number M);
b) sensor gains;
c) aerodynamic coefficients.

Such a parameterization allows the modern stability and control engineers to design a controller which assigns closed-loop poles in specified domains of the complex plane according to handling quality specifications, with robustness characteristics with respect to flight conditions and sensor failures [1].
In this paper we propose a new design method of robust stabilizing controllers for single-input-multi-output systems based on redundant compensators [2], [3] and on a computationally tractable robust stability test in parameter space with respect to assigned domain of the complex plane [4], [5].
Stabilizing compensators were introduced by Youla, Jabr and Bongiorno in 1976 [6]. Recently in [2] stabilizing compensators of assigned maximum order have been introduced. The design technique proposed in [2] allows the designer to assign closed-loop poles in specified domains of the complex plane and to have at disposal a specified number of degrees of freedom in order to meet other design specifications.
Moreover in the very last years the automatic control literature has grown rich with new methodologies for the robust stability analysis, mainly due to the influence of the work of Ackermann (1980) [7], and of the celebrated theorem of Kharitonov (1979) [8]. Recently in [4] a computationally tractable procedure in the case of linearly and non-linearly dependent coefficient perturbations has been proposed.
By jointly using the procedure proposed in [2] and [4] we design a controller which meets the nominal handling qualities requirements at all typical flight conditions (simultaneous stabilization). Then the resulting degrees of freedom will be utilized in order to assure robustness against sensor failure.

1. FLIGHT MECHANICS AND DEFINITIONS

This section contains a brief description of aeronautical terms used in this paper and the mathematical model of the aircraft fully parameterized with respect to the most significant parameters.

Notations

c	mean aerodynamic (geometric) chord
g	acceleration of gravity
h	altitude
I_y	pitching moment of inertia
m	mass (airplane)
q	pitch rate
$\bar{q}$	dynamic pressure
S	reference (wing) area
T	engine thrust
V	true speed
α	angle of attack
ϑ	pitch attitude angle
φ	bank angle
β	sideslip angle
δ_e	elevator control
δ_T	thrust control

1.1 The model

Now we briefly outline the decoupled longitudinal equation of motion of an aircraft. Most aircraft dynamics texts e.g. [9], [10], give more detailed versions of the derivation of this equation.
In the polar coordinate velocity form, the equations of the longitudinal motion of the rigid aircraft, symmetrical with respect to X-Z plane and flying at small sideslip and bank angles, can be written as follows

$$\dot{\alpha} = -\frac{\bar{q}S}{mV} C_L + q + \frac{g}{V}\cos(\vartheta-\alpha) - \frac{T\sin\alpha}{mV} \tag{1}$$

$$\dot{V} = -\frac{\bar{q}S}{m} C_D + g\sin(\alpha-\vartheta) + \frac{T}{m}\cos\alpha \tag{2}$$

$$\dot{q} I_y = \bar{q} S c C_m \tag{3}$$

$$\dot{\theta} = q \tag{4}$$

where $\bar{q}=\rho V^2/2$ is the dynamic pressure, and the air density ρ can be expressed as $\rho=\rho(h)$. C_L, C_D, and C_m are the lift, drag and pitching moment aerodynamic coefficients.
Assuming that the normal acceleration a_n, the longitudinal acceleration a_x and pitch rate q are available for the measurements by using two accelerometers and one gyro respectively, the output equation can be written as

$$y_1 = a_x = -\frac{\bar{q}S}{m}(C_D\cos\alpha - C_L\sin\alpha) - g\sin\vartheta + \frac{T}{m} \tag{5}$$

$$y_2 = a_n = \frac{\bar{q}S}{m}(C_L\cos\alpha - C_D\sin\alpha) - g\cos\vartheta \tag{6}$$

$$y_3 = q\,. \tag{7}$$

By denoting with $x=(\alpha, V, q, \vartheta)^T$ the state vector, $u=(\delta e, \delta T)^T$ the control vector and with $y=(a_x, a_n, q)^T$ the output vector, the open loop model can be linearized with respect to equilibrium state and control vectors x_o and u_o respectively.
For each flight condition inside the flight envelope of the airplane, the model obtained by linearization can be written as:

$$\frac{d}{dt}\begin{bmatrix}\alpha\\ q\\ V\\ \vartheta\end{bmatrix} = \begin{bmatrix} Z_\alpha & 1 & Z_V & -\sin(\vartheta-\alpha)g/V \\ M_\alpha & M_q & M_V & 0 \\ X_\alpha & 0 & X_V & -g\cos(\vartheta-\alpha) \\ 0 & 1 & 0 & 0 \end{bmatrix}\begin{bmatrix}\alpha\\ q\\ V\\ \vartheta\end{bmatrix} + \begin{bmatrix} Z_{\delta e} & 0 \\ M_{\delta e} & 0 \\ X_{\delta e} & X_{\delta T} \\ 0 & 0 \end{bmatrix}\begin{bmatrix}\delta e\\ \delta T\end{bmatrix} \tag{8}$$

$$\begin{bmatrix} ax \\ an \\ q \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ v \\ \vartheta \end{bmatrix} + \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & 0 \\ 0 & 0 \end{bmatrix} \tag{9}$$

In Fig. 1 the open-loop system including actuator dynamics and sensors is represented.

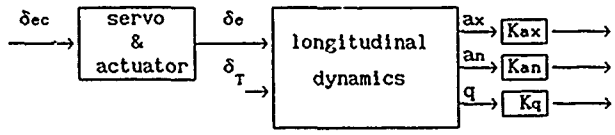


Fig. 1

2. THE STABILITY AUGMENTATION PROBLEM

Generally, high performance requirements in terms of better efficiency (reduction in fuel consumption) and manoeuvrability, impose intrinsic instability on the aircraft. Then a Stability Augmentation System is required for its stabilization. Moreover, airworthiness rules (MIL-F-8785 (ASG), 1969) require that the aircraft is safely controllable by the pilot without any exceptional piloting skill. These requirements concern with the dynamic aircraft behavior and are referred in literature as handling qualities.
With reference to assigned flight condition, the handling qualities are expressed in terms of damping coefficients (ζ_{min},ζ_{max}) and natural frequencies (ω_{min},ω_{max}). Then the S.A.S. design problem can be formulated as follows: design a control system which assures that

a) the closed loop poles lie in the prescribed region of the complex plane;

b) the desired performances are guaranteed for assigned flight conditions (simultaneous stabilization);

and

c) provides some kind of fault-tolerance against incidents like sensor failure.

In order to meet a) and b) requirements, a procedure has been proposed in [11] and [12] based on a new parameterization of all stabilizing compensators [2] and a robust stability test [4], [5].
In this paper the above parameterization will be presented for single-input-multi-output systems as the basis of a low level control system which provides simultaneous stabilization and fault tolerance against sensor failure.

2.1 All stabilizing compensators of maximum order ν

Let A be a real nxm matrix, and i a nonnegative integer; with $S_i(A)$ we denote the real (n+i-1)x(mxi) matrix

$$S_i(A) := \begin{bmatrix} A & 0 \; . \; 0 & \dots & 0 \; . \; 0 \\ 0 \; . \; 0 & A & & \vdots \\ \vdots & \vdots & & 0 \; . \; 0 \\ 0 \; . \; . & 0 \; . \; . \; . \; . \; . & & A \end{bmatrix} . \tag{10}$$

Let

$$a(s) := a_1 + a_2 s + \dots + a_{n+1} s^n = w_n^T(s) a \tag{11}$$

be a polynomial in the indeterminate s of degree n at the most, where

$$a := (a_1 \; a_2 \; \dots \; a_{n+1})^T \; ; \tag{12}$$

$$w_n(s) := (1 \; s \; \dots \; s^n)^T . \tag{13}$$

If

$$c(s) := w_{n+m}^T(s) c = a(s) b(s) \tag{14}$$

denotes the product of two polynomials a(s) and b(s), of maximum degree n and m respectively, it is routine to verify that

$$c = S_{m+1}(a)b = S_{n+1}(b)a \ . \tag{15}$$

Now let $\alpha(s)$ and $\beta_i(s)$, i=1,..,r, be polynomials of degree ν at the most

$$\alpha(s) = \alpha_1 + \alpha_2 s + \ . \ . \ + \alpha_{\nu+1}s^{\nu} = w_{\nu}^{T}(s)\alpha \ , \ \alpha\in R^{\nu+1} \tag{16}$$

$$\beta(s) = (\beta_1(s) \ . \ . \ \beta_r(s)) = \beta_1^T + \beta_2^T s + \ . \ + \beta_{\nu+1}^T s^{\nu} = w_{\nu}^T(s)\begin{bmatrix} \beta_1^T \\ . \\ \beta_{\nu+1}^T \end{bmatrix} , \beta_i\in(\beta_{i1}\ldots\beta_{ir})^T, \tag{17}$$

and $b_i(s)$, i=1,..,r, polynomials of degree n at the most

$$b(s) = (b_1(s) \ . \ . \ b_r(s))^T = b_1 + b_2 s + \ . \ . \ + b_{n+1}s^n \tag{18}$$

By using the above notations, we can write the generalized Diophantine equation:

$$d(s) = a(s)\alpha(s) + \beta^T(s)b(s) \tag{19}$$

in the following alternative form:

$$\left(S_{\nu+1}(a) \quad S_{\nu+1}(B)\right)\begin{pmatrix}\alpha \\ \beta\end{pmatrix} = S\begin{pmatrix}\alpha \\ \beta\end{pmatrix} = d \tag{20}$$

where

$$B = \begin{bmatrix} b_1^T \\ . \\ b_{n+1}^T \end{bmatrix} \ ; \quad \beta = \begin{bmatrix} \beta_1 \\ . \\ \beta_{\nu+1} \end{bmatrix} \tag{21}$$

The following Theorem holds.

Theorem 1. If $a_{n+1}\neq 0$, a(s) and b(s) are coprime and $\nu\geq\nu_o-1$, where ν_o is the observability index of a state realization of b(s)/a(s), then

$$rank\ S = n + \nu + 1 \ . \ \blacksquare \tag{22}$$

The proof of the above theorem is omitted and can be found in [13].

Now consider the feedback system in Fig. 2

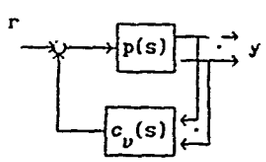


Fig.2

where p(s) and $c_{\nu}(s)$ are given by

$$p(s) = \frac{b(s)}{a(s)} \ , \ b(s)\in P^r \ ; \tag{23}$$

$$c_{\nu}(s) = \frac{\beta^T(s)}{\alpha(s)} \ , \ \beta(s)\in P^r. \tag{24}$$

The transfer function of the feedback system in Fig.3 is given by

$$W(s) = p(s)(1+c(s)p(s))^{-1} = \frac{\alpha(s)b(s)}{\alpha(s)a(s)+\beta^T(s)b(s)} = \frac{\alpha(s)\ b(s)}{d(s)} \qquad (25)$$

By assigning closed-loop poles in a specified stability region of the complex plane one can solve, by virtue of theorem 1, the generalized Diophantine equation (19) in terms of the compensator c(s) of assigned order ν. If $\nu \geq \nu_o - 1$ the designer has at disposal $(\nu+1)r-n$ degrees of freedom which can be utilized in order to optimize other design performance indices.
The set $\mathcal{C}$ of all stabilizing compensators $c_\nu(s)$ of order $\nu \geq \nu_0 - 1$ which ensures that the poles of the feedback system lie in a specified stability region $\mathcal{D}$, is given by

$$\mathcal{C} = \left\{ \frac{w_\nu^T(s)(B_1 d + B_2 z)}{w_\nu^T(s)(A_1 d + A_2 z)};\ d \in R_D^{n+\nu+1},\ z \in R^{\nu+1-n} \right\} \qquad (26)$$

Where $R_D^{n+\nu+1}$ denotes the set of vectors $d \in R^{n+\nu+1}$ such that the zeros of the associated polynomial d(s) are in the stability region $\mathcal{D}$.
The matrices:

$A_1 \in R^{(\nu+1)x(n+\nu+1)}$, $B_1 \in R^{r(\nu+1)x(n+\nu+1)}$, $A_2 \in R^{(\nu+1)x((\nu+1)r-n)}$, $B_2 \in R^{r(\nu+1)x((\nu+1)r-n)}$,

by virtue of theorem 1 are given by

$$\begin{bmatrix} A_1 \\ B_1 \end{bmatrix} = T\,(ST)^{-1}, \quad T = \begin{bmatrix} I_{n+\nu+1} \\ \hline 0 \end{bmatrix} \qquad (27)$$

$$\begin{bmatrix} A_2 \\ B_2 \end{bmatrix} = \begin{bmatrix} (ST)^{-1}\left[-S\begin{bmatrix} 0 \\ \hline I_{(\nu+1)r-n} \end{bmatrix}\right] \\ \hline I_{(\nu+1)r-n} \end{bmatrix}; \qquad (28)$$

$z \in R^{(\nu+1)r-n}$ is the vector of completely free parameters.

3. ROBUSTNESS WITH RESPECT TO FLIGHT CONDITIONS

The first design objective will be to design a feedback controller which meets the nominal handling quality requirements at all typical flight conditions inside the aircraft flight envelope (simultaneous stabilization).
Formally the problem of simultaneous stabilization can be formulated as follows:
Given a set of μ nominal plants P_i, obtained by linearization around specified nominal flight conditions

$$\dot{x} = F_i x + G_i u, \quad i=1,..,\mu \qquad (29)$$

and a set of μ regions $\mathcal{D}^i$ in the complex plane, corresponding to handling quality requirements at each flight condition, find a feedback compensator in the set (26) which stabilizes simultaneously the set of plants (29) with respect to the corresponding $\mathcal{D}^i$ region, $i=1,..,\mu$.

Following the results in the previous section each plant $P_i(s)$ can be $\mathcal{D}^i$-stabilized with a feedback compensator of the set

$$\mathcal{C}^i = \left\{ \frac{w_\nu^T(s)(B_{1i} d_i + B_{2i} z_i)}{w_\nu^T(s)(A_{1i} d_i + A_{2i} z_i)};\ d_i \in R_D^{n+\nu+1},\ z_i \in R^{\nu+1-n} \right\} \qquad (30)$$

then simultaneous stabilization can be achieved if and only if there exist d_i whose zeros belong to $\mathcal{D}^i$ and z_i such that

$$\begin{bmatrix} A_{1i} & A_{2i} \\ B_{1i} & B_{2i} \end{bmatrix} \begin{bmatrix} d_i \\ z_i \end{bmatrix} = \begin{bmatrix} A_{1h} & A_{2h} \\ B_{1h} & B_{2h} \end{bmatrix} \begin{bmatrix} d_h \\ z_h \end{bmatrix}; \quad i=1,..,\mu-1, \quad h=i+1. \tag{31}$$

By using the results in [4] and [5], the region $\mathcal{D}^i$ can be approximately expressed as the union of a specified number q_i of regions $\mathcal{D}^i_j$.

$$\tilde{\mathcal{D}}^i = \bigcup_{j=1}^{q_i} \mathcal{D}^i_j \subseteq \mathcal{D}^i \tag{32}$$

3.1 Approximate linear parametrization of handling quality requirements.

By using decoupled linearized models of longitudinal and lateral directional dynamics, with reference to each specified flight condition, the handling qualities are expressed in terms of closed-loop pole locations with an assigned tolerance in the complex plane.
As reported in [14] there are three flight handling quality levels, which one can use to classify the dynamic behaviour of the aircraft. These levels are characterized by maximum and minimum allowable values of damping ratio ζ and natural frequencies ω_n of the modes associated to the linearized model (ζ_s, ω_{ns} for the short period and ζ_p, ω_{np} for the phugoid mode respectively).
The military specifications do not contain requirements for the location of additional closed-loop poles originating from actuator or feedback dynamics. In order to keep them fast enough and separate from short period poles, an additional closed-loop pole location has been taken into account. The extreme value ω_d of ω_n for these additional poles can be chosen in order to mantain a bandwidth limitation below the first structural mode frequency.
Then, for a specified handling quality level, the region $\mathcal{D}^i$ can be characterized by the following inequalities:

$$\begin{aligned} &\zeta^-_{si} \le \zeta_{si} \le \zeta^+_{si} \quad ; \quad \zeta^-_{pi} \le \zeta_{pi} \le \zeta^+_{pi} \\ &\omega^-_{nsi} \le \omega_{nsi} \le \omega^+_{nsi} \quad ; \quad \omega^-_{npi} \le \omega_{npi} \le \omega^+_{npi} \end{aligned} \tag{33}$$

and

$$\zeta \ge \zeta^-_{si} \quad , \quad \omega^+_{nsi} < \omega_{ns} \le \omega_d \tag{34}$$

for the additional closed-loop poles originating from actuator or feedback dynamics.

With reference to MIL specs and additional requirements, each region $\mathcal{D}^i$ turns to be the union of three compact unconnected domains.

$$\mathcal{D}^i = \mathcal{D}^i_p \cup \mathcal{D}^i_s \cup \mathcal{D}^i_{ad} \, , \tag{35}$$

that is, the closed-loop characteristic polynomial $d_i(s)$ can be expressed as the product of three factors:

$$d_{ip}(s) = s^2 + 2\zeta_{pi}\omega_{npi}s + \omega^2_{npi} \tag{36}$$

whose zeros belong to $\mathcal{D}^i_p$;

$$d_{is}(s) = s^2 + 2\zeta_{si}\omega_{nsi}s + \omega^2_{nsi} \tag{37}$$

whose zeros belong to $\mathcal{D}^i_s$, and by denoting with $l = n+\nu-4$ the degree of the third factor

$$d_{iad}(s) = \begin{cases} \prod_{1}^{l/2} (s^2 + 2\zeta_{ik}\omega_{nik}s + \omega^2_{nik}) & l \text{ even} \\ \prod_{1}^{(l-1)/2} (s^2 + 2\zeta_{ik}\omega_{nik}s + \omega^2_{nik})(s+\lambda) & l \text{ odd.} \end{cases} \tag{38}$$

By denoting with

$$\pi_i := (\pi_{i1},\pi_{i2},\ldots,\pi_{i(n+\nu)}) = (\zeta_{pi},\omega_{npi},\zeta_{si},\omega_{nsi},\ldots,\zeta_{ik},\omega_{nik},\ldots,\lambda), \tag{39}$$

the closed-loop characteristic polynomial can be written as:

$$d_i(s) = d_{ip}(s)\, d_{is}(s)\, d_{iad}(s) = a_1(\pi)+a_2(\pi)s+\cdots+s^{n+\nu};\ \pi\in\Pi, \tag{40}$$

where the set Π is defined by (33) and (34).
Eqn. (40) characterizes the family of all polynomials whose zeros belong to $\mathcal{D}^i$, with nonlinearly depending coefficient perturbations.
The problem of an approximate linear parametrization of the domain $\mathcal{D}^i$ can be viewed as the problem of robust $\mathcal{D}^i$-stability of uncertain dynamical systems with nonlinearly depending characteristic polynomial coefficient perturbations. Such a problem is the subject of continuing research in the very last years. A proposal of solution can be found in [11].
Since characteristic polynomial coefficients are nonlinear functions of parameters π_{ij}, $j=1,\ldots,n+\nu$, as it has been shown in [11], only an approximate characterization of the stability domain in the complex plane, with an assigned degree of accuracy, can be given.
Following the results proposed in [11], partition the set Π in the parameter space $\mathbb{R}^{n+\nu}$ by means of convex polytopes Π_j, $j=1,\ldots,q$. Let $\delta(\Pi_j)$ be the diameter of the subset Π_j and let

$$\mathcal{P}_q=\{\Pi_1,\Pi_2,\ldots,\Pi_q\} \tag{41}$$

be the succession of such partitions, such that

$$\max_{\Pi_j\in\mathcal{P}_q}\delta(\Pi_j) > \max_{\Pi_j\in\mathcal{P}_{q+1}}\delta(\Pi_j) \tag{42}$$

and

$$\lim_{q\to\infty}\max_{\Pi_j\in\mathcal{P}_q}\delta(\Pi_j) = 0. \tag{43}$$

Denote with $v_{ij}, i=1,\ldots,2^{n+\nu}$ the vertices of the polytope Π_j.

Let $a(v_{ij})$ be the value of the function $a(\cdot):\Pi\to\mathbb{R}^{n+\nu}$ and consider the set in the coefficient space

$$\mathcal{H}_j = \left\{ h_j:\ h_j=\sum_{i=1}^{2^{n+\nu}} a(v_{ij})\lambda_i,\ \lambda_i\geq 0,\ \sum_{i}^{2^{n+\nu}}\lambda_i=1 \right\}. \tag{44}$$

Denote with

$$\mathcal{R}_q = \{\mathcal{H}_1,\mathcal{H}_2,\ldots,\mathcal{H}_q\}, \tag{45}$$

the succession in the coefficient space associated to the succession $\mathcal{P}_q$ in the parameter space. Taking into account that the real vectorial function $a(\pi)$ is a continuous one and that the set Π is a compact one, it is possible to prove that

$$\bigcap_{q=1}^{\infty}\bigcup_{j=1}^{q}\mathcal{H}_j = a(\Pi). \tag{46}$$

The set $\mathcal{H}_j$ characterizes a family of polynomials expressed as a convex combination of vertex polynomials:

$$d_{ij}(s) = w_{n+\nu}^T(s)D_{ij}\lambda_j,\ \lambda_i^{(k)}\geq 0,\ \sum_{i}^{2^{n+\nu}}\lambda_i^{(k)}=1,\ j=1,\ldots,q \tag{47}$$

where the columns of the matrix D_{ij} are the coefficients of the vertex polynomials.
In order to test the $\mathcal{D}^i$-stability of the families (47), the robust $\mathcal{D}$-stability test

proposed in [5] can be used. By denoting with $\partial\mathcal{D}^1$ the boundary of $\mathcal{D}^1$, a necessary and sufficient condition for the $\mathcal{D}^1$-stability is

i) $d_{1j}(\hat{s})\neq 0 \ \forall \hat{s}\in\partial\mathcal{D}^1$

ii) $\sup_{\hat{s}\in\partial\mathcal{D}^1} \left|\phi_\theta\left[d_{1j}(\hat{s})\right]-\phi_\theta\left[d_{1k}(\hat{s})\right]\right|<\pi, \ \theta=\arg(d_{11}(\hat{s})), \ \forall i,j=1,\ldots,2^{n+\nu}, i>j,$

where $\phi_\theta(x)$, $\theta\in[-\pi,\pi)$, denotes $\arg(x\, e^{-j\theta})$.

The set D_i of matrices D_{ij}, $j=1,\ldots,q$ which satisfy *i)* and *ii)*, gives a linear characterization of the region $\mathcal{D}^1$. The number q_i of matrices D_{ij} is q at the most. Clearly by increasing q a better characterization can be obtained. By specifying the degree of accuracy required, the number q can be selected by using further results proposed in [5]. Then the handling qualitiy requirements are characterized by the set of matrices D_i.

3.2 Simultaneous stabilization.

The results of previous subsection allow to formulate problem (31) as follows:

Problem (*simultaneous stabilization*).

Find indices $j\in\{1,\ldots,q_i\}$ and $k\in\{1,\ldots,q_h\}$ such that:

$$\begin{bmatrix} \tilde{A}_{1ij} & A_{21} \\ \tilde{B}_{1ij} & B_{21} \end{bmatrix} \begin{bmatrix} \lambda_{ij} \\ z_{ij} \end{bmatrix} = \begin{bmatrix} \tilde{A}_{1hk} & A_{2h} \\ \tilde{B}_{1hk} & B_{2h} \end{bmatrix} \begin{bmatrix} \lambda_{hk} \\ z_{hk} \end{bmatrix}; \ i=1,..,\mu-1, \ h=i+1, \tag{48}$$

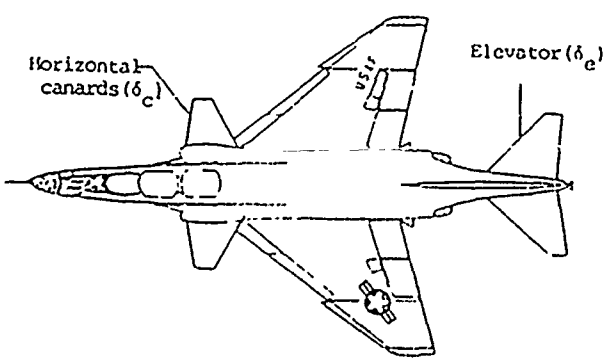
where $\tilde{A}_{1ij}=A_{11}D_{ij}$ and $\tilde{B}_{1ij}=B_{11}D_{ij}$. The problem (48), with the further constraints

$$\sum_{k=1}^{2^{n+\nu}} \lambda_{ij}^{(k)}=1, \ \lambda_{ij}^{(k)}\geq 0, \ j=1,\ldots,q, \ i=1,\ldots,\mu \tag{49}$$

can be solved as a standard linear programming problem.

3.3 Example.

We refer to the F4-E aircraft as in the work of Ackermann [1].
The F4-E is a military aircraft which is destabilized by horizontal canards. A simplified 3-rd order model including the short period longitudinal mode and actuator dynamics in sensor coordinates, normal acceleration a_n and pitch rate q respectively is considered with reference to four typical flight conditions FC-i.



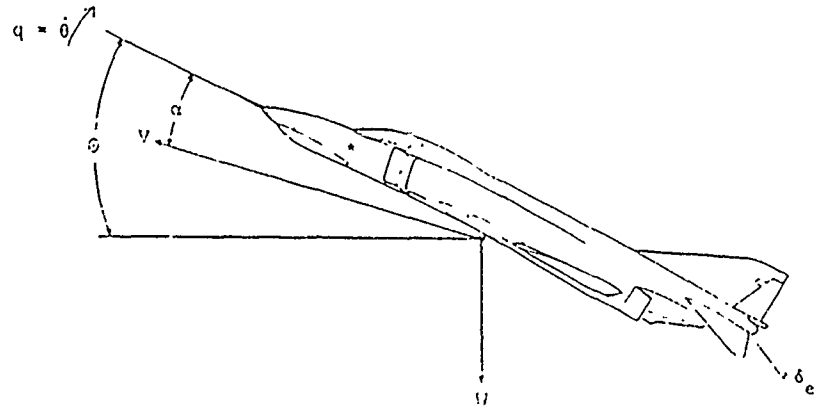


Fig. 3 F4-E with canards

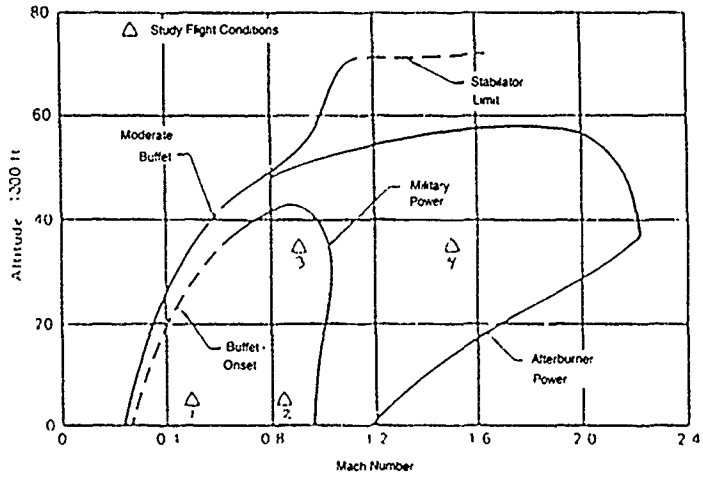


Fig. 4 Flight envelope and operating points

$$F_i = \begin{bmatrix} a^i_{11} & a^i_{12} & a^i_{13} \\ a^i_{21} & a^i_{22} & a^i_{23} \\ 0 & 0 & -14 \end{bmatrix}; \quad G_i = \begin{bmatrix} b^i_1 \\ 0 \\ 14 \end{bmatrix}, \quad i=1,\ldots,4 \tag{50}$$

where the entries of the matrices F_i, G_i are given in the following table [1].

	FC 1	FC 2	FC 3	FC 4
Mach	0.5	0.85	0.9	1.5
Altitude	5000'	5000'	35000'	35000'
a_{11}	-0.9896	-1.702	-0.667	-0.5162
a_{12}	17.41	50.72	18.11	26.96
a_{13}	96.15	263.5	84.34	178.9
a_{21}	0.2684	0.2201	0.0820	-0.6896
a_{22}	-0.8512	-1.418	-0.659	-1.225
a_{23}	-11.39	-31.99	-10.81	-30.38
b_1	-97.78	-272.2	-85.09	-175.6

The regions $\mathcal{D}^i$, i=1,..,4 are described by

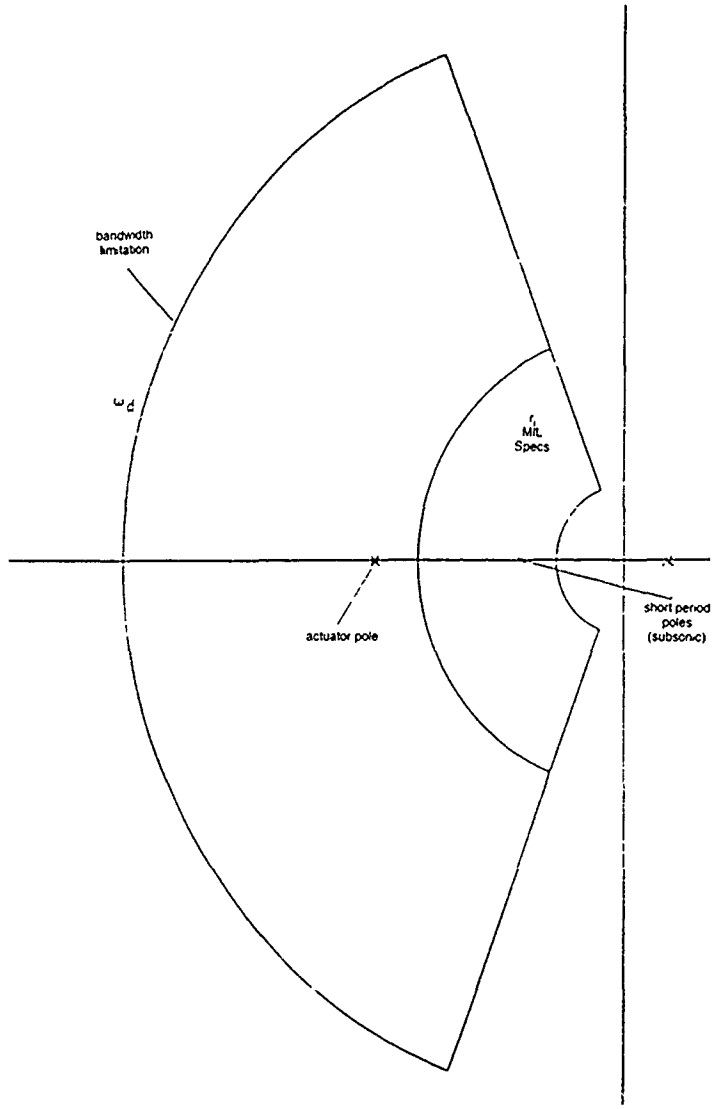


Fig. 5 Required closed loop pole regions

$$0.35 \leq \zeta_{si} \leq 1.3, \ \forall i$$
$$\omega^{-}_{nsi} \leq \omega_{nsi} \leq \omega^{+}_{nsi} \tag{51}$$

where

Natural frequency (rad/sec)	FC 1	FC 2	FC 3	FC 4
ω^{-}_{ns}	2.02	3.50	2.19	3.29
ω^{+}_{ns}	7.23	12.6	7.86	11.8

and ω_d = 70 rad/sec.

By using the procedure illustrated in the previous subsection, we found that a good approximation of regions $\mathcal{D}^i$, i=1,..,4, is obtained with q_1=6, q_2=11, q_3=2, q_4=2. By selecting a first order compensator (ν=1) and solving problem (48)-(49), the following compensator has been obtained

$$c(s) = \frac{\beta^T(s)}{\alpha(s)} = - \frac{((0.0465s+5.8519)\ (0.751s+33.556))}{s + 22.852} . \tag{52}$$

4. ROBUSTNESS AGAINST SENSOR FAILURES.

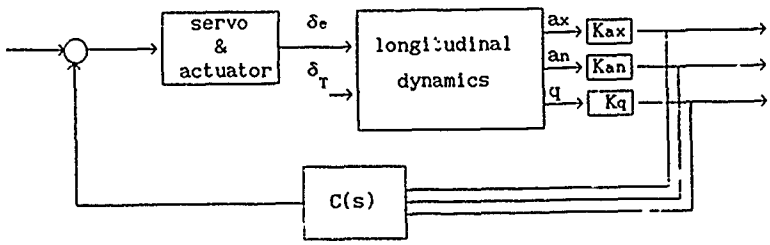


Fig. 6

To increase the reliability in the presence of system vulnerability, the control system will have to provide some kind of fault tolerance. It is well known that redundancy is the basic ingredient to build a reliable control system.
For the design problem, two basic approaches have been studied in the past few years:

a) passive fault tolerance by means of robust control techniques
b) active approach by means of an on line reconfiguration of the control law.

The approach proposed in this paper is of type a).

With reference to Fig.6, a failure of the accelerometer or a gyro is equivalent to a reduction of the respective gain K_{ax}, K_{an} or K_q from the nominal values to zero or some value in between. Then the sensor failure problem can be viewed as a robust stability problem of a linear dynamical system with structured linear perturbations. With reference to this the closed-loop characteristic polynomial can be written as

$$d(s,z,k) = a(s)\alpha(s,z) + \beta^T(s,z)\mathrm{diag}\{k\}b(s) \tag{53}$$

where $k=(K_{ax}\ K_{an}\ K_q)^T$ and

$$c_\nu(s,z) = \frac{\beta^T(s,z)}{\alpha(s,z)} \in \mathcal{C} \tag{54}$$

has been designed in order to meet MIL specs.
The vector z of compietely free parameters is selected by solving the optimization problem

$$\begin{cases} \max\limits_{z} \| k - \bar{k} \|_\infty^W \\ d(z,k) \in \mathbb{R}_D^{n+\nu+1} \end{cases} \tag{55}$$

where $\bar{k}$ denotes the nominal sensor gains and $\|.\|_\infty^W$ denotes the weighted l_∞ norm.
This can be accomplished by using the robust stability test proposed in [4] and [5] at each optimization step. By denoting with

$$d_i(s,z) = d(s,z,k_i), \quad i=1,\dots,8 \tag{56}$$

where k_i denotes the i-th vertex of the politope in the sensor gain space $\mathcal{K}$, the robust stability test can be formulated as follows:

$$d_i(s,z) \neq 0 \quad \forall \hat{s} \in \partial\mathcal{D} \tag{57}$$

$$\sup_{\hat{s}\in\partial\mathcal{D}} \left|\phi_\theta\left(d_i(\hat{s},z)\right) - \phi_\theta\left(d_j(\hat{s},z)\right)\right| < \pi \ , \theta = \arg(d_1(\hat{s},z)), \quad \forall i,j=2,\dots,8,\ i>j. \tag{58}$$

In the following example we show that by iterarively using the above test it is possible to describe the real $\mathcal{D}$-stability region in the space $\mathcal{K}$, and to determine the sensor redundancy degree to achieve fault tolerance.

4.1 Example

We refer to the F4-E aircraft, with reference to the subsonic cruise flight condition (Mach=0.85, h=5000 ft).

$$F_2 = \begin{bmatrix} -1.702 & 50.72 & 263.5 \\ .2201 & -1.418 & -31.99 \\ 0 & 0 & -14 \end{bmatrix}; \quad G_2 = \begin{bmatrix} -272.2 \\ 0 \\ 14 \end{bmatrix}; \quad x = \begin{bmatrix} an \\ q \\ \delta e \end{bmatrix} \tag{59}$$

The MIL specs for the short period mode in the specified flight condition are:

$$3.50 \leq \omega_{sp} \leq 12.6 \ , \quad 0.35 \leq \zeta_{sp} \leq 1.3 \tag{60}$$

the poles added by the actuator must belong to a region characterized by

$$12.6 < \omega_n \leq 70 \ , \ \zeta \geq 0.35 \tag{61}$$

In Fig.7 the stability region in the sensor gain space, in the case of ν=0 (static compensator) is reported. The stability region coincides with that in [1].

Remark. Note that the procedure in order to improve robustness against sensor failure has been presented for a zero order compensator which is different from the one which assure simultaneous stabilization. This only in order to show that the proposed procedure allows to obtain the same results in [1]. Such procedure does not present any limitation in the case of higher order compensator and is the subject of work in progress.

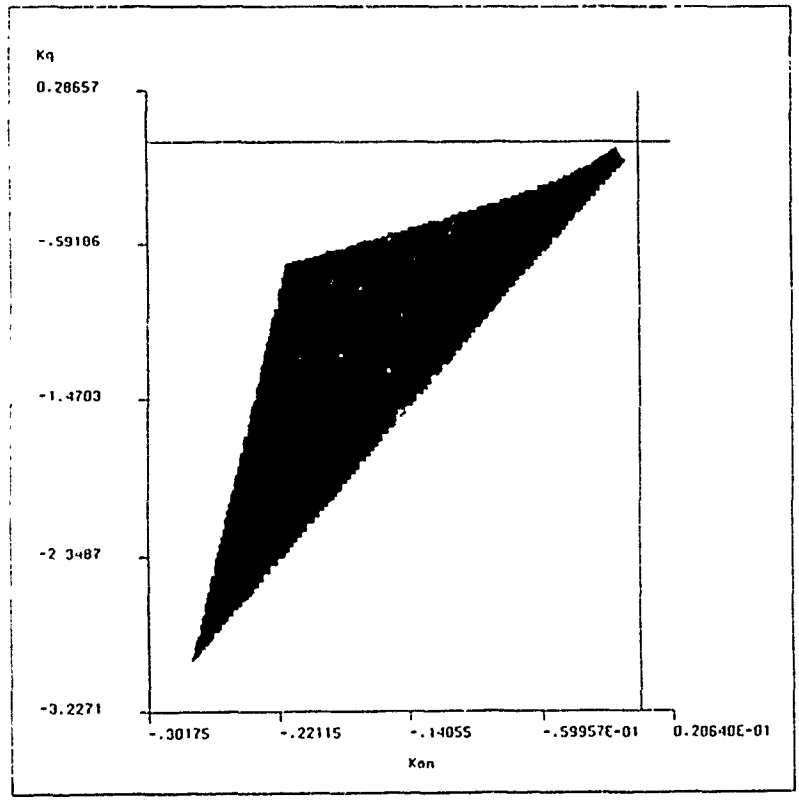


Fig 7 Stability region in sensor gain space

5. CONCLUSIONS

In this paper a new design procedure of all stabilizing feedback controllers for single-input multi-output systems has been presented as the basis of a low level flight control system. This parametrization allows to design a feedback controller which assures that MIL specifications are respected for all typical flight conditions inside the aircraft flight envelope (simultaneous stabilization), and to have at disposal a specified number of degrees of freedom in order to improve robustness against sensor failure. The procedure has been tested with reference to the F4-E military aircraft which is destabilized by horizontal canards. A simplified 3-rd order model including the short period longitudinal mode and actuator dynamics in sensor coordinates has been considered. Simultaneous stabilization has been achieved with respect to four typical flight conditions.

6. REFERENCES

[1] J. Ackermann, Robust control system design, AGARD-AG-289, 1987.

[2] G. Celentano and G. De Maria G., A new linear parameterization of all stabilizing compensators for single input single output plants. *Proc. IEE part D* ,1989.

[3] G. Celentano and G. De Maria, Effective stability margin with respect to specified stability regions. *ICCON '89*, WA-2-2.

[4] A. Cavallo, G. Celentano and G. De Maria, Robust stability analysis of uncertain linear time invariant dynamical systems. *28th CDC*, Tampa, 1989.

[5] A. Cavallo, G. Celentano and G. De Maria, Robust stability analysis of polynomials with linearly dependent coefficients perturbations. *Accepted for publication on IEEE Trans. Autom. Contr.*, 1990.

[6] D. C. Youla, H. A. Jabr and J. J. Bongiorno, Modern Wiener-Hopf design of optimal controllers, part II. *IEEE Trans. Automat. Contr.*, AC-21, 319-338, 1976.

[7] J. Ackermann, Parameter space design of robust control systems. *IEEE Trans. Automat. Contr.*, 25, 1058-1072, 1980.

[8] V. L. Kharitonov, Asymptotic stability of an equilibrium position of a family of systems of linear differential equations. *Differential Equations*, 14, 1483-1485, 1979.

[9] B. Etkin , Dynamics of atmospheric flight. J.Wiley & sons, 1972.

[10] J. Roskam, Airplane flight dynamics and automatic flight controls. Parts I,II. Roskam aviation and engineering corporation, Ottawa, Kansas, 1982.

[11] A. Cavallo, G. De Maria and L. Verde, Robust analysis of handling qualities in aerospace systems. *To be presented at IFAC World Congress 1990*, Tallin, USSR.

[12] A. Cavallo, G. De Maria and L. Verde, Robust parameter design of flight control systems. *AMST 90, Bradford, UK, 1990.*

[13] A. Cavallo, G. De Maria and L. Verde, Robust flight control system design with fault tolerance against sensor failure. *Submitted for presentation at 29th CDC, 1990.*

[14] Flying qualities of Piloted Airplanes. MIL-F-8785B (ASG), 7 Aug. 1969

COMPUTER AIDED DESIGN AND SIMULATION OF THE AUTOMATIC APPROACH AND LANDING PHASE OF A COMBAT AIRCRAFT

by
Dr.-Ing. (USA) F. D. Langer
MESSERSCHMITT-BÖLKOW-BLOHM GMBH
Aircraft Division
8000 München 80, Postfach 801160
FRG

INDEX:

ABSTRACT:

1. INTRODUCTION
2. STATEMENT OF THE PROBLEM AND SOLUTION ANSATZ
3. THE ARCHITECTURE OF THE CONTROL SYSTEM FOR AUTOMATIC APPROACH AND LANDING
4. THE AIRCRAFT'S DYNAMICS AND THEIR LINEARIZATION
5. SYMBOLIC PROGRAMMING USING AN EXPERT PROGRAM
6. HARDWARE ASPECTS OF THE SYMBOLIC MANIPULATION PROGRAM MACSYMA
7. CONCLUSION

SUMMARY

Complex control systems like those employed in modern aircraft can be efficiently designed and simulated with the aid of artificial intelligence tools. In the present paper, it is dicsussed how a symbolic manipulation program can be used to automate the steps which are necessary to design and simulate a control system. The landing phase of the MRCA TORNADO is taken as an illustrative example. The automation of the development phase of a control system has the advantage of reducing the workload of design engineers by doing repetitive, tedious, time-consuming, and error prone tasks on the computer while letting the respective engineers concentrate on more important design issues.
In the initial design phase, a six-degree-of-freedom model is derived for the approach and landing mode of the aircraft configuration under consideration. The resulting non-linear equations of motion are linearized around suitably spaced points of the flight trajectory. Finally, control systems design methods are applied to the linearized set of equations to generate a control algorithm that satisfies prespecified goals such as minimum gain and phase margin, maximum overshoot and settling time while maintaining a locally stable, non-linear system under small perturbations around operating points on the flight trajectory. Digital computer simulations of the closed-loop system, which consists of the non-linear aircraft model and the control algorithm designed as discussed above, serve to verify the system performance against specified values.

The derivation and subsequent linearization of the aircraft's equations of motion is a tedious and time-consuming task which is subject to many errors, if it is done by hand. This can be more efficiently done on a computer which executes a symbolic manipulation program like MACSYMA. It is shown in this paper that a symolic manipulation program can be employed as an integrated tool to derive the equations of motion, to linearize them around a prespecified operating point, and to produce source code for digital computer simulations of the closed-loop system. Limitations of symbolic manipulation programs are discussed, as well.

The above mentioned approach can be implemented on computer systems ranging from main frames to high performance personal computers, thus reducing investment costs. Furthermore, this approach can be adapted to other control systems with minor modifications, after it has been implemented for a particular control system. Therefore, the cost-to-benefit ratio is very high.

1. INTRODUCTION

Computer aided design tools are used in many engineering applications to reduce the workload of design engineers by doing repetitive, tedious, time-consuming, and error-prone tasks on a computer system so that these engineers are free to concentrate on more important design issues.

In the present paper, it is shown that an expert program [1] can be used as a central development tool for the design, analysis, and documentation of a flight control system. The expert system serves to, Fig. 1

- o symbolically derive the non-linear equations of aircraft dynamics [2] and to optimize the resulting algebraic expressions with respect to speed of computation;
- o symbolically linearize the non-linear dynamic equations around arbitrary operating points;
- o symbolically derive transfer functions of the linearized dynamic equations in minimal form [3];
- o to export the optimized, non-linear, dynamic equations (in FORTRAN or C) for integration into a simulation environment;

- o to export transfer functions and/or linearized aircraft dynamics to a control systems analysis package [5] for the computation of feedback gains,
- o to export the above mentioned equations to a text formatting system (TEX), [4]) for documentation purposes.

The approach presented in this paper offers several advantages over conventional design approaches:

- o If the non-linear equations of aircraft dynamics are both programmed and debugged in MACSYMA, there is no need to debug the offsprings of it, i.e. the linearized aircraft dynamics [6, 7], the transfer functions and the parameters, which are exported to a simulation package. Thus, the overall debugging time during software development can be significantly reduced.
- o The costs for documentation can be significantly reduced, as well, since the mathematical equations and the corresponding results which are coded in MACSIMA are self-explanatory, see section 5, and are automatically converted to TEX format [1].

The automatic approach and landing of a combat aircraft is chosen as an arbitrary example of a complex control system which can be more efficiently designed and analyzed with the proposed expert program.

The organization of this paper is as follows:
In section II, the control problem is stated, and methods for solution are discussed. In section III, the architecture of the overall flight control system for automatic approach and landing is outlined. In section IV, the aircraft dynamic equations, the linearization and computation of transfer functions is reviewed. In section V, the symbolic programming of equations presented in section IV is shown. In section VI, computer hardware for executing MACSYMA is discussed. Final remarks are given in section VII.

2. STATEMENT OF THE PROBLEM AND SOLUTION ANSATZ

Clear and unlimited visibility seldom exists in Europe, especially in Central Europe, where bad weather predominates and winter brings only 8 hours of daylight in 24.

Automatic approach and landing systems were devised, for use in the civilian and military aviation community, which generate virtual beams defining a nominal and safe approach and landing trajectory for the aircraft. One of the first of such a system is the ILS (Instrument Landing System) [8] which is over 40 years in use now and is going to be replaced in the near future by either a MLS (Microwave Landing System) [9] or a differential GPS (Global Positioning System)[10]. While a detailed discussion on the difference of the above mentioned systems is beyond the scope of this paper and is done elsewhere [8 - 10], it suffices to note that the ILS allows only straight-in approach and landing manoeuvres while the MLS and differential GPS does both straight-in and curved ones.

Consider the curved approach and landing manoeuvre which is shown in 3D in Fig. 2a, and in the corresponding azimuth and elevation planes in Fig. 2b and Fig. 2c respectively. The procedure for automatic approach and landing is as shown below.

CURVED APPROACH AND LANDING IN THE AZIMUTH PLANE:

In the azimuth plane, the aircraft flies in the autopilot track acquire mode up to position A, where it enters the coverage zone of the automatic approach and landing system. This zone is defined by a circle of constant radius centered about the airport. From position A to D, curved approach autopilot control laws are engaged. In the straight pre-turn segment, the aircraft follows a straight line flight path and rolls into the bank angle which is required for circular segment BC. In segment BC, the aircraft does a steady, circular turn up to position C. In the straight-line segment CD, the aircraft rolls out from the turn's bank angle, levels and follows a straight-line path corresponding to an ILS-type landing.

CURVED APPROACH AND LANDING IN THE ELEVATION PLANE:

In the elevation plane, the aircraft's autopilot is engaged in a barometric height hold mode up to position C. In segment CD, the aircraft reduces height up to touchdown point D. Depending on the landing category which the aircraft is cleared for, the pilot has to perform the final landing procedure manually, see Table 1. This is true for landing categories I and II where the autopilot is disengaged at decision heights of 50 m or 30 m and runway visual range of 800 m or 400 m defined by ICAO [11]. In landing category III, the aircraft is automatically guided down to touchdown on the runway.

THE ASSOCIATED CONTROL PROBLEM:

The control problem associated with automatic approach and landing can now be stated:

Design a control within the overall control systems architecture of the aircraft, see section 3., which

- o provides an accurate tracking of the virtual approach and landing beam, especially of the final landing segment,

o satisfies the air vehicle spec with respect to overshoot and settling time during the segments for curved approach and landing.

Landing Category	Decision Height		Runway Visual Range	
	m	(ft)	m	(ft)
I	60	(200)	800	(2600)
II	30	(100)	400	(1200)
IIIa	0		200	(700)
IIIb	0		50	(150)
IIIc	0		0	0

Table 1: Decision Heights and Runway Visible Range for Different Landing Categories (according to ICAO [11])

SOLUTION ANSATZ FOR SOLVING THE ASSOCIATED CONTROL PROBLEM

Various approaches exist to treat the control problems associated with automatic approach and landing. - However, a detailed discussion on them is beyond the scope of this paper. -

McRuer et al. [12] present a detailed design procedure for an automatic approach and landing system which receives ILS-type sensor information. They design the corresponding control system in the frequency domain.

Merriam and Ehlert [13] design an automatic control system for landing a high-speed jet airplane on the deck of an aircraft carrier by selecting an appropriate performance index and solving the associated optimal control problem. Control laws for presently used military aircraft are confidential in nature [14]. It should be mentioned that the expert system, which is proposed in this paper is intended to be used for the verification of existing control laws specified in [14] and to experiment with other control laws for automatic approach and landing [12, 13].

3. THE ARCHITECTURE OF THE CONTROL SYSTEM FOR AUTOMATIC APPROACH AND LANDING

The architecture of the overall flight control system for automatic approach and landing is shown in Fig. 3. It consists of the main blocks aircraft, primary flight control system and guidance system.
The aircraft is represented by its actuation units, the control surfaces, its airframe dynamics, and its corresponding motion sensors. The primary flight control system forms an inner control loop and the guidance system an outer one. The primary flight control system consists of a corresponding lateral and longitudinal controls system and has as basic functional and operational tasks:

- to establish an equilibrium point of motion by operating point control and
- to restore a disturbed aircraft to its equilibrium point and/or to regulate its departure from the operating point condition [12].

The guidance control system has the task of generating appropriate flight path command for the aircraft to follow the virtual approach and landing beam.

From the above shown architecture of the overall flight control system, it is evident that the design of control laws for automatic approach and landing is not an isolated design issue but rather has to be done in the framework of the overall control system. Therefore, the guidance system may have the task to augment the primary flight control about regions in the aircraft's permissible flight envelope.

An important point in the design of autopilot functions for automatic landing for military aircraft is that some of the corresponding autopilots lack the ability of a direct link to the yaw axis in order to compensate for gust disturbances, for example. If this is true, the respective aircraft may not be cleared for category III landings, see Table 1.

4. THE AIRCRAFT'S DYNAMICS AND THEIR LINEARIZATION

Consider Fig. 4 which shows an aircraft with associated coordinate systems, components of the state vector and control inputs to the control system. It should be noted that throughout this paper, variable names and indices are used according to air vehicle specification LN 9300 [17].

Let a body coordinate system (BCS) with axes X, Y, Z be aligned with the principal axes and originating in the center of gravity (c.o.g.) of the aircraft. Furthermore, let an inertial coordinate system (ICS) with axes X_g, Y_g, Z_g be located with its origin in the c.o.g. of the aircraft. Let the state vector of the system consist of components of translational and rotational motion of the aircraft. Let the components of translational motion of the state vector be the vector $\vec{r}$ with components (x, y, z) in the BCS and vector $\vec{v}$ with components (u, v, z) in the BCS of the aircraft. Similarly, let the components of rotational motion of the state vector be the vectors $\vec{\Theta}$ and $\vec{\Omega}$.

The vector $\vec{\Theta}$ has as components the Euler angles which describe the orientation of the BCS relative to the ICS. The vector $\vec{\Omega}$ has as components the angular velocities in the BCS (p, q, r).

The controls of the aircraft consist of the surface deflections δ_A, δ_B, δ_E, δ_F, δ_R which represent the angle of deflection of aileron, air brakes, elevaters, flaps, rudder and the power plant input δ_{RPM} representing the change of power plant revolutions per minute.

The assumptions for the derivation of the dynamics of the aircraft are that:

i) the aircraft is a rigid body,

ii) the air flow along the aircraft is quasi-steady, and

iii) the mass of the aircraft remains constant for the duration of the dynamic analysis.

It is shown in appendix A that the equations of dynamics of the aircraft from [18] can be written in vector matrix form as:

$$\underline{\dot{X}} = f_1(\underline{X}, t) + f_2(\underline{X}, U, t) \qquad \ldots(1)$$

where $\underline{X} = [x\, y\, z\, u\, v\, w\, \varnothing\, \Theta\, \psi\, p\, q\, r]^T$

is the state vector

and $\underline{U} = [\delta_A \delta_B \delta_E \delta_F \delta_{RPM}]^T$ is the control input vector of the control system.

LINEARIZATION

The nonlinear equations of motion (1) can be linearized by expanding its corresponding left and right hand sides in a Taylor series about an operating point given by

$$\underline{X}_o = [x_o\, y_o\, z_o\, u_o\, v_o\, w_o\, \varnothing_o\, \Theta_o\, \psi_o\, p_o\, q_o\, r_o]^T$$

$$\underline{U} = [\delta_{A_o} \delta_{B_o} \delta_{E_o} \delta_{F_o} \delta_{RPM_o}]^T$$

and neglecting high-order terms.

Let $\underline{X} = \underline{X}_o + \delta x + 0(\delta \dot{x})$

and $f(\underline{X}, \underline{U}, t) = f(\underline{X}_o, \underline{U}_o, t_o) + f_{\underline{X}} \delta \underline{X} + f_{\underline{U}} \delta \underline{u} + 0(\delta x, \delta u)$

with $f\underline{X} = \frac{\delta f}{\delta \underline{x}}|_{\underline{x}_o, \underline{u}_o}$ and $f\underline{U} = \frac{\delta f}{\delta \underline{u}}|_{\underline{x}_o, \underline{u}_o}$

Then, the linearized equations of motion are: $\delta \underline{\dot{X}} = f_{\underline{x}} \delta \underline{X} + f\underline{U} \delta \underline{U} \qquad \ldots(2)$

The output of the system $\delta \underline{Y}$ is found by premultiplying the state vector $\delta \underline{X}$ by the output matrix C.

$$\delta \underline{Y} = C \delta \underline{X} \qquad \ldots(3)$$

Single-input-single output SISO transfer functions of the linearized set of equations (2) and (3) are found by Laplace transforming equation (2) and substituting the result into equation (3):

$$\delta \underline{Y} = C(sI - f\underline{x})^{-1} f\underline{u} \delta \underline{U} \qquad \ldots(4)$$

where I is the identity matrix and "s" the Laplace operator. Values of matrices C and $f\underline{u}$ for some commonly used transfer functions are given in appendix B.

5. SYMBOLIC PROGRAMMING USING AN EXPERT PROGRAM

The non-linear dynamic equations of the aircraft, equ. (1), the corresponding linearized equations, equ. (2), and the transfer functions of interest, equ. (4) are coded in the MACSYMA symbolic language. MACSYMA code for the non-linear equations of motion and their linearization is shown in Appendix C. This approach is more efficient and less time-consuming than computing the linearized equations and the respective transfer functions manually [13, 14], and cross-checking the results afterwards by tediously computing the sum of rows and columns [13, 14]. Furthermore, if the non-linear equations of the aircraft have to be modified to include additional terms, the computation of the linearized equations and of the respective transfer functions has to be repeated. This is easier accomplished with the aid of a computer system executing the MACSYMA symbolic language.

The linearization around different operating points can be studied in more detail if this is done symbolically as opposed to numerically. The simplification routines of MACSYMA alleviate the computational burden of computing the simplest mathematical expression for the equations of interest. This is necessary fo iteratively computing flight trajectories of the closed loop system shown in Fig. 3. For iterative computations, the MACSYMA internal code (LISP) is not suited. However, a set of specialized set of subroutines exists to export MACSYMA expressions into a subroutine coded in C or FORTRAN.

A closed loop simulation of the system shown in Fig. 3 can be accomplished with minimal effort by providing a central numerical integration routine, which calls subroutines representing the aircraft, the primary flight control system and the guidance loop. The latter subroutines are generated from symbolic expressions coded in MACSYMA.

For documentation purposes, an interface from MACSYMA to the text formatter TEX [4] exists [1]. Using this feature, intermediate results from symbolic computation can be automatically converted to TEX format.

With the above mentioned features, a development tool can be devised using the expert program MACSYMA as a central routine, see section 1.

6. HARDWARE ASPECTS OF THE SYMBOLIC MANIPULATION PROGRAM MACSYMA

The expert program MACSYMA is ported to many computer systems ranging from main frames to high performance personal computers. Constraints are imposed on the computer system by the amount of RAM space available. There should be enough space to store intermediate expression which becomes large when forming Jacobian matrices, as in equ. (2), or by computing the inverse of matrices as in equ. (4).

Since computer systems of the present time can be interconnected via Ethernet, a central development tool for control system does not necessarily need to be implemented on a single, dedicated computer system as done by Colgren [15]. Colgren describes in his paper [15] the development of a workstation which integrates the design, analysis, and simulation methods used for flight control system analysis.

7. CONCLUSION

A method is presented for using the expert program MACSYMA as a central development tool for the analysis and simulation of complex control systems. It is shown that the expert program can be used to:

- o efficiently compute the linearized dynamic equations of the aircraft,
- o compute the corresponding transfer functions of the linearized system,
- o export optimal expressions of equations to other languages which are used for iteratively computing flight trajectories of the corresponding closed-loop system, and
- o automatically convert symbolic expressions to expressions for the text formatter TEX.

The expert system is intended to be used for the verification of existing control laws for automatic approach and landing and for the design and testing of optimized control laws, as well. The use of such an expert system for control system analysis purposes roughly saves two third of the time over conventional design approaches.

REFERENCES

1. "MACSYMA Reference Manual", Doc. no. SMI 0506030011, Symbolics Inc. March 1986.
2. "LN 9300", Normenstelle Luftfahrt, 1970.
3. Kailath. "Linear Systems", Prentice-Hall, 1980.
4. Knuth, K. E. "The TEXbook", Addison-Wesley, Reading, 1986.
5. "RASP User Manual", DFVLR Oberpfaffenhofen, West-Germany.
6. "Aerodynamic Equations for Rig Simulation Purposes", PANAVIA Report G/6000/6512/A, July 1975.
7. "Reductions of Aerodynamic Equations of Motion to State Variable Form for Rig Simulation Purposes", PANAVIA Report G/6500/1054/A, June 1978.
8. Kayton, M. and Fried, W. "Avionics Navigation Systems", New York : Wiley, 1969.
9. Redlien, H. and Kelly, R. "Microwave Landing Systems, the New International Standard", In Advances in Electronics and Physics, vol.57, New York : Academic Press, 1981.
10. Schanzer, G. "Einsatzmöglichkeiten von Satellitennavigation", Symposium Satellitennavigation in der Flugführung, Braunschweig, July 1989.
11. "Aeronautical Telecommunications", vol. I, 3rd Edition, ICAO, Montreal, Canada, July 1972.
12. McRuer, D. et al. "Aircraft Dynamics and Automatic Control", Princeton University Press, Princeton, New Jersey, 1973.
13. Merriam, C.W., III, and F.J. Ellert, "Synthesis of Feedback Controls Using Optimization Theory - An Example", IEEE Trans. Automatic Control (1963), 89-103
14. "MRCA-AFDS Control Laws for Series A/C", GEC Avionics, Rochester, Kent, England, 1990.
15. Colgren, R.D. "Integrated Control System Design and Simulation", CDC Conference on Robotics, pg. 561-565

Appendix A: Derivation of Dynamics of the Aircraft's Frame

$\dot{X} = f[x(t), u(t), t]$

$$\frac{d}{dt}\begin{bmatrix} x \\ y \\ z \\ u \\ v \\ w \\ \Phi \\ \Theta \\ \varphi \\ p \\ q \\ r \end{bmatrix} = \left[\begin{array}{cc|cc} 0 & \frac{1}{m} I_3 & & \\ & & 0 & \\ 0 & -\frac{1}{m} WW & & \\ \hline & & 0 & B^{-1} \\ 0 & & & \\ & & 0 & -J^{-1}WWJ \end{array}\right] \begin{bmatrix} x \\ y \\ z \\ u \\ v \\ w \\ \Phi \\ \Theta \\ \varphi \\ p \\ q \\ r \end{bmatrix} + \left[\begin{array}{cc} 0 & \\ & 0 \\ \frac{1}{m} I_3 & \\ \hline & 0 \\ 0 & \\ & J^{-1} \end{array}\right] \left[\begin{array}{l} F_x [u, w, \dot{w}, q, T(u), T(\delta_{RPM}), \delta_B, \delta_E, \delta_F] \\ F_y [v, p, r, \delta_A, \delta_R] \\ F_z [u, w, \dot{w}, q, T(u), T(\delta_{RPM}), \delta_B, \delta_E, \delta_F] \\ \hline L [v, r, p, \delta_A, \delta_R] \\ M [u, w, \dot{w}, T(u), T(\delta_{RPM}), \delta_B, \delta_E, \delta_F] \\ N [v, p, r, \delta_A, \delta_R] \end{array}\right]$$

with $I_3 = \begin{pmatrix} 1 & 0 & \\ & 1 & \\ 0 & & 1 \end{pmatrix}$, $J = \begin{pmatrix} I_x & 0 & \\ & I_y & \\ 0 & & I_z \end{pmatrix}$, $B^{-1} = \begin{pmatrix} 1 & 0 & -\sin\Theta \\ 0 & \cos\Phi & \sin\Phi\cos\Theta \\ 0 & -\sin\Phi & \cos\Phi\cos\Theta \end{pmatrix}$

$$WW = \begin{pmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & \tan\Theta\sin\Phi & \tan\Theta\cos\Phi \\ 0 & \cos\Phi & -\sin\Phi \\ 0 & 0 & \frac{\cos\Phi}{\cos\Theta} \end{pmatrix}$$

Appendix B: The Choice of Matrices C and $f\underline{u}$ for Commonly Used Transfer Functions

Transfer Function	Matrix $f\underline{u}$	Matrix C
$\frac{u(s)}{\delta_E(s)}$	The column of $f\underline{u}$ corresponding to δ_E	[000 1 0 0000000]
$\frac{w(s)}{\delta_E(s)}$	The column of $f\underline{u}$ corresponding to δ_E	[00000 1 000000]
$\frac{\Theta(s)}{\delta_E(s)}$	The column of $f\underline{u}$ corresponding to δ_E	[0000000 1 0000]
$\frac{\Phi(s)}{\delta_R(s)}$	The column of $f\underline{u}$ corresponding to δ_R	[000000 1 00000]
$\frac{\varphi(s)}{\delta_R(s)}$	The column of $f\underline{u}$ corresponding to δ_R	[00000000 1 0000]

APPENDIX C

```
/*      Symbolic programming of the non-linear, dynamic equations                    */
/*      of the airframe, their linearization, computation                            */
/*      of transfer functions of the linearized system and                           */
/*      export of matrix entries to a C language interface                           */
/*                                                  --- Langer, March 1990           */

write file ("Six DOF.sym");

showtime: true; dynamolloc: true;     /* program switches                           */

/*      Definition of non-linear, dynamic equations: d/dt C = f1(X,t)+f2(x,u,t)      */

/*      Definition of Euler Angles                                                   */

Euler Angles:= block ([c-phi, c_th, c_psi, s_phi, s_th, s_psi],

c-phi: los (phi), c_th: los (theta), c_psi: los (psi),
s-phi: sin (phi), s_th: sin (theta), s_psi: sin (psi),

b: matrix    (    [1:   s_th/c.     th * s_phi,      s-th/c.     th * c_phi     ],
                  [0:   c_phi,                       s_phi                      ]
                  [0:   0,                           c_phi/c_th                 ],
bin v: matrix(    [1:   0,          0,               -s_th                      ]
                  [0:   C-phi,      s-phi * c_th,                               ]
                  [0:   -s_phi      C_phi, *c_th                                ]));

ev. (enl. ang., simp, detout);

/*      Definition of State Vector                                                   */

sv: matrix ([x], [y], [z], [u], [v], [w], [phi], [th], [psi], [p], [q], [r]);

/*      Definition of Inertial Matrix J                                              */

J:      Matarix ( Ix, o, o], [0, IY, 0], [0, 0, IZ]);

/*      Definition of Cross Product Operator WW                                      */

 WW:   matrix ([0, -r, q], [r, 0, -p], [-q, p, o]);
 tmp:  -inverse (J). WW. J j

 /*     matrix f1 (x, t) of equation (1)                                            */

 f1: matrix ( [0, 0,  0, 1/m,      0,          0,          0, 0, 0, 0,      0,          0,]
              [0, 0, 0,  0,        1/m,        0,          0, 0, 0, 0,      0,          0],
              [0, 0, 0,  0,        0,          0,          0, 0, 0, 0,      0,          0],,
              [0, 0, 0,  b[1,1]/m, b[1,2]/m,   b[1,3]/m,   0, 0, 0, 0,      0,          0],
              [0, 0, 0,  b[2,1]/m, b[2,2]/m,   b[2,3]/m,   0, 0, 0, 0,      0,          0],
              [0, 0, 0,  b[3,1]/m, b[3,2]/m,   b[3,3]/m,   0, 0, 0, 0,      0,          0],
              [0, 0, 0,  0,        0,          0,          0, 0, 0, bin v [1,1], bin v [1,2], bin v [1,3],],
              [0, 0, 0,  0,        0,          0,          0, 0, 0, bin v [2,1], bin v [2,2], bin v [2,3],],
              [0, 0, 0,  0,        0,          0,          0, 0, 0, bin v [3,1], bin v [3,2], bin v [3,3],],
              [0, 0, 0,  0,        0,          0,          0, 0, 0, tmp [1,1],   tmp [1,2],   tmp [1,3],],
              [0, 0, 0,  0,        0,          0,          0, 0, 0, tmp [2,1],   tmp [2,2],   tmp [2,3]],
              [0, 0, 0,  0,        0,          0,          0, 0, 0, tmp [3,1],   tmp [3,2],   tmp [3,3],]),

 /*     matrix f2 *x, u, t( in equation (1)                                         */
 jinv:  inverse (7);
```

```
f2:
matrix (
        [0,    0,    0,    0,          0,          0],
        [0,    0,    0,    0,          0,          0],
        [0,    0,    0,    0,          0,          0],
        [1/m,  0.    0,    0,          0,          0],
        [0,    1/m,  0,    0,          0,          0],
        [0.    0,    1/m,  0,          0,          0],
        [0,    0,    0,    0,          0,          0],
        [0,    0,    0,    0,          0,          0],
        [0,    0,    0,    0,          0,          0],
        [0,    0,    0,    jinv [1,1], jinv [1,2], jinv [1,3],
        [0,    0,    0,    jinv [2,1], jinv [2,2], jinv [2,3],
        [0,    0,    0,    jinv [3,1], jinv [3,2], jinv [3,3],]);
inp:matrix([FX], [FY], [FZ], [L], [M], [N]);
/*      forces and moments :                                          */
/*depends(inp[1], [sv[4], sv[6], diff(sv[6],t), t_u, t_δrpm, δ_B, δ_E, δ_F],
      inp[2], [sv[5], sv[10], sv[12], δ_A, δ_R],
      inp[3], [sv[4], sv[6], diff(sv[6],t), t_u, t_δrpm, δ_B, δ_E, δ_F],
      inp[4], [sv[3], sv[10], sv[12], δ_A, δ_R],
      inp[5], [sv[4], sv[6], diff(sv[6],t), t_u, t_δrpm, δ_B, δ_E, δ_F],
      inp[6], [sv[3], sv[10], sv[12], δ_A, δ_R]);

/*      right hand side of equ. (1)                                   */
      rhs: f1. sv + f2 inp
/*      Taylor series expansion of forces and moments                 */
...
```

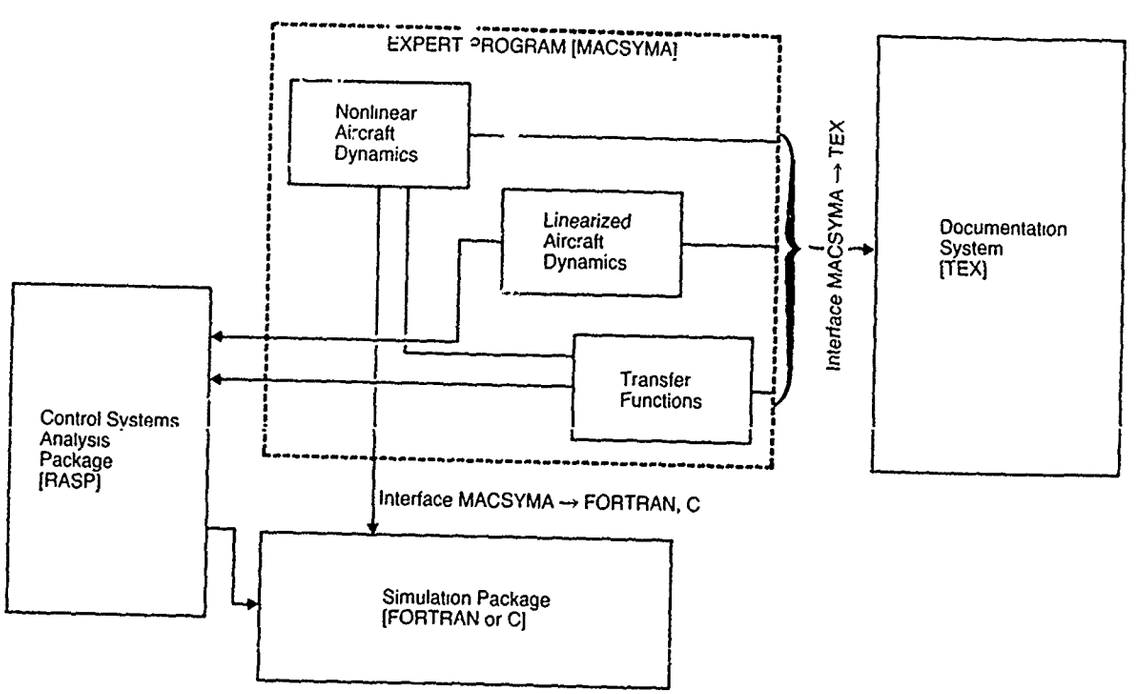


Fig. 1: The Expert Program MACSYMA as a Central Development Tool

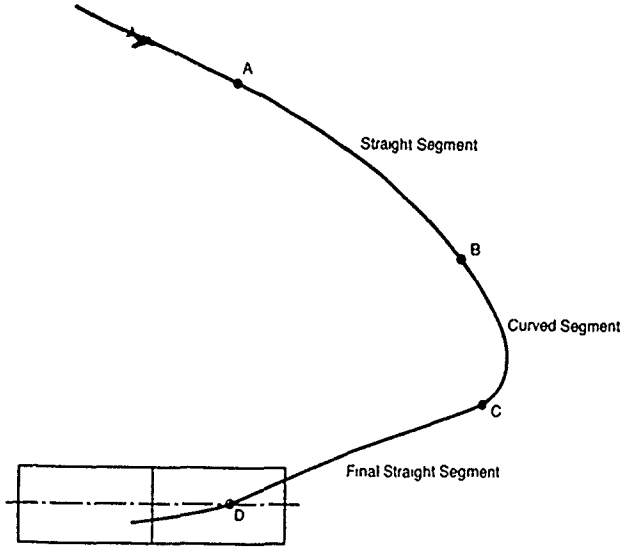


Fig. 2a: Curved Approach and Landing Trajectory in 3 D

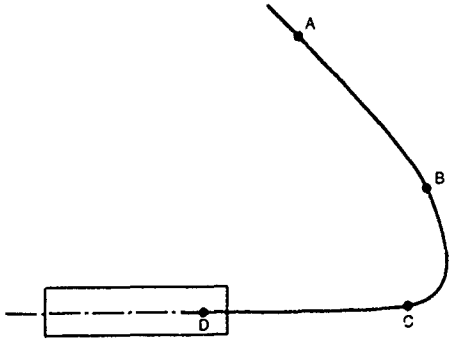


Fig. 2b: Curved Approach and Landing Trajectory in the Azimuth Plane

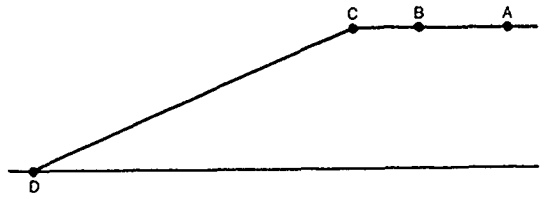


Fig. 2c: Curved Approach and Landing Trajectory in the Elevation Plane

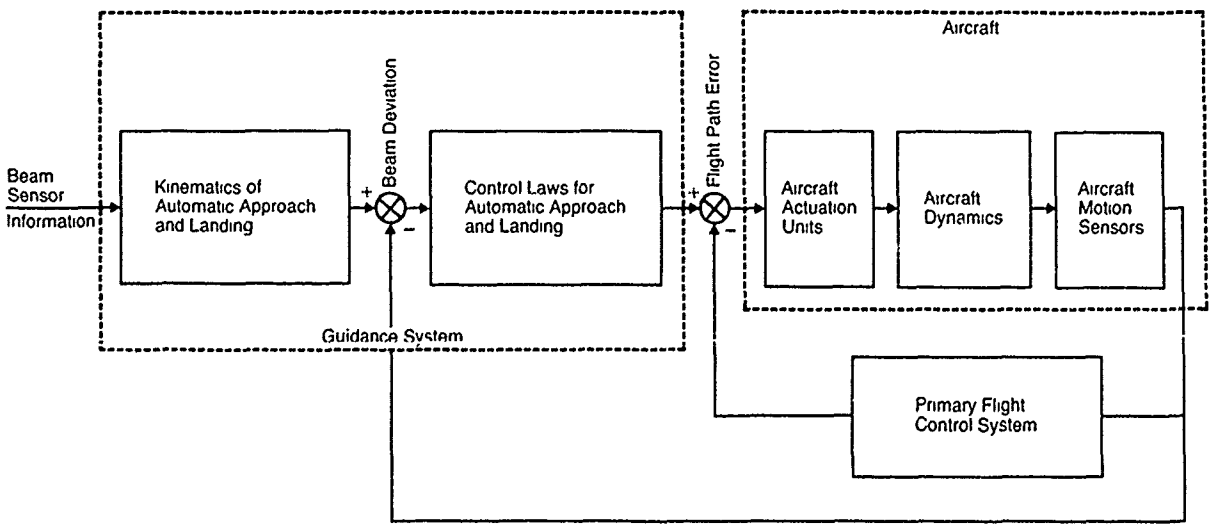


Fig. 3: Architecture of the Overall Flight Control System for Automatic Approach and Landing

δ_A: Angle of deflection of aileron
δ_B: Angle of deflection of dive brakes
δ_E: Angle of deflection of elevator
δ_F: Angle of deflection of flaps
δ_R: Angle of deflection of rudder
δ_{RPM}: Change of power plant revolutions per minute

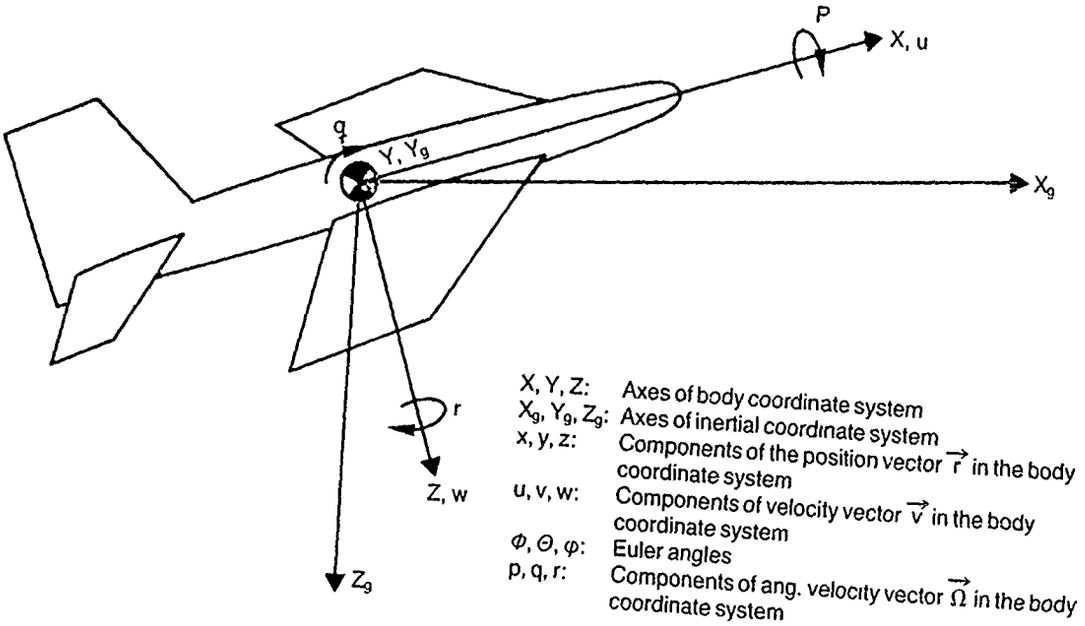


Fig. 4: Coordinate Systems and Components of Velocity and Ang. Velocity Vectors of an Aircraft

CONCEPTION D'AUTODIRECTEURS RIGIDES

par
F. Bertrand et M. Breuzet
THOMSON-CSF, Division Electronique de Missiles
23-27, rue Pierre Valette
92242 - Malakoff - Cedex
France.

.1. RESUME

Le développement des capteurs d'imagerie infrarouge ces dernières années, parallélement à celui des techniques de traitement d'image, amène à concevoir des autodirecteurs optroniques dans lesquels la partie optomécanique est très simplifiée.

La spécification d'autodirecteur rigide (sans stabilisation optomécanique de la ligne de visée) nécessite une simulation globale du comportement du missile car c'est celui-ci qui assure le ralliement angulaire sur la cible.

Une simulation numérique a été mise en place, qui permet de définir les paramètres principaux du capteur : le champ, la résolution, le temps d'intégration, compte-tenu des caractéristiques du missile, de la cible et de la mission. Le guidage de l'engin est entiérement assuré par des algorithmes de traitement d'image assurant la poursuite d'une cible sur fond structuré. Ces algorithmes sont, au stade actuel, représentés par un modèle comportemental.

.2. INTRODUCTION

Pour répondre à des besoins opérationnels de plus en plus exigeants, les autodirecteurs sont devenus de plus en plus complexes, ce qui s'est traduit par une augmentation notable de la part prise par ce sous-ensemble dans le coût global d'un missile.

C'est dans le but de réduire le coût et l'encombrement d'un autodirecteur à imagerie infrarouge qu'il a été entrepris une étude sur le concept d'autodirecteur rigide.

Dans ce concept, le dispositif optomécanique de stabilisation de la ligne de visée est supprimé, le capteur infrarouge étant rigidement lié au missile. Ce dernier devient ainsi l'unique moteur de rotation de l'axe de visée.
L'image délivrée par le capteur IR doit fournir toutes les informations nécessaires au guidage de l'engin.

Dans le cas étudié d'un guidage en navigation proportionnelle, les informations à connaitre sont :

- mouvements relatifs du missile et de la cible
- mouvements absolus du missile

Le premier point revient à un calcul d'écartométrie, ce qui est réalisé par des traitements d'image classiques. L'autodirecteur doit, de ce fait, conserver la cible dans son champ de vision pendant toute la durée du vol, ce qui nécessite une couverture de champ élevée pour encaisser les mouvements relatifs du missile et de la cible.

Le deuxième point relève d'un calculateur de traitement d'image spécifique, qui ne peut fonctionner que sur un fond structuré.

Les algorithmes de traitement d'image permettant d'extraire les informations de guidage à partir de l'image ont été largement étudiés ces dernières années.

Les travaux effectués dans ce domaine restent le plus souvent limités à une approche théorique, sans prise en compte de contraintes opérationnelles.

On peut citer, par exemple, les travaux de A. Bruss et B. Horn [1] ou de K. Prazdny [2] qui proposent des méthodes de calcul des trois composantes du vecteur de rotation et des cosinus directeurs du vecteur de translation d'une caméra par rapport à un scène structurée.
Ces méthodes nécessitent la détermination du champ de vitesse des points de l'image (optical flow field).

Dans la méthode développée par H. Zinner [3], le calcul de ces mêmes paramètres est effectué directement, à partir des gradients spatiaux et temporels de l'image. La scène observée est supposée plane, ce qui permet de déterminer des paramètres supplémentaires, fonctions de l'attitude du porteur par rapport à la scène.

L'étude présentée dans cet article a pour but de déterminer le domaine opérationnel d'emploi d'un autodirecteur rigide.

Elle est limitée au domaine Air-Sol ou Sol-Sol.

Les cibles aériennes ont en effet été abandonnées en raison de leur grande évolutivité, entraînant des vitesses de rotation élevées du capteur susceptibles de générer du flou, et des champs importants que doit couvrir ce dernier pour compenser le traînage.

Une simulation a été entreprise à partir d'un modèle comportemental des traitements d'image, basé sur un modèle mathématique ainsi que sur le comportement réel de calculateurs de traitement d'image d'écartométrie existants.

Ce modèle sert de base à une simulation paramétrée globale, incluant l'autopilote et la cinématique du missile.

Afin de couvrir un domaine opérationnel suffisamment large, deux types de cellules aérodynamiques aux caractéristiques trés différentes ont été étudiées :

* Sous-munition à faible constante de temps et manoeuvrabilité élevée

* Bombe lourde à constante de temps élevée et faible manoeuvrabilité

La simulation a deux objectifs principaux :

* dimensionner le capteur

* étudier l'influence des perturbations sur le guidage et notamment sur la distance de passage

Ceci est obtenu en faisant varier les différents paramétres de la boucle, notamment le nombre de trames analysées par le traitement d'image, les temps de retard liés aux capteurs utilisés et les amplitudes de bruit.

On commencera par décrire le modèle utilisé pour la simulation.

On présentera ensuite le résultat des simulations dans les deux configurations opérationnelles retenues.

On examinera enfin l'apport d'une girouette au concept.

.3. DESCRIPTION DU MODELE UTILISE

3.1. DESCRIPTION GENERALE DE LA BOUCLE

Le modèle retenu pour faire l'évaluation des performances de l'autodirecteur rigide, est un modèle plan à deux dimensions.

Ce modèle est valable dans l'hypothèse où les deux plans de navigation du missile (tangage et lacet) sont parfaitement découplés. Ceci est réalisé si le missile est équipé d'une stabilisation en roulis parfaite.

Les positions du missile et du but dans le plan de guidage sont repérées par leurs coordonnées respectives xm, ym, xb, yb dans un référentiel absolu arbitraire. Les notations angulaires sont définies sur la figure 1.

Les quatre modules principaux mis en oeuvre dans la simulation sont (fig. 2) :

* l'autodirecteur rigide, comportant l'écartométre ainsi qu'un calculateur mesurant la vitesse absolue de rotation du porteur

* l'autopilote

* la cinématique

* l'interface entre l'autodirecteur, le calculateur et l'autopilote, assurée par l' élaborateur d'ordre.

3.2. MODELE COMPORTEMENTAL DE L'AUTODIRECTEUR RIGIDE

Le modèle de l'autodirecteur rigide est relativement simplifié par rapport à un autodirecteur stabilisé.

Il ne rentre, en effet, aucun élément de stabilisation mécanique dans ses composants.

Ce modèle est purement phénomènologique, dans la mesure où la simulation entreprise ne comprend pas la partie traitement d'image.

On rappelle que la loi de navigation proportionnelle nécessite d'appliquer à l'autopilote un ordre proportionnel à la vitesse de rotation de la droite missile-but.

Cette vitesse est composée de la vitesse angulaire de rotation de la droite missile-but dans le référentiel lié au missile (égale à la dérivée de l'écartométrie) et de la vitesse absolue de rotation du missile.

L'autodirecteur rigide a donc pour fonctions de calculer la dérivée temporelle de l'écartométrie ainsi que la vitesse de rotation du porteur dans le référentiel absolu lié au fond de scène observé.

Le modèle correspondant doit :

* mettre en forme le signal d'écartométrie issu de l'image fictive et dériver ce signal.

* mettre en forme le signal de vitesse de rotation issu de l'image fictive.

* assurer l'accord de phase et de gain entre les deux signaux, afin de réaliser leur sommation cohérente.

3.2.1. Ecartometre

Le modèle de l'écartométre est basé sur les performances de modules existant, et ne nécessite donc pas des hypothèses phénomènologiques comme c'est le cas pour le calculateur des paramètres du mouvement.

L'écartométre réalise les opérations suivantes à partir du signal d'écartométrie cinématique (figure 3):

* retard :
Le retard est constitué d'un retard de calcul et d'un retard d'une demi période trame, correspondant à la formation d'image.

* échantillonage :
L'échantillonage est effectué à la période trame.

* bruitage :
Un bruit de mesure à densité de probabilité gaussienne peut être rajouté. Ce bruit est principalement du à la quantification spatiale de l'image, et est donc lié à la résolution du détecteur.

* écrêtage :
Le signal est écrêté lorsque la cible passe en dehors des limites du champ.

* dérivation :
Le guidage en navigation proportionnelle nécessite une dérivation, effectuée numériquement sur deux trames.

* correction de phase :
Un correction de phase est effectuée pour assurer l'accord de phase avec le calculateur. Cette correction est réalisée par un simple retard.

* blocage et gain :
Le signal est enfin bloqué et soumis à un gain K_e représentant le gain statique de l'écartométre.

3.2.2. Calculateur des paramètres du mouvement

Un certain nombre d'hypothèses sont utilisées pour modèliser le calculateur. Ces hypothèses sont issues de modèles mathématiques, dont ceux développés par H. Zinner [1] et S. K. Shee [4]. En particulier, on admet que le calcul des paramétres du mouvement peut être effectué au minimum sur 2 trames.

Les opérations effectuées par le calculateur à partir du signal de vitesse de rotation issu du module cinématique sont (figure 3):

* retard :
Le retard est constitué d'un retard de calcul et d'un retard de phase dépendant du nombre de trames analysées.
Si ntrames est le nombre de trames analysees, et tr la periode trame, le temps d'observation de l'image est:

$$(ntrames - 1) \times tr$$

Le retard de phase est égal à la moitié de cette valeur.

* échantillonage :
Effectué à la période trame

* seuillage
La valeur du seuil est liée à la résolution r du détecteur par :

$$s = r/4 \times (ntrames - 1) \times tr$$

* bruit de quantification :
Un bruit de quantification à densité de probabilité gaussienne peut être rajouté. Ce bruit est lié à la quantification spatiale de l'image. Il est exprimé en vitesse de rotation.

* bruit de calcul :
Ce bruit est du à la mauvaise qualité de l'image(flou) et à la précision de l'algorithme de calcul.
La précision est proportionnelle à la vitesse de rotation du porteur, et inversement proportionnelle a la racine du nombre de trames étudiées [4].
L'écart type de bruit est donc mis sous la forme :

$$\%err \cdot \dot{\Theta} \cdot \sqrt{2} / (ntrames)^{\frac{1}{2}}$$

où %err représente le pourcentage d'erreur sur la mesure de la vitesse de rotation absolue du porteur ($\dot{\Theta}$) lorsqu'on analyse deux trames.
Ce pourcentage est un paramètre de la simulation.

* blocage d'ordre 0 et gain :
Le gain du calculateur doit être égal au gain de l'écartométre pour assurer l'accord de gain.

3.3. MODELE DE L'AUTOPILOTE

Le module autopilote permet de simuler la réponse aérodynamique de la cellule lorsqu'on transmet une commande aux gouvernes.

La cellule représentée est très générale. La fonction de transfert simulée permet de modèliser des configurations aérodynamiques très générales.

Les limites du modèle sont les suivantes :

* la simulation plane est bien adaptée aux missiles du type cruciforme, ayant deux plans de voilure identiques.

* Les coefficients aérodynamiques sont constants au cours du vol.

3.4. MODULE CINEMATIQUE

Le module cinématique a pour fonction de déterminer l'écartométrie cinématique à partir des informations de vitesse de rotation et d'accélération latérale du porteur, connaissant sa position initiale et connaissant la position de la cible à chaque itération.

Ce module a également pour fonction de calculer la distance de passage et de provoquer l'arrêt de la boucle en fin de guidage.

3.5. ELABORATEUR D'ORDRE

Le module élaborateur d'ordre a pour fonction de mettre en forme le signal provenant de l'autodirecteur pour commander l'autopilote de manière à réaliser la loi de guidage :

$\Gamma_{MN} = A\ V_M\ d\eta_B / dt$ pour une navigation proportionnelle sans biais avec un gain A
et
$\Gamma_{MN} = A\ V_M\ d\eta_B / dt + B$ pour une navigation proportionnelle avec un biais B.

L'élaborateur d'ordre réalise également un filtrage passe bas de la tension de commande.

.4. SIMULATION

Ce paragraphe décrit les résultats de la simulation d'un vol accroché sur une cible fixe dans les deux cas opérationnels retenus (Bombe et sous-munition).

4.1. CAPTEUR

Deux types de capteurs IR sont envisagés.
Il s'agit de détecteurs matriciels à grand nombre de points susceptibles de couvrir un champ horizontal de +/- 15 degrés sans dispositif de balayage :

* matrice MCT 256 x 256 pixels, période trame 5ms, résolution 2 mrad.

* matrice PtSi 512 x 512 pixels, période trame 20ms, résolution 1 mrad.

La période d'échantillonage (= période trame) est liée au temps d'intégration du détecteur, qui est plus important pour un détecteur PtSi.

4.2. DONNEES QUANTITATIVES POUR L'ECARTOMETRE

Les principales caractéristiques de l'écartométre sont présentées dans le tableau 1 pour les deux types de détecteurs envisagés.

Détecteur	Temps de calcul Tce	Période d'échantillonage Tr	Retard total Tce + ½ Tr
MCT 256 x 256 pixels	20 ms	5 ms	22,5 ms
PtSi 512 x 512 pixels	20 ms	20 ms	30 ms

Tableau 1

Un retard supplémentaire doit être ajouté aux valeurs indiquées pour réaliser l'accord de phase avec le calculateur.

Le gain statique de l'écartométre est 1.

Les valeurs des écarts-type de bruit de quantification sont présentées dans le tableau 2 pour les deux types de détecteur, avec une quantification de l'écartométrie de ¼ de pixel.

Détecteur	Quantification de l'écartométrie	ecart-type de bruit
MCT	0,5 mrad	0,144 mrad
PtSi	0,25 mrad	0,072 mrad

Tableau 2

4.3. DONNEES QUANTITATIVES POUR LE CALCULATEUR

Le calculateur est supposé générer la vitesse de rotation absolue de la sous-munition à partir de l'image, avec un retard de calcul de 40 ms (compte tenu de la complexité de l'algorithme de calcul utilisé).

Pour obtenir le retard total, il faut ajouter le retard de phase dépendant du nombre de trames analysées.

Le retard total est dans tous les cas, supérieur à celui de l'écartométre ce qui impose une correction de phase. Les valeurs des retards et de la correction de phase de l'écartométre sont indiquées dans le tableau 3.

Capteur	Période trame	Retard Ecartométre	Nombre de trames	Retard Calculateur	Correction phase
PtSi	20 ms	30 ms	2	50 ms	20 ms
			3	60 ms	30 ms
			4	70 ms	40 ms
MCT	5 ms	22,5 ms	2	42,5 ms	20 ms
			3	45 ms	22,5 ms
			4	47,5 ms	25 ms

Tableau 3

Les perturbations introduites sont:

- le bruit de quantification dont l'écart type de bruit est exprimé en vitesse de rotation (tableau 4).

Détecteur	Quantification	Ecart-type de bruit
MCT	0,1 rad/s (5,7 °/s)	0,03 rad/s (1,65 °/s)
PtSi	0,0125 rad/s (0,7 °/s)	3,6 mrad/s (0,2 °/s)

Tableau 4

- le bruit lié à la mauvaise qualité de l'image et à la précision de l'algorithme qui est paramétré.

- le seuil de mesure, lié à la quantification des écarts angulaires, qui dépend du nombre de trames étudiées (tableau 5).

Détecteur	Seuil ntrames = 2	Seuil ntrames = 3	Seuil ntrames = 4
MCT	5,7 °/s	2,85 °/s	1,425 °/s
PtSi	0,7 °/s	0,35 °/s	0,175 °/s

Tableau 5

4.4. SIMULATION AVEC LA SOUS-MUNITION

4.4.1. Donnees sur l'autopilote

L'ensemble de l'autopilote a une fonction de transfert approximativement équivalente à un 2ème ordre d'amortissement x = 0 7 et de pulsation propre Ω = 25 rad/s.
L'incidence lag moyenne est : λ = 0,7 s.

L'ensemble de ces valeurs correspond à un missile ayant d'importantes possibilités de manoeuvre.

4.4.2. Configuration opérationnelle

La sous-munition accroche la cible à 1000m en distance oblique à une altitude de 270m. Sa vitesse est 280 m/s. Les directions de la vitesse initiale du missile et de l'axe de visée sont confondus et parallèles au sol.
L'écartométrie à l'accrochage est de 15 degrés (fig. 4a).

4.4.3. Fonctionnement sans perturbations sur cible fixe

Le guidage est effectué avec un gain de navigation proportionnelle de 3.

Afin d'arrondir la trajectoire et de limiter le facteur de charge, un biais de +5°/s est appliqué. Le calculateur analyse 2 trames.

Un premier calcul a été effectué sans perturbations.

Au moment où la cible est accrochée et où le guidage commence, l'écart angulaire entre la cible et l'axe de visée du missile est 15°, ce qui correspond à appliquer un échelon d'écartométrie (ou une impulsion en dε/dt).

Ceci explique un pic en vitesse de rotation missile (qui est également la vitesse de rotation de l'axe de visée puisque le détecteur est rigidement lié au missile) aux alentours de 36°/s avec une stabilisation finale vers 10°/s (fig. 5).

Le facteur de charge, quant à lui, se maintient aux alentours de 5g jusqu'à la fin du vol (fig. 6). Ceci est caractéristique d'une navigation proportionnelle biaisée qui impose une trajectoire circulaire (accélération latérale constante).

L'écartométrie décroît rapidement au départ puis régulièrement jusqu'à des valeurs négatives (fig. 7).

La distance de passage est de l'ordre de quelques µm, ce qui découle de l'abscence de perturbations.

La vitesse de rotation élevée atteinte à l'accrochage nécessite de considérer son influence sur la qualité de l'image.

Si on tolére un pixel de flou, on constate que la vitesse relative du capteur et du fond de scène observé ne doit pas dépasser 6°/s pour un capteur PtSi à temps d'intégration 20 ms.

Pour un détecteur MCT à temps d'intégration 5ms, la tolérance au flou raméne la limite à 45°/s.

Cette constatation impose le choix d'un capteur MCT pour cette application.

4.4.4. Fonctionnement avec perturbations sur cible fixe

Les conditions de la simulation sont :

- capteur : MCT
- erreur commise sur la mesure de vitesse de rotation : 5% (pour 2 trames analysées)
- biais de navigation proportionnelle : 5 °/s

Le tableau 6 résume les résultats obtenus lorsqu'on fait varier le nombre de trames analysées en présence de l'ensemble des perturbations décrites précedemment.

On trouve en première colonne le nombre de trames analysées par le calculateur. La deuxième colonne fournit la distance de passage moyenne sur 20 vols, et la troisième colonne l'écart-type associé. La quatrième colonne présente le facteur de charge maximum observé sur l'ensemble des vols.

Nombre de trames	Distance de passage	Ecart-type	Facteur de charge max
2	0,57 m	0,4	6g
3	0,35 m	0,3	6g
4	0,3 m	0,2	6g

Tableau 6

On constate une amélioration des performances avec le nombre de trames analysées, imputable à la diminution de l'erreur de mesure (dans un rapport inversement proportionnel à la racine du nombre de trames analysées).

Bien qu'on puisse encore diminuer le bruit de mesure en analysant plus de trames, ceci aurait pour conséquence une augmentation du retard du calculateur, et donc du traînage de la boucle de guidage. Ce problème devient prépondérant, surtout dans une configuration opérationnelle sévère (voir plus loin), par rapport à la réduction du bruit.

4.4.5. Analyse des résultats

Les performances de l'autodirecteur rigide sont limitées, dans le cas de la sous-munition, par la vitesse de rotation élevée que peut atteindre le capteur par rapport au fond de sol.
Ceci conduit à retenir pour cette application un capteur MCT à temps d'intégration court.
Le champ du capteur est dimensionné par l'écartométrie initiale (à l'accrochage), cette valeur initiale n'étant pas dépassée en valeur absolue.

4.5. SIMULATION AVEC LA BOMBE GUIDEE

Il s'agit d'étudier le comportement d'une bombe lourde avec une manoeuvrabilité limitée, en vol accroché sur une cible fixe.

4.5.1. Modèle de l'autopilote

L'autopilote de la bombe peut être représenté, de manière simplifiée, par un système du deuxième ordre avec un amortissement typique de x = 0,9 et une pulsation propre de 1,8 rad/s. L'incidence lag moyenne est Λ = 2 sec.

4.5.2. Configuration opérationnelle

La bombe accroche la cible à une distance de 3000 m avec un écart initial entre la ligne de visée et la direction de la cible de 15°. Elle a une vitesse de 150m/s.

Le guidage s'effectue en navigation proportionnelle dans un plan horizontal (plan de lacet).

4.5.3. Fonctionnement sans perturbations sur cible fixe

Le guidage est effectué en navigation proportionnelle avec un gain de navigation de 3 sans perturbations. Aucun biais n'est appliqué.

Le calcul effectué montre que le facteur de charge ne dépasse pas 0,6g tandis que la vitesse de rotation de la bombe n'excède pas 4°/s (fig. 8).

La distance de passage est de l'ordre de quelques μm.

Il apparaît, contrairement au cas de la sous-munition, que la limitation au guidage est due à la borne inférieure de la plage de bon fonctionnement du calculateur et non plus à la limite supérieure.

Ceci découle directement de la lenteur de manoeuvre de la bombe.

C'est pourquoi la solution retenue pour la suite des calculs est un capteur PtSi, qui permet d'observer des vitesses faibles (< 6°/s) en raison de son temps d'intégration élevé.

4.5.4. Fonctionnement avec perturbations sur cible fixe

Une série de calcul a été menée avec pour bases les caractéristiques suivantes :

- Détecteur : PtSi
- Gain de navigation proportionnelle : 3
- Biais de navigation : sans

Les paramètres d'entrée sont :

- Le nombre de trames analysées
- Le pourcentage d'erreur sur la mesure du calculateur quand on analyse deux trames

et les résultats :

- La moyenne des distances de passage pour 20 calculs
- L' écart-type associé
- Le facteur de charge maximum observé sur l'ensemble des vols

L'écartométrie maximum reste dans tous les cas égale à la valeur initiale de 15° en valeur absolue. Les résultats sont résumés dans le tableau 7.

La source de perturbation principale est le seuil de mesure. C'est ce qui explique une distance de passage importante en l'abscence de bruit de mesure, qui ne tient pas seulement à la présence du bruit de quantification.

Le seuil peut être réduit en augmentant le nombre de trames analysées. Ceci a cependant pour conséquence un traînage plus important de la boucle de guidage, de sorte qu'on observe une distance de passage optimum pour 3 trames. C'est cette valeur qui a été conservée pour étudier le système avec un bruit d'erreur de mesure (10% sur 2 trames).

On constate que ce bruit n'a d'influence que sur l'écart-type.

Nombre de trames	%erreur	Distance de passage	Ecart-type	Vitesse rotation max	Facteur de charge max
2	0	0,5 m	0,3	4°/s	0,6 g
3	0	0,2 m	0,2	4°/s	0,6 g
4	0	0,3 m	0,15	4°/s	0,6 g
3	10	0,2 m	0,25	4°/s	0,6g

Tableau 7

Cette faible influence tient à la bonne précision du calculateur pour mesurer de faibles vitesses.

4.5.5. Analyse des résultats

Le fonctionnement de la bombe est limité par la borne inférieure de la plage de fonctionnement du détecteur et non plus par la borne supérieure.

Ce cas est préférable parce que le seuil de fonctionnement peut être réduit alors que la limite supérieure est irréductible.

Pour diminuer le seuil de fonctionnement, il faut observer l'image sur un temps suffisamment long. Deux solutions sont envisageables :

- Matrice PtSi
- Matrice MCT avec post-intégration sur plusieurs trames

Dans les deux cas, il est de toute manière possible d'effectuer les calculs de traitement d'image sur plusieurs trames, afin d'augmenter leur précision, et leur capacité à estimer de faibles vitesses de déplacement, en restant toutefois dans la limite due au traînage (optimum 3 à 4 trames pour le PtSi).

On se trouve dans un cas particulièrement favorable au fonctionnement de l'AD rigide, où ne se pose pas le problème du flou et où les distances de passage en présence de perturbations restent dans des limites acceptables pour cette application (< 1m).

Comme dans le cas de la sous-munition, le champ du capteur est dimensionné par l'écartométrie à l'accrochage.

.5. APPORT DE LA GIROUETTE

La girouette est une structure articulée qui s'oriente suivant la direction du vecteur vitesse du missile.

Dans le cas où le capteur est solidaire d'une girouette, celui-ci est découplé des mouvements d'incidence, qui peuvent être particulièrement gênants, dans le cas de la sous-munition.

C'est ce qu'on constate lorsqu'on impose à la sous-munition un fonctionnement dans une configuration opérationnelle sévère. Une configuration opérationnelle délicate mais cependant réaliste est par exemple représentée par le cas de la figure 9, où la sous-munition accroche la cible à 500m en distance oblique à une altitude de 150m.

Dans cette configuration, l'écartométrie initiale est de 17,5°. Pour assurer la couverture de champ, il est possible de décaler la ligne de visée d'une dizaine de degrés vers le sol par rapport à l'axe du missile. Ceci sera confirmé par la suite.

Si on considère la figure 10 on constate que la prise d'incidence de la sous-munition, au moment de l'accrochage, qui permet au missile d'atteindre un facteur de charge latéral de 10g, s'accompagne de vitesses de rotation missile très élevées (> 180°/s).

Ces vitesses sont incompatibles avec le fonctionnement du capteur aussi bien PtSi que MCT, en raison du flou qu'implique une telle vitesse de défilement du fond de scène. Il en résulte une période "aveugle" de l'ordre du 1/10ème de seconde pendant laquelle le capteur ne peut délivrer les informations de guidage.

D'autre part, on a constaté qu'un écrêtage de la vitesse de rotation dans des limites supportables par un capteur MCT, s'accompagnait d'un traînage de la boucle de guidage, d'où résultaient des distances de passage importantes.

Le facteur de charge, quant à lui, atteint rapidement 10 g et s'y maintient jusqu'à la fin du vol (fig. 11). Ceci est favorable à l'utilisation d'une girouette.

En effet, la vitesse de rotation du vecteur vitesse, qui est la vitesse de rotation vue par le capteur, est le rapport de l'accélération latérale sur la vitesse missile. On voit clairement que la vitesse de la girouette ne dépassera pas 20°/s dans le cas de la sous-munition.

Il semble donc intéressant d'étudier le guidage avec girouette dans ce cas.

5.1. MODELE DE LA GIROUETTE

La girouette est supposée découpler parfaitement les mouvements d'incidence dans le plan de guidage. Ceci implique un débattement de la girouette par rapport à l'axe missile égal à l'angle d'incidence maximum (fig. 12).

L'angle d'incidence α , se déduit de l'accélération latérale par :

$$\alpha = \Lambda \, \Gamma_{MN} / V_M$$

où Λ est l'incidence lag.

La girouette suivant exactement les mouvements du vecteur vitesse, l'optical flow généré au niveau du capteur, solidaire de celle ci permet de déduire la vitesse de rotation absolue du vecteur vitesse, égal à la dérivée de la pente aérodynamique.

Comme l'écartométrie est d'autre part mesurée entre la droite missile-but et la vitesse de l'engin, la somme :

$$\dot{\epsilon} + \dot{\theta}$$

fournit bien la vitesse de rotation absolue de la droite missile-but.

Il n'y a donc aucune modification de principe dans le cas d'un fonctionnement avec girouette :

- l'entrée du calculateur est la dérivée de la pente aérodynamique (issue du module cinématique)

- l'entrée de l'écartométre est l'écartométrie cinématique calculée sans girouette à laquelle on ajoute l'angle d'incidence.

5.2. SIMULATION AVEC LA SOUS-MUNITION

La configuration opérationnelle retenue est la configuration de la figure 9.

Les conditions de la simulation sont les suivantes :

- pas d'écrêtage des tensions de commande

- filtrage de la tension de commande : BP 10 Hz

- biais de NP : 10°/s

Un premier calcul a été effectué sans perturbations.

Les figures 13 à 16 présentent respectivement les évolutions temporelles, de la vitesse de rotation de la girouette, du facteur de charge de l'écartométrie, et de l'angle d'incidence.

Comme attendu, on constate que la vitesse de rotation du capteur reste sensiblement voisine de 20°/s.

L'écartométrie, quant à elle, évolue pratiquement linéairement de sa valeur initiale à une valeur nulle, ce qui justifie le calage de la ligne de visée à un valeur intermédiaire entre les deux extrêmes.

On observe enfin que l'incidence prise par le missile n'excède pas 15 degrés.

Un deuxième calcul a été effectué en introduisant les bruits de quantification et le bruit de mesure du calculateur.

Les conditions sont :

- écart type du bruit de mesure à 10% de la vitesse de rotation (pour 2 trames analysées)

- nombre de trames analysées : 4

On observe une distance de passage moyenne de 0,22m avec un écart-type de 0,14.

Les figures 17 à 20 présentent l'allure de l'évolution des différents paramètres pour un vol perturbé.

5.3. ANALYSE DES RESULTATS

Les bons résultats obtenus sont liés au fait que la vitesse de la girouette reste faible, ce qui implique une bonne précision du calculateur.

Pour assurer un bon fonctionnement en vol perturbé, il est nécessaire d'assurer un débattement de la girouette de l'ordre de 17 degrés (fig. 20).

Dans le cas d'une généralisation à deux axes (site et gisement), il faut concevoir une girouette 2 axes à débattement important.

Un autre problème est celui des vibrations dues à des résiduels aérodynamiques de la girouette.

La solution girouette reste donc hypothétique dans le cas de la sous-munition.

.6. CONCLUSION

a) Le premier objectif de la simulation était de dimensionner le capteur. Les points essentiels qui ressortent de l'étude paramétrique sont les suivants :

* L'écartométrie ne dépasse pratiquement pas sa valeur initiale au cours du vol.

Le champ du capteur est donc essentiellement dimensionné par la valeur nécessaire à l'acquisition en début de vol, compte tenu des erreurs de pointage.

* La vitesse de rotation de l'AD rigide par rapport au fond de sol est le facteur dimensionnant pour le temps d'intégration du détecteur et pour la résolution.

On peut définir une plage de fonctionnement (théorique) de l'autodirecteur selon le type du détecteur.

Cette plage a une limite inférieure (ou seuil) égale au rapport de la quantification angulaire du capteur, imposée par la résolution, sur le temps d'observation de l'image.

Ce seuil est imposé par le calculateur des paramètres du mouvement.

La limite supérieure est imposée par le flou induit sur l'image.

Il résulte de ces considérations qu'un engin ayant de faibles vitesses de rotation, comme la bombe guidée, nécessite un temps d'observation de l'image élevé pour obtenir une précision suffisante sur la mesure des faibles vitesses.

Un engin doué d'une grande manoeuvrabilité, comme la sous-munition, nécessite au contraire de faibles temps d'intégration, de manière à éviter le phénomène de flou et à encaisser les pics de vitesse.

Le capteur MCT est donc bien adapté pour la sous munition, tandis que le capteur PtSi peut être employé dans la bombe.

Les simulations entreprises ont finalement permis le choix d'un capteur pour chaque application.

b) Le deuxième objectif de la simulation était d'observer l'influence de diverses perturbations sur le guidage notamment:

- le temps de retard du calculateur (temps de calcul et nombre de trames analysées).

- les bruits de mesure et de quantification du calculateur et de l'écartométre

Les simulations montrent des performance acceptables en terme de précision de guidage, avec des distances de passage largement inférieures au métre, compte tenu de la variation des différents paramètres.

c) Enfin le dernier point concernait l'apport d'un d'une girouette au fonctionnement de l'autodirecteur.

L'intérêt de la girouette n'apparait clairement que pour la sous-munition dans une configuration opérationnelle sévère.
Il ressort des simulations entreprises, que la girouette est une solution particulièrement efficace à ce problème, son rôle étant de découpler les mouvements de prise d'incidence responsable des pics de vitesse de rotation au moment de l'accrochage.

d) Il reste à connaitre de manière quantitative la précision des algorithmes de traitement d'image, ce qui ne pourra être obtenu qu'en remplacant le modèle phénomènologique par le traitement d'image réel.

Une simulation complète 2 axes comportant le logiciel de traitement d'image réel, est en cours de réalisation et permettra de confirmer précisément la validité du concept d'autodirecteur rigide.

REFERENCES BIBLIOGRAPHIQUES

[1] A. R. Bruss and B. K. P. Horn, "Passive navigation", Computer Vis. Graph. Im. Proc. 21, 3-20 (1983)

[2] K. Prazdny, "Determining the instantaneous direction of motion from optical flow generated by a curvilinearly moving observer", Com. Graph. Proc. 17(3), 238-248 (1981)

[3] H. Zinner, "Determining the kinematics parameters of a moving imaging system by processing spatial and temporal intensity changes"
J. Opt. Soc. Am. A3, 1512-1517, (1986)

[4] Shantana K. Shee, "Analysis and implementation of direct passive navigation". Applications of artificial intelligence V (1987).

REMERCIEMENTS

La présente étude a été soutenue par les services officiels français (DRET : Direction ces Recherches Etudes et Techniques).

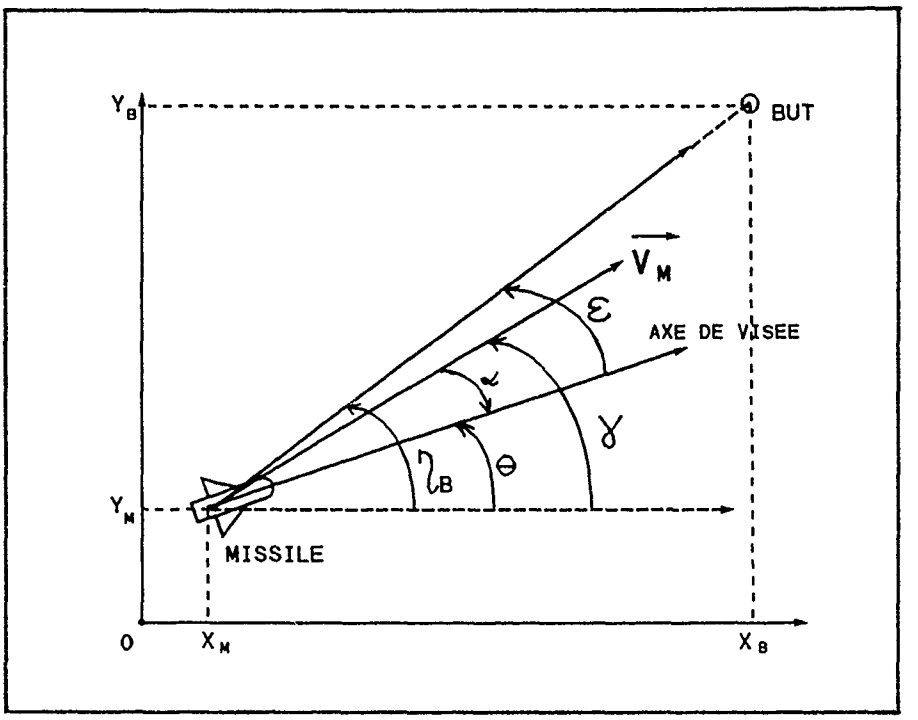


Figure 1
Notations angulaires

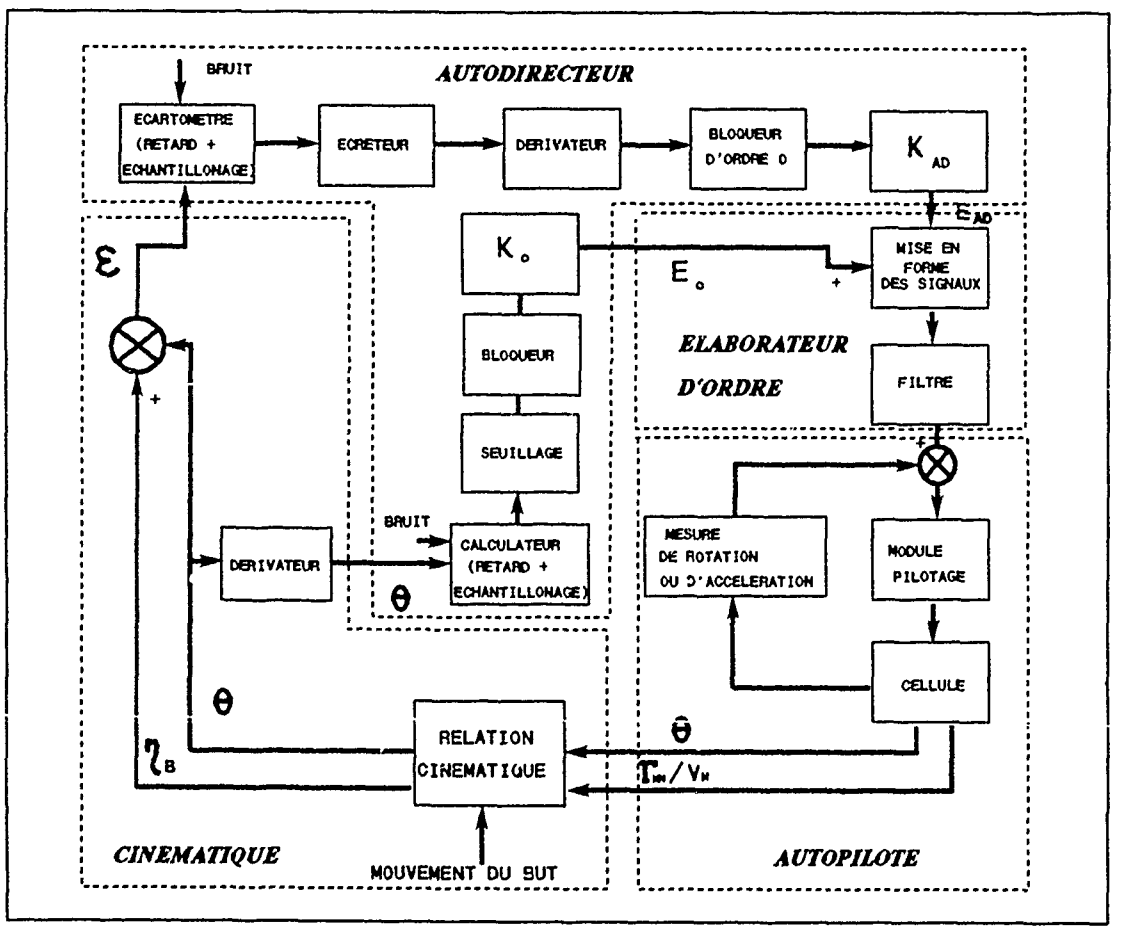


Figure 2
Boucle complète

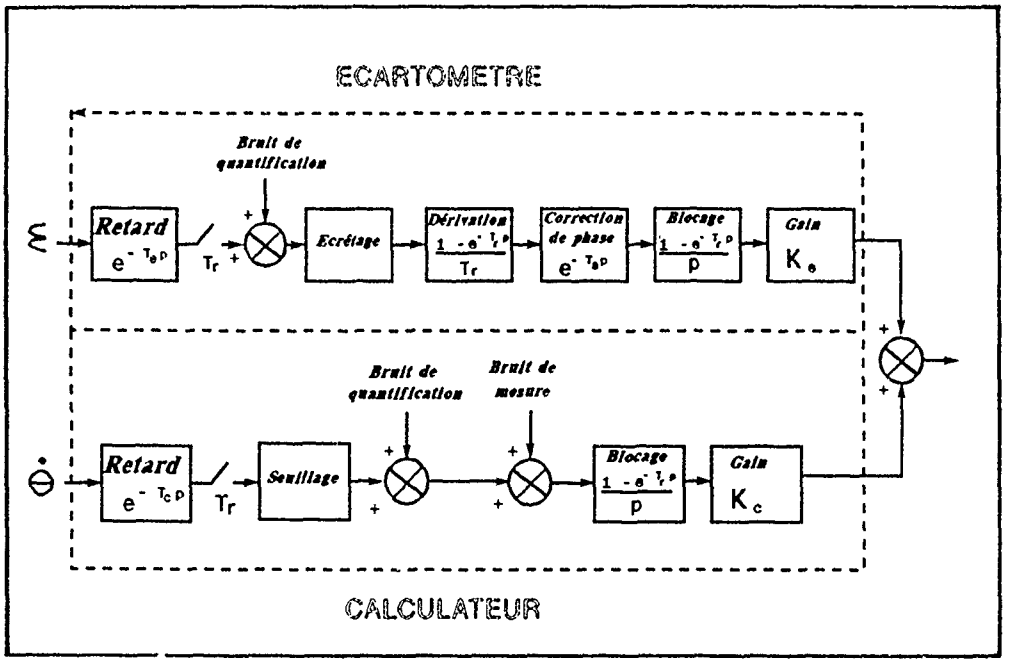


Figure 3
Modèle comportemental de l'autodirecteur rigide

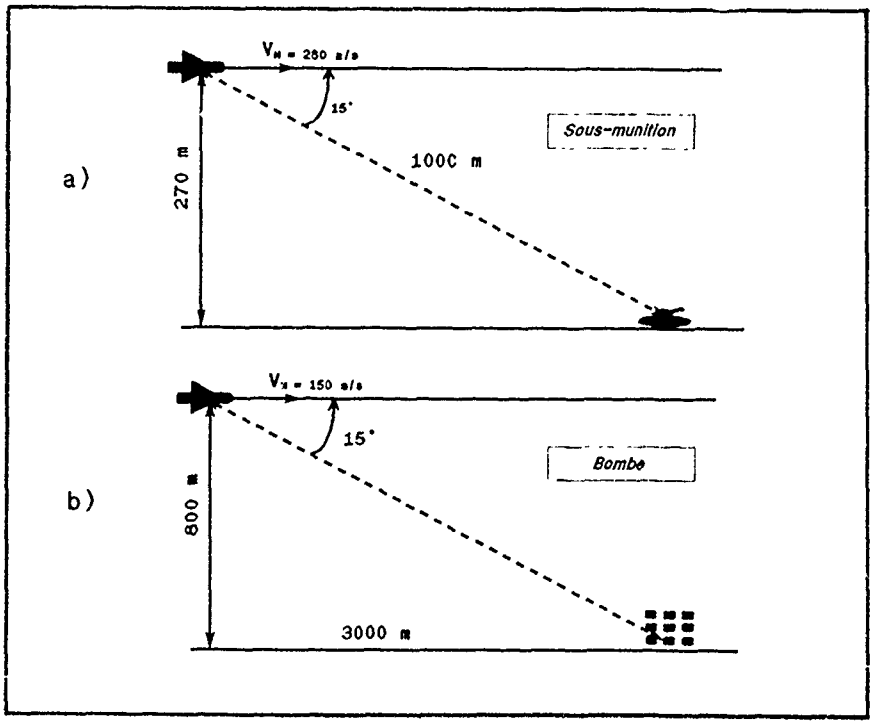


Figure 4
Configuration opérationnelle pour la sous-munition et la bombe

Sous-munition sans perturbations

Figure 5

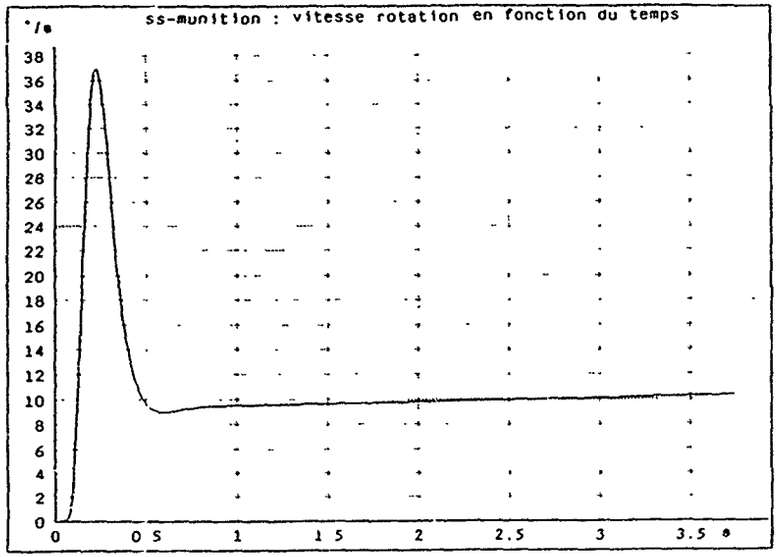
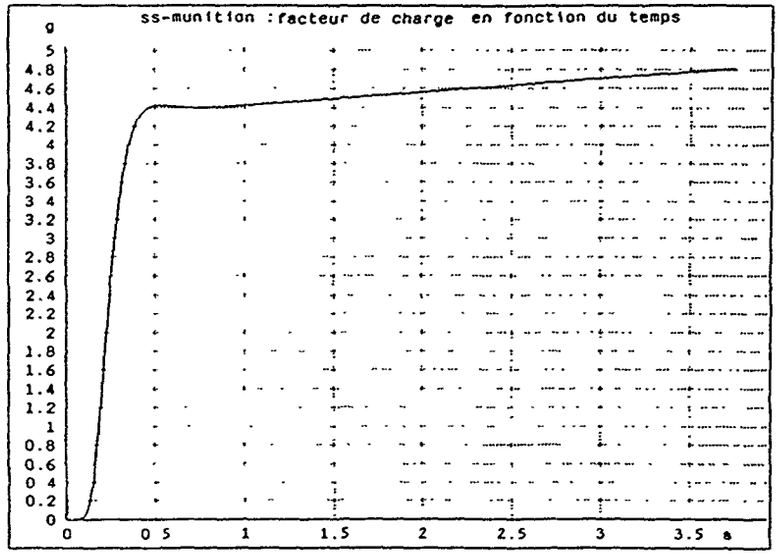
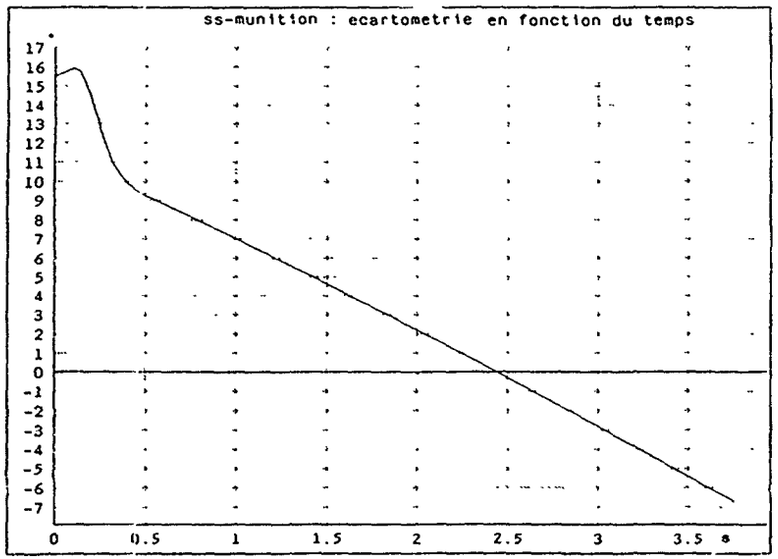


Figure 6



Sous-munition sans perturbations

Figure 7



Bombe guidée sans perturbations

Figure 8

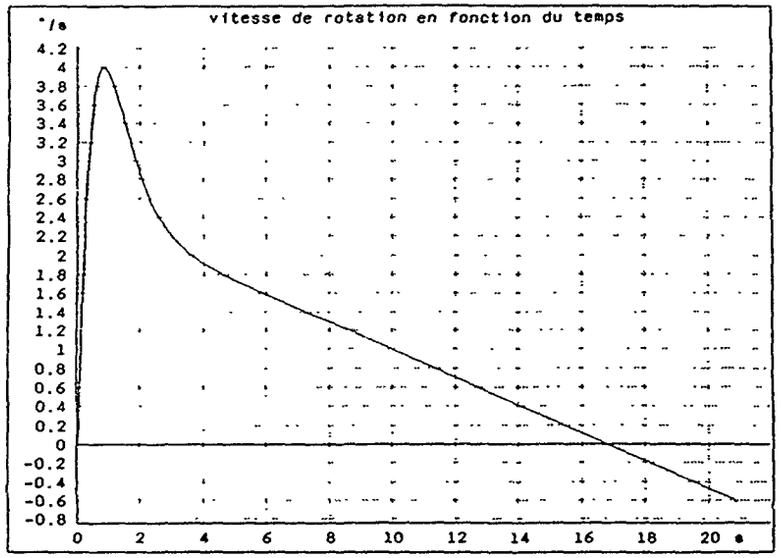


Figure 9

Configuration opérationnelle pour la sous-munition avec girouette

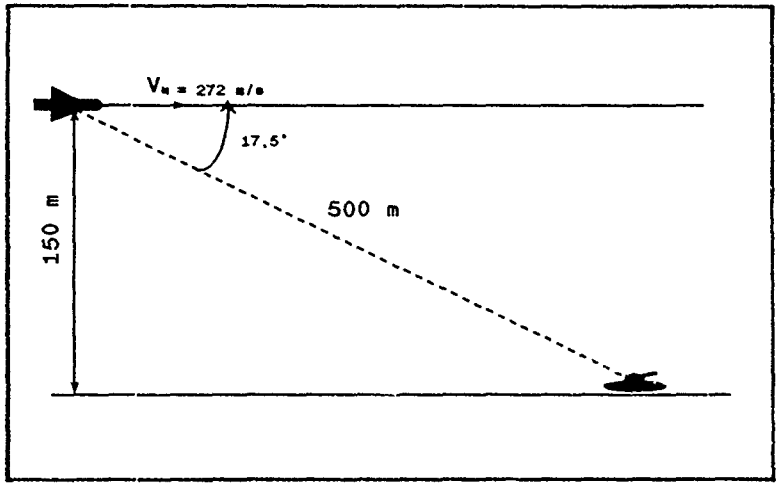
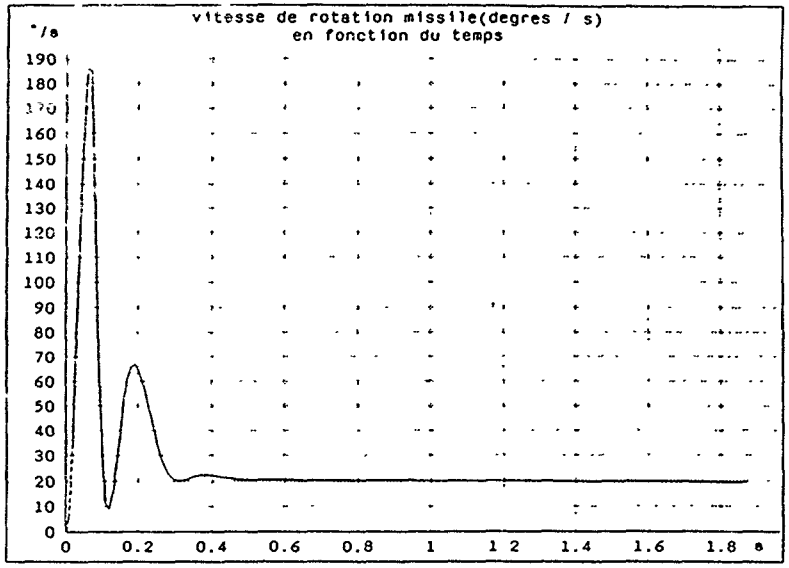


Figure 10



Sous-munition sans girouette dans une configuration opérationnelle sévère

Figure 12

Modèle de la girouette

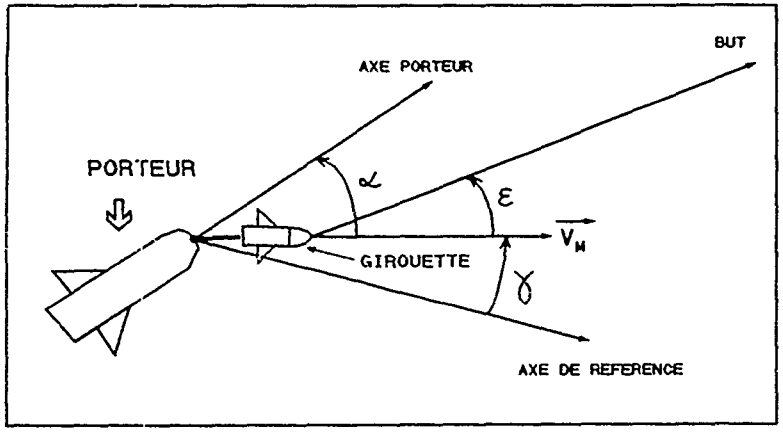


Figure 11

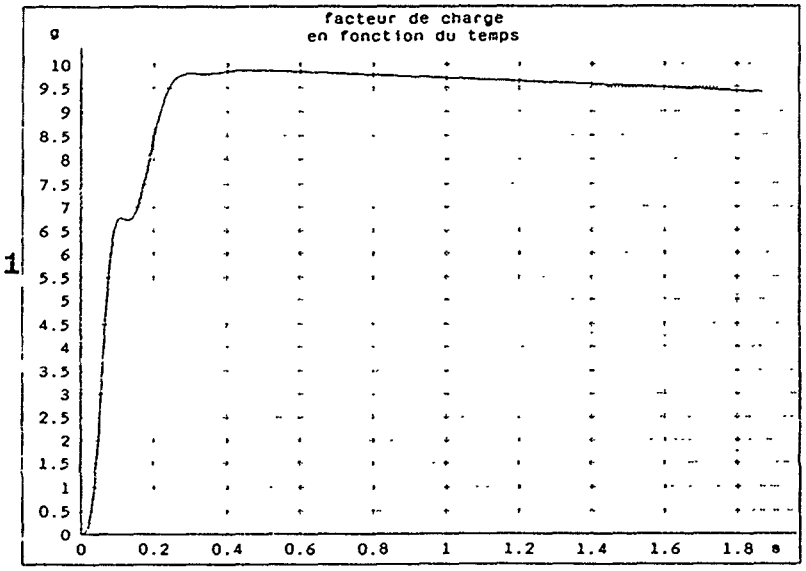


Figure 13

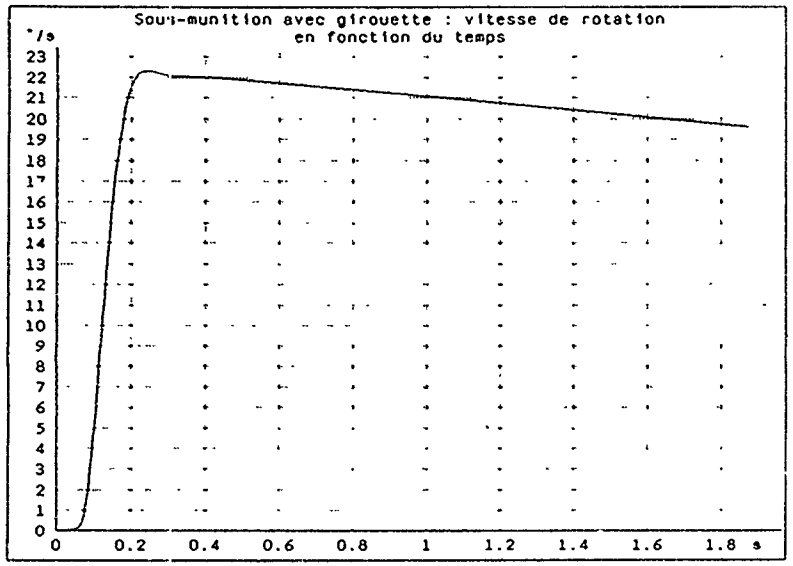
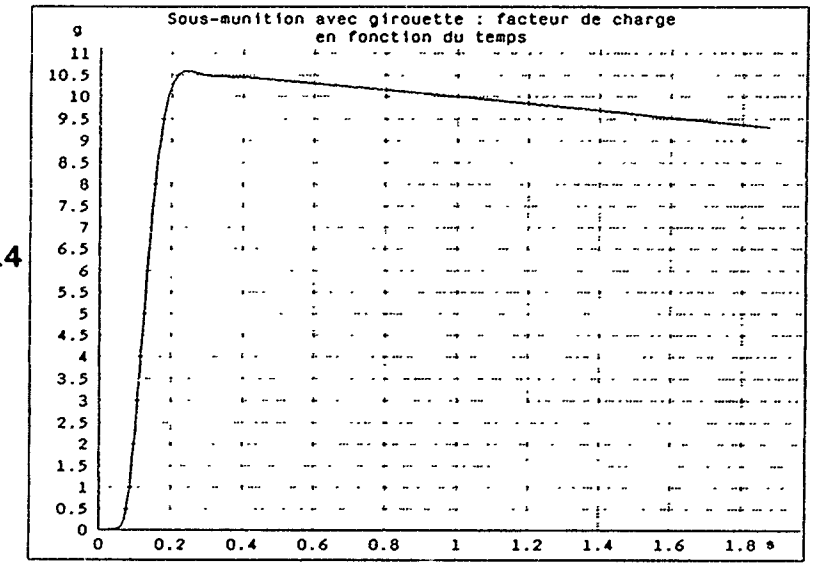


Figure 14



Sous-munition avec girouette sans perturbations

Figure 15

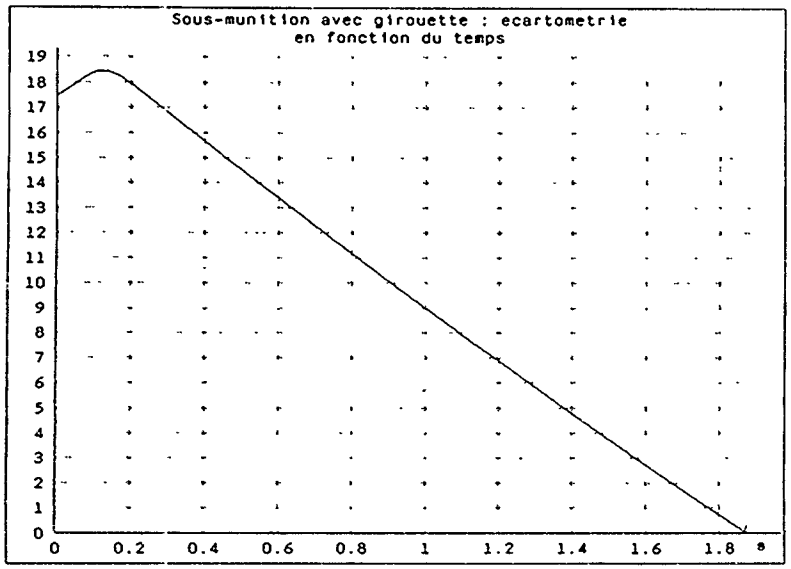


Figure 16

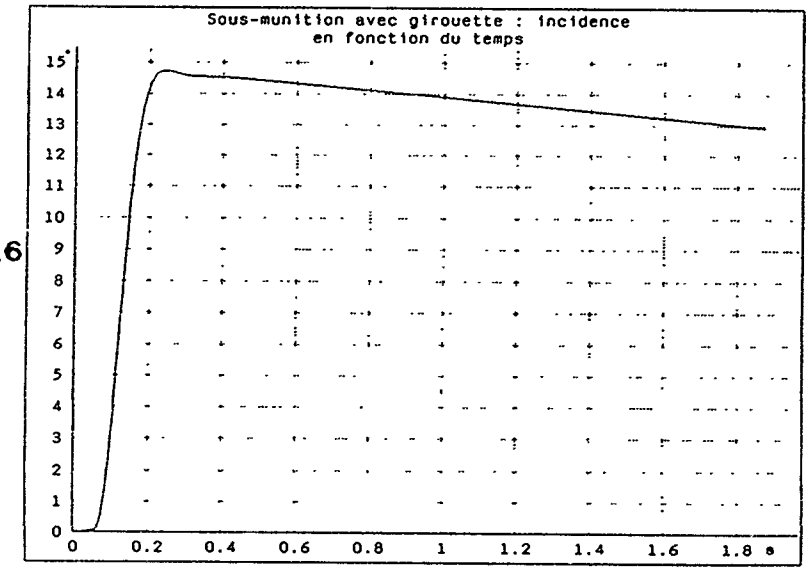


Figure 17

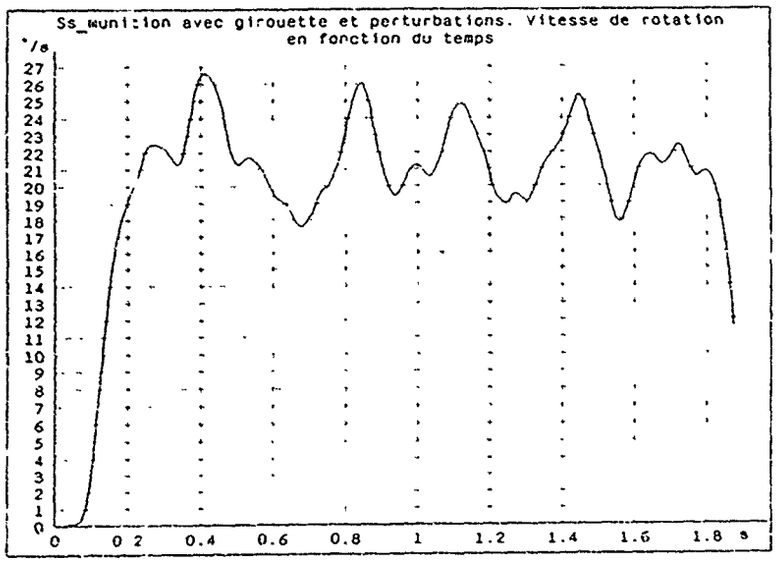
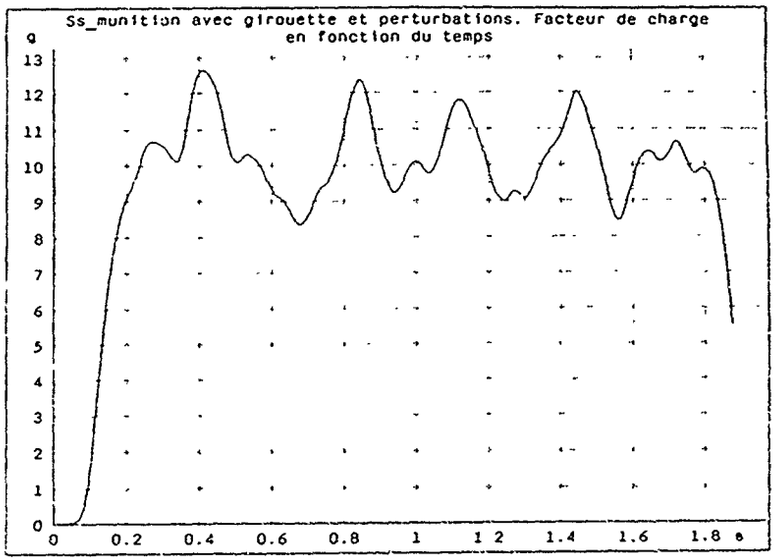


Figure 18



Sous munition avec girouette et perturbations

Figure 19

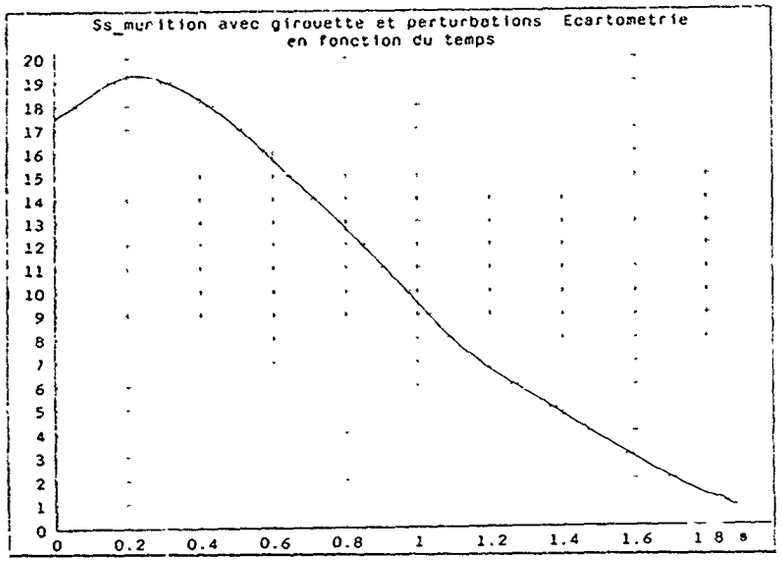
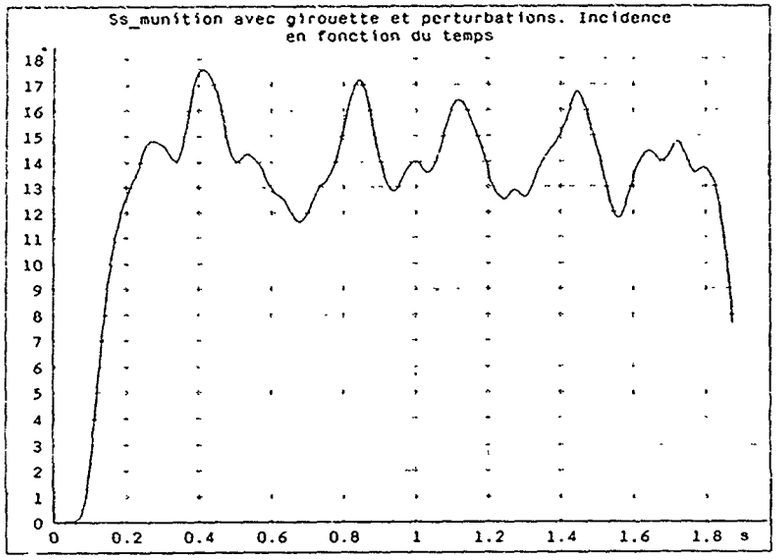


Figure 20



COMPUTER SIMULATIONS FOR THE DEVELOPMENT OF MISSILE NAVIGATION SYSTEMS

by

D.Berton, D.Duhamel and **G.Cuvelier**
SAGEM
Missile and Space Guidance Systems Unit
Eragny R & D Center
F-95612 Cergy-Pontoise Cedex
France

ABSTRACT

This paper gives an overview of general aspects of conception and utilization of simulators for the development of missile Navigation Systems. First, specific needs for simulators arising during the various stages of a project (Concept Definition, Specifications, Algorithms Development, Software Validation, Mission Requirements Analysis) are detailed.

Then, the simulators currently operational in SAGEM, and used to assist engineers in the development of modern missile Guidance Systems, are described.

The philosophy of utilization, as well as the different methods and tools available to exploit the vast amount of data delivered by simulators are detailed. Particular emphasis is given to the obtention of true and accurate models of the systems being simulated.

Finally, an example of the use of a simulator is given. It deals with the development of a new algorithm for automatic alignment of an airborne missile Navigation System, before the launching of the missile.

1 - INTRODUCTION

Modern Inertial Guidance Systems (IGS) for missiles, as other key guidance and control equipments, are becoming increasingly complex. Their accuracy has improved as well as their reliability. Their design must often lend itself to high yield and low cost manufacturing and they integrate many different technologies. The combination of high precision mechanics, optics, optronics, accurate analog electronics, fast digital circuitry and processors, together with embedded software implementing state of the art algorithms in signal processing, automatic control and navigation is common nowadays for almost any type of missile application.

The development and use of IGS simulators are major factors allowing the reduction of the cost of development and maintenance of IGS and associated test equipments and the reduction of development time.

Moreover, as experience has proved, the adequate use of computer simulations is more profitable than just a design aid.

It is an excellent mean to foresee needs of future equipments and components, and to ensure that initial design specifications are suited to the intended missile mission. It also allows a much higher optimization of the IGS with respect to constraints due to the weapon system. Because the simulator takes, as an input, mission trajectories and environmental conditions, the design of the IGS will fulfill the mission's goals as closely as possible.

Simulators allow the analysis of a complex mission, during which several IGS and different modes of utilization are encountered. A global simulation allows one to determine accurately the effect of each individual parameter on the mission's effectiveness and consequently to obtain optimal design, as compared to methods in which each individual equipment is specified separetely and complex interac ons are not taken into account.

The use of simulators also authorizes global and fast explorations of design tracks. Many solutions are assessed within short deadlines. Worst case analysis are performed. Once again, the globality of the approach must be stressed.

Finally, the potential for further and more elaborate uses of simulators is high. The integration of modern CAD tools (e.g. in automatic control) and of symbolic computation softwares with more classical computer simulations opens new areas of development and promises additional performance and user friendliness in the future.

SAGEM's Missile Division undertook, 4 years ago, the development of a simulator for IGS. Capitalizing on many previous efforts in less general and powerful simulators, a global set of integrated computer programmes, allowing comprehensive utilizations, has been developped and has been fully operational for the last 2 years. This simulator was developped with the support of the "Délégation Générale de l'Armement - Direction des Engins - STEN - Bureau Guidage/Pilotage" under contract 86-70-200. The "Laboratoire de Recherches Balistiques et Aérodynamiques (LRBA)" acted as technical programme coordinator.

2 - NEEDS ANALYSIS

2.1 General

From initial concept exploration to retrofit and maintenance of an IGS, many different engineering and design activities are carried out. The aids a simulator offers vary from one stage of the IGS life to another. The different utilizations of a simulator can be split up into five categories, drawn from SAGEM's experience.

The first type of use of the simulator concerns the Conceptual Design phase, in which the architecture of the IGS is defined, after a wide range of different possibilities have been explored.

The second group allows to write thorough technical specifications of equipment(s) to be developped.

An other type of use is the development of algorithms which will allow optimal performance of the IGS, for a given set of sensors and associated electronics and a specific mission.

A fourth domain of application of the simulator is the validation of on board real time software, implementing the above mentioned algorithms.

Finally, most of the remaining uses of the simulator concern the evaluation of the behaviour of the IGS for different mission profiles that are necessary to achieve given objectives.

We will now analyze each of these applications in greater detail.

2.2 Concept Definition phase

The Concept Definition phase of the Inertial Guidance System of a missile consists in considering the different organizations of the IGS which potentially fulfill the user's needs and in selecting the most promising of these organizations, based on a combination of cost / performance / soundness of design / reliability / technical challenge / ... criteria.

In this phase the following are usually defined :

- The basic structure of the IGS (gimballed or strap-down Inertial Measurement Unit (IMU), basic algorithms, ...).

- The updating sensors (if needed)

- The technology and type of inertial components (gyroscopes or gyrometers and accelerometers).

- The topology of these sensors on the cluster.

- The shock isolator's transfer function.

- The thermal regulation.

.
.
.

Obviously, the definition of the IGS structure requires to apprehend its function within the globality of the Weapon System. Parameters like volume and weight have to be approximately defined at the begining of this phase. It is equally important to obtain typical mission profiles, mission constraints due to operational requirements and the totality of data influencing the missile guidance function (type of carrier and its navigation system, available external aids, possibilities for the system's initialization, ...).

Once again, the more global the need, the more technically viable and cost-effective the solutions to be considered and assessed, and the more innovative organizations of the navigation and guidance function are likely to arise.

2.3 Technical Specifications Realization

The writing of detailed technical specifications is a necessary task, which must be achieved before the actual design of the IGS begins. It obviously takes place after the concept definition phase.

The technical specifications define the IGS accuracy in term of equivalent gyro drift or accelerometer bias. They thus differ from the prime contractor need-specifications which express a functional need (knowledge of position, speed, attitude, ...) of the vehicle for a given mission. Once they are completed, the design engineer is able to allocate an error budget to each of the IGS sub-assemblies (inertial components, coding electronics, thermal regulation, embedded software, ...), to the calibration process, to the aging of the IGS components between recalibrations, ...

Writing thorough technical specifications requires a very good knowledge of the error models governing each component of the system, and is consequently a task reserved to the equipment manufacturer. Only the use of the right models and the knowledge of adequate values of the parameters of the error models allow to optimize the design of the IGS. The skill of the design engineer deriving these technical specifications is to use error models close enough to physical models, allowing to release the constraints on the IGS design as much as possible. Because the mathematical relationships mapping navigation performance to IGS parameters are complex and non-linear, the use of a simulator to specify the IGS is mandatory.

The derivation of preliminary technical specifications is equally of the highest importance, especially when it concerns the development of costly new inertial components for future applications. The simulator is then again an invaluable tool. It allows, for a given structure of the error model of the inertial component, to determine the relative weight of each of the error model parameters on a performance criterion (for example the missile final position error). The component designer is able to link error model parameters to physical characteristics of the component. He then has accurate performance goals and is able to realize trade-offs, decreasing some performance whilst increasing other more easily available.

2.4 Development of Algorithms

A missile navigation system consists basically of a set of inertial components and associated electronics (the Inertial Measurement Unit (IMU)), of additional sub-systems (GPS receiver, TERCOM*, ...) which allow the updating of the navigation filter and the calibration of the IMU, and of embedded computer(s) on which various algorithms are implemented.

They include :

- compensation algorithms, which correct the effects of sensor errors. The more accurate the error model, the more efficient is the error compensation,

- calibration algorithms, which estimate the parameters of the error model by comparing navigation data outputted by the navigation filter and those coming from the updating means. Typically, these algorithms are based around an Extended Kalman Filter and operate either in static conditions (recalibrations during periodic check-up of the IGS, zero-velocity updates whenever the missile is immobile before launch), or in dynamic conditions if sufficient external measurements are provided to the IGS,

- alignment algorithms, similar in principle to calibration algorithms, but whose aim is to estimate the 3 Euler angles φ_X, φ_Y, φ_Z, which define the orientation of the reference navigation frame with respect to the local geographic frame,

- inertial navigation and aided inertial navigation algorithms, which integrate the specific forces measured by the IMU added to the local gravity field.
Once again, a Kalman Filter authorizes the fusion of inertial data with external measurements.

Because most of these algorithms are complex, their behaviour cannot be entirely predicted with analytical methods. Very often the mathematical model used is a simplified version of a more realistic one because of computing time constraints. The robustness of these algorithms to variations in model parameters, noise level, ... must be determined. However, their performance must be assessed early enough and the possibility of divergence has to be considered. The trimming of their governing parameters must be realized, generally through a trial and error procedure.

The SAGEM IGS Simulator allows a thorough check-up of the algorithms in various operational conditions. Accurate IMU models generate realistic data, feeding the algorithms under test. These algorithms are themselves part of a global structure (other algorithms, data processing and interpretation modules, statistical analysis, ...). The trajectory, environmental conditions, type of IMU, frequency at which the algorithms will be running, ... are set by the operator.

The simulator is generally used early in the algorithm development process, once the equations have been derived and preliminary analysis undertaken.

* TERrain COntour Matching

2.5 Embedded software test and validation

Once an algorithm has been derived, tuned, and its performance has been checked, the next step is generally to implement this algorithm into the IGS embedded computer. Risks of software errors, either in the core of the programme - the mathematical equations -, or in the sequencing of tasks by the real-time Operating System, or within data exchange modules or other parts of the programme, must be minimized.

Very often the software functions operate in conditions not encountered on the ground, and to which the IMU cannot be submitted, like a combination of rotation rate and linear acceleration. Sometimes, the function of the algorithm is to estimate an internal parameter of the IGS, unknown to the operator. The risk of residual software errors would thus not be negligible without the use of a specific test tool.

A version of the IGS Simulator, RTIS (Real Time IGS Simulator) has been developed to allow the test, the tuning-up and the validation of embedded softwares. It delivers data similar to those of a true IMU, in an user-defined environment.

RTIS permits one to tune up and test on-board software without the actual IMU. This is of great interest because software development represents an increasing part of the global development work for modern IGS. RTIS allows embedded software to be tested in parallel with its realization, without having to wait for the availability of the IMU, thus diminishing costs and improving quality. While not eliminating the need for a final check-up of the software when the IMU is available, RTIS permits one to test the software in many different controlled IMU configurations with a high degree of flexibility. The knowledge of the simulated IMU allows a thorough check of the software, including for IMU characteristics at the margin of their specifications. This has proved an invaluable help, in particular to control calibration algorithms. RTIS simulates the various functional channels of the IMU and the sequence of operations, which is extremely useful for testing software without putting the sensitive parts of the IMU at risk.

2.6 Global analysis and mission profile requirements

Navigation and guidance are vital functions for all weapon systems. Most of the time, these functions are not realized by only one subsystem but by a set of them. Its configuration varies during the course of the mission, as does the mode of operation of each equipment.

A submarine launched missile will depend on the several submarine IGS, the speedometer, the stellar updates, ... for its initialization.

The accuracy of an air-launched missile depends on the initialization of the aircraft IGS(s), as well as the aircraft trajectory which affects the performance of its navigation, the autonomous alignment filter which initializes the missile IGS with the help of the aircraft IGS, the missile IGS itself, the missile updating means and, possibly, its terminal guidance equipment.

Whereas the specification of an isolated equipment, or the development of some algorithms do not always require one to take into account the globality of the mission, a full simulation is necessary at some stage. This one should take place a first time at the beginning of the weapon system development in order to bound the performance of the various sub-systems and to determine the criticity of their key parameters. It utilizes simplified models and trajectories. Towards the end of the design phase a more detailed global simulation is also needed. It allows one to obtain with accuracy the navigation performance of the whole weapon system, taking into account the complex interactions that might occur.

A typical example where global simulations are necessary stems from on-board alignments, in which the missile IGS utilizes the carrier IGS as an updating means. In this case, global simulations during the alignment itself are necessary to show the observability of the filter and to determine the trajectory profiles which permit its convergence. The outcomes of these simulations will be later translated into mission requirements to the aircraft pilot.

3 - GUIDANCE SYSTEM SIMULATOR DESCRIPTION

3.1 General

The IGS simulator consists of a set of software modules, written in FORTRAN and running on 3 HP 9000-835 S computers.

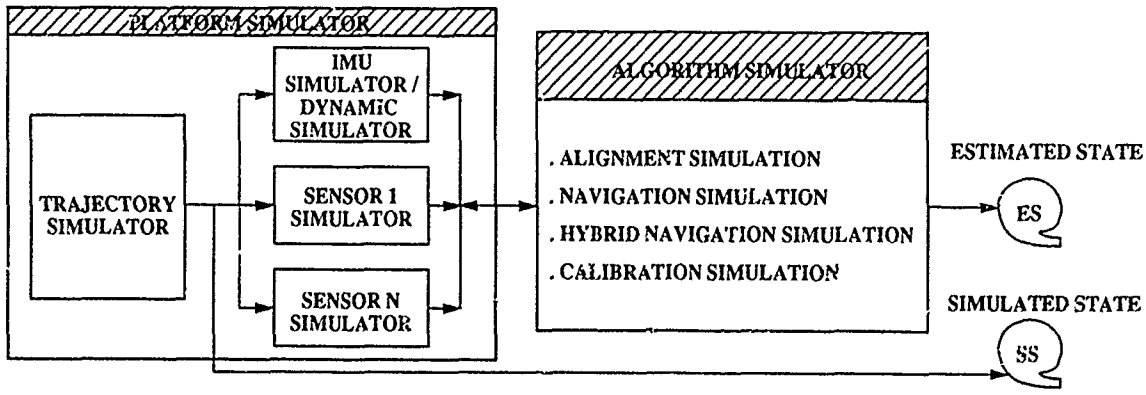
The overall size of the simulator is approximately 200 000 lines of source code, including comments. The run-time version typically takes up 4 M Bytes of RAM when loaded in central memory.

For each application, users build their own version of the simulator corresponding to their particular needs. A special purpose user programming language allows them to organize their own structure from an initial framework, picking specific modules from several libraries.

3.2 Functional description

The schematics hereunder represents the functional organization of the simulator. It is composed of two main sub-functions, the platform, or Inertial Measurement Unit (IMU) simulator and the algorithms simulator.

The platform simulator represents the physical phenomena that take place inside the Navigation System. Detailed functioning of inertial components and other sensors, thermal characteristics, dynamic and vibrations effects are taken into account.



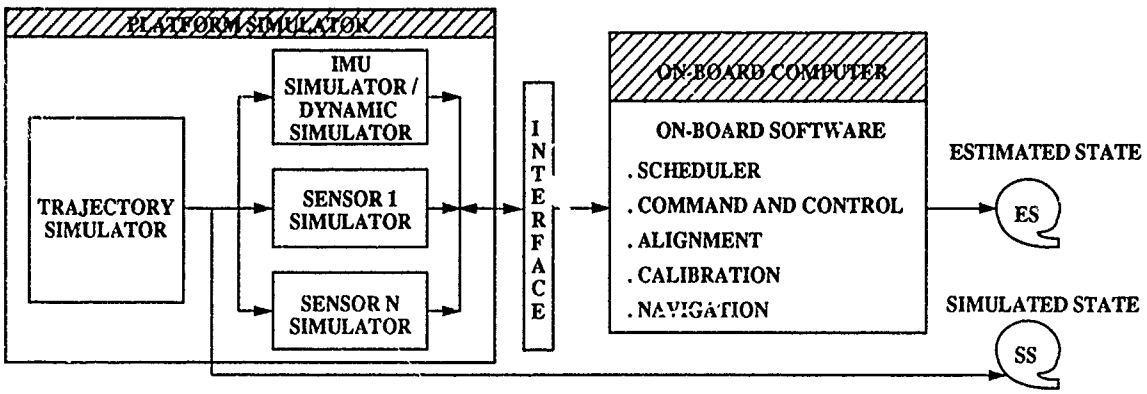
This "sensing" part is fed in by the output of the trajectory simulator, which elaborates stimuli that excite the various error models and differential equations of the platform simulator. The trajectory simulator gets its input from analytically generated trajectories or from data files.

The algorithm simulator represents the on-board real time software implemented by the IGS or the missile computers. These algorithms transform raw data generated by inertial components and other sensors, while submitted to given environmental conditions and for a specific trajectory, into useful information. They include the computation of missile position, velocity and attitude (navigation algorithms), the determination of the attitude with respect to the local geographic frame of the IMU cluster (alignment algorithms), the identification of the parameters of the IMU error models (calibration algorithms), ...

Typically, the behaviour of the IMU and associated algorithms is analyzed by comparing data obtained either directly from the trajectory generation programme, or from the platform and algorithm simulators.

Because of the requirements of many applications, the models used in the platform simulator are sophisticated and, consequently, the computing power needed is high. This is specially true for dynamic analysis, because the behaviour of inertial components and platform mechanics is simulated at a high frequency.

A modified version of the IGS Simulator, called "Real Time IMU Simulator" (RTIS) is specially dedicated to real-time embedded software test and validation, in a "software in the loops" configuration. In this case (see schematics below) a simplified version of the platform simulator is connected to a specific interface. This allows to output data identical, not only in type and accuracy, but also in temporal and electrical characteristics, to the data delivered by a real IMU.

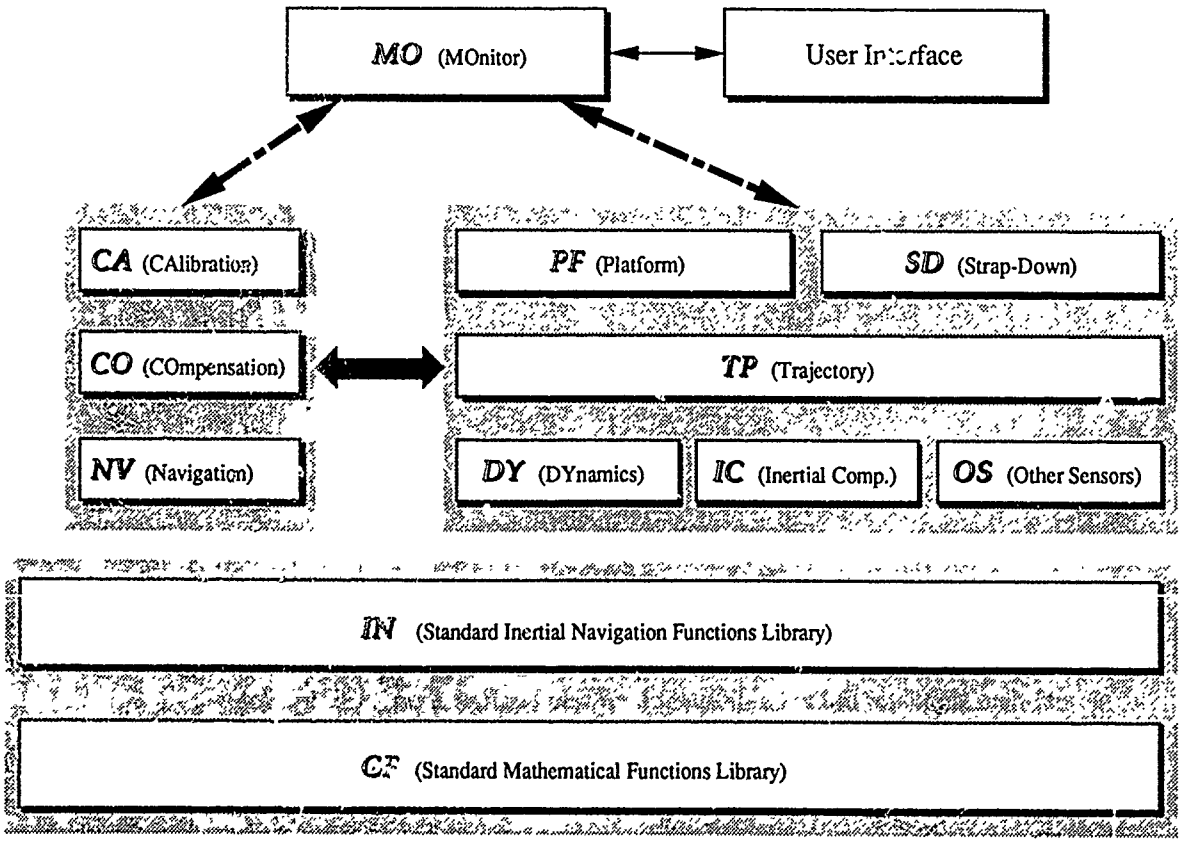


3.3 Software organization

The simulator consists of approximately 200 software modules integrated within a global structure, outlined in the schematics below.

The main parts of this software are :

- the platform simulator, which includes the simulator of gimballed and strap-down IMU, the trajectory generation, the representation of different inertial components (dry-tuned gyros, ring laser gyros, pendulous accelerometers, PIGA accelerometers, ...), the simulation of other sensors and sub-systems (GPS receiver, radio-altimeter, electro-optical devices, ...), and the simulation of dynamic phenomena within the IMU,



- the algorithm simulator, which includes the simulation of compensation algorithms, of calibration and of aided and autonomous alignment algorithms, and of navigation algorithms (pure inertial or aided inertial navigation),

- the "Inertial Navigation" (IN) library which contains modules frequently used in navigation algorithms, like gravity field models or transfer matrices between different geographic frames,

- the "General Fortran" (GF) library which contains reusable mathematical functions like matrices operations, statistical operations, Kalman filter implementation,...

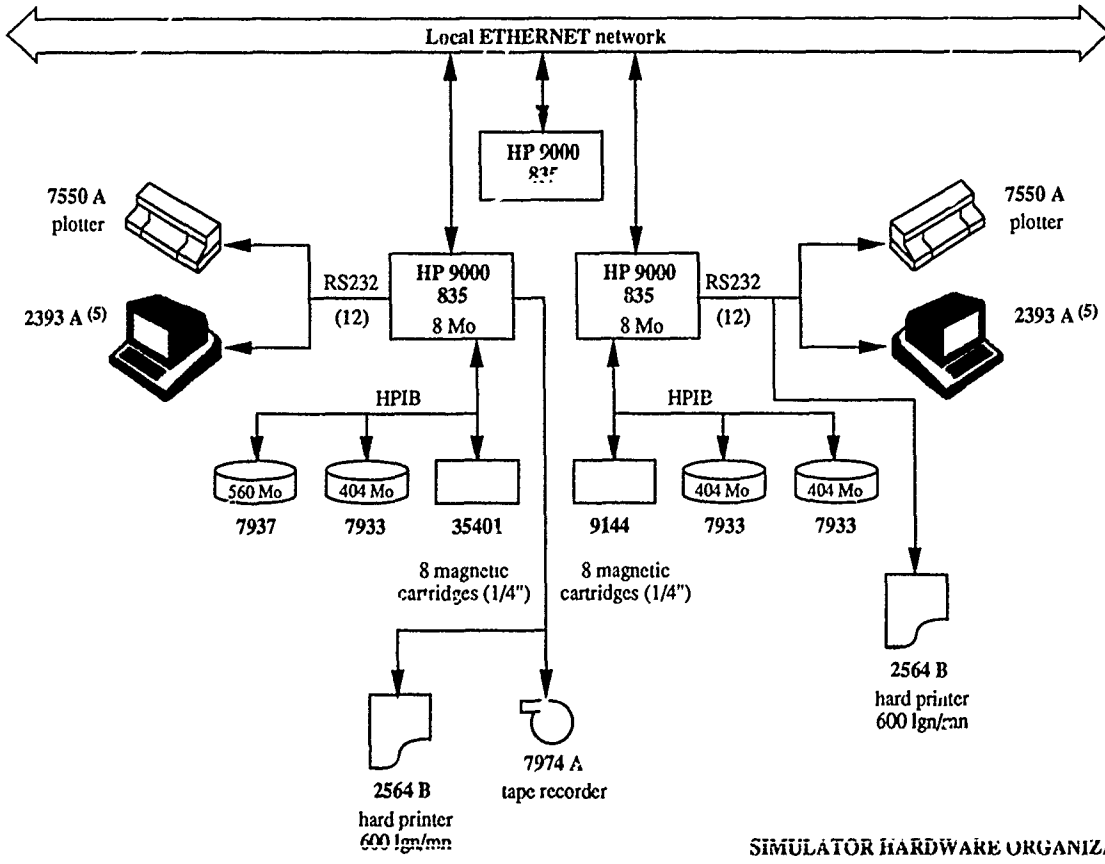
The operator uses those libraries to help him adapt the simulator structure to its needs.

- the Monitor module is a special purpose tool which helps to implement the desired configuration of the simulator,

- the user interface authorizes sophisticated data display and storage functions.

3.4 Hardware organization

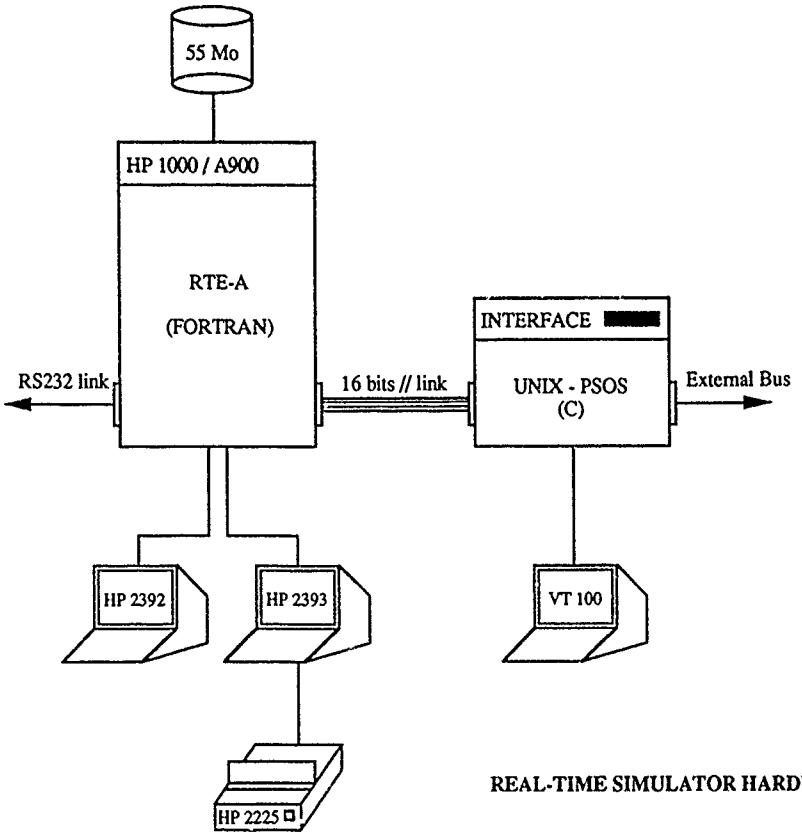
The 3 HP 9000-835 S on which runs the simulator are connected via a local area network. The Figure hereunder shows the structure of the network and the peripherals available.



SIMULATOR HARDWARE ORGANIZATION

The real-time simulator utilizes a different hardware configuration, sketched below.

The HP 1000 - A900 computer is well suited to real-time applications. The specific interface is a proprietary design based around a VME bus.



REAL-TIME SIMULATOR HARDWARE ORGANIZATION

4 - METHODOLOGY OF USE

4.1 General

A major aspect of the methodology of use of the IGS Simulator, the adequation between a given application and the associated simulator configuration, has been described in chapter 2.

We now give more details about the various levels of model accuracy available and show how such models are obtained. The different techniques of use of the simulator are then depicted.

4.2 Adaptation of models to specific uses

Wherever possible, the mathematical models used should be as simple as possible. This brings the following advantages :

- reduction of computing time,

- no need for elaborate knowledge of the physical system,

- easier interpretation.

For preliminary studies and concept explorations sophisticated models are generally not useful. On the other hand, the specification of a component or a sub-assembly and global performance evaluations require a fully realistic description of the physical processes.

Below are listed examples of the various degrees of complexity of the models implemented in the simulator.

Sensor simulation (inertial sensor as gyrometer, gyroscope or accelerometer, measurement hardware as GPS receiver)

- input output relationship ("black box" approach),
- input output relationship including dynamic modeling, such as electronic frequency response,
- full internal process description, manipulating the true physical characteristics of the sensors,
- incremental or continuous form of output data.

Platform simulation (gimballed units)

- kinematic modeling, assuming perfect caging loops,
- dynamic modeling taking into account gimbals inertia and friction torques,
- added thermal modeling of the cluster and inertial components

Environment description

- no vibrations,
- sinusoïdal vibrations (one or several pulsations),
- random noise with any power spectral density,
- constant or evolving external temperature.

Algorithms simulation

- Euler's integration, Runge-Kutta or Quaternion technique for navigation equation integration,
- for strap-down IMUs, modeling of high frequency phenomena like coning, sculling, Body Coriolis, ...,
- earth modeling WGS 72 or WGS 84,
- modeling of gravity field (constant module, Somigliana-Tchebichev model, Markov model for deflections, GEM10, GRIM 2).

4.3 Model determination and refinements

Basically, a simulator is a huge collection of mathematical formulae or models. However the definition of these models requires :

- that they fit to physical processes,
- correct numerical values. These can be obtained from :
  . CAD means for parameters as lengths, inertia, areas, ...,
  . theoretical analysis for parameters like electronic gain, time-constant, time-lag of a control-loop,
  . experiments on real equipments, for inertial components stability, thermal constants, ...

Knowledge of models comes from SAGEM's experience on inertial techniques acquired throughout the last 40 years.

It also profits of follow-up, done either by SAGEM or by its customers, on operationnal equipments. This is particularly important for realistic studies concerning aging, because theoretical results are poor.

Model refinement works guaranty the necessary representativity of models. They are integrated in the following framework :

--> collection of data,

--> model refinement,

--> control of the refined model accuracy by comparison between simulated and actual data.

4.4 Processing and interpretation of simulated outputs

The 3 widely used techniques recalled here are :

- partial derivatives method,

- Monte Carlo technique,

- single runs.

4.4.1 Use of partial derivatives

This powerful tool exhibits the influence on target miss distance E of one error cause pi such as initial misalignment or inertial component elementary error (bias, scale factor, ...).

The ratio $Si = \frac{E}{pi}$ with other error sources pj = 0 (for $j \neq i$) is called partial derivative of the error E with respect to the parameter pi.

Generally, E has 2 components in a local horizontal frame and we distinguish :

$$Sxi = Ex/pi$$
$$Syi = Ey/pi \qquad \text{with } E^2 = Ex^2 + Ey^2$$

This technique is used for most of the applications of the simulator, the knowledge of partial derivatives allows to compute the missile ECP.

Under appropriate assumptions, the relationship between ECP and partial derivatives is :

$$ECP = 0.563\ Max\ (\sqrt{\sigma_x}, \sqrt{\sigma_y}) + 0.614\ Min\ (\sqrt{\sigma_x}, \sqrt{\sigma_y})$$

where σ_x, σ_y are eigenvalues of $\sigma = M \Sigma M^t$

with $M = \begin{pmatrix} Sx_1, ..., Sx_n \\ Sy_1, ..., Sy_n \end{pmatrix}$, n is number of involved errors

$$S = \begin{pmatrix} E(p_1^2) & \dots & E(p_1\ p_n) \\ E(p_n\ p_1) & & E(p_n 2) \end{pmatrix}$$ where E (pi) is the expected value of pi

4.4.2 Monte-Carlo Technique

This technique is used to qualify the global performance of a system with input random variables. These variables characterize :

- dispersion of components (manufacturing, calibration),

- noise of components (random walk, random bias ...),

- measurement noise (random error associated with updating means like GPS or TERCOM),

- noise due to structural vibrations,

- initial values of navigation parameters.

Probability density functions (Pdf) associated with these errors are generally zero mean Gaussian or uniform. However, specific Pdf, resulting for example of screening and trimming of inertial components in manufacturing, can be taken into account.

The desired global performance, for instance :

- ECP (it is generally obtained more accurately than with the previous method because correlations between individual error terms are simulated),

- Navigation (aided / unaided) performance,

- Alignment, Gyrocompassing performance,

is obtained with several runs. The behaviour of the sample variance s^2 with respect to the actual variance σ^2 for 80 % confidence, and 20 runs is given by :

$$0.83 \quad < \quad \frac{\sigma^2}{s^2} \quad < \quad 1.27 \, .$$

This allows sufficient precision for primary evaluation with acceptable computing time. If more precision is necessary, generally during the ultimate phase of validation of a system, the number of runs must be increased.

See [SHA 83] for further details.

4.4.3 Single runs

They are used to solve particular problems, mainly for algorithms development, such as :

- determination of the filtering time necessary to ensure low residual filtering error or sufficient observability,

- definition of manoeuvres for improved observability,

- analysis of the influence of noise (for instance gyrolaser random walk effect on calibration or alignment accuracy),

- analysis of thermal aspects during start-ups.

5 - APPLICATIONS

5.1 General

In this section, we discuss briefly two examples of in-flight alignment which illustrate the benefits of simulation for inertial navigation systems development.

The first example - case A - deals with an extensively studied alignment algorithm ([BAR 80], [POR 81]). This algorithm is an optimal recopy which uses aircraft master IMU velocities as updates. The second example - case B - presents a global approach whose aim is to fuse data provided by the aircraft and missile navigation systems. We explain the structure and give comments on this filter and show when and how the simulations helped in its development.

5.2 Classical technique for in-flight alignment (case A)

Because of the non-linearity of the navigation equations, an Extended Kalman Filter (EKF) in which the equations are linearized around the current estimate is required. The carrier aircraft velocity, which has a good enough resolution, is taken as an observation. The missile IMU parameters of interest are part of the state vector. The vibrations between the two IMU's, and the instrument quantization are considered as measurement noise and the model errors are considered as state noise. A functional block diagram of the system is given in Figure 1.

For good alignment performance, it was shown that alignment must begin 400 seconds before launch and that a manoeuvre must occur during this period (see Figures 2 and 3).

- Theoretical considerations

Missile and aircraft state models are derived from the fundamental equation of dynamics :

For the aircraft : $\dot{X}_A = F_A(X_A) + Q_A$ (1)

where X_A is the aircraft state vector.

Q_A is a zero mean, Gaussian white noise process.

For the missile, equation is the same with subscript M :

$$\dot{X} = F_M(X_M) + Q_M \quad (2)$$

Aircraft's updates are provided by GPS or TERCOM.

This Extended Kalman Filter gives the best estimates $\widehat{X_A}$ of the aircraft state taking intc account external measurements.

Missile updates Z_M are derived from the aircraft estimate. The effects of lever arm between the two IMU's are modeled.

$$M\widehat{X_A} = Z_M = H_M X_M + L + W_M$$

where M selects appropriate updates,

H_M is the observation matrix,

L is lever arm compensation,

W_M is the measurement zero mean, white Gaussian noise process.

5.3 Global filtering as a new concept

Performance improvement is achieved by processing both aircraft and missile measurements simultaneously.

From case A, we derive the following ideas leading to global filtering :

- it is necessary to model the whole system composed of :

  . aircraft and missile IMU's,

  . coupling arm between IMU's,

  . external updates as GPS or TERCOM,

- extended Kalman filtering is a suitable formalism,

- the global filter is implemented in the most powerful computer,

- for performance enhancement, global filtering must operate as long as possible. Consequently we distinguish three phases :

  . from take-off (including gyrocompassing) to missile launch, this filter ensures optimal navigation for the whole system thanks to external measurements,

  . just before launch, only aircraft states stay active in the aircraft computer while the missile navigation algorithm is initialized with the last values of the missile state given by the global filter,

  . after launch, there is no difference between this solution and case A.

- for similar conditions, the global filter method must be at least as good as the classical technique (case A).

- **Theoretical considerations**

They are issued from the previous discussion. A functional block diagram of the system is given Figure 4, which must be compared with Figure 1.

The global filter has a larger state X_C with :

$$X_C = (X_A, X_M)^t ,$$

with state equation derived from (1) and (2).

Measurements Z_C splits into two parts :

- external measurements GPS, TERCOM,

- internal measurements which indicate physical coupling between IMU's before launch. so that :

$$Z_C = H_C X_C + W_C$$

where $$H_C = \begin{pmatrix} H_R & 0 \\ C & -C \end{pmatrix}$$

$H_R = 0$ when external measurement are not available.

C indicates the coupling.

- **Qualitative analysis**

When $H_R = 0$, the two IMUs play identical roles. In fact, at a given time, the filter automatically selects the best IMU based on the estimation error covariance matrix.

Because the filter possesses maximum information about the global navigation system (aircraft, missile), the estimates it delivers are better than those of the best individual filter (see figure 5).

When $HR \neq 0$, external measurements are provided both to the aircraft and to the missile. However, the noisy lever arm gives rise to different levels of estimation accuracy (see figures 6 and 7).

5.4 Results and conclusions

This concept reveals an increased alignment performance of 10 to 40 % with respect to classical techniques (see figure 8).

Nevertheless, preliminary studies conducted with other tools as BASILE (*) and MAPLE (**) permitted fast design on simpler linear steady-state IMU models.

Once the EKF was defined using covariance matrix analysis, simulations allowed comprehensive evaluation on the global filter structure, its tuning, the potential interest of aircraft manoeuvres, and finally performance assessment.

6 - CONCLUSION

Many years of experience have proved the interest of using simulators during the various stages of the development of navigations systems.

Improvements in computer hardware and software, as well as a better knowledge of the models of the systems studied allow to get more results of better quality from simulators. The increased complexity of the navigation function of global weapon systems, the use of even more sophisticated algorithms and their implementation with real-time embedded software require this additional potential.

However, the benefits that simulators deliver are not freely given. The development of simulators is expensive, and requires a lot of work in system modeling and identification. Good software design, employing a structured approach and maximizing modularity, gives a high degree of versatility and consequently allows one to reduce the complexity and the cost of evolution and adaptation of simulators.

Examples of this versatility have included, in our case :

- the development of a satellite trajectory simulator for autonomous satellite navigation studies [BER 89]. It profited of complete gravity field modeling and of the mathematical library of the IGS simulator.

- the development of a tactical missile gimballed seeker simulator, for studies concerning future Infra Red seekers for high velocity missiles.

Many more applications are currently under way. The analysis of new concepts for missile guidance which rely heavily on electro-optical means, image processing and multisensory data fusion is one of them.

(*) BASILE : CAD on automatic control - Developped at INRIA (France)

(**) MAPLE : Software for symbolic computation developped at the University of Waterloo (Canada)

ACKNOWLEDGMENTS :

The authors wish to thank P. Lodola for his contribution to the paper. They also acknowledge I. Lanoe, J.M. Chevignon, R. Goulette and P. Seligman for their precious help.

REFERENCES

[SHA 83] : Shakarian A.

Application of Monte-Carlo Techniques to the 757/767 - Autoland dispersion analysis by simulation.

AIAA 83.

[BAR 80] : Bar-Itzhack, I.Y. and Porat, B., "Azimuth Observability Enhancement During Inertial Navigation System In-Flight Alignment", Journal of Guidance and Control, Vol. 3, July-Aug. 1980, pp. 337-344.

[POR 80] : Porat, P. and Bar-Itzhack, I.Y., "Effect of Acceleration Switching During In-Flight Alignment", Journal of Guidance and Control, Vol. 4, July-Aug. 1981, pp. 385-389.

[BER 89] : D. Berton, T. Codron, R. Horak, S. Salle

Expert System For Kalman Filter Supervision - Application to autonomous satellite navigation

AGARD GC Panel 48th Symposium - Lisbon 1989.

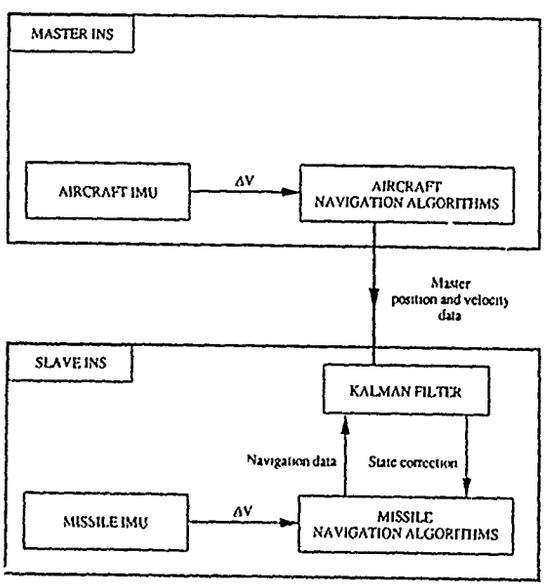


FIGURE 1

Block diagram for alignment operation (Case A)

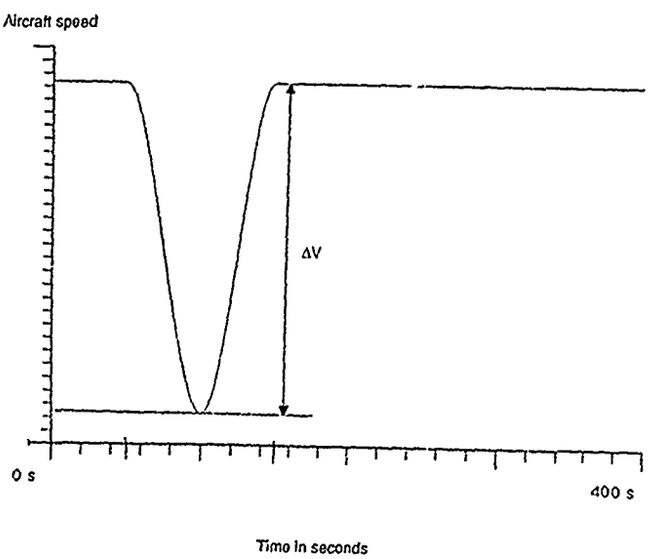


FIGURE 2

Aircraft manoeuver

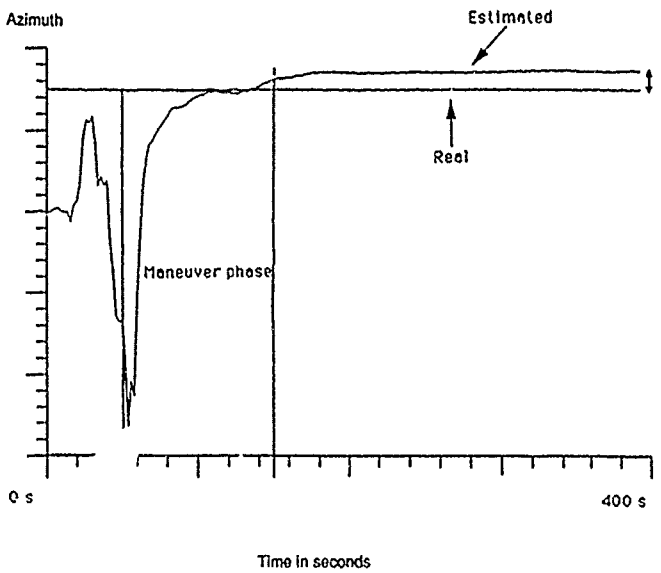


FIGURE 3

Real and estimated azimuth

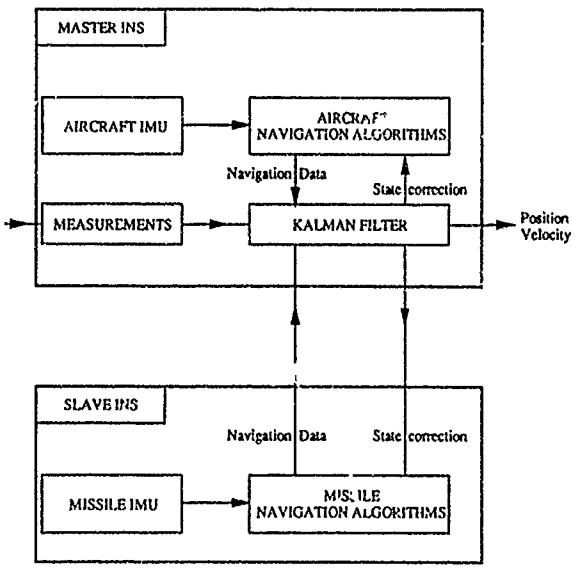
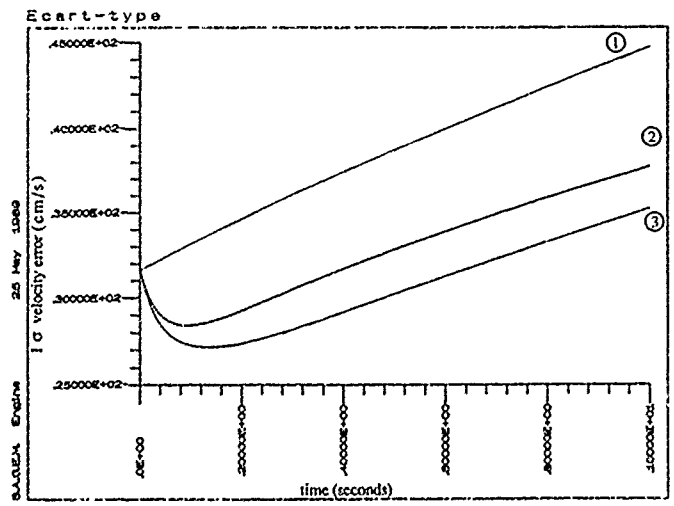


FIGURE 4

Block diagram for alignment operation with global filter (case B)



① : one INS (aircraft) alone

② : global filtering : missile velocity error (NB : missile process is noiser than aircraft process)

③ : global filtering : aircraft velocity error.

FIGURE 5

Covariance of velocity error for unaided IMU's (simplified trajectory - linear steady state models)

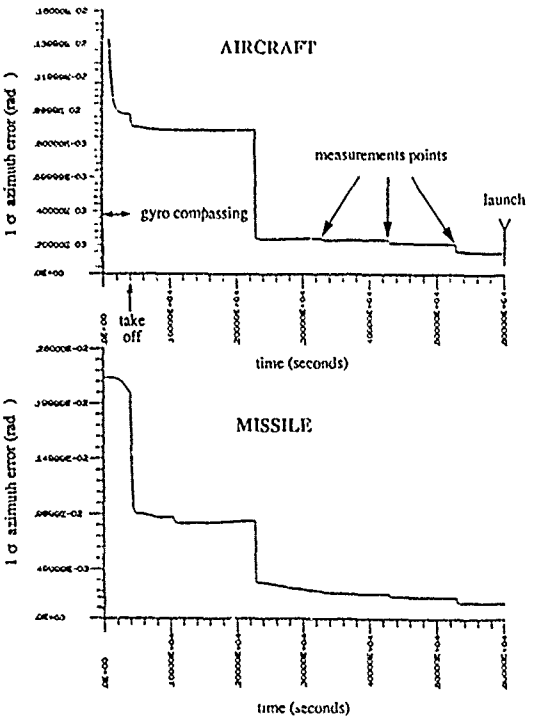


FIGURE 6

FIGURE 7

Influence of external updates on missile and aircraft azimuth errors covariance

$$\text{ratio} \quad \frac{\text{missile azimuth error (1 } \sigma\text{) (global filter)}}{\text{missile azimuth error (1 } \sigma\text{) (classical filter)}}$$

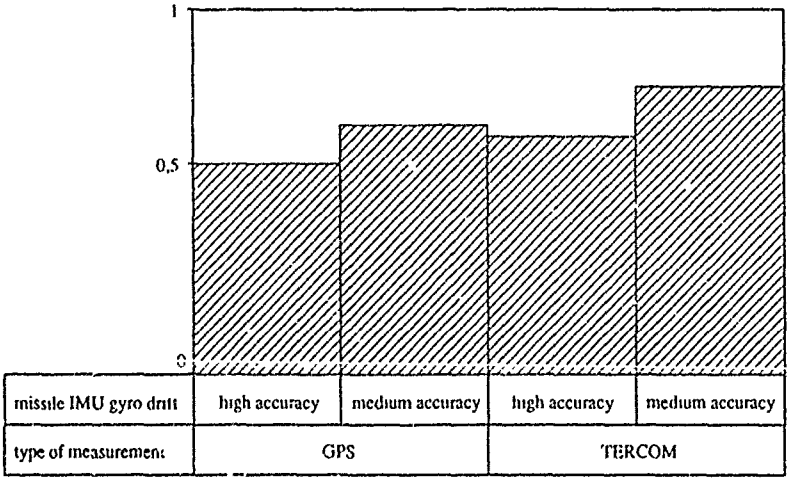


FIGURE 8

Results

GUIDED WEAPON SIMULATION
THE 'SBGL' DEVELOPMENT EXPERIENCE

José Luis Quesada Rodríguez
I N T A
Crta. de Ajalvir, km 5, 28850-Torrejón de Ardoz-MADRID (Spain)

Ricardo Mínguez and Pedro Segurola
S E N E R
Raimundo Fernández Villaverde, 65, 28003-MADRID (Spain)

ABSTRACT

This paper discusses the works in the simulation area carried out in the development of a laser guided weapon system (named SBGL) for the Spanish Ministry of Defence.

Simulation has been used at two levels:

- Computer integral simulation, with all subsystems algorithmically simulated. A remarkable point is the use of the same high-level software which was afterwards embedded in the SBGL real time processor. This allowed a very early testing of both design and codification of the software, independently of the hardware development, and a considerable reduction of the hardware- software integration problems.

- Simulation in testing facility, for testing and validating the different subsystems and the whole system.

The simulation facilities and the software tools used for their adjustment and validation are described, as well as the data reception and analysis capabilities.

Finally, the flight data and simulation results are compared. As a conclusion it is stated that the model developed is accurate, and that the complete simulation facility is a powerful tool for the design and validation of guided weapons.

1.- INTRODUCTION

Simulation as a project tool began to be used from the early stages of the project of a laser guided weapon developed by SENER Ingeniería y Sistemas, under contract with the Spanish Ministry of Defence.

The simulation tests and other validation tests were carried out with the cooperation of INTA (Instituto Nacional de Técnica Aeroespacial), wich also provided its facilities and testing equipment.

At the beginning, the emphasis was placed in the aerodynamic aspects, simulating the remainder of the system with theoretical and simplified models. As the project went ahead and the different subsystems were defined, its modelling became more detailed taking into account the real characteristics and behaviour of each element.

In many cases, simulation helped to specify the characteristics of elements and subsystems, to test the validity of the assumed solutions, or to choose among the different possible alternatives. Nevertheless, simulation has been most intensively used in the design and validation of the guidance and control algorithms, as well as in the development of the SBGL software on board.

Part of the simulation software was later on used at the testing facility for the validation and calibration of the real flight hardware, and for the hardware in the loop testing.

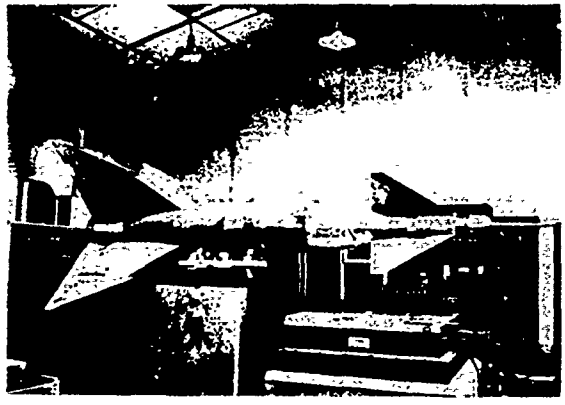
From an operational point of view, the following characteristics of the simulation performed in this project can be pointed out:

- Both versions, computer simulation and testing facility simulation, are identical. Each case has its own peculiarities according to the collateral functions to be performed and the execution environment, but the simulation itself remains the same.
- There is only one software on board. It means that in simulation, as well as in the weapon microprocessor, the code is exactly the same.
- The auxiliary programmes for analysing simulation, tests or real flight data, are also common.

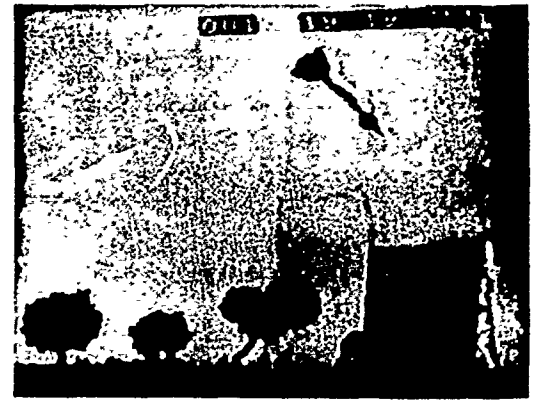
This homogeneity in data processing has simplified the study of further simulation stages and, above all, it has helped to obtain the necessary feedback from the flight and testing facility tests towards computer simulation, in order to achieve a maximum reliability.

The evaluation and validation of the whole SBGL system was made by the Monte Carlo method. Numerous simulations in different flight conditions, randomly generated, were performed. Likewise, the final validation of each prototype to be tested was carried out submitting it to several simulated flights in the probable environment of the nominal flight data.

As a summary of the level reached by the simulation it can be pointed out that, for several flights and launchings, the behaviour and the impact error predicted through simulation were accurately reproduced, with deviations less than ten per cent. A photography of the SBGL and a snapshot showing the impact moment are given in Figure 1.



SBGL prototype at the vibration test



SBGL prototype reaching the target in a real launching

Figure 1

The developed computer programme was given the name SIMTRA (SIMulation of TRAjectory), and initially was loaded on a CYBER-172 computer. Nevertheless, when the idea of using simulation as a development system for the software on board arose, and taking into account the characteristics of the weapon microprocessor, the whole simulation was implemented in a PC.

2.- INTEGRAL COMPUTER SIMULATION

Figure 2 shows the block diagram of the SIMTRA simulation programme. The modelling blocks of elements and subsystems of the SBGL (at the left side of the figure) have been separated from those representing the software on board for guidance and control (at right).

The modules that make up the simulation are described below, as well as the modelling performed in each case. It has been intended to reproduce the real operation of each element with a reasonable reliability, according to the needs of the project. In some cases, it has been necessary to develop detailed mathematical models of subsystems clearly nonlinear.

2.1- Aerodynamic model

The first approximation to the aerodynamic behaviour of the cruciform and canard configuration chosen for this project was made on a theoretical basis. Later on, numerical results and other qualitative aspects deduced from the various tests performed in the wind tunnel and in real flight were incorporated into the model.

Since several dynamic effects related to the rolling motion could not be well identified and quantified in the wind tunnel tests, this part of the simulation is the one that has undergone more modifications and improvements as a consequence of the real flight tests.

The following characteristics of the developed aerodynamic model can be pointed out:

- Breakdown of the whole configuration into separated geometrical elements: tail wings, fore wings and body.
- Analytic formulation of the lift and drag curves and aerodynam'. centre position of each element, as a

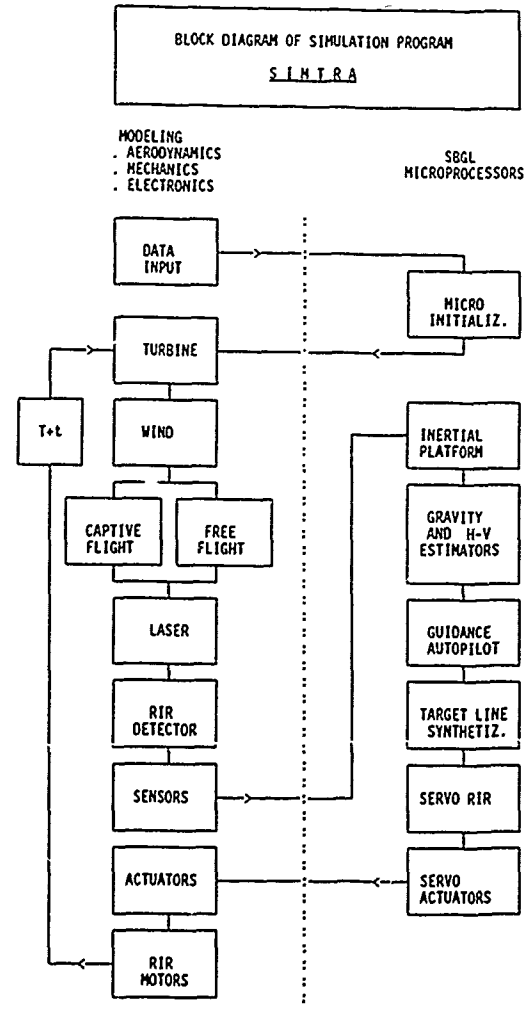


Figure 2

function of its geometry, aerodynamical parameters and specific flight conditions.
- The resulting forces and moments in this superposition have allowed an easy representation of the dynamic effect arisen from the interaction between the fore and tail wings, which usually occurs in this type of configurations.
- This analytical and parameterised approach has helped in the study of different alternatives and in the optimisation of the aerodynamic design.
- On the other hand, from the point of view of computer efficiency, it is also very useful to consider the aerodynamic characteristics in an analytical way.

2.2.- Flight dynamics and cinematics

For the calculation of the free flight path, by integration of the differencial equations of the SBGL motion, Hamilton's quaternions have been used.

The integration time step has been chosen in order to ensure the necessary accuracy, and also to facilitate the execution of the software on board at the same frequency as in the weapon microprocessor.

2.3.- Captive flight path

Since several software on board functions are performed during the captive flight (i.e. inertial platform and its initialisation and updating), it has been also necessary to simulate the flight of the SBGL carrier aircraft.

The captive flight path is defined through a series of points among which an interpolation with second derivative continuity is done. Course angle, height and speed are given for each point. Aerodynamic characteristics of the carrier aircraft are also given in a very simplified way.

Attitude angles of the aircraft are calculated so that all lateral forces are cancelled out, and the angle of attack produces the required normal acceleration for the defined flight path.

2.4- Wind and turbulence

The average wind is calculated according to a wind profile either defined by the user, or obtained from the MIL-STD-210B standard. The turbulence is modelled according to the MIL- 8785C standard.

2.5- Turbine

A turbine produces the mechanical energy necessary for the actuators and for the electric generator powering all the electronic systems.

Rate and torque of this turbine are calculated according to the flight conditions, height and speed, and the corresponding load (energy necessary for the electronics and for the actuation of control surfaces).

The characteristic curves of the turbine have been obtained from wind tunnel tests, and they have been verified in several tests in both captive and free flight.

2.6.- Target designation and laser radiation

In this part, the process of target designation is modelled. The relative positions of designator, target and weapon are taken into account, as well as the transmittance properties of the atmosphere and the reflectance of the target itself. In this way the radiant energy reaching the RIR detector is calculated.

2.7.- RIR detector

This subsystem basically consists of a quadrant detector with sophisticated amplification electronics, placed on a double gimbal that allows the required orientation of the line of sight in any weapon roll position.

The motion of the detector is produced by two motors placed on both gimbal handles, controlled by the software on board. During the target searching phase, the detector performs an azimuthal scanning of the earth's surface. Once the target is acquired, the detector tries to minimize the pointing error.

In the modelling of the detector itself, the laser energy arriving to each one of its four quadrants is calculated according to the SBGL location and to the detector orientation.

The response of the RIR detector, including the optic system and the corresponding amplifier, has been modelled and verified through numerous static and dynamic tests.

2.8.- Sensors and interfaces with the weapon microprocessors

All sensors included in the SBGL, like gyroscopes, accelerometers, total pressure probe, turbine tachometer, RIR resolvers, quadrant detector, temperature sensors, etc., have been modelled so that their analogic response takes into account:
- a bias or zero error
- a scale factor error
- a first order delay
- a random noise

In order to achieve the highest similarity between simulation and the real process, the way in which the output of each sensor is read by the weapon microprocessor has also been taken into account. The simulation model supplies to the software on board the measured values, putting the binary value produced by the converters at the corresponding port of the weapon microprocessor.

The quadrant detector modelling is more difficult, because the simulation must reproduce the interrupts generated by the reception of the laser pulses arriving asynchronously with the simulation process.

2.9.- Actuators

The mechanism of each actuator is basically composed of a reduction chain and a couple of clutches which transmit the turn of the turbine to the control surface. The deflection is controlled by means of the activation time of the clutch.

In the modelling of the four actuators the dynamics of the whole mechanism has been accurately described (turbine, reducer, screw gear, clutch, reducer and control wing).

Each control surface moves according to the pulses sent by the microprocessor, which determine the actuation time of the corresponding clutches.

2.10.- Seeker servo motors

The motors that move the quadrant detector have been modelled according to their mechanical and electric characteristics. The control of each motor is performed according to the feeding voltage sent by the microprocessor.

2.11.- Simulation results

A flexible system to obtain the appropriate results on each case has been included into the simulation programme, so that numeric or graphic results can be produced. Some graphic outputs are shown in Figure 3, with the pitch and yaw angles and the target line angles obtained from a real launching, and those produced at the corresponding simulation.

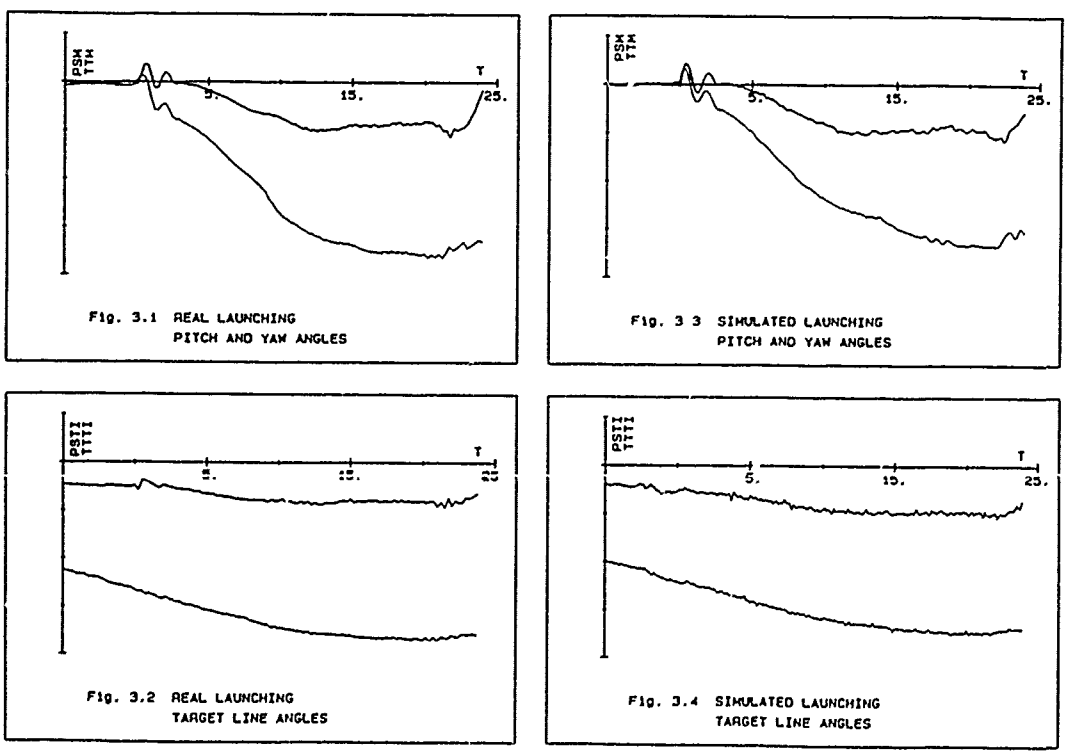


Fig. 3.1 REAL LAUNCHING PITCH AND YAW ANGLES

Fig. 3 3 SIMULATED LAUNCHING PITCH AND YAW ANGLES

Fig. 3.2 REAL LAUNCHING TARGET LINE ANGLES

Fig. 3.4 SIMULATED LAUNCHING TARGET LINE ANGLES

Figure 3

3.- HARDWARE CONFIGURATION AND SOFTWARE ON BOARD FUNCTIONS

The characteristics of the hardware embedded in the SBGL and the software on board functions are overviewed in this section.

3.1.- Hardware configuration

The SBGL process hardware is made up of two systems closely connected through a shared memory that allows to synchronise both processes and to exchange information.

Each one of the systems is formed by:
- Microprocessor and numeric coprocessor
- EPROM memory for resident code
- RAM memory for process data
- Shared RAM memory
- EEPROM memory for process and telemetry parameters

Furthermore, each system has its own set of hardware elements for driving the sensors and the actuators: timers, A/D, D/A and V/F converters, as well as parallel I/Os.

One of the microprocessors is named 'Micro Pilot' and the other 'Micro Optic', the former acting as master.

3.2.- Micro tasks

The micro optic performs the tasks related with the seeker motion control as well as the laser reception, and sends the line of sight parameters to the micro pilot, which is in charge of guidance and control operations for the SBGL flight.

Moreover the micro pilot performs the input/output operations: maintenance control, RS232 communication for the EEPROM updating and telemetry.

3.3.- Synchronisation between micro pilot and micro optic

The whole process of both micros is performed in a continuous way, developing the corresponding calculations in cycles of constant duration.

The synchronisation is performed through an interrupt signal sent by the micro pilot and the functional cooperation is achieved through the shared memory.

3.4.- EEPROM data

The EEPROM memory is addressed to store two types of information:
- Parameters that define and control the whole process on board, for instance cycle duration, time constant of the digital filters, autopilot gains, reception margins of the laser pulse, and so on.
- Parameters defining the information to be transmitted through telemetry, according to the requirements of each test.

The simplicity and flexibility in the definition of these parameters was of great importance to facilitate the design and preparation of the tests, as well as the study and analysis of the results.

3.5.- Telemetry

The whole telemetric process is carried out by the micro pilot:
- Generating the telemetric frames according to the transmission strategy indicated by the EEPROM data.
- Moving the telemetric data to the transmission hardware.

4.- SOFTWARE ON BOARD DEVELOPMENT

The final assumed methodology for the software on board development was intrinsically related to the SIMTRA simulation programme, allowing the global solution of a whole set of problems, ranging from the guidance and control algorithms development, up to the functional design, codification and validation of the software.

In order to avoid the problems and limitations typically inherent to the embedded software coded in low level language such as:
- small portability and, therefore, difficulties in the access to other computers or development systems for checking the implemented algorithms,
- difficulties to incorporate changes in a quick and reliable way,
- progressive decay of the original design as a consequence of the changes,

- problems in the testing facility simulation to identify the cause of results not foreseen in the computer simulation,

and taking into account the hardware similarity between the popular PC and the SBGL microprocessors, as well as the availability of high level languages (FORTRAN and C), a specific methodology based on the SIMTRA simulation was developed.

So on the role played by the simulation was reinforced, not only as an instrument to analyse the SBGL dynamic behaviour, but also as a tool for the embedded software development.

The software on board was integrated within the simulation, using high level language for the SBGL algorithms and assembler language for sensors and actuators drivers, for which the simulation should provide the equivalent stimulus and receive the commands for the actuators.

The final designing and codifying tasks were performed after analysing the code performance and measuring the execution time of the algorithms. To install the PC code in the SBGL, an almost automatic procedure in the development station was established.

The result of this integration was very positive. In particular this method allowed not only checking the software on board even at exceptional conditions, but also to consider the simulation environment as an 'ad hoc' software development system.

5.- TESTING FACILITIES OVERVIEW

Figure 4 shows the configuration used for the SBGL tests.

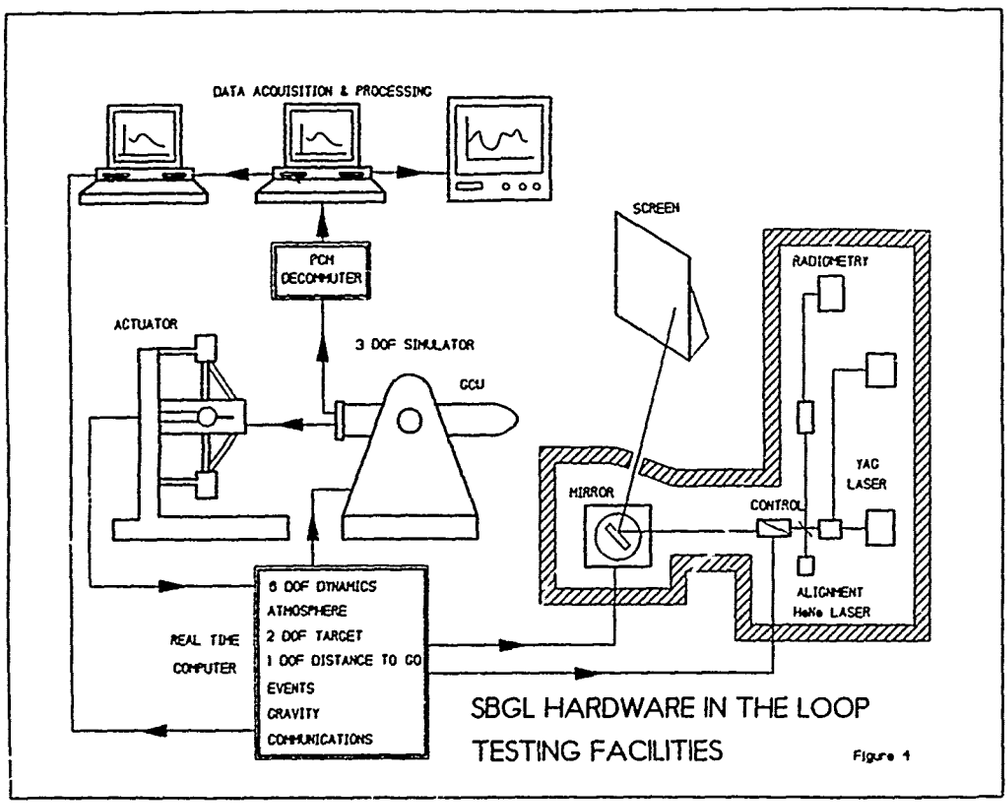


Figure 4

This hardware configuration is the result of an evolution along the project development. It has been used for isolated subsystems at different development levels (bread-boards, engineering and development models) and at different integration stages.

Concerning the software support, it is worthwhile saying that the increasing difficulties to solve the tuning, checking and control of devices involved in the testing loops, could only be overridden by developing a set of specialised software tools.

The subsystems that conform this installation are described below. Some very useful aspects throughout the tests programme development are pointed out.

- Real time computer

It is a GOULD computer, CONCEPT 32/9780 series, equipped with conventional D/A, A/D, and D/D converters, external interrupts and I/O communication ports for controlling the simulators and other peripheral equipments.

In this computer runs not only the mathematical simulation model of 6 DOF, controlling the closed loop tests, but also all the auxiliary software tools mentioned before.

- Rotational simulator

It is a 3 DOF CARCO simulator with continuous motion in the rolling axis, controlled by the real time computer.

- Target motion simulator

Two target motion simulators are available at the testing facility.

The first one is a 2 DOF CARCO composed by two rings coaxially mounted in the rotational simulator. The simulated target can be installed at a platform over the azimuth axis.

The second one, which was chosen for this project, is also a 2 DOF CARCO with a mirror projecting an IR beam over a screen placed in front of the rotational simulator.

- Laser target simulator

It is a YAG laser generator with a conditioning system which modifies the beam diameter and the laser output power, so that the apparent size of the spot and the power arriving into the seeker sensor simulate the real target.

As auxiliary elements, there are a red HeNe laser to line up the beam and a radiometric equipment for calibrating and monitoring the radiated power.

These equipments are shown in the shadowed area of the Figure 4.

- Aerodynamic torques simulator

There is a CARCO simulator capable of generating dynamically the required aerodynamic torques on the control surfaces.

Nevertheless, given the characteristics of the SBGL actuators, this simulator was replaced by passive simulators, which provide torques proportional to the control surfaces deflection.

- Data acquisition and process system

To improve the efficiency of the conventional method (on line paper strip recorder) for analysing the results of testing, the following acquisition chain was arranged:
. the embarked telemetry system
. a PCM ADS-100 decommuter
. a work station to store the telemetric data in real time
. a raster TV screen for real time monitoring
. a PC for graphical and numerical data analysis
. a communication line with the real time computer

The simulation data, dynamics and flight path, generated by the real time computer, are stored on a dedicated file. Once the test is finished, both data sets, the one coming from the prototype and the one coming from the simulation computer, are sent to the postprocessing and analysis equipment, where special data analysis and graphic edition programmes are available.

6.- FLIGHT TESTS EQUIPMENT

A summarised description of the equipment used in the flight tests is given below. The common use of some equipment and procedures in some tests performed both in flight and in testing facilities is pointed out. This has improved the reliability of flight tests and has helped in comparing the results obtained in flight and in testing facilities.

- Flight path tracking

Two trajectography systems have been used alternatively, according to the purpose of each test:
. an electrooptic system with automatic TV for videorecording the flight path data and the time basis
. radars with an analogue calculator attached

- Telemetric equipment

The same data acquisition and process system used in the simulation tests has been used here. The only difference is that the coaxial cable link between the telemetric system on board and the PCM decoder used at the testing facility have been replaced by a radio link between the transmitter on board and the ground receptor.

The advantages of using a single telemetry system are clear:
. The reliability of the system is increased, since it has been proved and debugged during the laboratory tests.
. The comparison between the results of ground and flight tests is made easier since both data sets are framed with the same format.
. The data processing and analysis time is considerably reduced.

7.- HARDWARE ADJUSTMENT TESTING

An important aspect for the success with the testing facilities simulation, both for the tuning up of the SBGL hardware and for the final hardware in the loop tests, is the correct calibration and tuning of the different elements in the installation. In this work, the possibilities of the testing facility computer itself were intensively used.

For this purpose, a series of SW tools were developed which allowed an early acknowledgment of those data in the installation which had been changed, and correcting them if necessary, or simply taking them into consideration for new tests.

The most significant example that can be pointed out is perhaps the one concerning the relative positioning between the target simulation screen and the seeker head placed in the 3DOF rotational simulator, which was realised with the own testing equipment and without any additional measuring instrument.

As a final result of all the adjustments and measurements, a data file with the parameters of the installation is edited for a later use in the tests.

8.- TESTING SOFTWARE

Thanks to the design made in the SIMTRA simulation programme, and to the new criteria emerging during the software on board integration, the implementation of the tests control software into the real time computer was also very simplified.

The modular design of the simulation programme allowed an almost direct implementation into the real time computer of all the algorithms related to aerodynamics, wind and turbulence, captive and free flight, target designation, laser radiation control and others, with the corresponding improvement in reliability and in time saving.

On the other hand, the design of the SIMTRA interfaces with the software on board makes it easier to programme the signals that the GOULD computer sends through D/A converters to the SBGL hardware in the closed loop tests (accelerations, pressures, etc.)

Moreover, the specific simulation signals for guiding all the elements in the installation, basically the rotational simulator and the target simulation system (laser emission, intensity regulation, and mirrors for positioning the spot), were added.

Additionally a series of software tools were prepared to direct and control the calibration and validation of SBGL hardware, previously to the final integration and to the hardware in the loop tests.

Data and results definition was kept identical to that used in the PC simulation, with which the data analysis task was homogenised.

8.1.- Hardware calibration and validation

The design made for the data loading into the EEPROM memory made it easier to design and prepare the tests for the validation and calibration of the subsystems because it allowed
- the quick definition of the parameters values to be tested, and
- the selection of the most appropriate results for each test.

The most relevant tests carried out were:
- Calibration of the inertial platform.
- Adjustment and calibration of the servo actuators.
- Characterisation of the seeker and quadrant detector.
- Adjustment and calibration of the seeker servos.

The results obtained in each case were taken into account for improving the simulation, and for defining the SBGL parameters to be resident in the EEPROM memory of each prototype.

8.2 - Validation of the whole SBGL system

The methodology followed to set up the software on board as well as to validate the hardware components and subsystems, has contributed to improve the prototypes reliability, and to reduce considerably the HW-SW integration problems, achieving a correct running in the hardware in the loop tests in a very short time.

As a consequence of the results obtained in these tests, of course, some analysis and adjustments of parameters did have to be made to improve the guidance and control, as well as the actuators servo or the seeker servo. But even in these cases, the modifications were verified in simulation, before being introduced into the SBGL microprocessor.

9.- DATA ANALYSIS

As it has already been indicated, the definition of the results to be obtained in each test was made through the EEPROM memory. For the reception of these results a system was designed which, directly connected to the telemetry output, stores the information in a data base and simultaneously displays, in real time, the alarm status and the graphical and numerical evolution of a limited number of variables.

By means of a postprocess, the information received from the weapon microprocessor is combined with that information produced by the real time computer, so having a complete view of the test performed. Graphics or printouts can be obtained immediately.

For more sophisticated analysis it is possible to obtain segments of information in files that directly feed to other numerical analysis programmes like MATLAB or MATRIX.

The same methodology is used in the real flight tests, with the only difference, physical but not logical, of the radio-telemetry transmitter and receiver.

10.- MONTE CARLO SIMULATIONS

Due to the complexity of the system, the diversity of working conditions, and the amount of random elements, the final validation of the system must be done in an statistical way, by the Monte Carlo method, with a high number of simulations with randomly generated data.

With this purpose a preprocessor was prepared which, starting from the probabilistic data distribution, generates the particular data for each case. The results of these simulations are stored in a file for a later statistical analysis.

This methodology was applied both for the computer simulation and for the testing facility simulation.

In the first case, and according to its corresponding degree of reliability, the random data generation affected practically all the data groups:

- SBGL characteristics
  - . mass data and aerodynamical parameters
  - . turbine
  - . quadrant detector and seeker servos
  - . servo actuators
  - . sensors (gyroscopes, accelerometers, potentiometers, and A/Ds)
- wind data, intensity, direction, and turbulence
- captive flight and ejection data
- target data, position, and reflectance characteristics

The most representative result of this study refers, logically, to the impact error. Figure 5 shows the probability distribution obtained in a Monte Carlo study with 350 samples.

When a specific prototype has to be validated for a determined mission the random generation must affect only to the external data, that is:

- wind data
- captive flight data

In this case the number of tests is limited by the time and the work involved in preparing each test, and especially because of the useful life of some components. In the particular case of a prototype that was to be tested over a real target, 26 closed loop simulations were done at the testing facility and their results appear in Figure 5.

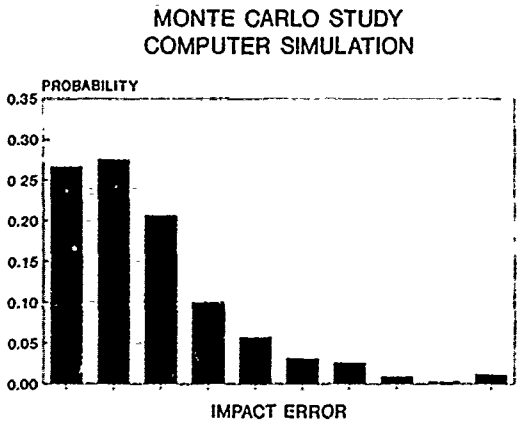


Fig. 5.1

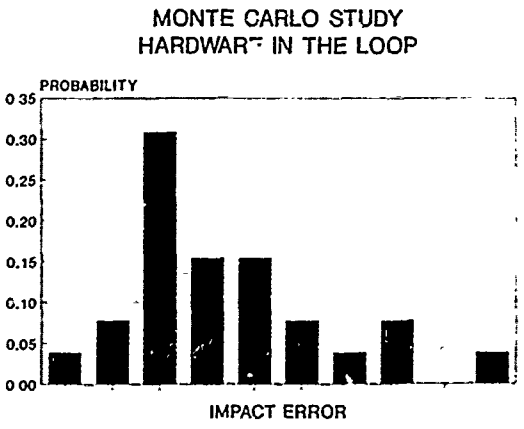


Fig. 5.3

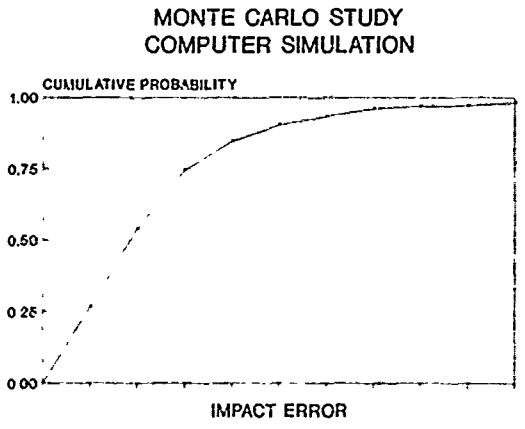


Fig 5.2

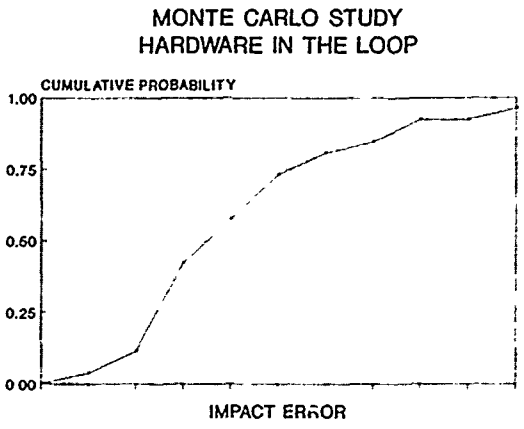


Fig. 5.4

Figure 5

Comparing this impact error distribution with that obtained in the computer simulation, the deviation or systematic error introduced by the residual disarrangement in the installation, basically in the target simulation can be noted.

11.- CONCLUSIONS

The results comparison between simulation and real flights briefly exposed in this article allows to qualify as 'very satisfactory' the degree of similarity with reality reached by the SIMTRA simulation.

Nevertheless, and in spite of the interest in this aspect, we want to make a special emphasis in the great importance that simulation must have in these type of projects for reducing costs as well as for developing them faster. Our own experience in this project has led us to this conclusion.

In the beginning, simulation was mainly used to check the flight performances of the aerodynamic configuration designs, but many other mechanical designs, and even electronic ones, which were assumed along the project, were included in the simulation 'a posteriori'. This means that simulation came 'behind'.

With the progress of the project, we learned that simulation must go 'before', verifying and validating the design from the conceptual phase. And not only the design, but also the hardware and software subsystems themselves, so that the final integration of the whole system can be reduced to an almost routine task.

GWSIM
A Computer Based Design and Simulation Package for Land Based and Air Weapon Systems

P.M.G.Silson BSc,PhD,AMIEE.
B.A.White MSc,PhD,MIEE.

Control and Guidance Group,
Royal Military College of Science (Cranfield),
Shrivenham, WILTS. ENGLAND

Abstract

This paper describes and demonstrates a weapon system design and simulation package developed at the Royal Military College of Science (RMCS). The package has been developed for use both as a research and development design tool as well as a Computer Aided Learning (CAL) / demonstration package.

The package is currently used in projects as diverse as guided weapon systems, VTOL aircraft control systems, helicopters, and autonomous land vehicles systems.

In this paper the use of the package in design and simulation of a medium range ground to air missile is described and demonstrated.

Introduction

GWSIM was originally developed as a design simulation package for weapons control systems, but has since expanded into a computer aided learning (CAL) package for teaching of weapons control system design at both undergraduate and post graduate level. The package is currently used on IBM PC or equivalents (figure 1) but is readily transportable to other systems.

The package is menu driven, making it self explanatory and user friendly and, after long development, the package is virtually "user proof". That is to say, any keyboard input to the simulator will result either in a sensible action or in a warning message and a request to the user to enter something else. This feature separates GWSIM from many current simulator systems that are both user unfriendly and prone to respond to false keyboard inputs by either crashing or locking up.

The package has been designed in such a way as to allow the inclusion of any control system model simply by the insertion of a single module containing all the system equations and parameters. This makes it extremely versatile, the adaptation to different system applications being simply a matter of providing the dynamic equations.

The package in its teaching form allows a student to apply control design techniques such as Nyquist, Bode or root locus to a weapon control system design and then be able to immediately realize time responses from their design in order to evaluate the overall performance of their design.

In the particular example discussed in this paper a facility has been added to allow the time simulator outputs to be used to drive a simple mechanical model of a CLOS guided missile. This was intended to allow the student to observe the physical actions of the missile as it responded under closed loop control. Thus the effects of incidence lag, weather cock frequency, non-minimum phase lateral acceleration etc. are seen directly.

The package was originally produced as a control system design package and has included a data logging module to allow capture of data from real systems under test to allow system identification and model validation to be accomplished off line.

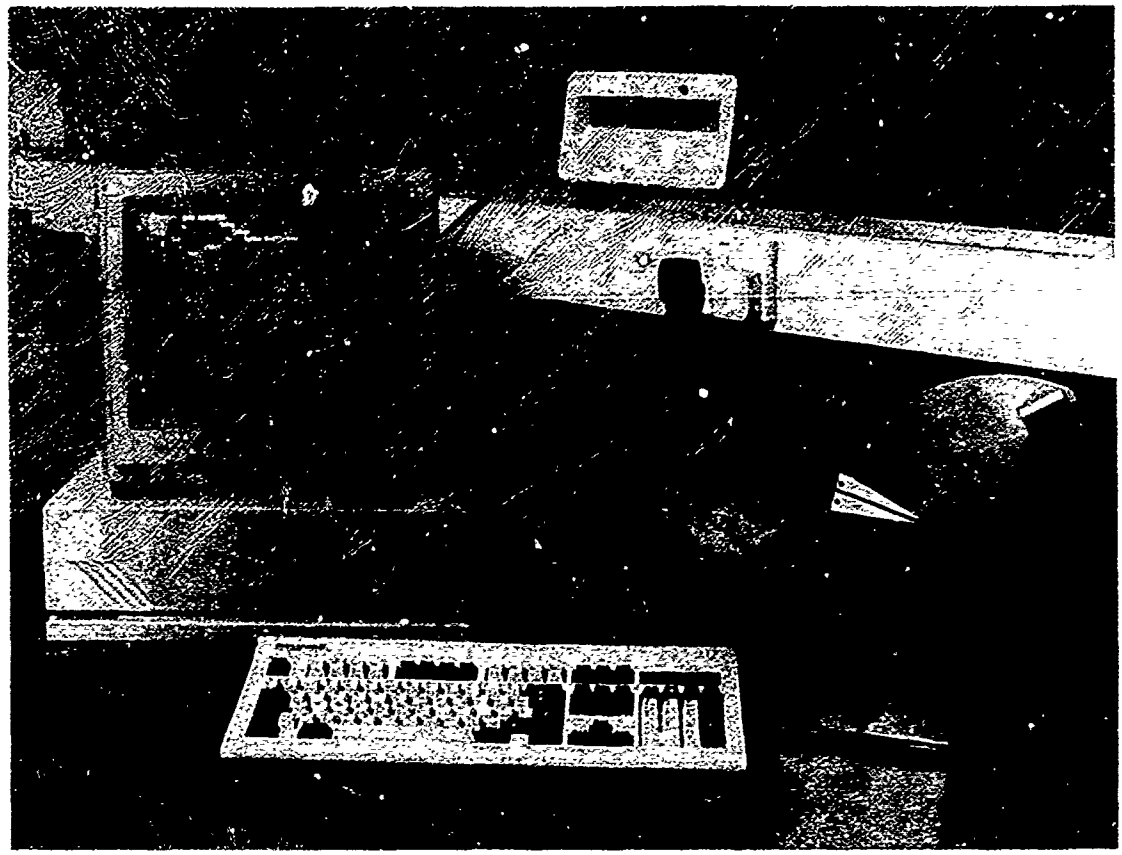


Figure 1: GWSIM Hardware

Structure of GWSIM

The software for GWSIM is written in a deliberately modular form in order to allow easy restructuring of the package towards various applications.

GWSIM consists of a number of design, simulation, and data handling modules linked by a screen menu handler. Modules currently available on the simulator include:

1. System equations module, containing all the system equations and state labels of the particular application.
2. Linear modelling and design. Frequency design suites using Bode, Nyquist and inverse Nyquist techniques, as well as a root locus design package.
3. Non-linear modelling and simulation. A time response simulation suite with facilities for inclusion of amplitude, time and environmental non-linearities as well as stochastic effects.
4. Coefficient handling. Allows the dynamic variation of any system coefficient during simulation as well as a facility to store/retrieve default or current values.
5. Data acquisition and storage. A high speed analogue/digital data collection package and interface for acquisition of test data, on up to 64 channels, for model validation and estimation.
6. Demonstrator Units. A facility for driving demonstration models using test or simulation data as a teaching aid.

A particular application is installed into the simulator by the insertion of a single module containing the system dynamic equations with a set of state, output and coefficient labels. These reference labels are carried throughout the simulation software and appear as labels on menus, graphical output etc. The system equations module also contains the description of any non-linearities in the model for use in the time response simulator.

The example discussed later in this paper is a simplified two dimensional application of a CLOS missile system however a general six degree of freedom model has been written as a linkable module.

This module contains all the necessary axis transformations to view the model in vehicle reference or earth (inertial) reference axies. These transforms are implemented in both quarternian and direction cosine form for use depending on the particular application. The motion equations module for the full six degree of freedom model consists of a set of forces (aerodynamic, frictional, thrust etc.) applied to the 3 dimensional model with the mass and inertia matrix defined.

For any particular application the menu driver is used to link the required design/simulation packages to the basic system equations as needed, thus tailor made generation of new application packages is straightforward.

System Models

System models are presented to the package as a single insertable module containing the state equations of the system plus piecewise linear or polynomial gain non-linearities and spectral characteristics of noise sources.

The system model discussed later in this paper is a ground based, medium range anti-aircraft missile developed during a weapon system design project. This model includes the dynamics of the fin actuation systems, the dynamic and aerodynamic equations of the missile and autopilot and on board instrumentation, and the kinematic and dynamic equations of the guidance loop and tracker. The model also includes kinematic equations of simulated targets for later evaluation of the overall designed systems performance.

Any number of model parameters (such as aerodynamic derivatives, missile size and mass distribution, velocity profile, instrumentation and control system gains etc.) can be varied dynamically during the design and simulation processes and these are specified in the system equations module. To each of the variable coefficients and system outputs is attached a label, this label along with the coefficient value is stored globally and is therefore accessible to any program module subsequently linked into the simulator.

The variable coefficients and state initial conditions are also allocated default values in this routine.

Non-linear elements, such as fin angle/rate saturation, velocity / altitude dependant aerodynamic derivatives and radar glint noise, are specified in this module although these are either fixed or linearised when used in the linear design package.

Noise sources, such as radar glint, are represented by shaped white noise specified by a frequency / phase characteristic applied to a Gaussian noise generator. Gain non-linearities are specified either as a piecewise linear characteristic (such as simple saturation) or as a polynomial representing the gain characteristic. Multiplicative, dual valued non-linearities as well as time delays, quantization and sampling delays and many other non-linear elements can be included, although these are not applicable to the example shown.

A sub-module is linked to the system equations module that generates standard input functions such as steps, ramps, parabola and sinewaves etc. however non-standard inputs may be either generated from within the module or imported in from data files of, for instance, measured test data.

In the example discussed later, one of the inputs to the system is the line of sight angle of a typical target. This input is generated internally by including the kinematic equations of motion of the target and simulating these as part of the system dynamic equations.

Integration Routines

The simulation equations are presented as a set of non-linear state equations with gain coefficients set by the fixed model coefficients as well as the variable design parameters. Initial conditions are given to each of the system states and each state derivative transferred to an integration package. The iteration interval and number of graphics data points produced are set to a default value in the equations module, but there is also a menu facility for interactively varying these.

In the majority of simulations of reasonably well conditioned feedback systems it has been found that non-recursive (Runge-Kutta) type numerical integration is adequate although facilities do exist within the simulator to use more complex recursive algorithms. It has been found in practice that the simulator processing time is considerably increased with the use of recursive integrators and that, in the majority of applications, the results are not improved over those of the Runge-Kutta algorithms. One possible exception to this is when open loop integration is required, for instance, when simulating inertial navigation equations.

The integration routine produces a set of current system states, these are transferred back to the equations module to gene ·te the required system outputs which are stored in a global data file. Each stored output variable also carries with it its individual label assigned to it in the equations routine for use by the graphics routine and menu handler.

Menu Handler

For a particular application various modules are link together via a menu handler.

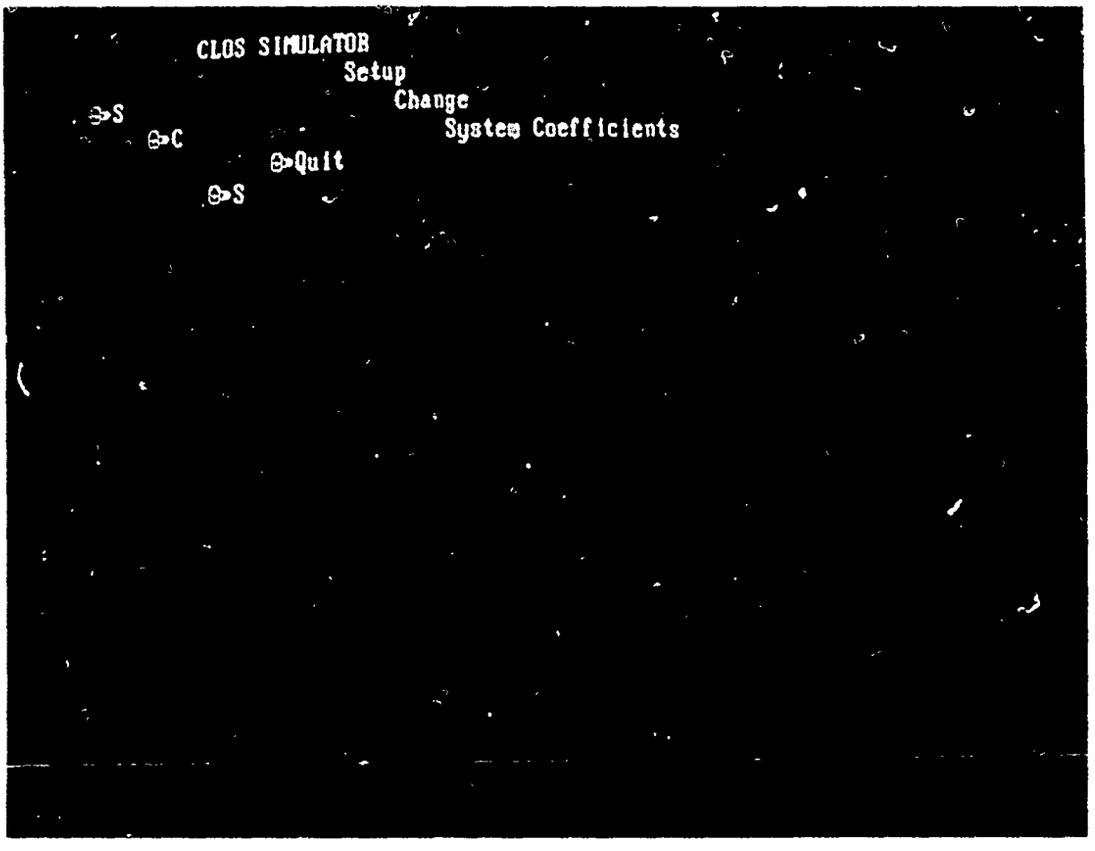


Figure 2: Typical Menu Screen

Figure 2 shows a typical menu screen. The cursor is driven from the normal keyboard screen editing keys ↑↓←→, *pageup*, *top* etc. or by a mouse.

The software is very tolerant of input data format types and will interpret keyboard input data into the correct form required by the simulator. For instance either 10.000 , 10. , 10 , 10E1 , 10.E1 , 10.0e1 will be interpreted correctly as a real number.

If an integer is required then the previous example would also be accepted, but an input of 10.01 would result in an error message saying "Integer number required, please input new value".

Graphics Output

The graphics handling module is common to both the time response simulator and the frequency response / root locus modules. It is linked to any other module via the menu handler and interfaces data via global data files or arrays.

The label data carried with the output data points is accessed firstly by the menu handler to request data variables to be displayed from the user and then by the graphics module to provide axis labelling / titling to the screen output.

The graphics module provides autoscaling facilities, zoom, multiple plots on common axies and two or three dimensional graph plotting. The frequency design packages output data contains flags to specify the type of output format required (Bode or polar etc.). Typical graphical output is shown in figures 6 and 7.

A hard copy dump facility is also provided which produces a pixel file of the graphics screen which is outputted directly to a printer port to a dot matrix or laser printer.

Data Acquisition

The data acquisition interface, although not as yet used on the guided missile system, does bear mention. The data acquisition interface module consists of a number high speed analogue/digital (AD) inputs ports, programmable digital input/output (PIO) ports and high speed digital to analogue (DA) output ports. These are set up to collect test data from a system under test, either under control of the package (as a scheduler or closed loop controller) or simply as a data logger. The test data is initially stored in high speed RAM and then filtered, processed and transfer onto Winchester drives or magnetic tape for later retrieval for off-line model estimation / evaluation and model comparison / validation.

A typical application of this module would be to record system input demands and output responses from vehicle test manoeuvres and then to apply either model identification or model validation algorithms off line. This system is current being applied to a ground based autonomous vehicle study being carried out for the Royal Signals & Radar Establishment by RMCS.

GWSIM Application CLOS Missile System

An example of the use of the package in the design of the servo-mechanisms, autopilots and guidance systems of a medium range ground to air missile will illustrate some of the features of GWSIM. The missile system used in this example does not represent any particular current system, but is a typical or generic model of the system type. This allows students using the simulator as a CAL package to have a feel for typical performance constraints that may be met on a real system.

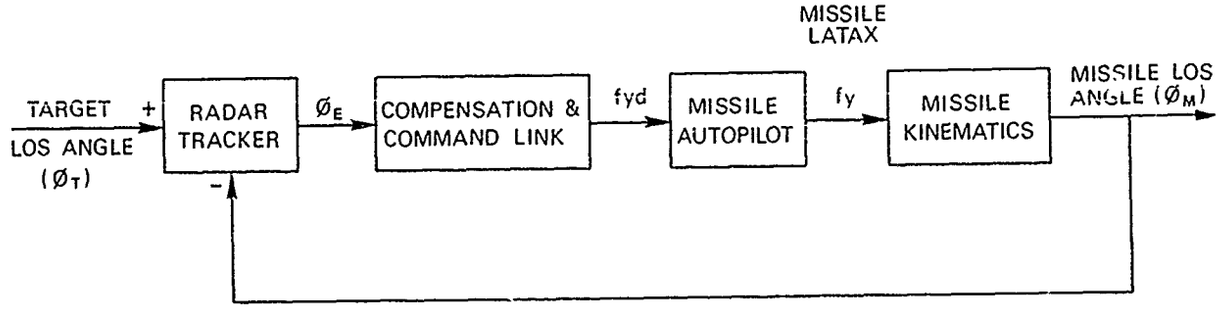


Figure 3: CLOS System

Figure 3 shows a block diagram of the CLOS system guidance loop in this example. The specific technologies involved in, for instance, the command link, target and missile tracker etc. are unimportant to the student at this stage, since they are only being asked to design the various control system parameters on a predefined model. The simulator does however allow the inclusion of more detailed models of tracker dynamics etc. for specific design studies.

The missile aerodynamic coefficients and mass/inertia matrices are all interactively variable, allowing the student firstly to study the effects of changing various parameters on the default system model and secondly to allow him to simulate complete' · different missile systems within the same simulator. For instance the system being discussed in ' s paper is a ground launched supersonic CLOS anti-aircraft system, but by changing the various mass, inertia and aerodynamic derivative coefficients a sub-sonic anti-tank system could equally well be simulated.

In this application the student is given the mass, inertias and aerodynamic derivatives of a typical system, and is asked to meet certain design criteria in terms of miss distance for the outer (guidance) control loop. He is restricted to varying certain parameters such as

- Fin servo-mechanism position control loop gain.
- Autopilot pitch rate gyroscope gain.
- Autopilot lateral accelerometer gain.
- Accelerometer positioning relative to the missile C of G.
- Kinematic stiffness of the guidance loop.
- Pole/zero separation and centre frequency of the forward path lead network.

Linear System Design

Figure 4 shows a schematic of the missile autopilot installed in the simulator.

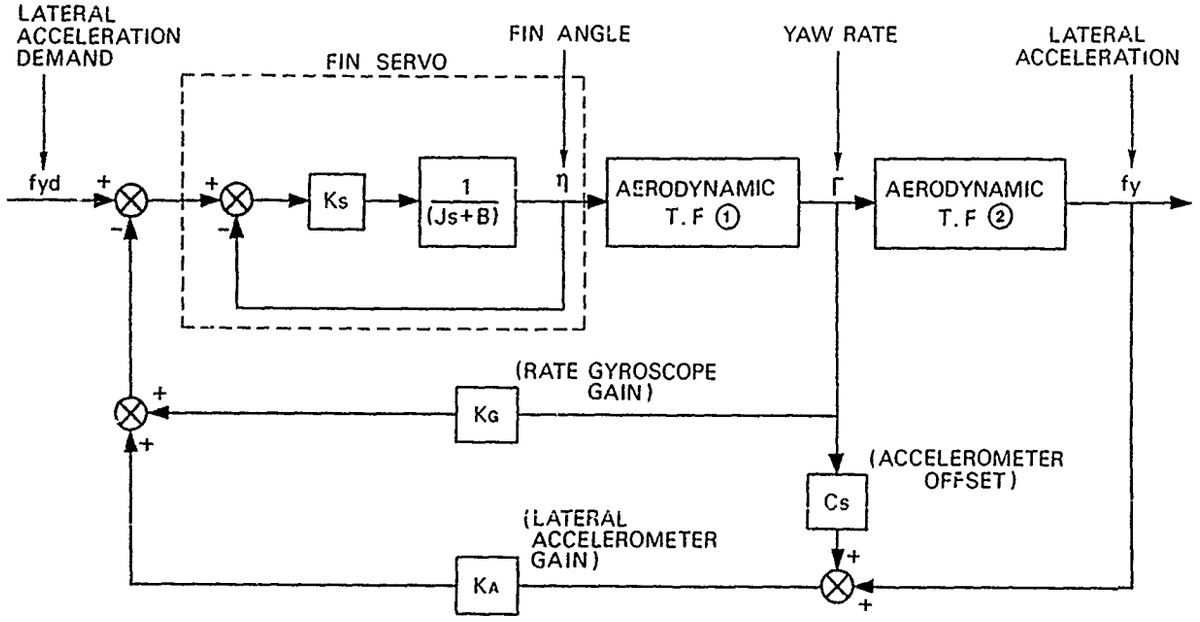


Figure 4: Autopilot Schematic

where

$$\frac{r(s)}{\eta(s)} = \frac{\eta_\eta s + (\eta_v y_\eta - \eta_\eta y_v)}{s^2 - (y_v - \eta_r)s + (y_v \eta_r + U\eta_v)}$$

and

$$\frac{f_y(s)}{r(s)} = \frac{U(\eta_v y_\eta - \eta_\eta y_v)}{\eta_\eta s + (\eta_v y_\eta - \eta_\eta y_v)}$$

The performance specification for the miss distance of the CLOS system against a specific target is estimated by the student and gives a bandwidth requirement for the guidance loop of 10 rads/sec.

This in turn gives rise to a bandwidth specification on the autopilot of 60 rads/sec and similarly this puts a closed loop bandwidth requirement on the fin servo-mechanism of 200 rads/sec.

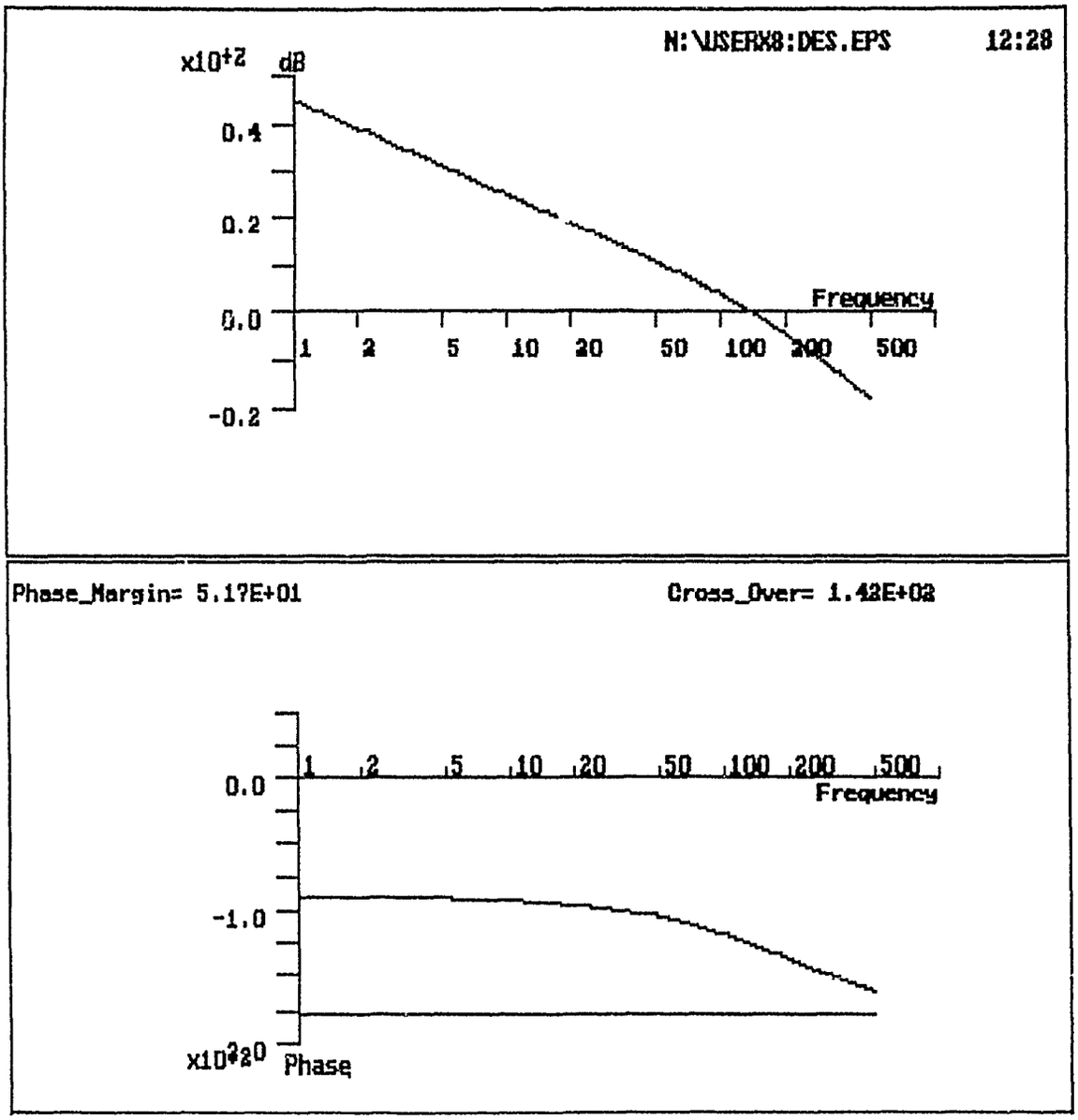


Figure 5: Open Loop Frequency Response of Fin Servo-mechanism

Fin Servo Loop Design

The first exercise normally given to the student is the design of the fin servo loop gain to achieve a required bandwidth with reasonable damping ratio (of say 0.5). Figure 5 shows the open loop Bode plot produced by the frequency response design package. It can be seen that the package also presents on the Bode plot the gain and phase margins as well as the gain crossover frequency.

From this the student can then ascertain the servo loop gain (K_s) to give a phase margin of roughly 50° (or a closed loop damping ratio of 0.5). Having chosen the fin servo loop gain the student can examine firstly the closed loop Bode plot and secondly the step response of the fin servo to check his design.

This first design exercise involves only a simple second order closed loop system and is useful in:

- Introducing the student to the CAL package with a simple example.
- Allowing the student to design a closed loop system and to be able to check the resulting closed loop frequency and time responses, having a reasonable idea what the responses should look like beforehand.
- Allowing the student to relate frequency domain and time domain specifications eg. bandwidth (from the frequency response) and time to first peak (from the step response).

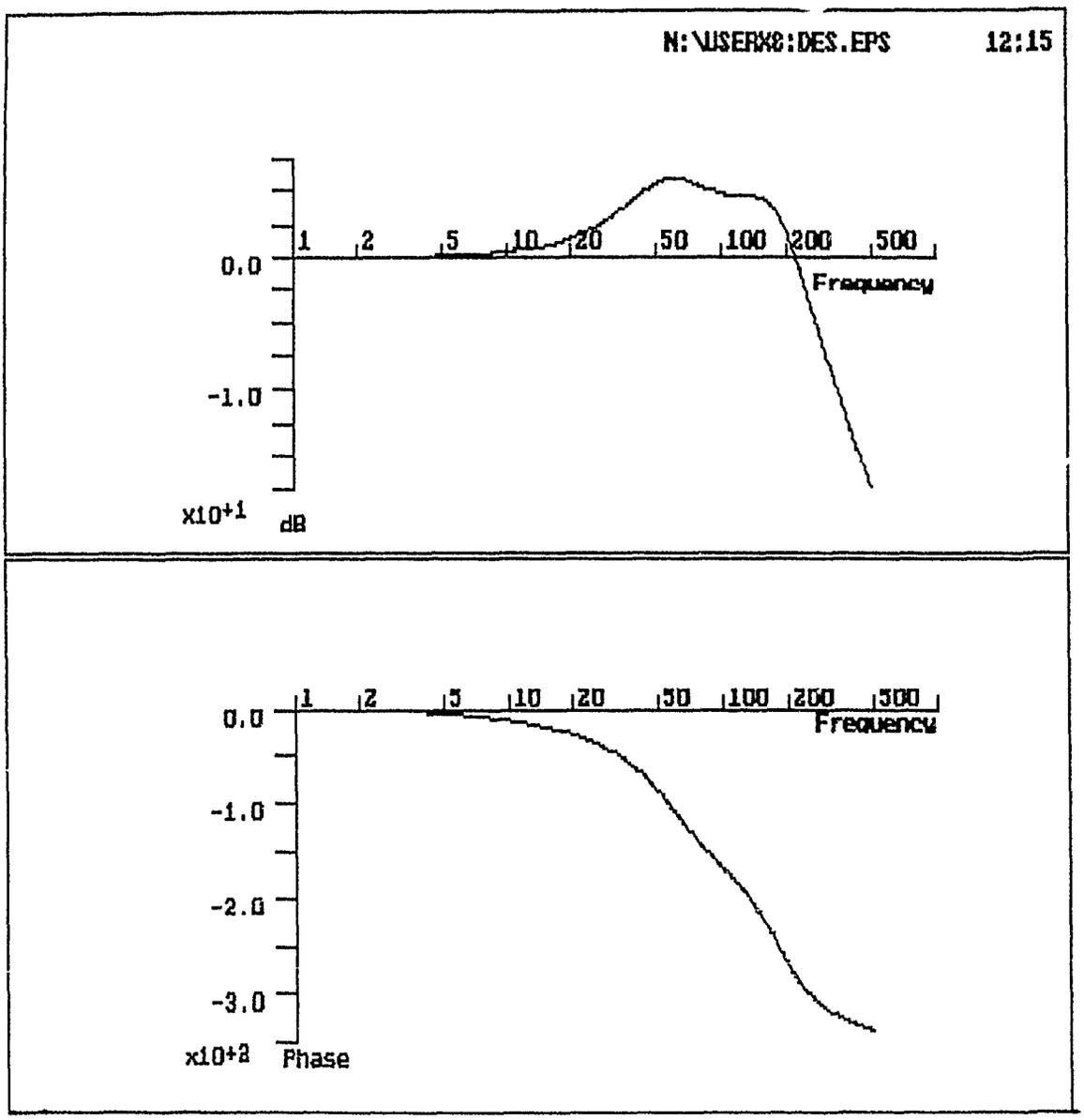


Figure 6: Closed Loop Frequency Response of Autopilot

Design of Autopilot Loop

Having completed the design exercise on the fin servo mechanism and having checked that the bandwidth and damping ratio specifications have been met, the student then goes on to design the autopilot. Initially the student is asked to meet the performance specification with accelerometer only feedback (K_G and C set to zero). It is soon apparent that it is not possible to obtain a stable autopilot response and that yaw rate gyro feedback is required in the autopilot controller.

Again the student can examine the effect of various rate gyro gains and accelerometer gains on the open and closed loop frequency responses. In the this particular system it is possible to produce a stable design but still not possible to meet the design specification in terms of bandwidth because of the low damping of the missile weathercock mode. The student now has to go on to examine the effect of moving the accelerometer ahead of the missile centre of gravity giving yaw angular acceleration feedback to augment the angular rate feedback. The final design closed loop frequency response and closed loop step response are shown in figures 6 and 7.

This design has a bandwidth of roughly 60 rads/sec and an effective damping ratio of 0.45.

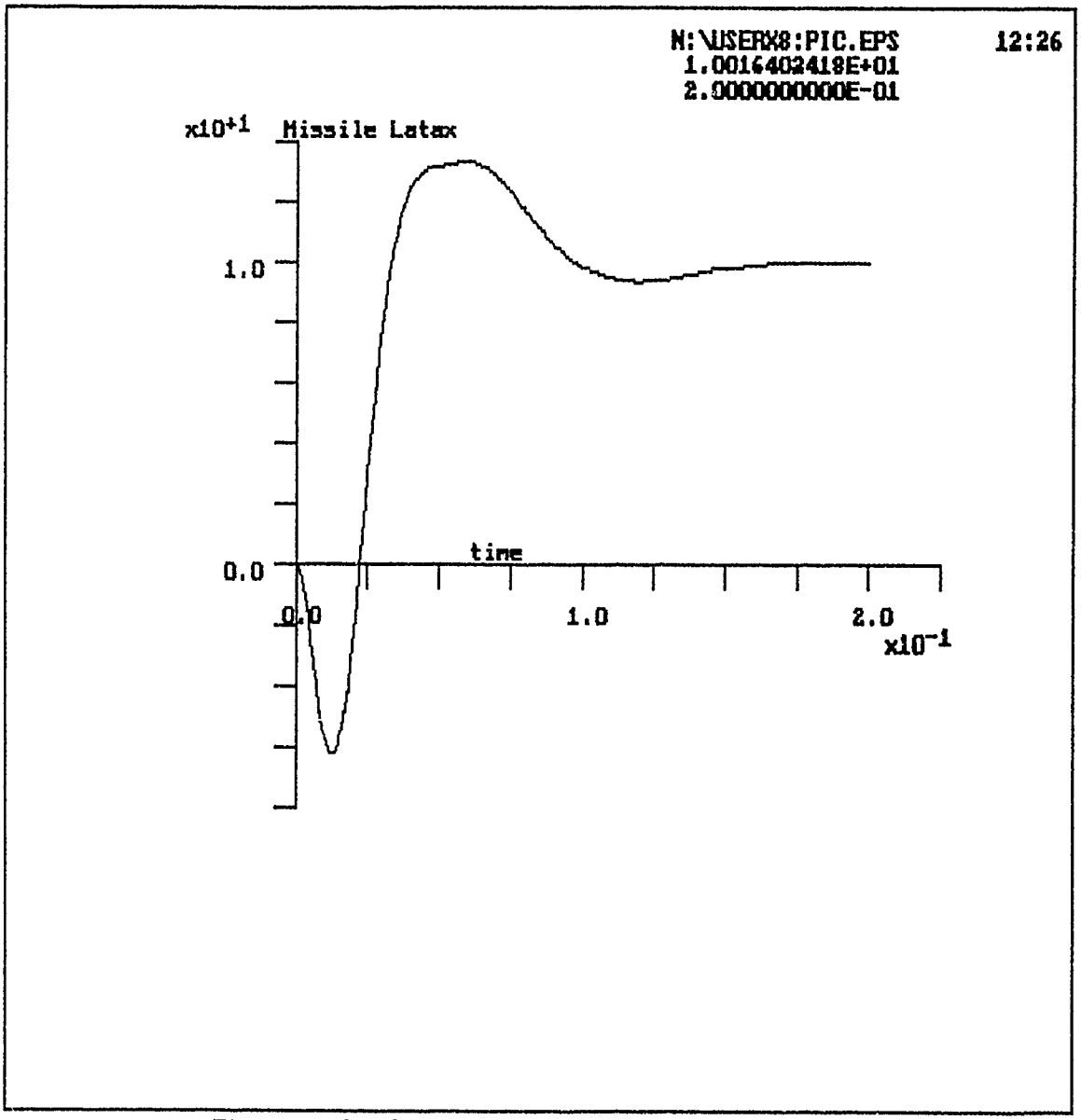


Figure 7: Closed Loop Step Response of Missile Autopilot

Design of Guidance Loop

Having met the design specification on the missile autopilot the guidance loop has to be designed.

It is not possible to do this without substantial forward path lead compensation since the missile kinematics in the guidance loop (figure 3) are inherently type 2 in nature. Again open and closed loop frequency plots are used to design the forward path lead networks, beam stiffness etc. to accomplish the desired bandwidth in the guidance loop. A closed loop guidance system can be obtained with a bandwidth of 10 rads/sec and an effective damping ratio of 0.4.

Linear System Performance Evaluation

Closed loop time responses can now be taken to test the miss distance performance against various target trajectories. Output can also be obtained of the true target and missile tracks in space to examine typical engagement trajectories (figure 8).

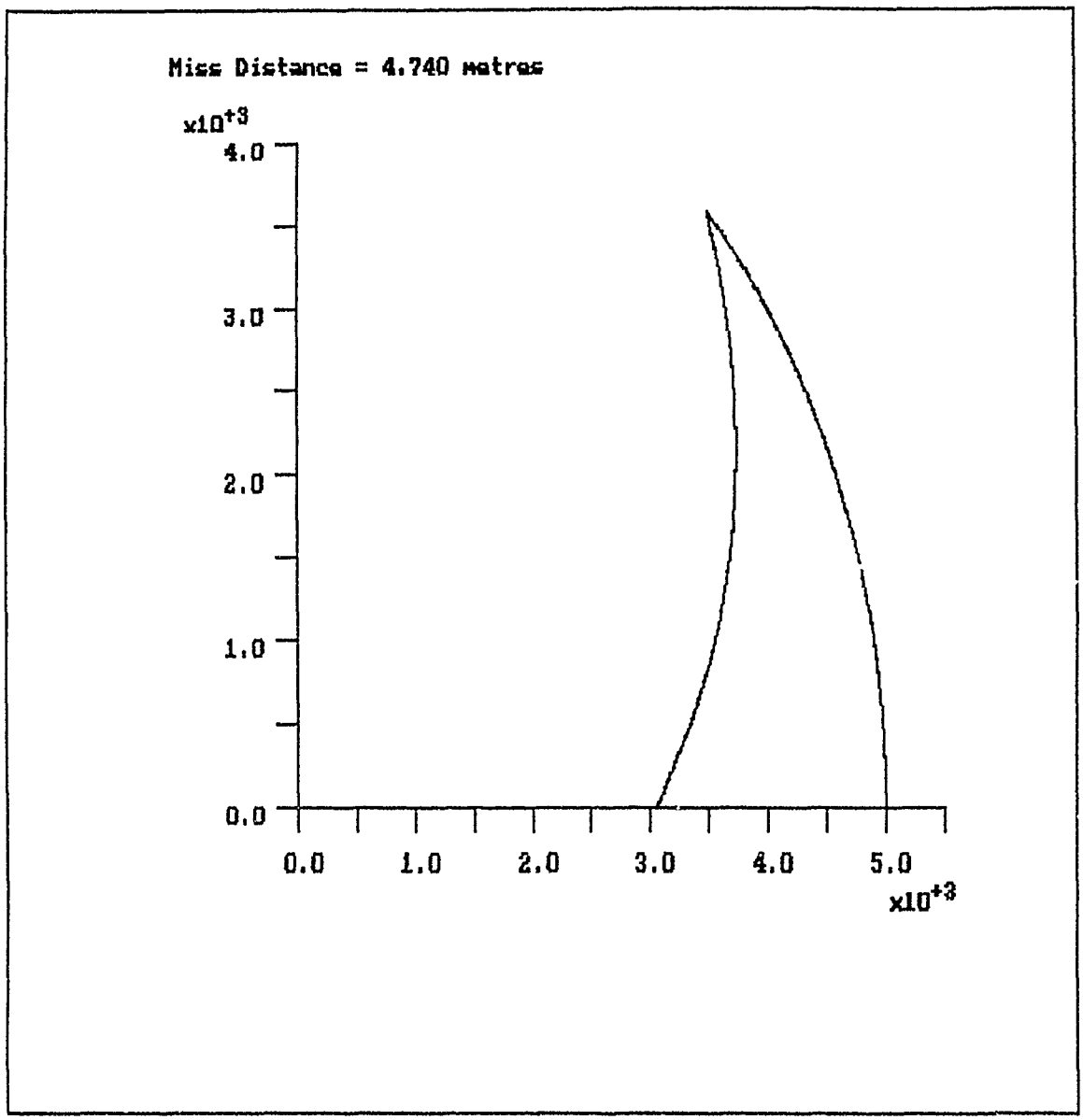


Figure 8: Typical Missile/Target Trajectories

Non-linear Modelling and Simulation

Having produced a satisfactory design on the linearized models the effects of various non-linear elements in the system can be examined. Three particular non-linearities are of interest in this system, firstly the effects of fin servo position (angle) and rate saturation on the autopilot stability, secondly the effect of altitude/velocity dependant aerodynamic derivatives on the autopilot, and lastly the effect of these and radar glint noise on the overall performance of the weapon system in terms again of miss distance against particular targets.

Model Demonstrator

This particular design exercise is included in a CAL package used during a guided weapon control system design course aimed at both military and industrial staffs. A mechanical model of the missile system has been built (figure 1) with facilities for physically representing fin and body motions and linear acceleration and velocity vectors of the missile. This mechanical model is linked to the simulation package via analogue and digital interfaces so that the graphically presented data can also be directed to the demonstrator model. This is useful for showing, in real or scaled time, the effects of, for instance, a lateral acceleration demand to the closed loop autopilot by the guidance system on the body motions

of the missile. The physical effects of non-linearities on the missile dynamics are very effectively visually demonstrated in this way.

Summary

GWSIM is a versatile CAD/CAL tool that may be used on many specialised weapons control system problems. It is part of an on going development program of weapon control system simulators at the Royal Military College of Science.

Because of the modular nature of the program software, development of new dedicated simulator systems is straightforward. The system is extremely user friendly and user proof and because of this makes it a useful computer aided learning tool for demonstration/teaching of complex control system design for users who are not necessarily fully computer literate.

One particular application has been discussed here but GWSIM is currently being used in the design of a wide range of application from an autonomous land vehicle system to a fly by wire VTOL aircraft autopilot.

It is hoped in the near future to include system models modules of various generic guided weapons to offer a broad based design package as well as a general computer aided learning package. Currently models have been written for the simulator of CLOS ground to air systems (the example discussed in this paper) CLOS anti-tank systems, active and passive homing GW systems, VTOL aircraft and wheeled and tracked land vehicles.

The package is currently installed in stand alone and networked IBM PCs and Sun workstations. The software is, however, transferable to many other systems.

THE MICROCOMPUTER AS A TOOL FOR GUIDANCE AND CONTROL VISUALIZATION

by

Paul Zarchan
The Charles Stark Draper Laboratory, Inc.
555 Technology Square
Cambridge, MA 02139
United States

Abstract

This paper shows how simulation output can be generated and enhanced, in real time, with the computational horsepower and graphics visualization technology which is currently available with microcomputers. Examples are presented which demonstrate how microcomputer based technology offer the designer a visualization which not only gives a deeper insight into the problem being solved, but in addition allows and encourages rapid iteration in order to get an acceptable design.

Introduction and Overview

In the last five years we have witnessed a proliferation of desktop personal computers unimagined only a decade ago. The 32-bit 80386 and 68030 microcomputers are computationally as powerful as a mainframe was only 10 years ago. Currently $5000 of microcomputer provides about as much computational horsepower as a $500,000 super minicomputer[1]. The intent of this paper is to show how the power of the microcomputer can be harnessed by the missile guidance system engineer, not only to computationally solve useful guidance system related problems, but also to provide a visualization which can be used to speed up the design process.

The paper presents several interceptor guidance system related examples which, until recently, were normally solved on mainframes. It is first demonstrated that these examples can be made to work on microcomputers with CPU running times which are very attractive and turn around times (i.e. time for engineer to get the answer in a useful form) that are far superior to that offered by a time-shared mainframes. It is then shown how these answers can be enhanced, in real time, with the graphics visualization technology which is currently available with microcomputers. The enhanced answers will offer the designer a visualization which not only gives a deeper insight into the problem being solved, but in addition allows the user to rapidly iterate cases to get an acceptable design.

The first example presented is that of a rate gyro flight control system for a tactical radar guided homing missile. The purpose of the example is twofold. First it is used as a reference to compare answers and CPU timings from a variety of hardware platforms in the microcomputer, minicomputer and mainframe worlds. Next it will be shown how instantaneous graphical output from both a time and frequency point of view enables the designer to rapidly understand the influence of the autopilot gain on the relative stability and performance characteristics of the flight control system.

A second example considers a satellite in circular orbit. The paper first shows how the satellite can be simulated on a microcomputer. Next it is shown how commercially available mapping data bases can be incorporated in the satellite microcomputer simulation to provide geographical context to the resultant satellite ground tracks. Finally it is shown how linear and orthographic transformations of the mapping data and satellite trajectory provide complementary three-dimensional visualizations on a two-dimensional microcomputer screen.

A final example extends the satellite simulation to include a strategic surface-based interceptor pursuing the satellite. It is shown how the use of dialog boxes with edit fields and buttons can be used to input simulation data and provide the user with complex options in a "user-friendly" way. It is also demonstrated how the simultaneous presentation of information in different windows provides insight which is invaluable in understanding interceptor performance related issues and in visualizing the engagement.

Rate Gyro Flight Control System Example

In order to illustrate the use of graphics in an interactive microcomputer environment, a representative example, is taken from missile guidance and control. A rate gyro flight control system for a radar guided missile[2] is shown in Fig. 1. The purpose of this flight control system is to ensure that the achieved body rate follows the body rate command. The gain, K, provides unity transmission between input and output while the autopilot gain, K_R, influences the system dynamic response. In this flight control system the autopilot generated fin deflection command, δ_C, is sent to the actuation system. This electrical command is converted by the

actuator to a mechanical deflection, through an angle δ, of the missile's control surface. The control surface deflection causes the missile body to pitch. A rate gyro is used to measure the achieved body pitch rate thus completing the feedback path. In this simplified model the body pitching can be described by rigid body dynamics expressed as differential equations or in transfer function form as shown in Fig. 1.

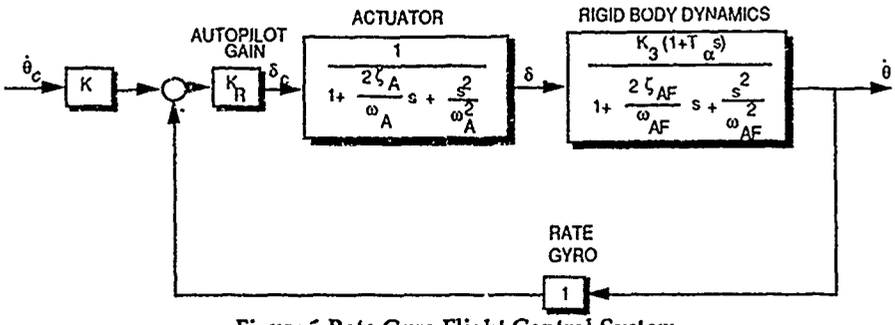


Figure 1 Rate Gyro Flight Control System

From Fig. 1 we can see that the differential equations that govern the system behavior are given by

$$\frac{d\theta}{dt} = -K_3\theta - K_3 T_\alpha \frac{d\theta}{dt}$$

$$\delta_c = K_R\left(K - \frac{d\theta}{dt}\right)$$

$$\frac{d^2\delta}{dt^2} = \omega_A^2\left(\delta_c - \delta - \frac{2\zeta_A}{\omega_A}\frac{d\delta}{dt}\right)$$

$$\frac{d^2\theta}{dt^2} = \omega_{AF}^2\left(\delta - \theta - \frac{2\zeta_{AF}}{\omega_{AF}}\frac{d\theta}{dt}\right)$$

where the autopilot gain, K, provides unity transmission and can easily be shown to be

$$K = \frac{1 - K_R K_3}{-K_R K_3}$$

Nominal parameter values for the rate gyro flight control system appear in Table 1.

Symbol	Name	Definition	Value
ζ_A	ZA	Actuator damping	.7
ω_A	WA	Actuator natural frequency	300 rad/sec
K_3	K3	Airframe gain	$-.2\ sec^{-1}$
T_α	TA	Airframe turning rate time constant	2 sec
ζ_{AF}	ZAF	Airframe damping	.1
ω_{AF}	WAF	Airframe natural frequency	10 rad/sec
K_R	KR	Autopilot gain	1.5 sec

Table 1 Nominal Rate Gyro Flight Control System Parameter Values

This flight control system can be simulated using FORTRAN and the second-order Runge-Kutta integration technique[3] for solving the preceding differential equations. The program listing of the rate gyro flight control system appears in Listing 1. We can see, that because of the high frequency actuator dynamics, a very small integration step size is required (H=.001 sec) to accurately numerically integrate the differential equations. The system differential equations appear after statement label 200. Special logic is included in the listing so that the answers are displayed every .005 sec.

```
      INTEGER STEP
      REAL K,KR,K3
      DATA ZA,WA,K3,TA,ZAF,WAF/.7,300.,-.2,2.,.1,10./
      DATA KR,THDC/1.5,1./
      K=(1-KR*K3)/(-KR*K3)
      DEL=0.
      DELD=0.
      E=0.
      ED=0.
      T=0.
      H=.0001
      S=0.
5     IF(T.GE.1.)GOTO 999
      S=S+H
      DELOLD=DEL
      DELDOLD=DELD
      EOLD=E
      EDOLD=ED
      STEP=1
      GOTO 200
66    STEP=2
      DEL=DEL+.1*DELD
      DELD=DELD+H*DELDD
      E=E+H*ED
      ED=ED+H*EDD
      T=T+H
      GOTO 200
55    CONTINUE
      DEL=.5*(DELOLD+DEL+H*DELD)
      DELD=.5*(DELDOLD+DELD+H*DELDD)
      E=.5*(EOLD+E+H*ED)
      ED=.5*(EDOLD+ED+H*EDD)
      IF(S.GE..004999)THEN
        S=0.
        WRITE(9,*)T,THD
      END IF
      GOTO 5
200   CONTINUE
      DELC=KR*(K-THD)
      DELDD=WA*WA*(DELC-DEL-2.*ZA*DELD/WA)
      EDD=WAF*WAF*(DEL-E-2.*ZAF*ED/WAF)
      THD=-K3*E-K3*TA*ED
      IF(STEP-1)66,66,55
999   CONTINUE
      PAUSE
      END
```

Listing 1 FORTRAN Simulation of Rate Gyro Flight Control System

The transient response of the rate gyro flight control system with a 1 deg/sec step input is shown in Fig. 2. From this figure we can see that initially the system output overshoots the input (i.e., output body rate reaches a peak of 4 deg/sec) but eventually follows the input. The response is stable and appears to be well behaved for the autopilot gain setting of K_R=1.5.

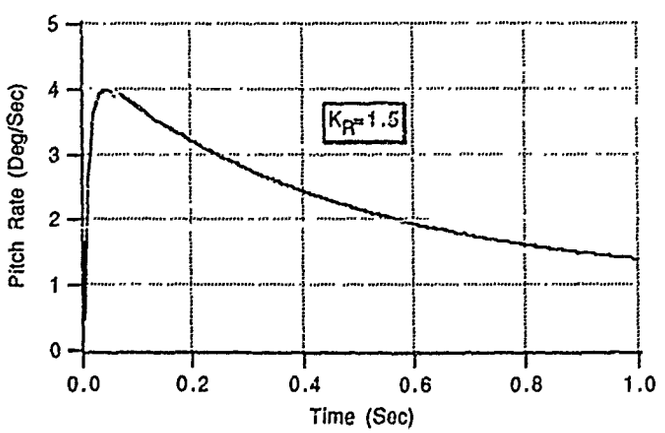


Figure 2 Nominal Response of Rate Gyro Flight Control System

FORTRAN Comparison

The simulation of the rate gyro flight control system, using the FORTRAN source code of Listing 1 was solved on microcomputers representative of the 8-bit, 16-bit, and 32-bit world and their running times were compared in the 1987 time frame.[2] The machines used in this comparison were the original IBM PC, an improved PC, an IBM AT, a Macintosh Plus, and Macintosh II microcomputers. The performance of the machines are compared with and without math coprocessors. Table 2 presents the running time comparisons.

Coprocessor	IBM PC	Improved PC	IBM AT	Macintosh Plus	Macintosh II
Out	520 s	75 s	39 s	61 s	15.4 s
In	--------	40 s	35 s	--------	7.4 s

Table 2 FORTRAN Running Time Comparison For Rate Gyro Flight Control System Example

Table 2 indicates that the original IBM PC is very slow, compared to the other machines on the rate gyro flight control system example. However, newer versions of the 4.77-Mhz, 8-bit IBM PC and clones are significantly faster (and less expensive too). For example, the IBM AT is about twice as fast as the improved IBM PC, and the Macintosh II is four times faster than the Macintosh Plus. Addressing the math coprocessor significantly improves the speed of both the Macintosh II and the improved IBM PC. However, addressing the math coprocessor on an IBM AT results in negligible speed improvement. The performance improvement for the IBM AT is not as significant because the math coprocessor operates at 4 Mhz whereas the machine is running at 6 Mhz. From Table 2 we can see that the 32-bit Macintosh II is nearly 35 times faster than the original 8-bit IBM PC. When the math coprocessor is addressed, it is nearly 70 times faster The current generation of 33 Mhz 80386 clones and 40 MHz 68030 based microcomputers are even faster than the Macintosh II. Clearly there have been many improvements since the introduction of the first IBM PC.

The sample problem was also run in FORTRAN on two super minicomputers and one mainframe computer. The running times are summarized in Table 3.[2]

IBM PC	IBM AT	Macintosh II	VAX/785	VAX/8600	IBM/3084Q
520 s	35 s	7.4 s	3.1 s	0.74 s	0.61 s

Table 3 Microcomputer, Minicomputer, Mainframe Running Time Comparison

In this table the running time for the larger machines corresponds to CPU time with a single-user load on a time-sharing system. Usually large machines are shared among many users, and the CPU time is indicative only of what the user is charged for a session. In addition, on large machines the turnaround time (the elapsed time it takes the user to get the output) may be hours, even though the CPU time may be in seconds. On a microcomputer the CPU time is the turnaround time. Nonetheless, Table 3 indicates that the Macintosh II is only 2.4 times slower than the VAX/785 and 12 times slower than the mainframe. Considering that the Macintosh II costs about $5,000, whereas the VAX/785 is about $250,000 and the IBM/3084Q is several million dollars, the comparison is more impressive. Most importantly, the sample rate gyro flight control system problem could be solved on a microcomputer in a very reasonable amount of time.

Open-Loop Transfer Function

Valuable information is available from the time-domain simulation of the system differential equations. However, additional information is also available from the system open-loop transfer function. The concept of the open-loop transfer function is the basis of feedback control systems analysis. While the whole open-loop transfer function is interesting, its frequency response characteristics are most useful to the designer when examined in the frequency domain. Both relative stability and robustness can be determined from an analysis of the magnitude and phase of the open-loop frequency response, and even more importantly, the designer can determine from it what changes to make in the system dynamics in order to achieve design goals.[4,5]

The open-loop transfer function is the transfer function around the loop when the loop is broken at a point. Although the loop can be broken anywhere, it is usually broken in series with some parameter whose value the designer can control to achiev a desired characteristic. For example, we can break the loop of a single-loop feedback contro_ system at the error signal as shown in Fig. 3.

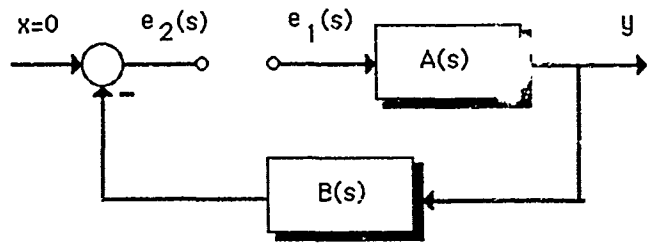


Figure 3 Sample Open-Loop System

In this case the open-loop transfer is defined as

$$HG(s) = -\frac{e_1(s)}{e_2(s)} = A(s)B(s)$$

In order to fully understand open-loop concepts, it is first required to understand the mechanics of finding the magnitude and phase of an open-loop transfer function. This can be done by replacing the complex frequency s in the transfer function with

$$s = j\omega$$

where

$$j=(-1)^{1/2}$$

Usually the magnitude of the open-loop transfer function is expressed in db where

$$db=20\log_{10}(\text{Magnitude})$$

and the phase is expressed in degrees.

With the open-loop transfer function other quantities are also important. For example, the gain margin gm is the value of additional gain required at the loop break (assuming the phase remains constant) to cause instability while the phase margin ϕ_{pm} is the amount of phase lag required at the loop break (assuming that the gain remains constant) to use instability. In addition to these margins, crossover frequencies are also of interest. The gain crossover frequency ω_{cr} is the frequency at which the open-loop magnitude is unity, while the phase crossover frequency ω_{180} is the frequency at which the open-loop phase is -180 deg. Both these crossover frequencies indicate the frequency of the ensuing oscillation in the time domain, should the system go unstable due to an increase in gain or decrease in phase.

In order to demonstrate the utility of the open loop transfer function, let us revisit the rate gyro flight control system of Fig. 1. Figure 4 shows the same system, except this time the loop is broken at the error signal. The loop is broken here because the designer can control the autopilot gain K_R.

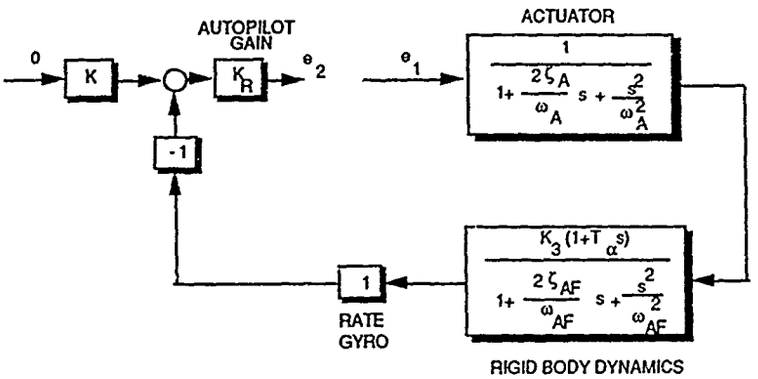


Figure 4 Open-Loop Model Of Rate Gyro Flight Control System

From the definition of open-loop transfer function, we can express HG(s) as

$$HG(s) = \frac{-K_3 K_R (1 + T_\alpha s)}{\left[1 + \frac{2\zeta_A s}{\omega_A} + \frac{s^2}{\omega_A^2}\right]\left[1 + \frac{2\zeta_{AF} s}{\omega_{AF}} + \frac{s^2}{\omega_{AF}^2}\right]}$$

By going to the complex frequency domain we can rewrite the open-loop transfer function as

$$HG(j\omega) = \frac{-K_3 K_R (1 + j\omega T_\alpha)}{\left[1 - \frac{\omega^2}{\omega_A^2} + \frac{j2\zeta_A \omega}{\omega_A}\right]\left[1 - \frac{\omega^2}{\omega_{AF}^2} + \frac{j2\zeta_{AF}\omega_{AF}}{\omega_{AF}}\right]}$$

where care has been taken in the preceding equation to separate the real and imaginary parts. The magnitude and phase of the open-loop transfer function can now be expressed as

$$|HG(j\omega)| = -K_R K_3 \sqrt{\frac{1+\omega^2 T_\alpha^2}{\left[\left[1-\frac{\omega^2}{\omega_A^2}\right]^2 + \left[\frac{2\zeta_A\omega}{\omega_A}\right]^2\right]\left[\left[1-\frac{\omega^2}{\omega_{AF}^2}\right]^2 + \left[\frac{2\zeta_{AF}\omega_{AF}}{\omega_{AF}}\right]^2\right]}}$$

$$\angle HG(j\omega) = \tan^{-1}\omega T_\alpha - \tan^{-1}\left[\frac{\frac{2\zeta_A\omega}{\omega_A}}{1-\frac{\omega^2}{\omega_A^2}}\right] - \tan^{-1}\left[\frac{\frac{2\zeta_{AF}\omega_{AF}}{\omega_{AF}}}{1-\frac{\omega^2}{\omega_{AF}^2}}\right]$$

Therefore the open-loop gain (magnitude) and phase can be expressed in conventional units as

$$\text{Gain} = 20\log_{10} |HG(jw)| \quad \text{(db)}$$

$$\text{Phase} = 57.3 \angle HG(j\omega) \quad \text{(deg)}$$

Designers have found several useful ways of displaying open-loop data. One of these ways is a Bode plot in which the magnitude, expressed in db, and phase, expressed in degrees, are displayed versus frequency on a logorithmic scale. The preceding equations were programmed in FORTRAN in order to generate a Bode plot for the rate gyro flight control

system and the resultant program appears in Listing 2. Note that in this program we are incrementally updating the frequency logorithmically and then solving for the magnitude and phase. This program runs quickly because integration is not involved.

```
      REAL K3,KR
      DATA ZA,WA,K3,TA,ZAF,WAF,KR/.7,300.,-.2,2.,.1,10.,1.5/
      DO 10 I=2,160
      W=10**(.025*I-1)
      XMAG1=SQRT(1+(W*TA)**2)
      XMAG2=SQRT((1-(W/WAF)**2)**2+(2*ZAF*W/WAF)**2)
      XMAG3=SQRT((1-(W/WA)**2)**2+(2*ZA*W/WA)**2)
      GAIN=20*LOG10(-K3*KR*XMAG1/(XMAG2*XMAG3))
      PHASE1=57.3*ATAN2(W*TA,1.)
      PHASE2=57.3*ATAN2(2*ZAF*W/WAF,1-(W/WAF)**2)
      PHASE3=57.3*ATAN2(2*ZA*W/WA,1-(W/WA)**2)
      PHASE=PHASE1-PHASE2-PHASE3
      WRITE(9,*)W,GAIN,PHASE
10    CONTINUE
      PAUSE
      END
```

Listing 2 FORTRAN Program to Generate Open-Loop Bode Plot

Figure 5 presents the resultant Bode plot, using the data generated by the FORTRAN program. Here we can see that the gain (or magnitude) peaks due to the low airframe damping (ζ_{AF}=.1) and then is quickly attenuated due to the dynamics of the actuator. The phase and gain margins are 75 deg and 17 db respectively. This means that if the system phase is decreased by 75 deg or if the system gain is increased by 17 db the system will go unstable. We can also see from Fig. 5 that the gain and phase crossover frequencies are 60 rad/sec and 302 rad/sec respectively. If the system goes unstable because of a decrease in phase, its frequency of unstable oscillation will be the gain crossover frequency. If the system goes unstable because of a gain increase, the frequency of the unstable oscillation will be the phase crossover frequency.

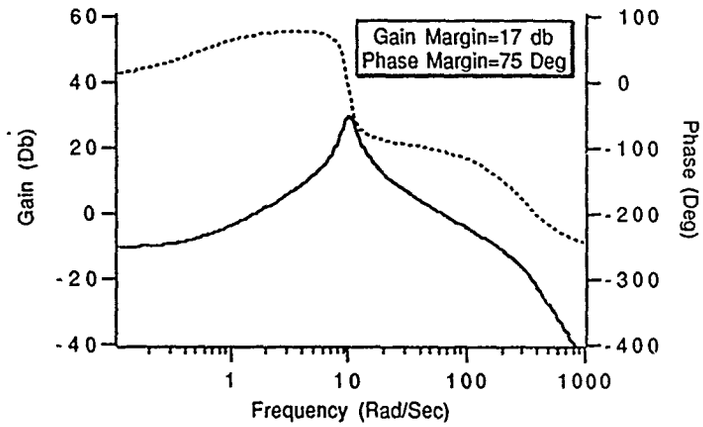


Figure 5 Bode Plot for Rate Gyro Flight Control System

Analysis and Verification of Open-Loop Results

The open-loop analysis of the previous section indicated that the system gain margin was 17 db. This means that if the gain K_R was increased by 17 db the system would go unstable. A gain increase of 17 db means that K_R must increase from 1.5 to 11 to destabilize the system. In other words,

$$20\log_{10}(K_{UNSTABLE}/1.5) = 17 \text{ db}$$

or

$$K_{UNSTABLE} \approx 11$$

In addition, the frequency response analysis indicated that the phase crossover frequency (i.e. frequency when phase is -180 deg) was 302 rad/sec. This means that if the rate gyro flight control system were destabilized by a gain increase, the system would oscillate at 302 rad/sec. Figure 6 shows that when the gain in the FORTRAN time-domain simulation of the rate gyro flight control system of Listing 1 is increased from 1.5 to 11 that the system breaks into growing oscillations at a frequency very close to the phase crossover frequency predicted by the frequency-domain analysis.

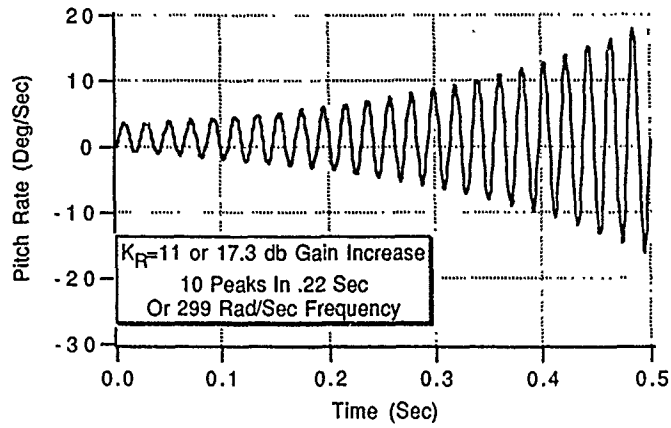


Figure 6 Flight Control System Goes Unstable If Gain Increased Too Much

Therefore this example demonstrates the relationship between the time or simulation domain and frequency or open-loop domain. Both time and frequency domain output information can easily be graphically incorporated into a microcomputer simulation.

We can also illustrate the concept of phase margin by first observing that an ideal delay can be represented by the transfer function

$$\text{DELAY} = e^{-sT}$$

Converting this representation to the complex frequency domain yields

$$\text{DELAY}(j\omega) = e^{-j\omega T} = \cos\omega T - j\sin\omega T$$

The magnitude and phase of the ideal delay is therefore

$$|\text{DELAY}(j\omega)| = (\cos^2\omega T + \sin^2\omega T)^{1/2} = 1$$

$$\angle\text{DELAY}(j\omega) = \tan^{-1}\left[\frac{\sin\omega T}{\cos\omega T}\right] = -\omega T$$

In summary, an ideal delay can be represented in the frequency domain as a function with unity magnitude and pure phase loss. The phase loss at 60 rad/sec (open-loop gain crossover frequency ω_{CR}) can be obtained from the preceding equation as

$$\text{DELAY PHASE LOSS} = -60T$$

Table 4 summarizes the phase loss of an ideal delay for various delay times.

T (sec)	Phase Loss (deg)
0.0	0.0
0.01	-34.3
0.022	-75.0

Table 4 Phase Loss From an Ideal Delay

We can see from Table 4 that a pure delay of .022 sec in the time domain results in a 75 deg phase loss in the frequency domain. Since the phase margin of the open-loop system (with the loop broken at K_R) is 75 deg, this means that if a pure delay of .022 sec were inserted in series with K_R, the system would go unstable and oscillate at a frequency of 60 rad/sec (open-loop gain crossover frequency). The rate gyro flight control time domain simulation of Listing 1 was modified to include a pure time delay of .022 sec and the system step response is shown in Fig 7. Here we can see that the system does go unstable at the predicted value of time delay and also oscillates at the predicted frequency.

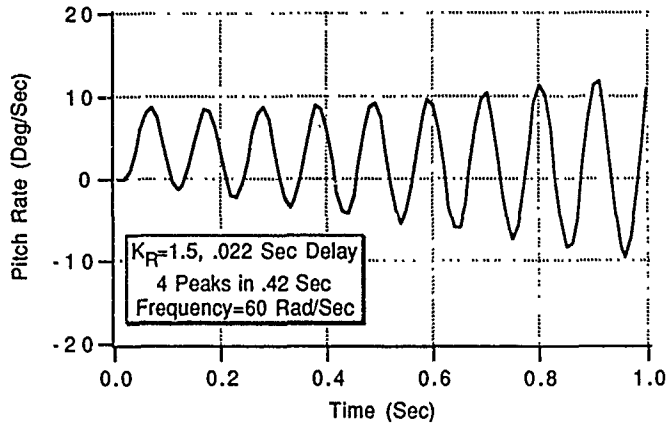


Figure 7 Decreasing the Phase Too Much Can Cause an Instability

The purpose of this section was to show the relationship, via an example, between the open loop frequency response and time domain simulation. The analyst uses both of these computerized methods of analysis for design because of the unique perspective that can be obtained from both the frequency and time domain. Both the time and frequency domain visualizations of the rate gyro flight control system can be presented simultaneously in different windows on a microcomputer screen so that the designer can rapidly iterate on acceptable values of autopilot gain.

Satellite Simulation

The purpose of this section is to provide a more dramatic example of how microcomputer based computation and graphics can be used to enhance the understanding of satellite dynamics. Let us begin by stating the satellite nonlinear differential equations. A convenient coordinate system for the simulation of a satellite is an Earth-centered Cartesian coordinate system as shown in Fig. 8. Since this coordinate system is fixed in inertial space (even though the earth rotates), all satellite acceleration differential equations can be integrated directly to yield velocity and position, without having to worry about Coriolis effects.

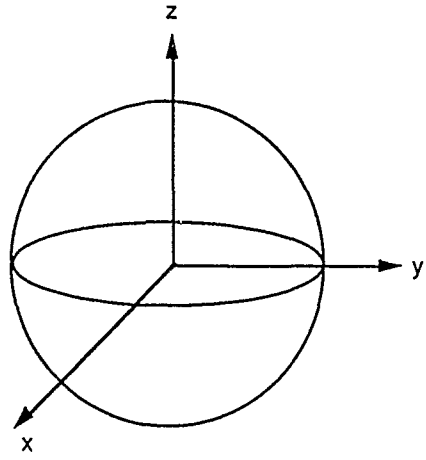


Figure 8 Earth-Centered Coordinate System

The differential equations describing the acceleration of a satellite in a gravity field can be derived from Newton's law of universal gravitation in the Earth-centered Cartesian coordinate system as[1,6]

$$\ddot{x} = \frac{-gm\,x}{(x^2+y^2+z^2)^{1.5}}$$

$$\ddot{y} = \frac{-gm\,y}{(x^2+y^2+z^2)^{1.5}}$$

$$\ddot{z} = \frac{-gm\,z}{(x^2+y^2+z^2)^{1.5}}$$

where x, y, and z are component distances of the satellite from the center of the Earth and gm is the gravitational parameter with value

$$gm = 1.4077 * 10^{16}\ ft^3 / sec^2$$

The velocity of a satellite in circular orbit is related to it's altitude according to[6]

$$V = \sqrt{\frac{gm}{a + alt}}$$

where alt is the altitude of the satellite, measured from the surface of the Earth, and a is the radius of the earth with value

$$a = 2.0926 * 10^7\ ft$$

Given the initial altitude, latitude and longitude of the satellite, we can express the initial location of the satellite in Earth-centered coordinates as

$$x(0) = (a + alt)\cos(lat)\cos(long)$$
$$y(0) = (a + alt)\cos(lat)\sin(long)$$
$$z(0) = (a + alt)\sin(lat)$$

where lat is latitude and long is longitude. The initial velocity components of the satellite in Earth-centered coordinates can be expressed in terms of the satellite velocity, location and inclination. For a satellite at a 90 deg inclination travelling in a prograde, ascending trajectory, the appropriate velocity initial conditions are

$$\dot{x}(0) = -V\sin(lat)\cos(long)$$
$$\dot{y}(0) = V\sin(lat)\sin(long)$$
$$\dot{z}(0) = V\cos(lat)$$

After integrating the satellite acceleration differential equations twice to get position, we must take Earth rotation into account. A coordinate frame moving with the Earth (x_e, y_e, z_e) is related to the inertial coordinate frame (x, y, z) according to

$$x_e = x\cos\omega t + y\sin\omega t$$
$$y_e = x\sin\omega t - y\sin\omega t$$
$$z_e = z$$

where ω is the rotation of the Earth with value

$$\omega = 360\ deg / 24\ hrs = 6.283185\ rad / 86400\ sec$$

The expressions for latitude and longitude can then be expressed in terms of the moving frame as

$$lat = \sin^{-1}\left[\frac{z_e}{\sqrt{x_e^2 + y_e^2 + z_e^2}}\right]$$

$$long = \tan^{-1}\frac{y_e}{x_e}$$

The FORTRAN code for a satellite in circular orbit, using the preceding differential equations and initial conditions, appears in Listing 3. From the source code we can see that the nominal case considered is that of the first 20,000 sec of a satellite travelling in a circular orbit at 1000 km altitude and 90 deg inclination. The acceleration differential equations, which are integrated using the second-order Runge-Kutta numerical technique, appear after statement label 200.

```
      REAL LATMDEG,LONGMDEG,LATM,LONGM,LATITUDEM,LONGITUDEM
      INTEGER STEP
      ALTMKMIC=500.
      LATMDEG=50.
      LONGMDEG=20.
      TF=20000.
      H=10.
      A=2.0926E+07
      GM=1.4077E+16
      W=6.283185/86400.
      T=0.
      LATM=LATMDEG/57.3
      LONGM=LONGMDEG/57.3
      ALTM=ALTMKMIC*3280.
      VS=SQRT(GM/(A+ALTM))
           'A+ALTM)*COS(LATM)*COS(LONGM)
      .     'ALTM)*COS(LATM)*SIN(LONGM)
      ZI      \LTM)*SIN(LATM)
      XM       SIN(LATM)*COS(LONGM)
      YMD- ..    (LATM)*SIN(LONGM)
      ZMD=VS*(  )(LATM)
101   CONTINUE
      IF(T>=(TF-.00001))GOTO 999
      XMOLD=XM
      YMOLD=YM
      ZMOLD=ZM
      XMDOLD=XMD
      YMDOLD=YMD
      ZMDOLD=ZMD
      STEP=1
      GOTO 200
66    STEP=2
      XM=XM+H*XMD
      YM=YM+H*YMD
      ZM=ZM+H*ZMD
      XMD=XMD+H*XMDD
      YMD=YMD+H*YMDD
      ZMD=ZMD+H*ZMDD
      T=T+H
      GOTO 200
55    CONTINUE
      XM=.5*(XMOLD+XM+H*XMD)
      YM=.5*(YMOLD+YM+H*YMD)
      ZM=.5*(ZMOLD+ZM+H*ZMD)
      XMD=.5*(XMDOLD+XMD+H*XMDD)
      YMD=.5*(YMDOLD+YMD+H*YMDD)
      ZMD=.5*(ZMDOLD+ZMD+H*ZMDD)
      XME=XM*COS(W*T)+YM*SIN(W*T)
      YME=-XM*SIN(W*T)-YM*COS(W*T)
      ZME=ZM
      LATITUDEM=57.3*ASIN(ZME/SQRT(XME**2+YME**2+ZME**2))
      LONGITUDEM=57.3*ATAN2(YME,XME)
      IF(LONGITUDEM>180)THEN
            LONGITUDEM=LONGITUDEM-360
      ENDIF
      ALTKM=(SQRT(XM**2+YM**2+ZM**2)-A)/3280.
      WRITE(9,*)T,ALTKM,LONGITUDEM,LATITUDEM
      GOTO 101
```

```
200     CONTINUE
        TEMPBOTM=(XM**2+YM**2+ZM**2)**1.5
        XMDD=-GM*XM/TEMPBOTM
        YMDD=-GM*YM/TEMPBOTM
        ZMDD=-GM*ZM/TEMPBOTM
        IF(STEP-1)66,66,55
999     CONTINUE
        PAUSE
        END
```

Listing 3 FORTRAN Satellite Simulation

By running the simulation of Listing 3 and projecting the results into longitude-latitude space in conjunction with a linear projection of a publically available world map data base[7] as a background, we can get a better graphical visualization. We can see from Fig. 9 that orbits do not overlap because of the rotation of the Earth. We can also see that in 20,000 sec the satellite went through three revolutions. The map provides important geographical context to the satellite simulation. Information missing from the linear mapping display of Fig. 9 is three-dimensional perspective. In addition, there appears to be confusion concerning the motion of the satellite at 90 deg latitude.

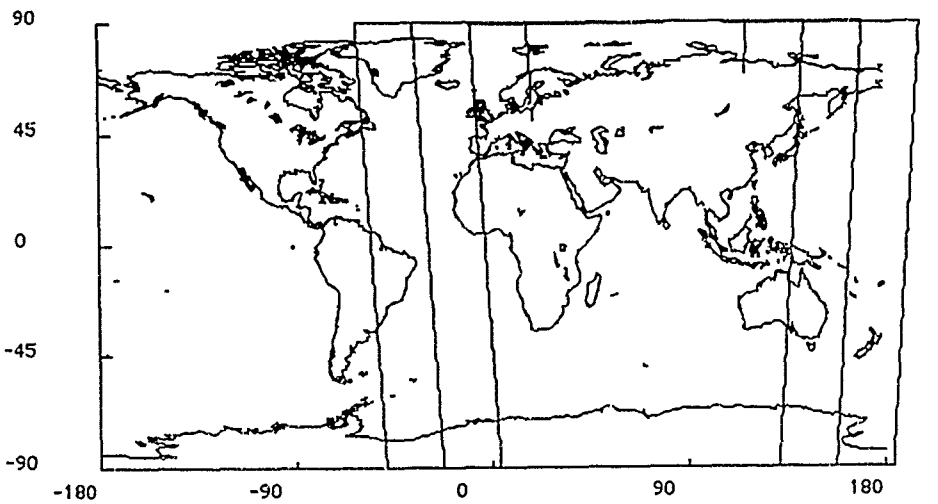


Figure 9 Ground Track of Satellite Motion Provides Geographical Context

Although orthographic mapping projections of the world are the least useful as maps because of the extreme distortion near the edges, they are useful in providing three-dimensional perspective on a two-dimensional microcomputer screen. For example, Fig. 10 with its mapping origin at 0 deg latitude and -45 deg longitude, provides an orthographic view of the same satellite trajectory of Fig. 9. The orthographic projection provides an excellent visualization for both the altitude and inclination of the circular satellite orbit. In addition, confusion concerning motion at 90 deg latitude in the linear display of Fig. 9 has been eliminated in the orthographic display. However, part of the trajectory is missing since an orthographic view can only show one hemisphere at a time.

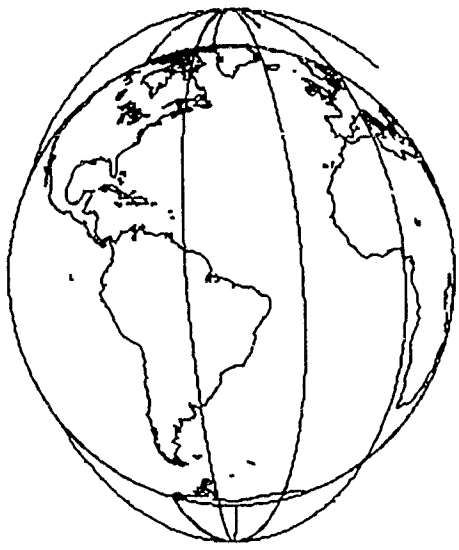


Figure 10 Orthographic View of Satellite Trajectory (Origin=-45 Deg Longitude and 0 Deg Latitude) Adds Perspective

By rotating the orthographic viewing angle we can obtain even more information about the trajectory. For example, if we want a view of the trajectory from infinity looking at the North Pole, we simply change the latitude origin of the map from 0 deg to 90 deg. The resulting North Pole orthographic view is shown in Fig. 11.

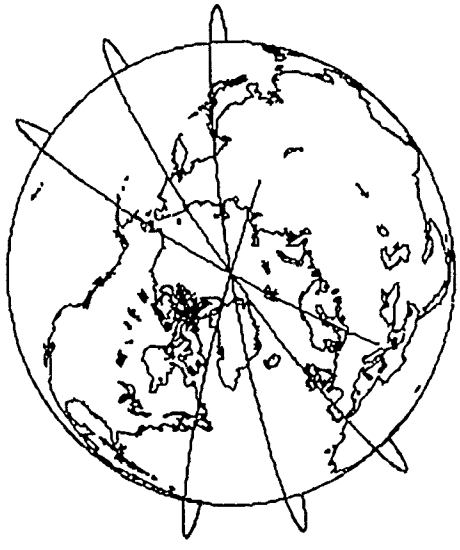


Figure 11 North Pole View of Satellite Trajectory (Origin=-45 Deg Longitude and 90 Deg Latitude)

We can see from Figs. 9-11 that microcomputer based graphics technology can add a new dimension to the visualization of trajectories.

Interceptor-Satellite Engagement Simulation

As a final example, let us consider extending the satellite simulation to include a strategic surface-based interceptor pursuing the satellite. Since this engagement simulation is more complex, easier ways of specifying large data sets and user options are required. In this scenario it is appropriate to borrow many of the user-interface concepts popularized by the Macintosh technology. For example, dialog boxes can be implemented as a "user-friendly" way of inputting data into a detailed engagement simulation. Figure 12 shows how the satellite orbital parameters can easily be specified with edit fields and buttons. The satellite location and inclination can be entered in the edit fields by use of an input pointing device known as a mouse. The type of orbit (i.e., prograde or retrograde) is specified by clicking on the appropriate button in the dialog box. When the user is satisfied with all the inputs, a simple mouse click on OK enters the data into the program. Recalling the dialog box from a menu, also controlled by

the mouse, allows the user to discover how the satellite orbital parameters influence interceptor performance

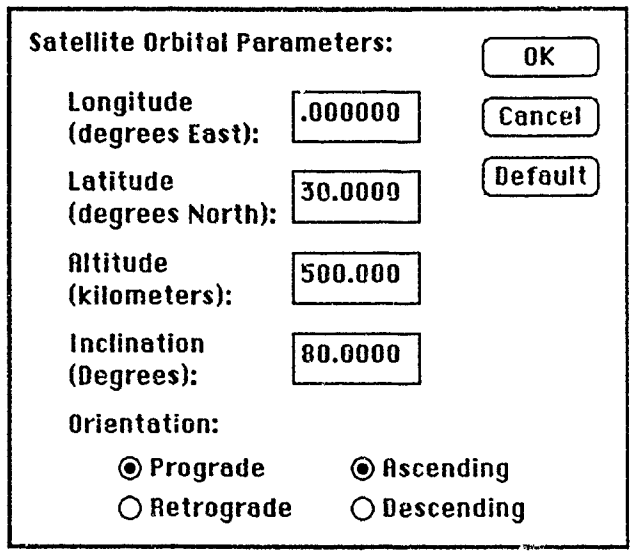


Figure 12 A Dialog Box Is A User-Friendly Way of Entering Data

The dialog box can also be used as a convenient way for providing the user with many complex options. For example, the use of buttons in the dialog box of Fig. 13 allows the user to choose between many sophisticated interceptor guidance laws. Edit fields are used to specify, in even greater detail, many guidance related parameters. Studies can be rapidly conducted in which the effectiveness of each guidance law is quantified.

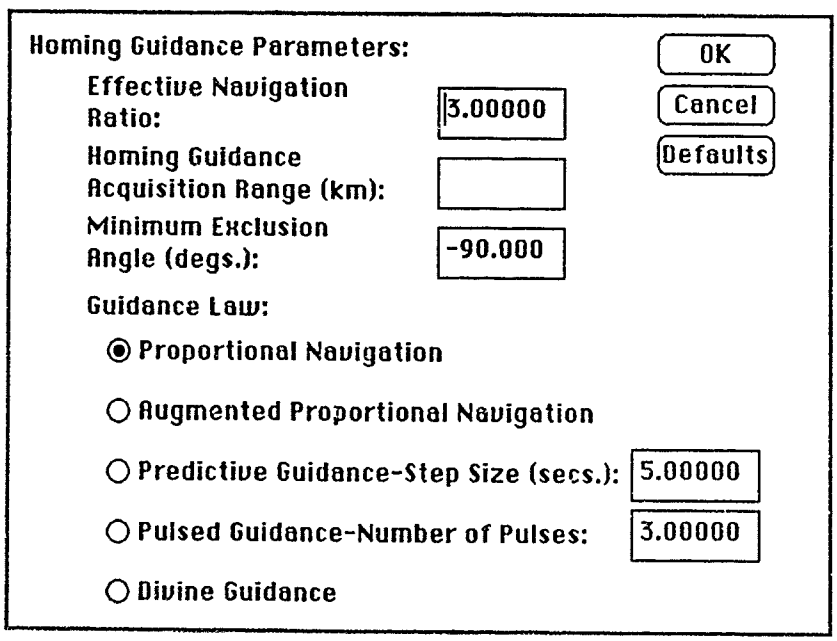


Figure 13 A Dialog Box Is A User-Friendly Way of Making Sophisticated Choices

Finally, after entering the data, the user needs a easy way to both visualize and understand the results of the simulation. Figure 14 presents a possible way of presenting some of the resultant data in different windows simultaneously. The "Ground Tracks" window presents a linear projection of a satellite (solid line) being pursued by a surface-based interceptor (partially dashed curve). A box at the top of the window presents simulation time, interceptor-satellite separation, the lateral divert required for this engagement, and interceptor altitude as the simulation is running. In order to convey perspective, an "Orthographic Projection" window simultaneously presents the same trajectory data. However, this time we get a better visualization of the three-dimensional aspect of the engagement. A "Global View" window provides a macroscopic view of the engagement using orthographic projection techniques. More precise altitude information concerning the interceptor and target can be found in the "Trajectories" window. We can see that the satellite is at constant altitude whereas the interceptor must climb to an altitude higher than the satellite and dive. The "Missile

Acceleration" window presents the required missile acceleration, which in this case was miniscule, to effect an intercept.

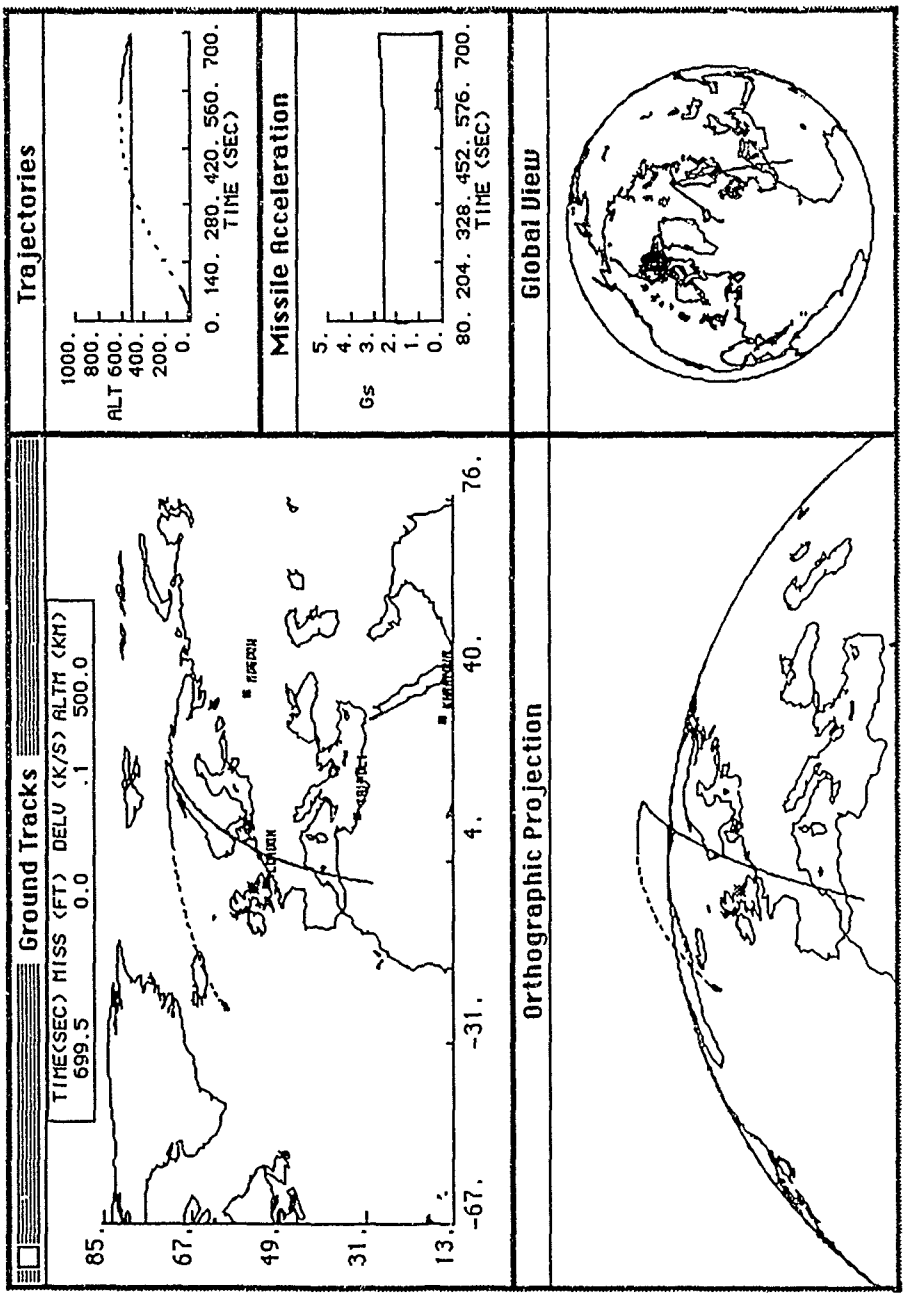


Figure 14 Simulation Data Can Be Presented Simultaneously In Different Windows

Summary

Several interceptor guidance system related examples have been presented. The paper first demonstrates that these examples can be made to work on microcomputers with CPU running times which are very attractive and turn around times (i.e. time for engineer to get the answer in a useful form) that are far superior to that offered by a time-shared mainframes. It is then shown how numerical output can be enhanced, in real time, with the graphics visualization technology which is currently available with microcomputers. Each of the examples demonstrates how the enhanced answers offer the designer a visualization which not only gives a deeper insight into the problem being solved, but in addition allows an engineer to rapidly iterate cases to get an acceptable design.

References

1. Zarchan, P., Tactical and Strategic Missile Guidance, Vol. 124, Progress in Astronautics and Aeronautics, AIAA, Washington, DC, 1990.

2. Zarchan, P. "Micros For Guidance and Control," Proceedings of AIAA Guidance and Control Conference, AIAA, Washington, D.C., Aug. 1987.

3 Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T., Numerical Recipes: The Art of Scientific Computation, Cambridge, University Press, London, 1986.

4. Bode, H. W., Network Analysis and Feedback Amplifier Design, D. Van Nostrand Company, Inc., Toronto, 1945.

5. Newton, G. C., Gould, L. A., and Kaiser, J. F., Analytical Design of Linear Feedback Controls, John Wiley & Sons, N.Y., 1957.

6 Bate, R. R., Mueller, D. D., and White, J. E., Fundamentals of Astrodynamics, Dover, New York, 1971.

7. Miller, R., and Reddy, F.,"Mapping the World in Pascal," BYTE, Vol. 12, Dec. 1987, pp 329-334.

A UNIFIED APPROACH TO SIMULATION SOFTWARE AND OPERATIONAL SOFTWARE

by
Dr. A. Mattissek
Fa. LITEF GmbH
Lörracher Str. 18
Postfach 774
D-7800 Freiburg i. Br.
Germany

Summary

Development of Avionic Software typically requires extensive simulation of avionic specific algorithms during system analysis phase. Traditionally these programs are written in FORTRAN. For the development of operational software the same algorithms are re-coded in assembly or a high level programming language.

This paper describes an approach where the "application software" is only coded once, using ADA as programming language, and the same code is then used for simulation as well as for the operational program.

To ease the translation from the existing FORTRAN simulation environment to ADA a FORTRAN to ADA translator was applied to automatically generate ADA versions of existing FORTRAN packages. The experience with such a translator is also presented.

1. Introduction

Standard definitions of the software development cycle such as that defined in the widely used DOD-STD 2167(A) are based on a phased approach, whereby the following activities are accomplished strictly sequentially with some allowed overlap:

- software requirements analysis
- top level and detailed design
- coding and unit testing
- formal qualification testing

The end product of this effort is, despite a bulky documentation, an operational program[1] in which three different types of software components may be identified, namely

1. the "application software"
   i.e. those portions of the code which implement the avionic specific mechanisations (e.g. algorithms for flight control, navigation, etc.).

2. the operating system
   which implements the low level, hardware related functions such as interface and device drivers or the task scheduling mechanism.

3. the built in tests
   typically comprising start up BIT, background BIT and system failure diagnosis.

Even though all three components are developed in parallel the methodology applied will differ. Our main concern is, of course, devoted to the application part of the operational program.

For avionic subsystems this part of the software typically comprises computations that require comprehensive investigations before formal software development can commence. The most valuable tool during this phase, which is generally referred to as the system analyis and design phase (figure 1), is a simulation program that simulates the avionic subsystem mechanisation as well as the environment. A minimal configuration of such a program is composed of the components

- input data generation
- avionic subsystem mechanisation
- output processing and performance presentation

More complex programs may be needed to simulate the complete flight control and autopilot loop or even the complete avionic system.

---

[1] We use the terminology "operational program" for the complete software embedded in the computer(s) of an avionic subsystem

Our development approach starts with the observation that in many cases the operational program and the simulation program have many functions in common, namely the "application software", i.e. the kernel of the avionic subsystem specific algorithms. From a different viewpoint we might consider the simulation program code of these function as a prototype of the application software part of the operational program.

Figure 2 quantifies the respective code portions for typical avionic subsystems under development at LITEF.

While traditionally simulation program and operational program implement the same application kernel in a different way, our attempt was to code and test this kernel only once and then use it for both types of software. This approach (called the "unified approach" here) offers great potential advantage:

By avoiding duplication of the coding and testing effort costs can be saved and the software development process can be shortened.

If we go the traditional way, deriving
a system mechanisation spec from the simulation code,
a software requirements spec from the mechanisation spec,
a design document from the software requirements spec and
the code from the design documentation,
new errors will be brought in during every phase. All of these errors are avoided in the unified approach.

Often parameter optimizations of the algorithms are ongoing while full scale software development has already started. Sometimes the definite choice of program parameters is determined through experiments performed with the completely integrated system in the laboratory or during field trials. For these cases it may be difficult to guarantee that the simulation program version and the operational program are kept consistent. This consistency problem is greatly reduced of both program share the same code kernel.

However, even with the advantages clearly visible, a fundamental technical question remains to be answered: Is it practicable to write programs that represent a fair compromise between the goals of simulation and of operational performance?

Simulation programs are designed and coded such that changes can be made with minimal effort. Thus they should be designed for maximum abstraction and transparency. Operational programs, on the other hand, are optimized according to constraints on CPU load and code size. Chapter 4 describes how this conflict was resolved for some particular projects.
For these projects ADA was chosen as programming language because ADA seemed to be powerfull enough to cover the needs of simulation and hard realtime software. But a transition to ADA introduces new risks:
FORTRAN simulation programs, in general, can be built on programs from similar applications and they can make use of the large number of existing FORTRAN program libraries. Mixing of ADA and FORTRAN code is risky and laborious. When converting to ADA it is therefore difficult to do so gradually and one is forced to convert the complete simulation program package in one step. Thus, a transition from FORTRAN to ADA might become a truly expensive venture for a sophisticated simulation environment. A potential solution to this problem is offered by FORTRAN to ADA translation programs.

When we decided to utilize a translator to our knowledge no experience with such a tool had at that time been reported, but there was a lot of general scepticism concerning the usefulness of such an approach. However most of the criticisms came from the viewpoint of ADA "purists" and seemed not to be directly applicable to the project described here.

Since the FORTRAN to ADA translator proved to be a useful instrument to generate a simulation environment in ADA in minimal time we include the discussion of the tool in this paper (chapter 3 and 5).

2. Project Description

At this point our experience with the methodology described here is based on two projects. Both are to develop attitude and heading reference systems.

One of the systems is an upgrade of an existing ship reference system primarily used for weapon stabilization but also serving as a backup compass unit.

The system comprises two MOTOROLA 68020 processors. Besides the free inertial functions the software employes a Kalman Filter to augment the AHRS with external reference information, primarily velocity as delivered by the ships log.

The project has been started in the mid of 1989. According to the standard phase model it is now in the transition from the preliminary design to the detailed design phase. The comlete code of the application software of one of the two processors has by now been developed employing the ideas described in this paper.

The other system under development essentially serves as an aircraft motion sensor unit. It provides angular rates, linear accelerations, attitude, heading and velocities to the

flight control system. Thus the software is classified as flight critical and encorporates the capability to detect, isolate and circumvent hardware failures. The system is based on a TMS 320C25 and a MOTOROLA 68020 processor.

Full scale development for this project started by the end of 1989. Even though only the Motorola 68020 processor will be programmed in ADA it is intended to develop a complete ADA simulation program which covers the functions of both processors. So far, the FORTRAN to ADA translator has been used to generate an ADA version of the application software allocated to the TMS processor as well as portion of the 68020 application code.

Both projects follow the development standard DOD-2167. The tool TEAMWORK with its elements
- structured analysis
- structured design
- ADA structured graphs

is applied for requirements analysis and design. While the TELEGEN II ADA compiler was used for the ongoing project phases, a cross compiler from System Designers Ltd. will be employed for final target code generation.

3. FORTRAN to ADA Translator

When the idea of applying a translator was concieved no adequate commercial tool was available. We therefore developed such a translator of our own.

The aim was not to write a universal tool to be applicable to virtually every FORTRAN program but rather to write a tool well suited to the our applications, dropping FORTRAN elements not important for the FORTRAN libraries used. Instead some effort was spent to introduce higher level ADA program features.
The translator has been written in PASCAL and comprises about 8000 lines of executable code. It takes a complete FORTRAN program contained in one file and a file of translator commands to generate various ADA program files as specified in the command file. Most translator commands are intended to support ADA program structuring. In essence these commands help to define ADA packages and the visibility of procedures and variables.

Packages are defined through an ADA like syntax, e.g.

```
PACKAGE my_package is

     PROCEDURE  proc_1;
     FUNCTION   fun_1;
          .
          .
END my_package;
```

Only one level of packages is allowed i.e. package nesting is not possible. A default package may be declared to comprise all those procedures not explicitely mentioned within a package definition. Within every package the procedures are sorted according to static program structure. Package specifications and bodies are written to separate files. The file names are automatically complemented by numbers indicating compilation order.

Other translator commands make it possible to define

- whether a procedure should be specified within a package specification or hidden in a package body.
- whether a procedure should be defined locally within another procedure such that the procedure is only visible to calling procedures.
- the maximum nesting level of procedures

To support an early identification of problems the translator analyses the static structure of the FORTRAN program and records structuring commands to the translator which are not compatible with the FORTRAN program structure, e.g. commands which lead to

- recursive use of packages
- procedures not visible to calling procedures

When such a problem is observed the translator reports it in an error message and applies a default strategy for program structure.

The translator commands for the handling of FORTRAN COMMON blocks allows similar flexibility as that provided for procedures. Thus one can define whether the COMMON block variables shall be defined
- in a dedicated package
- in the specification of a package where procedures using these variables are located
- on the lowest level possible within a package body

Again consistency of commands with program structure is checked by the translator and conflicts reported and resolved.

The barely constrained flexibility of FORTRAN to pass inconsistent procedure parameters imposes particular burden on the translator. FORTRAN library routines often make use of the fact that for arrays the dimensions of formal and actual procedures need not be the same. The translator checks for these discrepancies.

Whenever a procedure call with inconsistent array dimension in the parameter list is detected the translator generates a separate procedure which copies the actual parameters to the formal ones such that a syntatically correct ADA program is generated. Thus these auxiliary parameter copy routines tend to become very bulky. All arrays are defined to be of unconstrained array type.

For COMMON blocks no attempt was made to support the flexibility of FORTRAN. It is only checked whether the COMMON block variables are of same type. If this is not the case an error message is generated.

4 Experience with Unified Approach

We evaluated the impact of the unified approach from two viewpoints, namely by looking at top level program structure and at low level code. As ADA was particularly designed to help manage software complexity the top level viewpoint is of course more essential.

Top Level Design

Fig. 3 and 4 show the top level structure of operational and simulation software respectively. The examples are taken from ship reference system development project described in para 2. To make the basic concept more transparent, only the more simple program for one of the two processors is shown and BIT as well as operating function are left out. Besides the fact that both programs are structured according to the operating cycle of the functions to be executed, both approaches are different.
The operational software employes ADA tasking to implement the cycles. All functions with the same frequency are put together into the same task.

While real time interrupts arrive with 2048 Hz these interrupts are filtered by an interrupt service routine such that only each fourth interrupt is passed to the run time executive. Thus the fastest ADA task runs with 512 Hz cycle frequency. The slowest cycle (not shown) has 1 Hz repetition rate.

The simulation program, on the other hand, need not fulfill any real time requirements. Therefore, to ease debugging and testing, ADA tasking is avoided completely. The "main program" just calls the procedures cyclically.

While the unifying approach is not reflected on the highest design level, it is fully applied on the next lower design level. This level uses (non nested) ADA packages to implement program structure. Figure 5 shows the hierarchy of the main packages which contain the application part of the software for the single processor. Here, the hierarchy is defined by the ADA use clause. The same packages are used in the simulation program and in the operational software.

Thus the design follows the idea that the commonality between operational software and similation software should be implemented through a common pool of packages.
The design decision to use packages rather than tasks to encapsulate logically related functions stems from the fact, due to CPU load constraints, only very limited use of tasking is affordable. Therefore essentially only one task is spent for each cycle frequency.

For the particular application considered the concept also supports software re-use from project to project. On the top level as shown in Fig. 3 and 4 the design will depend on timing requirements imposed on the output signals. But on the package level all system dependent specifications have been pooled in the three packages CONSTANTS, TYPES, CALPROM. Through modification of these three packages the software may be adapted for any AHRS type system to constitute a complete set of free inertial navigation functions.

If one thinks about object oriented design, this approach is constrained to passive objects only. However, for embedded systems with less than 10 000 lines of executable ADA code of the application software this limitation seems to be quite adequate. In particular for formal V & V activities, as required for flight critical software, the clean and simple structure and the limited use of ADA tasking is advantageous.

Low Level Code

On the lowest code level the basic conflict between functional abstraction and code efficiency, with respect to CPU load, must be solved. For the systems considered where, due to weight and volume constraints, an additional CPU can not be afforded so duty cycle minimization is given very high priority.

But minimizing the computational effort does not necessarily lead to poor code. To illustrate this point we take a Kalman Filter as an example, an algorithm that is frequently found in avionic applications.

Fig. 6 shows an ADA version of the complete basic formula. All variables involved are either of vector or of matrix type. The operator overloading capability is used to keep the code very compact.

In a brute force approach one would implement the matrix and vector operations with the help of unconstrained array types as shown for the multiplication operator. In this case about 65% of the CPU time are spent just to evaluate the multiplications in the expression

```
PHI * P * TRANSPOSED(PHI)
```

The PHI matrix is typically sparse and its dimension and structure is defined in an early phase of the project. But PHI being sparse implies in our case that 80% of the CPU time is wasted to multiply elements of the P matrix by zero numbers.

However there is a simple and safe way to overcome this absurdity. We use a simple ADA program that reads the PHI matrix from a file and, based on the observed structure, automatically generates a suited ADA procedure for the matrix multiplication (see fig. 6: time optimal solution).

In this procedure nonzero elements are multiplied only. Furthermore index evaluation is shifted from run time to compile time. The automatically generated procedure is certainly not elegant ADA but still transparent, safe and very efficient. In fact this matrix multiplication needs only 7,5% of the CPU time as compared to the unconstrained array approach. Exploiting similar techniques further leads to a Kalman Filter implementation that is efficient as well as transparent.

In general, we found that a reasonable compromise between functional abstraction, as desired for the simulation program, and efficiency, as required for the operational software, is always easy to find in program segments which include a lot of real arithmetic. For these segments the real operations, whether executed by a numeric coprocessor or not, dominate the execution time.

Program segments without real arithmetic may require optimization steps which lead to less readable and flexible code.

5. Experience with FORTRAN to ADA translator

The requirements imposed on simulation software are, in general, less formal and explicit that those for operational software. In discussing the characteristics and limitations of the automatic FORTRAN to ADA translation process we also consider the generation of operational programs even though the requirements for the operational software are harder to meet. Please note that most topics considered are not tool specific but characterize the automatic translation process in general.

From any software generation process we must require that it produces code that is correct, transparent and efficient. Furthermore a useful tool should support top level design elements as well as detailed design. This implies, at least, that the tool may not constrain the flexibility and quality of the design. Furthermore the tool should actively support hirarchical program structure and should be able to map the decisions of the designer, concerning visibility of data and operations, to the resulting program structure.

As the tool actually employed does not support ADA tasking the design had to rely on ADA packages and subprogram nesting as major elements to express the structure of operations and data. For these the major constraints of the tool are that

- only one level of packages can be defined
- the nesting of procedures can not be commanded directly, the tool nests the procedures according to predefined rules

For the particular application we found:

Incorporation of tasking is not required for the code translator. Experience from other projects tells us that for hard real time applications with limited CPU resources the effective number of ADA task will be very limited. Thus the specification of ADA tasks can well be defined manually.

The missing ability of the tool to nest packages did not really confine the software design. In fact for programs of comparable size no definite need for nested packages can be seen. Nesting of procedures must be performed with care considering logic coherence, data visibility, code length etc. The rules applied by the translator are too schematic. Thus it may be practicable to restrict the translator to generating purely flat structures (no nesting) and to nest the generated ADA procedures manually after FORTRAN to ADA translation.

Despite the particular design aspects most of our findings reaffirmed the well known perception that program quality only remotely depends upon the language applied. Good FORTRAN code was translated to good ADA code while poor ADA code does not only produce poor ADA code but may also cause the translators to give up.

Particular problems which led to illegal ADA code were caused by:

- Inconsistent formal and actual procedure parameters, e.g. inconsistent index ranges of arrays.
- Inconsistent definition of variables of the same FORTRAN COMMON block within different procedures.
- Side effects in function, e.g. function parameters which are declared OUT or INOUT in the ADA program.
- Inadvisable use of GOTO statements which leads to illegal ADA code, where control is transferred into IF, CASE or LOOP statements.

For code sequences with little control portions the automatically derived ADA code is hardly distinguishable from manually generated code. Therefore mathematic library functions like matrix algebra or statistical analysis routines a particularly suited for automatic translation. On the contrary, for the I/O portions of a program the generated ADA code even though syntactically correct, is of little use. File opening and closing is handled differently in both languages, some FORTRAN FORMAT instructions have no direct equivalent in ADA. Basically the I/O concepts of FORTRAN and ADA are too different to allow a simple one to one translation, I/O modules had to be re-coded manually.

Another class of problems was identified when results of simulation runs of the FORTRAN and the corresponding ADA program version were compared against each other. For almost any initial attempt the results were different. When tracking down the source of the discrepancies it was found that FORTRAN programmers,' even though they believed they wer writing machine and compiler independent FORTRAN 77 compatible programs, very often relied on certain machine or compiler dependent features. A typical example is the use of the SAVE command. Even though it is not guaranteed by the language definition, most FORTRAN compilers save the values of local variables of a procedure even when the SAVE command is not used. Thus most FORTRAN programmers forget to use the SAVE command in cases where the values of local variables should be saved.

The original version of the FORTRAN to ADA translator considered variables not mentioned in a SAVE command to be truly local. After having spent a lot of debugging time with variables of undefined value the option SAVE_ALL_VARIABLES was included in commands list of the FORTRAN to ADA translator. This command causes the translator to generate ADA programs where all local variables are specified on package level. Obviously this is only a remedy for the debugging phase. For final program versions the SAVE command must be inserted into the FORTRAN source, where applicable, and the program retranslated.

Besides tasking and packages the elements

- generics
- exceptions
- operator overloading

are often quoted to be ADA specific advantages. For the automatically generated ADA code none of these elements were supported. But, even though program transparency may be improved somewhat through these features, they are considered less important for the given application.

Concerning safety aspects a more fundamental disadvantage of the automatic translation process is the inability to make full use of the strong typing concept of ADA.

A severe limitation is of very simple nature: The six character constraint of the FORTRAN program identifier names is inherited to the ADA program. To enhance code readibility and to conform to coding standards it may be necessary to revise the ADA program using self explanatory names.

Besides the basic conceptual findings a positive unexpected side effect was experienced when applying the translator: Through its independent FORTRAN source checking and problem reporting capability, the tool uncovered errors and deficiencies in the (already fully debugged) FORTRAN programs, e.g. FORTRAN syntax errors not notified by the FORTRAN compiler or variables with missing assignment of values.

6. Conclusions

For the project described we started with the requirement that the application part of the software should only be written once and that the same code should be used for simulation and analysis as well as for the embedded operational program.

This approach led to a library of common ADA packages which cover the complete application software. The commonality between simulation program and operational software was not apparent at the scheduler level.
Due to the limited CPU resources of the embedded system the ADA program does not exploit the full richness of ADA, but the following advantages were observed:

The more formal requirements imposed on the operational software enforce a more disciplined approach to simulation software development. The ADA simulation program was of considerable better quality (with respect to modularity, understandability,

changeability) than comparable FORTRAN predecessors. Machine independance i.e. portability also improved.

The unified approach removed the consistency problem between operational software and simulation software, thus making configuration control much easier.

The approach helps to reduce the barrier between system analysts and software engineers. Errors introduced through poor documentation of the algorithms to be implemented or misinterpretation of these requirements were avoided.

Reliable figures for CPU load and memory requirements being available in an early phase of the project reduces development risk significantly.

Even though by now our experience with the unified approach does not go far enough to quantify schedule reductions and development cost savings we feel that our initial exceptation have been confirmed. This approach is a measure to increase software development efficiency considerably.

Furthermore with the improved efficiency of code generated by future ADA compiler generations the limitations imposed on the operational ADA code will become a less important topic, such that the still existing conflict between program abstraction and code efficiency is likely to become insignificant.

Our experience with the FORTRAN to ADA translator may be summarized as follows:

At low level good FORTRAN code leads to ADA code of acceptable quality. For higher level program structure the translator used supports ADA packages and nesting of procedures. This is a reasonable compromise between the abilities of ADA and the flat structure of FORTRAN programs.

The translation process can be recommended for all standard FORTRAN programs, such as mathematical and engineering libraries, where the software is very likely to remain unchanged. However, effective use of the translator requires some experience. Thus application of the translator only pays off for programs of a certain minimal size. The break even point for cost will be somewhere in the order or
10 000 lines of executable FORTRAN code.

For operational software a translator could also help to cut down development time. But for this application the generated ADA code will, in general, be more of a prototype than a final program version.

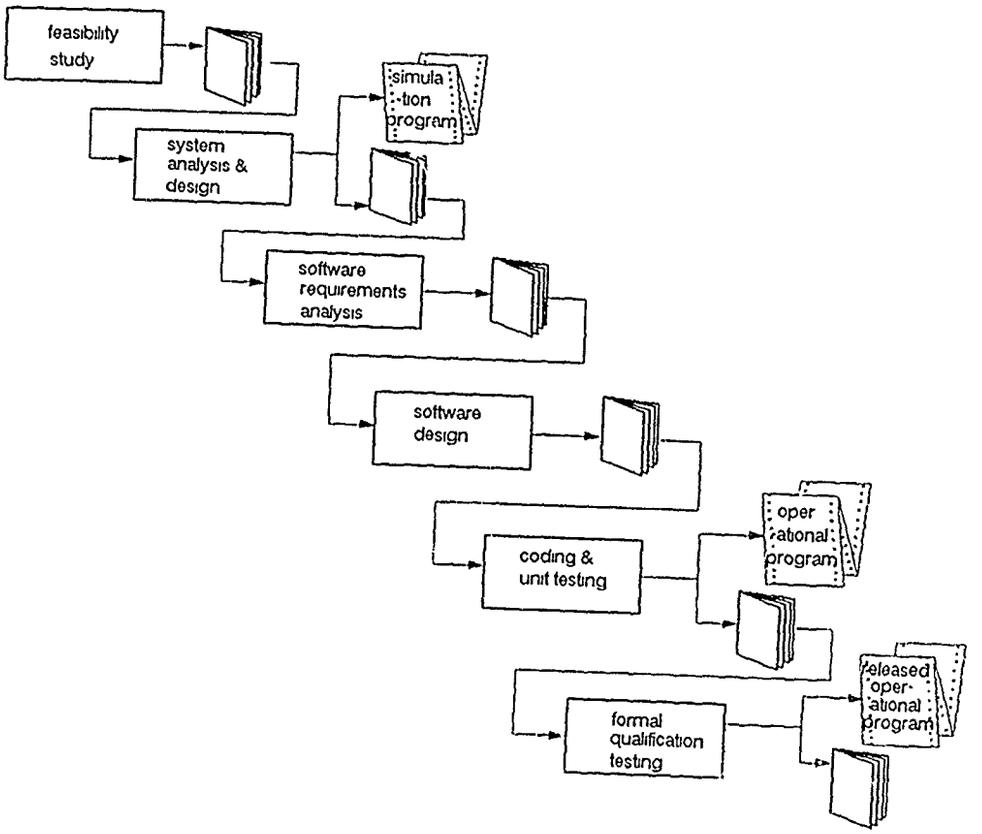


Fig. 1 : Software Development Phases

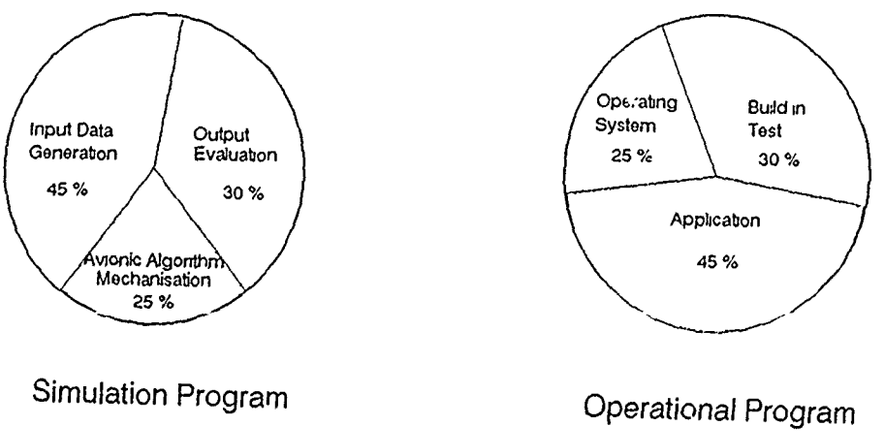


Fig. 2 : Code Breakdown

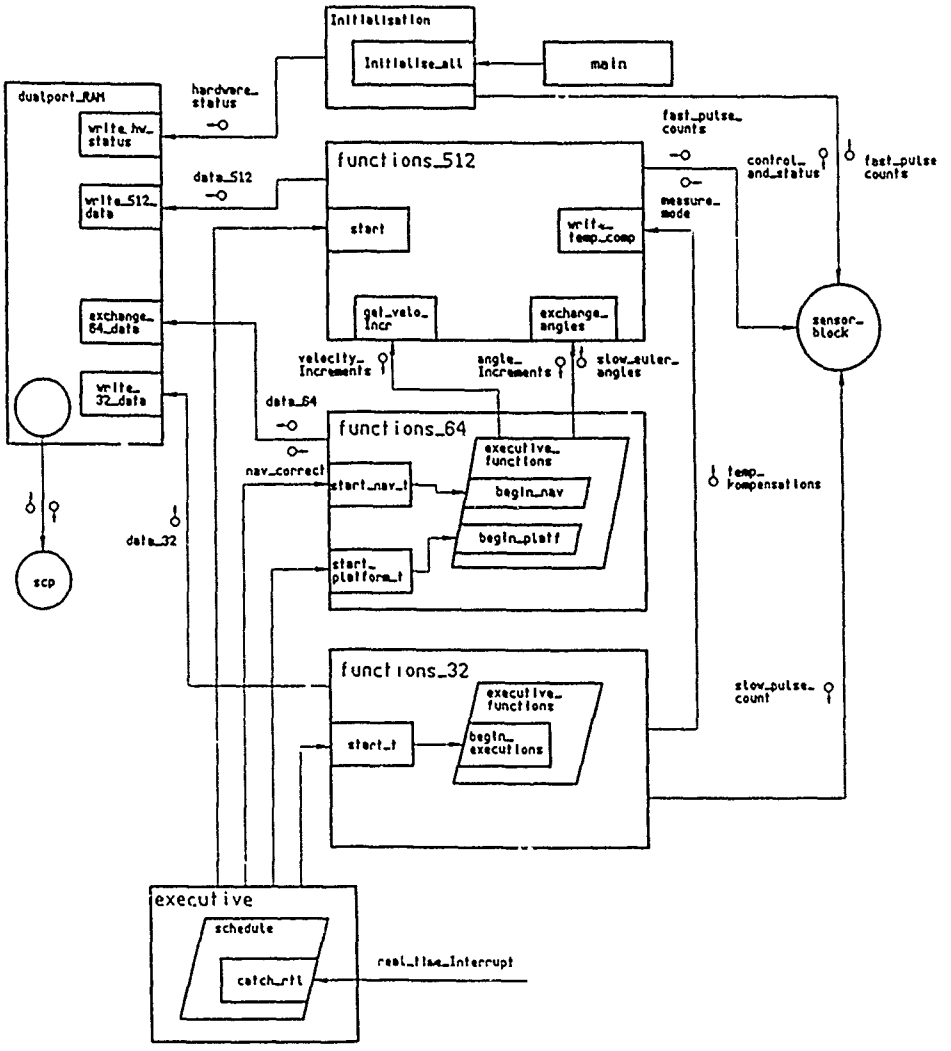


Fig. 3 : Operational Program Structure

```
procedure Insim is

Sensor_Period       : constant natural :=   1;  -- 512 Hz
Navigation_Period   : constant natural :=   8;  --  64 Hz
-- definition of longer periods

begin

   Time_Loop:
      for Cycle_Count in 0..Cycle_End loop

         if Cycle_Count mod Sensor_Period = 0 then
         -- call senquence of sensor related procedures
         end if;
         if Cycle_Count mod Navigation_Period = 0 then
         -- call senquence of quaternion integration procedures
         end if;
         if Cycle_Count mod Navigation_Period = Navigation_Period/2 then
         -- call senquence of navigation procedures
         end if;

         --  execution of slower cycles

   end Time_Loop;

end Insim;
```

Fig. 4 : Simulation Program

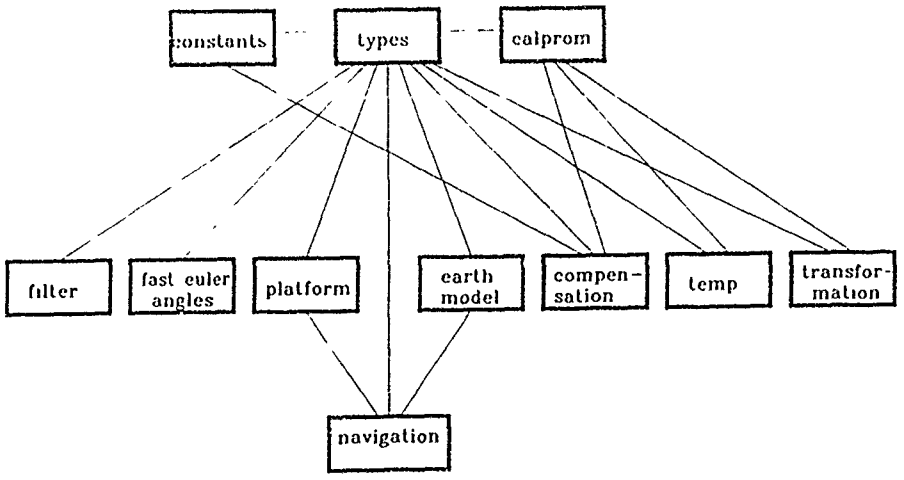


Fig. 5 : Common ADA Packages

```
FUNCTION "*"(A,B:MATRIX) RETURN MATRIX IS        -- brute force solution
   C : MATRIX(A'RANGE(2),B'RANGE(1));
BEGIN
   FOR I IN A'RANGE(1) LOOP
      FOR J IN B'RANGE(2) LOOP
         C(I,J):=0.0;
         FOR K IN A'RANGE(2) LOOP
            C(I,J):=C(I,J)+A(I,K)*B(K,J);
         END LOOP;
      END LOOP;
   END LOOP;
   RETURN C;
END "*";

FUNCTION "*"(A,B:M_14_14) RETURN M_14_14 IS      -- time optimal solution
   C : M_14_14;
BEGIN
 C(1,1):=A(1,1)*B(1,1)+A(1,3)*B(3,1)+A(1,5)*B(5,1)+A(1,6)*B(6,1)+A(1,7)*B(7,1);
 C(2,1):=A(2,2)*B(2,1)+A(2,3)*B(3,1)+A(2,4)*B(4,1)+A(2,6)*B(6,1)+A(2,7)*B(7,1);
 -- evaluation of all other components
 C(14,14):=A(14,13)*B(13,14);
 RETURN C;
END "*";

-- --------------  complete Kalman algorithm  ------------------------------
   -- covariance matrix
   P_EX := PHI*P*TRANSPOSED(PHI)+Q;
   H_P  := H*P_EX;
   K    := TRANSPOSED(H_P)*INVERSE(H_P*TRANSPOSED(H)+R);
   P    := P_EX-K*H_P;
   -- state vector
   X    := PHI*X+K*(Z-H*PHI*X);
-- ------------------------------------------------------------------------
```

Fig. 6 : Kalman Filter Example

A SIMULATION STUDY FOR ANALYSING PILOT'S RULE-BASED BEHAVIOR

Dr.-Ing. Bernhard Döring
Forschungsinstitut für Anthropotechnik (FAT)
Neuenahrer Straße 20
5307 Wachtberg-Werthhoven
F.R.of Germany

SUMMARY

In modern highly automated aircraft the pilot still plays an important role. He has, e.g., to monitor automated flight processes and to compensate malfunctions of the equipment. If he is trained to identify work situations and consciously relate them the appropriate action sequences, he exhibits rule-based behavior. To permit this type of training, work situations and appropriate action sequences have to be known in advance and the cockpit-interface must be designed to facilitate them. A simulation study has been accomplished to establish the task knowledge required by the pilot to make a correct automated landing approach and to identify information flow requirements for the pilot-cockpit interface. This paper describes the various steps of the study, starting with the definition of the problem and the development of a functional model. The model comprises a description of pilot behavior in terms of IF-THEN rules of a production system and of aircraft flight processes in the form of difference equations. To implement this model the simulation language SLAM was applied. SLAM elements used are explained. The resulting simulation generates the behavior of the pilot and the aircraft during the approach. Methods for validating the model are discussed. Finally, simulation results were analysed concerning information transmitted to the pilot and his control outputs.

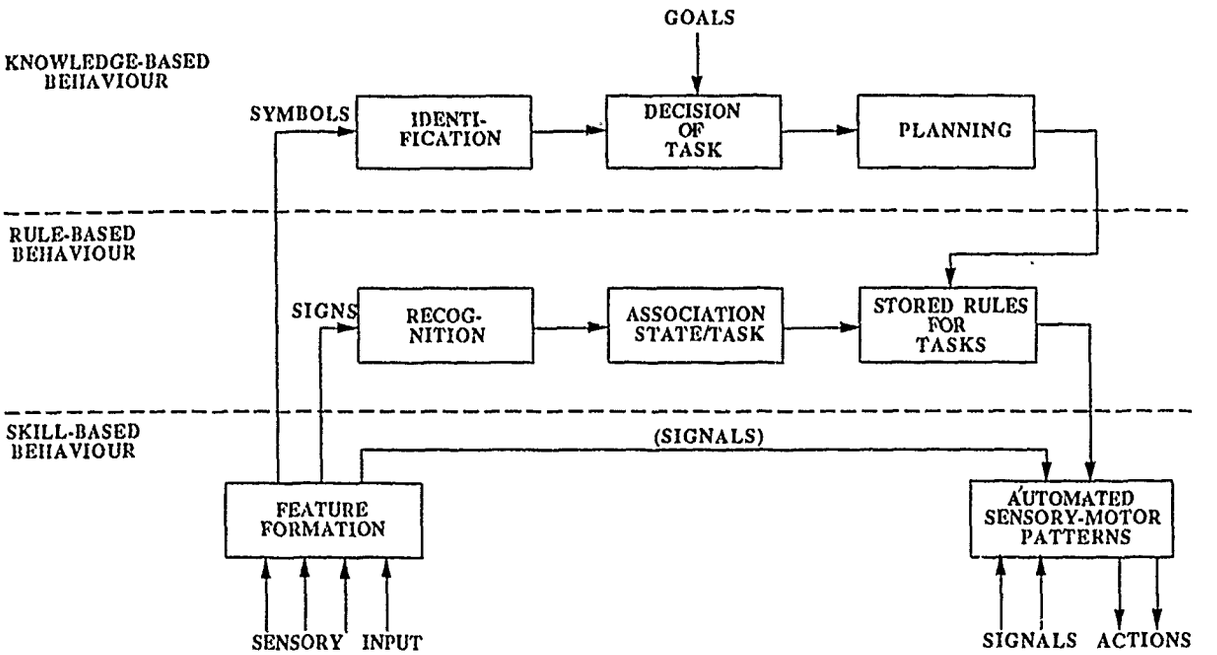
1. INTRODUCTION

Many system functions in modern civilian and military aircraft are performed automatically. But the pilot still plays an important role in those systems. He has to monitor, supervise, and control automated processes. The pilot is further responsible for detecting and diagnosing equipment malfunctions and operationally compensating for them. Because of the highly sophisticated technology currently being applied in system design, the pilot's performance may be limited in those aircraft with poor ergonomics design. To enable the pilot to perform tasks correctly during system operation, aircraft have to be designed with characteristics that match human abilities and capabilities. To have an impact on system design, special attention has to be given to human functions as early as possible during system development while it is still possible to influence major design decisions. To identify and analyse pilot activities and their impact on successful aircraft operation in the development phase, digital computer simulation has proven to be a powerful tool. Advanced simulation languages are very helpful in that analytic process as will be shown.

A simulation study will be described in which pilot behavior and aircraft processes during an automated landing approach were analysed and modeled. During that approach the pilot applies stored rules or procedures which he has acquired from training. He perceives information on aircraft states and approach situations in his environment. He relates that information to actions which he can apply to act on those states and situations. To describe the pilot behavior necessary for successfully performing those activities a scheme which Rasmussen [1],[2] has developed for categorizing operator behavior in process control can be applied. According to that scheme, performance behavior can be categorized as either skill, rule, and knowledge based (Fig.1).

The skill-based behavior represents sensory-motor activities of a pilot that are largely controlled unconsciously and performed automatically. This behavior is based on smooth, quasi-automated, and highly integrated behavior patterns that are acquired by intensive training and frequently repeated performances. The perceptual motor system acts as a continuous control system. For this control sensed information is perceived as time-space signals. Flying an aircraft manually is a typical example of this behavior type.

Rule-based behavior appears when the pilot accomplishes a sequence of actions that are controlled by a stored rule or procedure in a familiar work situation. The procedure may have been derived empirically during previous situations by means of training, observation, or instruction. The pilot consciously associates each of these situations with a distinct action sequence. He knows the relation between a situation and the belonging action se-



source: Rasmussen, 1983

Fig. 1: Levels of performance of skilled human operators [1]

quence explicitly and can describe it with an IF-THEN rule. At this behavior level, the perceived information is typically defined as a sign when it serves to activate or modify predetermined actions. Signs refer to situations by convention or previous experience. For example, if a certain situation appears which is defined through a distinct position during the landing approach, then the pilot activates the flaps of the aircraft to reduce its speed.

When in an unfamiliar work situation, e.g., in the case of an emergency, the pilot does not know any rule for controlling his environment, his performance must move to a higher conceptual level at which a goal-controlled knowledge-based behavior takes place. In such a situation, the pilot can explicitly formulate a goal, based on an analysis of the environment and his overall objectives related to mission fulfillment and system safety. Then a useful plan is developed by considering plan alternatives, checking the effectiveness of each alternative against the specified goal, and selecting the most appropriate alternative. At this behavior level, the pilot must represent the internal structure of the aircraft by a mental model which may take different forms. To be useful for causal functional reasoning in explaining unfamiliar situations, information must be perceived as symbols that refer to concepts related to functional properties and their internal conceptual representations.

For the simulation study an automated landing approach was chosen because the required pilot behavior for performing the approach can be categorized as rule-based behavior which is typical for highly automated man-machine systems (MMS). Monitoring, supervisory, and control tasks of the pilot during the approach are mainly based on predetermined procedures which he has learned during training. Therefore, he is familiar with such approach situations.

The goal of the study was the development of a simulation supported technique which can be used in early system development for determining required knowledge of the pilot in rule-based situations and human engineering design and arrangement requirements for the cockpit-interface. The study was accomplished according to the steps of the digital computer simulation. These steps can be arranged in a logical sequence that is described in detail in the literature, e.g., in [3],[4],[5]. Before describing the simulation study a short introduction to those steps will be given.

2. STEPS OF A SIMULATION STUDY

Generally, a simulation study proceeds in a logical sequence of individual steps during which conceptual and computerized models of the considered system are developed and simulation outputs and analysis data are generated. Also the study to be described proceeded in

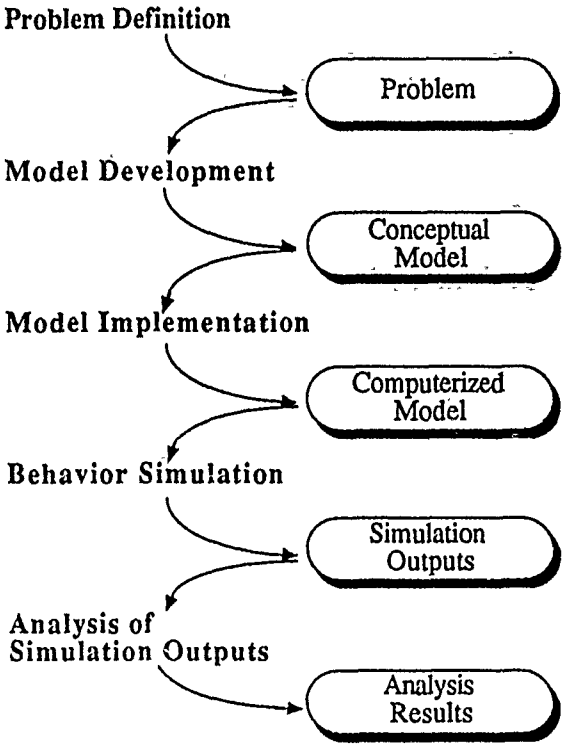


Fig. 2: Steps of a simulation study

this way. The essential activities necessary to produce the mentioned models and data are (Fig.2):

- Definition of the Problem,
- Development of the Model,
- Implementation of the Model,
- Simulation of System Behavior, and
- Analysis of Simulation Outputs.

Generally, a simulation study starts with a clear definition of the problem to be solved. The definition comprises a detailed description of the system to be studied as well as a formulation of problem objectives to be solved. Boundaries of the system and the level of modeling details are established according to those objectives. Problems which we considered in our study were related to a man-machine system which comprises the pilot as human element and aircraft subsystems as machine elements. Therefore, the problem definition has to represent a description of those system elements, i.e. of the pilot and the aircraft, including system performance measures such as task relevant data as problem solving objectives.

Once an initial problem definition is formulated, the development of the model begins. Developing a model of the considered system means describing relevant system elements and their behaviors excluding unnecessary details. The amount of detail which has to be considered should be based on the purpose for which the model is being built. During development of the model, input and performance data of considered system elements have to be defined and acquired. This process requires redefinition and redesign. Typically, the entire model building approach is performed iteratively. The final result of the model development phase is a mathematical-logical representation of the system, called a conceptual model [6], which can be exercised later in an experimental fashion on a digital computer.

After developing the conceptual model the next task is the implementation of the model on a digital computer. The implementation translates the developed conceptual model into a simulation program which is called the computerized model of the system under consideration [6]. In implementing a model, it is necessary to carefully select the computer language. Although a simulation model can be programmed using a general purpose language, e.g. PASCAL, there are distinct advantages to using a simulation language. In addition to the savings in programming time, a simulation language also assists in model formulation by providing a set of concepts for articulating the system description. In this study the simulation language SLAM (Simulation Language for Alternative Modelling) was used. Details of this language are described in chapter 3.3. An important part of the implementation is the verification of the simulation program. The verification task consists of determining that the computerized model represents the conceptual model within specified limits of accuracy.

When the simulation program has been implemented the next task is the actual simulation of the system behavior with the digital computer. System behavior is represented by state trajectories which are generated as simulation outputs. For example, a state trajectory of the considered pilot-aircraft system is depicted in Fig.9. One important task related to simulation outputs is the validation of the model built so far. The validation task checks if the computerized model is a sufficient accurate representation of the system behavior under consideration. It is in this phase that accumulated errors are discovered and final acceptance of the model must be achieved.

Simulation outputs describe the dynamic behavior of the system considered over time. Because in most cases it is neither useful nor possible to store all data that are generated during simulation, a simulation language like SLAM offers certain basic statistical functions for data reduction, aggregation, and documentation such as calculation of means and standard deviations, and certain formats presenting results such as histograms and certain sorts of plots. The statistical analysis of simulation outputs is similar to the statistical analysis of data obtained from an experiment with an actual system. The main difference with simulations is that the analyst has more control over the running of the simulation program. Thus he can design simulation experiments to obtain the specific analysis results necessary to answer the pertinent questions relating to the system under study. In this study, e.g., statistical data about the value ranges and use frequencies of cockpit interface information are recorded and analysed.

3. STEPS OF THE SIMULATION STUDY

Distinct steps of this study are described according to previously distinguished simulation steps. Attention is focused mainly on development and implementation of the model.

3.1. Definition of the Problem

The problem definition of a modeling process includes a statement of the phenomena of interest as well as a choice of performance measures [5]. Our phenomena of interest are highly automated flight processes of an aircraft and the corresponding tasks of the pilot during a landing approach. The basis of this simulation study was a real time flight simulator facility at the FAT. This facility simulates a twin engined HFB 320 Hansa executive jet manufactured by MBB, Hamburg, that is equipped with the Collins AP 104 autopilot and the FD 109 flight director. The simulator reproduces the approach to runway 25 of the Cologne-Bonn airport. Since the HFB 320 aircraft and runway 25 were already represented in the simulator these were accepted for analysis. The aircraft requires a crew consisting of pilot, co-pilot, and flight engineer. Only pilot tasks were analyzed. Fig.3 shows the corresponding instrument approach chart. It is assumed, that the initial position of the aircraft is above the VOR navigation station WIPPER at an altitude of 3000 ft, and that the autopilot has been engaged. With a left turn the aircraft has to reach the heading of 90^{o} which is required for intercepting the VOR radial of 150^{o}. Flying on this radial, the aircraft has to decrease its altitude to 2500 ft. On this altitude the glide path of runway 25 has to be approached with an interception heading of 190^{o}. On the glide path with a final heading of 249^{o} the aircraft descends with about 800 ft/min down to the middle marker. When this marker is reached the autopilot will be disengaged. At this event the analysis stops. Bad weather conditions were assumed for the approach so that the pilot could not rely on external vision and the approach was necessarily guided by an instrument landing system (ILS). The occurrence of aircraft malfunctions was not considered.

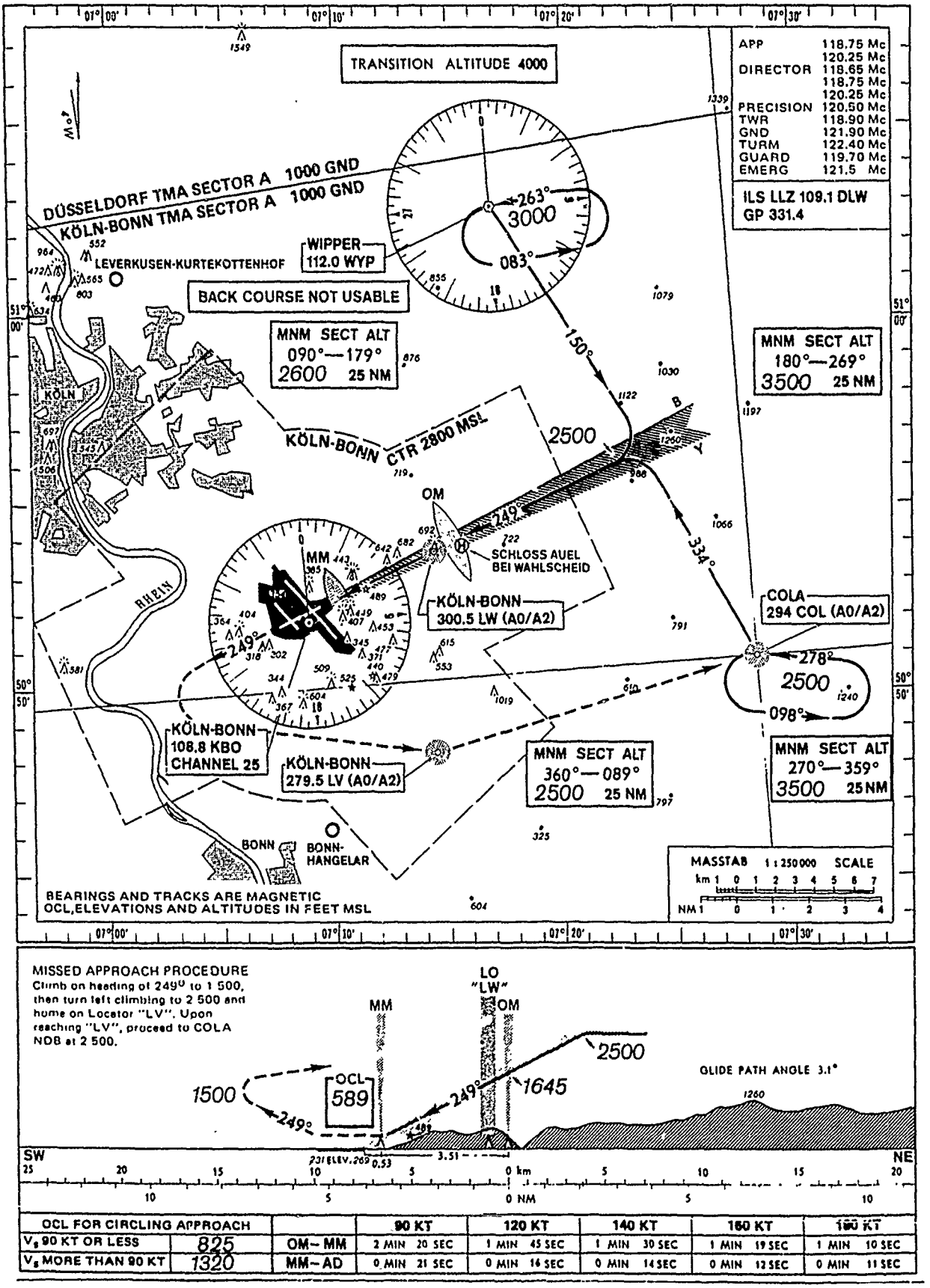
The simulation study should answer questions related to the specific task knowledge required by the pilot concerning correct approachs and to the information flow at the pilot-cockpit interface. This task knowledge is also needed, e.g., by system developers to determine the contents of pilot training programs. Also, the information flow helps to determine human engineering design and arrangement requirements for cockpit displays and controls. Specific questions which have to be answered are:

1. Which pilot tasks have to be performed during an autopilot controlled landing and in which approach segments to approach successfully?

2. Which values of which aircraft and approach state variables does the pilot need as inputs/outputs to perform his tasks successfully?

3. What is the frequency and sequence of use for those variables?

4. At which points in time and during which time intervals are those variables used?



OCL FOR CIRCLING APPROACH			90 KT	120 KT	140 KT	160 KT	180 KT
V_a 90 KT OR LESS	825	OM–MM	2 MIN 20 SEC	1 MIN 45 SEC	1 MIN 30 SEC	1 MIN 19 SEC	1 MIN 10 SEC
V_a MORE THAN 90 KT	1320	MM–AD	0 MIN 21 SEC	0 MIN 16 SEC	0 MIN 14 SEC	0 MIN 12 SEC	0 MIN 11 SEC

Fig. 3: Instrument approach chart of the airport Cologne-Bonn, runway 25

3.2 Development of the Model

An analysis was conducted to develop the conceptual model, i.e. the mathematical-logical behavior description of approach relevant systems and the pilot. Approach relevant systems from which the aircraft system receives and/or to which it gives information are VOR

and NDB navigation stations, the outer and middle markers, the localizer, glide path transmitter, and the tower (Fig.3). By relating aircraft position to ground stations and runway, actual bearings and distances between them and the aircraft can be determined. During an ILS approach, the course of events depends on aircraft motion relative to ground stations and the runway of the airport. Variables which characterize aircraft flight parameters are, e.g., heading, indicated air speed, and vertical speed. These variables determine the position of the aircraft in x, y, and z coordinates over time.

To identify system functions and especially pilot tasks, the ILS-approach was partitioned into 12 approach segments, partly listed in Table 1. Three task priorities and five task categories were established to identify required monitoring, supervisory, and control tasks:

First task priority:
- adjusting tasks, e.g., to set up the autopilot with new desired courses or headings;
- activating tasks, e.g., to change autopilot mode, flap position, or gear state;
- special tasks, e.g., verbal communication when performing outer marker check or receiving landing clearance.

Second task priority:
- checking tasks, e.g., to compare current and desired values of indicated air speed, heading, altitude, etc. during a cross-check.

Third task priority:
- monitoring tasks, e.g., to systematically observe the indicated altitude during descent or heading during approach to an interception heading.

The first task priority includes adjusting, activating, and special tasks which are required for a successful approach. These tasks have the highest priority because they have to occur at special approach points. If they are not performed at those points the approach progress may become incorrect. For instance, during the approach segment "Approach to In-

Table 1: Approach segments and corresponding pilot tasks (in part)

Approach to Interception Heading 90°	Adjust Heading Marker Position 170° Adjust Digital Course 150° Activate Lateral Mode VOR Adjust Heading Marker Position 90° Adjust Engine RPM 80 % Perform Cross Check Monitor Flight Attitude Monitor Heading
On Interception Heading 90°	Perform Cross Check Monitor Course Deviation Monitor Capture Indicator
Interception of Radial 150° of VOR-Station	Perform Cross Check Monitor Flight Attitude Monitor Course Deviation Monitor Heading
On Radial 150° of VOR-Station	Activate Flap Position 20° Adjust Vertical Speed 800 ft/min Activate Vertical Mode VS HOLD Adjust Vertical Speed 350 ft/min Activate Vertical Mode ALT HOLD Adjust Heading Marker Position 190° Activate Lateral Mode HDG Perform Cross Check Monitor Altitude
Approach to Interception Heading 190°	Adjust Digital Course 249° Activate Nav Receiver II Activate Lateral Mode APPR Perform Cross Check Monitor Flight Attitude Monitor Heading ...

terception Heading 90°" (Tab.1), the pilot has to adjust the heading marker position to the value of 170° at the beginning, and later to 90°. He has to adjust the digital value of the course to 150°, to activate the autopilot lateral mode VOR-APPROACH, and to adjust engine revolutions per minute to 80 %.

Tasks of the second task priority are required to ensure system safety. During all segments the pilot ought to check aircraft state variables to detect malfunctions of aircraft subsystems. For the checking procedure, it is assumed that he performs a cross-check during which he compares desired and actual values of indicated air speed, flight attitude, heading, altitude, vertical speed, and engine revolutions per minute. The cross-check is interrupted if tasks of the first priority have to be performed.

Monitoring tasks have the third priority. They are required to verify correct autopilot operations, especially during approach transition phases in which the autopilot is controlling distinct aircraft state variables, e.g., heading variation or vertical speed. The pilot should monitor aircraft state variables over a certain time period, e.g., headings during heading changes, altitudes during altitude changes. It is assumed that these tasks have the lowest priority because malfunctions of aircraft subsystems are normally already detected during the cross-check.

For each approach segment, lower and upper limits of aircraft state and approach variables which are specific to a segment were determined. Such variables are, e.g., air speed, vertical speed, heading, altitude, flight attitude, course and glide slope deviation. This information represents that knowledge which the pilot has to retrieve out of his memory or from the instrument approach chart when he performs checking and monitoring tasks.

By using specified task categories, 42 different pilot tasks were identified for the ILS-approach. Pilot task performance was considered to be normative, i.e. what the pilot should do during the approach. Each task was described by a behavioral verb, which was used in the task classification above and which indicated the nature of activity being performed during that task and by that information or state variable that was being acted upon by the pilot. Additionally, the time required by the pilot to perform that task was estimated. Basic data for time estimations were taken from the timeline analysis program of Boeing [7]. Operating times for tasks such as adjusting heading marker position, digital course, and vertical speed were measured in our flight simulator facility. The task duration required by the pilot to perform a task was modeled by a normal distribution function characterized by mean and standard deviation. For describing the identified tasks in detail, a production system approach [8],[9] was used. The major elements of a production system are a database, a set of productions, and a control system. The productions operate on the database. Rouse [5] defines a production as a situation-action pair where the situation side is a list of things to watch for in the database and the action side is a list of things to do. Actions resulting from a production change the database. The control system determines which applicable productions should be applied and ceases computation when a termination condition on the database is satisfied. Possible control strategies are discussed, e.g., in [9].

In applying the production scheme to a pilot task, the situation side represents actual values of those variables which the pilot has to perceive from his cockpit environment or to retrieve from his memory. The action side describes the actual values of those variables which he has to act upon when performing a task. The situation side of productions specifies signs which are the typically perceived information at the rule-based level of behavior, while the action side specifies the predetermined manipulations which are activated when situation specific signs appear [1],[2]. The situation and the action side represent the task input and output, respectively. It is assumed that input and output are separated in time by the task duration, i.e. the time the pilot needs to perform the task. In this study, applicable productions were determined by querying expert pilots, and analysing approach procedure descriptions [10] and records obtained with our flight simulator.

To explain the structure of task specific productions in some detail, only the adjusting task 'Adjust Heading Marker Position' (AD HMP) is described. That task appears twice during the segment "Approach to Interception Heading 90°" and once during the segment "On Radial 150° of VOR-Station". It appears at distinct points of time at which distinct actions, i.e. the values 170°, 90°, 190° of the heading marker position hmp have to be entered into the autopilot. At those points, the situation can be characterized by the approach segment and specific values of the heading hd(t), the heading marker position hmp(t), and the altitude alt(t) at time t. Denoting the duration of the task AD HMP by d(AD HMP), the following productions can be established:

IF (approach segment = 'Approach to Interception Heading 90°' .AND. hd(t) > 198° .AND. hmp(t) ≠ 170°)
THEN 'Adjust hmp(t + d(AD HMP)) = 170°',

IF (approach segment = 'Approach to Interception Heading 90°' .AND. hd(t) < 198° .AND. hmp(t) ≠ 90°)
THEN 'Adjust hmp(t + d(AD HMP)) = 90°',

IF (approach segment = ' On Radial 150^{o} of VOR-Station' .AND. alt(t) ≤ 2500 ft .AND. hmp(t) ≠ 190^{o})
THEN 'Adjust hmp(t + d(AD IIMP)) = 190^{o}'.

By describing each pilot task with productions that are specific to a special approach segment and/or aircraft state, the large amount of knowledge required by the pilot during the landing approach could be clearly and completely structured and specified.

Flight processes were described in a simplified form. Because an essential goal of the study was to determine the information flow at the pilot-cockpit interface, only those state variables displayed in the cockpit and relevant to pilot tasks were modeled. To model relevant state variables and their value changes over time, a set of difference equations was used. For example, to compute the actual x, y, and z coordinates posx, posy, and posz of the aircraft position at time t_n, the values of heading hd, indicated air speed ias, and vertical speed vs at time t_{n-1} were used:

$$
\begin{aligned}
posx(t_n) &= posx(t_{n-1}) + (t_n - t_{n-1}) * ias(t_{n-1}) * \sin(hd(t_{n-1})),\\
posy(t_n) &= posy(t_{n-1}) + (t_n - t_{n-1}) * ias(t_{n-1}) * \cos(hd(t_{n-1})),\\
posz(t_n) &= posz(t_{n-1}) + (t_n - t_{n-1}) * vs(t_{n-1}),\\
hdv(t_n) &= hdv(t_{n-1}) + (t_n - t_{n-1}) * hdvv(t_{n-1}),\\
hd(t_n) &= hd(t_{n-1}) + (t_n - t_{n-1}) * hdv(t_{n-1}),\\
ias(t_n) &= ias(t_{n-1}) + (t_n - t_{n-1}) * iasv(t_{n-1}),\\
vs(t_n) &= vs(t_{n-1}) + (t_n - t_{n-1}) * vsv(t_{n-1}),
\end{aligned}
$$

The value of the rate-of-turn at time t_n is $hdv(t_n)$ and $(t_n - t_{n-1})$ represents the time interval. Values at the last computation time t_{n-1} are represented by $posx(t_{n-1})$, $posy(t_{n-1})$, $posz(t_{n-1})$, $hdv(t_{n-1})$, $hd(t_{n-1})$, $ias(t_{n-1})$, $vs(t_{n-1})$; $hdvv(t_{n-1})$, $iasv(t_{n-1})$, and $vsv(t_{n-1})$ are the variation rates of hdv, ias, and vs at time t_{n-1}.

The general relationships between model state variables are shown in Fig.4. Starting with aircraft state variables the aircraft position is determined. Actual bearings and distances to the stations can be derived by comparing the actual aircraft position with positions of navigation stations. Actual course and glide slope deviations are determined by comparing actual bearings with those bearings desired according to the approach path. For modeling the autopilot control, thresholds of aircraft state, aircraft position, and approach variables are used. When a corresponding state variable reaches such a threshold then a state event occurs causing a change to the values of other state variables, e.g., when during altitude change the required altitude is reached, vertical speed is set to zero.

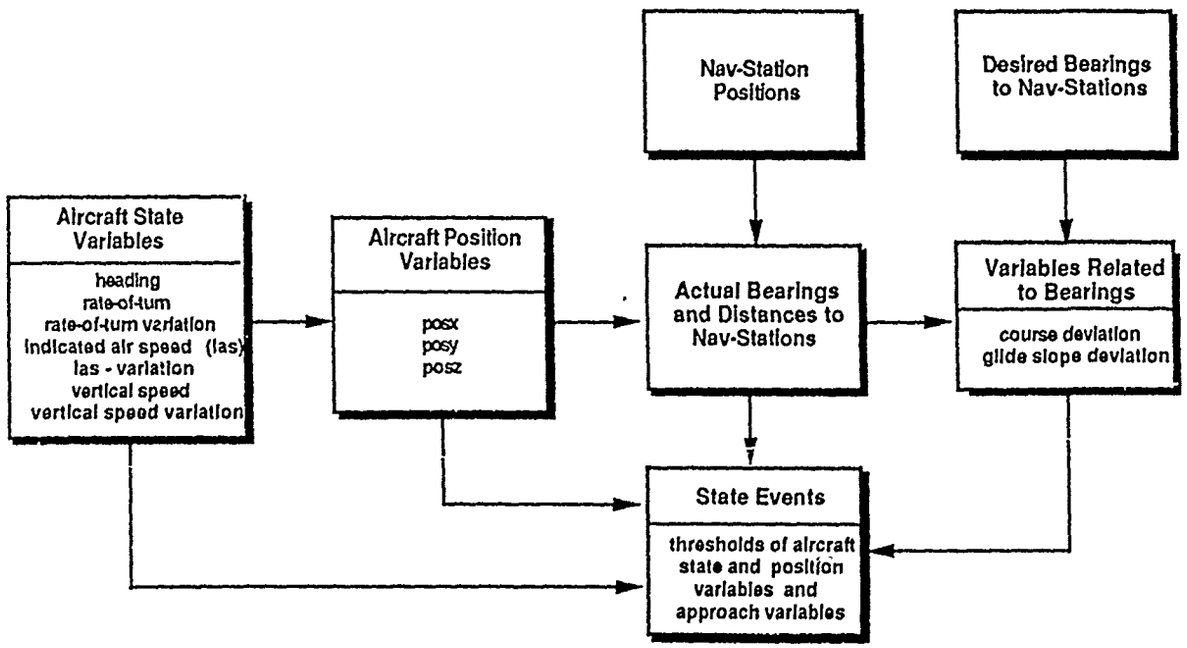


Fig. 4: Relationships between model state variables

The structure of the conceptual model that results from applying previously described elements of a production system to pilot tasks and modeled aircraft and approach related processes is depicted in Fig.5. It can be seen that changes of the database are not only induced by productions which characterize pilot tasks but also by aircraft and approach processes. According to the three task priorities the set of productions is partitioned in three subsets which represent the task categories: adjusting, activating, special task, cross check task, and monitoring tasks. The control system selects a production subset according to the actual aircraft state and approach situation and its priority. Within a subset the first rule of which the condition part is matched will be activated.

3.3. Implementation of the Model

Once the conceptual model has been developed, the next step in the simulation study is the implementation of that model. This step transforms the conceptual model into the computerized model. The simulation language SLAM (Simulation Language for Alternative Modeling) was used for the implementation. SLAM was selected mainly because it provides a conceptional framework for implementing both continuous and discrete systems and combinations of them. A detailed description of the various modeling possibilities of SLAM is given by Pritsker [4].

A continuous model is coded in SLAM by specifying differential or difference equations which describe the dynamic behavior of state variables. These equations are coded by the modeler in the SLAM subroutine STATE in FORTRAN. State variables described in the subroutine STATE are automatically updated by SLAM to calculate their values within an accuracy specified by the modeler.

For modeling discrete systems SLAM offers the possibility to apply an event-orientated and a process-orientated view. With the event orientation of SLAM, the modeler defines the events and the potential changes to the modeled system when that event occurs. The mathematical-logical relationships, prescribing the changes associated with each event type, are coded by the modeler in the SLAM subroutine EVENT in FORTRAN. The executive control program of SLAM controls the simulation by advancing time and initiating calls to the event subroutine at the proper points in simulated time. Hence, the modeler is completely relieved of the task sequencing events to occur chronologically.

The process orientation of SLAM employs a network structure which consists of specialized symbols called nodes and branches (see e.g. Fig.6). These symbols model elements in a process such as resources, queues for resources, activities, and entity flow decisions. The modeling task consists of combining these symbols into a network model which pictorially represents the system of interest and its processes. Entities in the system (such as people and items) flow through the network model. With special nodes, values which can be generated in a user-written function USERF are assigned to attributes of entities. To perform process simulation, each network element has to be coded by a special statement. An input file of the SLAM simulation package stores all those network statements which are examined con-

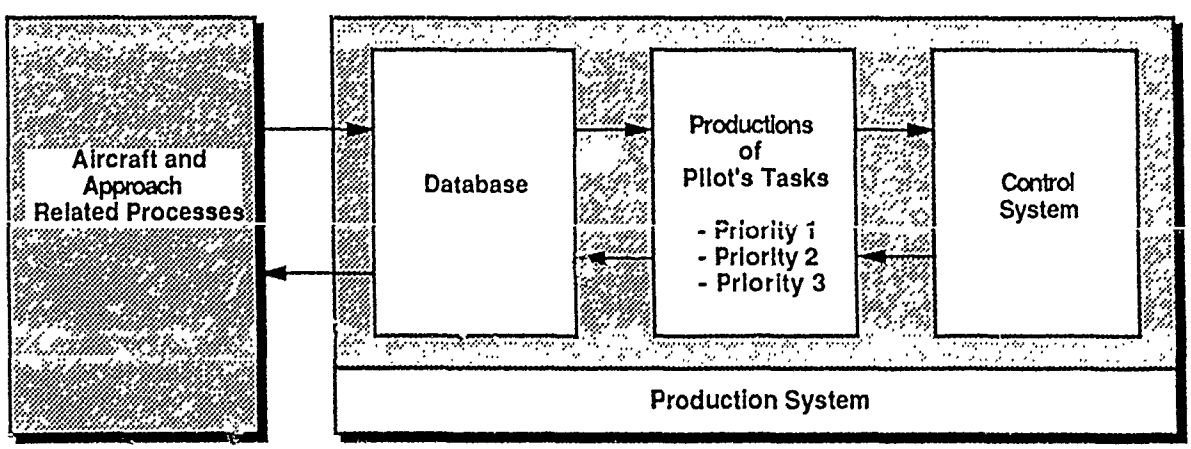


Fig. 5: Structure of the conceptual model based on a production system

cerning their correctness at the beginning of a simulation run.

In combined discrete-continuous models, the independent variable, e.g., time, may change both discretely and continuously. The view of a combined model specifies that the system can be described in terms of entities, their associated attributes, and state variables. The behavior of the system model is simulated by computing the values of state variables in small time steps and by computing the values of attributes of entities at event times.

The computerized model of the ILS-approach comprises a continuous part which is described in the SLAM subroutine STATE, an event-related part realized in the subroutine EVENT, and a network part with connections to the user-written function USERF.

With the continuous part in subroutine STATE, the aircraft related and approach related processes are modeled by means of difference equations. SLAM offers a set of state variables called SS(.) variables for describing those processes. Aircraft and approach state variables described previously (Fig.4) are defined by using these SS(.) variables. The equations in chapter 3.2 can be expressed as follows:

```
SS(1) = SSL(1) + DTNOW * SIN(SSL(5)),
SS(2) = SSL(2) + DTNOW * COS(SSL(5)),
SS(3) = SSL(3) + DTNOW * SSL(7),
SS(4) = SSL(4) + DTNOW * HDVVL,
SS(5) = SSL(5) + DTNOW * SSL(4),
SS(6) = SSL(6) + DTNOW * IASVL,
SS(7) = SSL(7) + DTNOW * VSVL.
```

by setting, e.g., the variables $SS(1) = posx(t_n)$, $SS(2) = posy(t_n)$, $SS(3) = posz(t_n)$, $SS(4) = hdv(t_n)$, $SS(5) = hd(t_n)$, $SS(6) = ias(t_n)$, $SS(7) = vs(t_n)$, $DTNOW = (t_n - t_{n-1})$, $HDVVL = hdvv(t_{n-1})$, $IASVL = iasv(t_{n-1})$, and $VSVL = vsv(t_{n-1})$. All SSL(.) variables in the equations represent the values of the corresponding SS(.) state variables at the last time t_{n-1}.

The combination of the network part and the function USERF models discrete changes of variables characterizing pilot task performance. Generally, the network simulates the flow of temporary entities through processes from their arrival to their departure. In our case, entities represent requests for performing a pilot task. The pilot is regarded as a resource and regular network activities represent his tasks. An entity that flows through the network occupies the resource and activates that activity which is selected in the function USERF. The selection occurs according to the task priority and the actual approach situation and aircraft state.

The network which models pilot tasks and their previously defined priorities is depicted in Fig.6. It consists of three partial networks labeled A, B, and C. Each part represents tasks of a different priority: A) First priority tasks are adjusting, activating, and special tasks; B) Cross check tasks are second priority; C) Monitoring tasks are third priority. The partial networks are controlling the simulation by working concurrently. Each partial network consists of a combination of nodes and branches. In general, an entity is moving in a cyclic manner from an ASSIGN node to a PREEMPT node, further to a FREE node and back to the ASSIGN node. The path between the PREEMPT node and the FREE node represents the performance of a single pilot task. All three partial networks are similar in structure. In an ASSIGN node the characteristics of a task are determined by calling the user-written function USERF. In that function task specific productions are selected and activated. Normally, a branching activity that needs no time leads to the PREEMPT node whose only function is to capture the resource according to the distinct task priority. The activity between PREEMPT and FREE node is used to simulate the task duration. Because in all partial networks PREEMPT nodes require the same resource, i.e. the pilot, the different priorities of PREEMPT nodes which correspond to task priorities control which of the three task categories is performed. E.g., adjusting tasks interrupt cross check and monitoring tasks; cross check tasks interrupt monitoring tasks. When a task activity between a PREEMPT node of a lower priority and a FREE node has been activated and a task is requested at a PREEMPT node of a higher priority, then the activated task activity is interrupted and the task activity with higher priority starts. The partial network A includes one or two EVENT nodes in some branches which connect the network part with the EVENT orientated part of the model

that will be described later. The FREE nodes release the captured resource so that it is free for performing other tasks.

As mentioned above the ASSIGN node connects the network model with the function USERF. Determined in this function (Fig.7) are the actual approach segment and the corresponding upper and lower limits of state variables, the requested pilot task and its action, and the task duration. The task specific production in USERF is chosen in the following way: Ac-

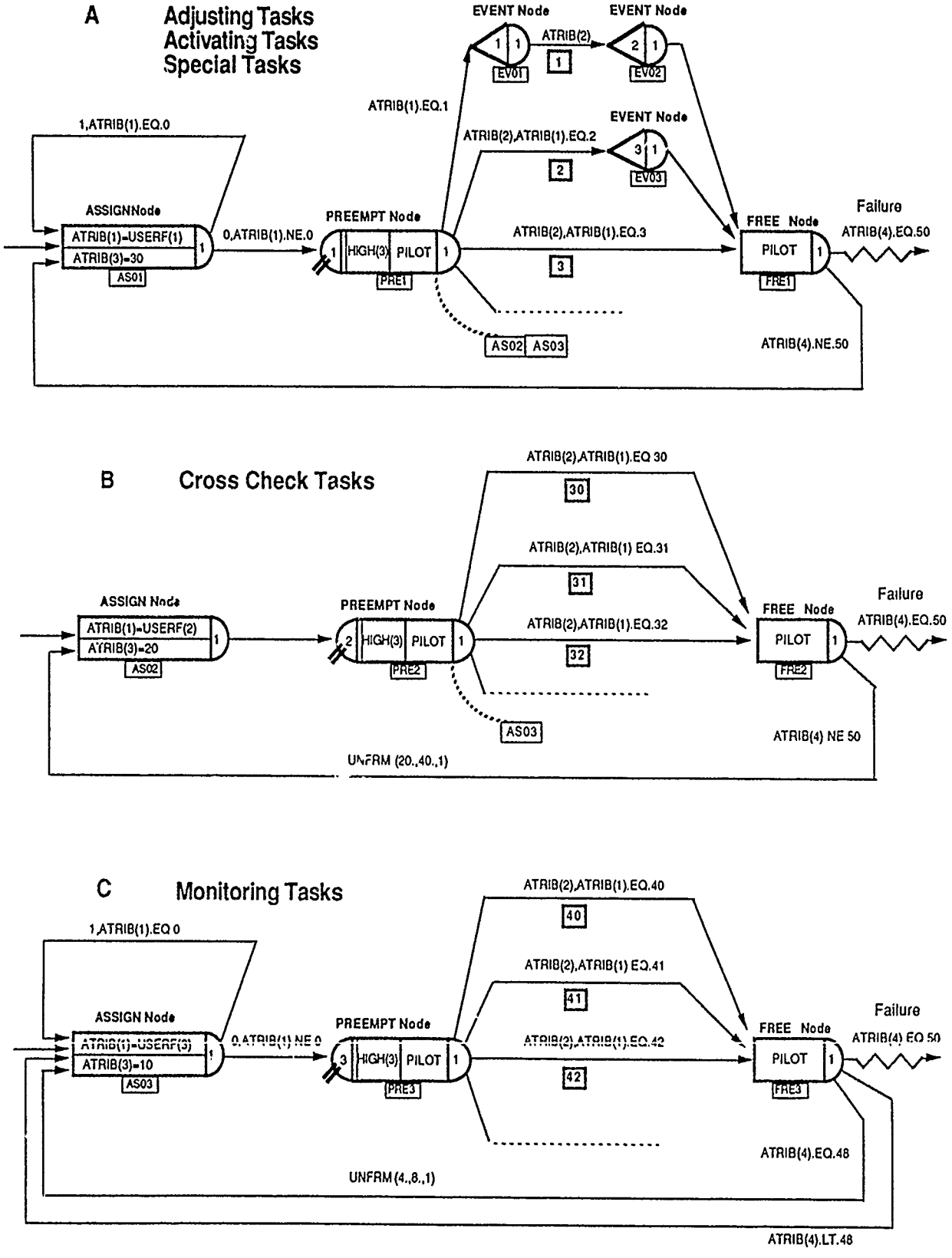


Fig. 6: SLAM network for controlling tasks selection

cording to the priority of a task request, a corresponding part of USERF is selected. In this part the production selected is that for which the situation side is matched first.

```
      FUNCTION USERF (IFN)

C***  DATABASE ***

      COMMON ...

C***  PRODUCTIONS FOR SELECTING THE APPROACH SEGMENT  ***

      IF ( APPROACH SITUATION = 'X' )
            THEN (  APPROACH SEGEMENT = 'APPR SEGM NAME X'
                    SETTING OF DESIRED VALUES OF STATE VARIABLES
                    FOR APPROACH SEGEMENT 'X')
      IF ( APPROACH SITUATION = 'Y' )
            THEN (  APPROACH SEGEMENT = .... )


C***  SELECTION OF THE TASK TYPE ***

      GOTO (1000,2000,3000), IFN

C***  PRODUCTIONS FOR SELECTING ADJUSTING, ACTIVATING, ***
C***  AND SPECIAL TASKS                                 ***

1000  USERF = 0
      IF ( APPROACH SEGMENT = 'X' .AND. FLIGHT SITUATION = 'Y' )
            THEN (  TASK = 'ADJUST STATE VARIABLE SS'
                    USERF = 1
                    ATRIB(2) = TASK DURATION
                    ACTION = ACTION OF TASK
                    RETURN)
      IF ( APPROACH SEGMENT = 'Y' .AND. FLIGHT SITUATION = 'Z' )
            THEN (  .....
                    RETURN)

          RETURN

C***  PRODUCTIONS FOR SELECTING CROSS CHECK TASKS ***

2000  IF (ATRIB(4) EQ.30)
            THEN (  TASK = 'CHECK STATE VARIABLE SS'
                    ATRIB(2) = TASK DURATION
                    USERF  = 30
                    ATRIB(4) = 31
                    IF ( SS.LT.SSLOW .OR. SS.GT.SSHIGH) ATRIB(4) = 50
                    RETURN)
      IF (ATRIB(4).EQ.31)
            THEN ( .....
                    RETURN)

      RETURN

C***  PRODUCTIONS FOR SELECTING MONITORING TASKS ***

3000  USERF = 0
      IF ( APPROACH SEGEMENT = 'X' .AND. FLIGHT SITUATION = 'Y' )
            THEN (  TASK = 'MONITOR STATE VARIABLE SS'
                    USERF = 40
                    ATRIB(2) = TASK DURATION
                    IF (SS.LT.SSLOW .OR. SS.GT.SSHIGH)        ATRIB(4) = 50
                    IF (ABS(SS-SSDESIRED)/DD.GT.15.)          ATRIB(4) = 48
                    RETURN)
      IF ( APPROACH SEGEMENT = 'Y' .AND. FLIGHT SITUATION = 'Z' )
            THEN (  .....
                    RETURN)

      RETURN
```

Fig. 7: Schematic structure of the user function USERF(IFN)

A regular SLAM activity (denoted in Fig.6 by the activity number in a rectangle) is used to model time duration which the pilot needs to perform the task. The task duration of a specific task when performed is a sample of a normal distribution function and is determined in the function USERF that is activated in the corresponding ASSIGN node.

EVENT nodes are included in the network model to interface that part of the model with discrete time events. Such an event occurs, e.g., after adjusting and activating tasks are accomplished. They lead to modifications of aircraft related processes which are modeled in the subroutine STATE. The actual modification of state variables is specified by the user in the subroutine EVENT. The EVENT node causes the subroutine EVENT to be called. In contrast to time events that occur when an entity reaches an EVENT node in the network, so-called state events occur when specified state variables cross prescribed thresholds. To model such events, the state-event feature of SLAM which also activates subroutine EVENT is used. In the ILS model, for instance, a state event occurs when a distinct bearing to a navigation station is reached. The user has to specify in the subroutine EVENT that now the heading changes with a turn rate of $3^{o}/s$.

Fig.8 shows the implementation of the conceptual model based on elements of a production system (Fig.5) with subroutines, functions, and the network elements of SLAM. It can be seen that the COMMON block of SLAM represents the database of the implemented production system. Changes of database states are induced by aircraft related and approach related processes that are coded in subroutines STATE and EVENT. In the user-written function USERF, productions which describe the pilot's rule-based behavior are defined by means of situation-action rules. The network part of SLAM constitutes the control system which selects a task category and its actual production according to the task priority and the actual approach and aircraft state. A task activity of the network simulates the task duration.

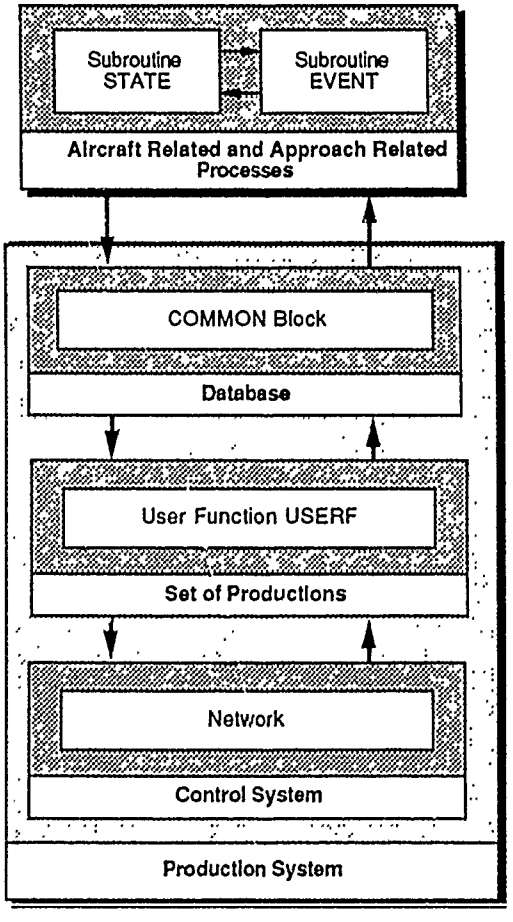


Fig. 8: Implementation of the conceptual model based on a production system with SLAM elements.

3.4. Simulation of the ILS-approach

After the computerized ILS-approach model has been implemented, the digital computer is used to simulate system activities. Generally, simulation implies an exercising of the computerized model to generate a chronological succession of state descriptions, i.e. of values of relevant state and task variables describing system behavior. Fig.9 shows an example

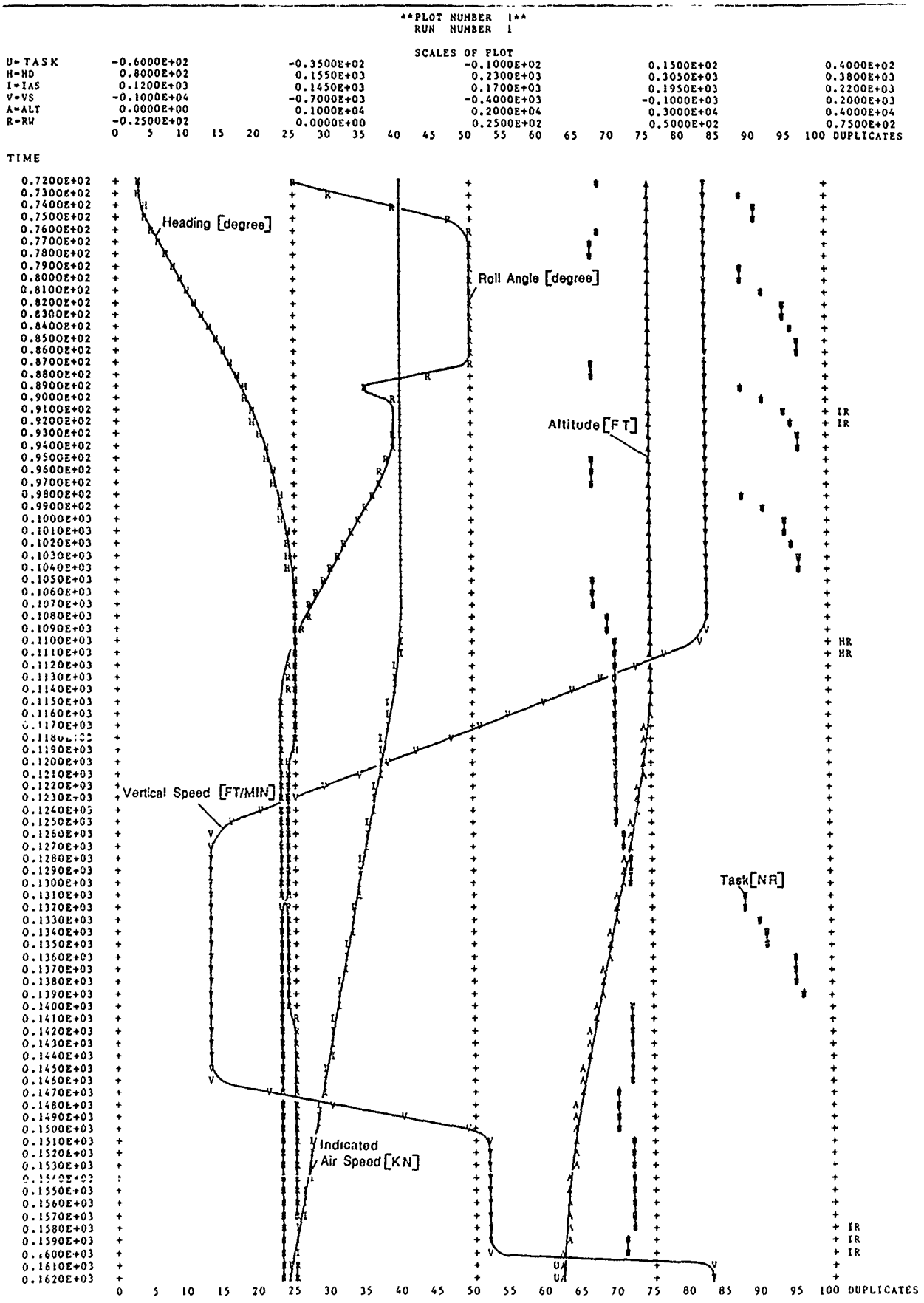


Fig. 9: State trajectory of heading (H), roll angle (R), altitude (A), vertical speed (V), indicated air speed (I) and the task time line (U).

of a plotted state trajectory with the state variables heading, roll angle, altitude, vertical speed, and indicated air speed. Additionally, the corresponding task time-line with pilot tasks coded by numbers on the head-line is shown on the right side of the plot. The

length of each line segment represents the task duration. Since the model uses probabilistic distributions for, e.g., task durations, the generation of simulation data is statistical in nature. Thus, simulation runs must be repeated many times to obtain a sufficient number of state and task trajectory samples in order to estimate average performance within reasonable confidence limits. 30 ILS-approaches were simulated with the computerized model in order to get corresponding trajectories and task timelines (traces).

An important task related to simulation outputs is the validation of the model. Experimental validation of a model involves using the model to predict performance and then empirically determining how close predictions are to actual occurrences [5]. The pilot's rule-based behavior, aircraft processes, and approach events were simulated with the computerized process model. Modeled pilot behavior could be validated against performance of a real pilot in our flight simulator. Aircraft processes and event time points would be validated if simulated values of appropriate measures closely match measured values of piloted runs in our realistic flight simulator. Ten real time ILS-approaches were performed with one experienced pilot in the flight simulator. Pilot and simulator activities observable in the cockpit were recorded on video-tape for later analysis. Data obtained during the approaches constituted the basis for validating the model.

From the various validation techniques which are applicable [11], we considered face validity, simulation output traces, internal validity, and event validity. For evaluating the face validity of the model, pilots familiar with the ILS-approach were asked whether the conceptual model was reasonable. Particularly, production rules of tasks, details of flight processes described, and interrelations between tasks and flight processes were discussed and corrected in this way. Simulation output traces were used to check the computerized model, i.e. pilot tasks as represented by the network were plotted to determine whether the simulation program correctly corresponded to the network and logic of those tasks. Deviations found in the model were modified appropriately. Internal validity was assessed by comparing the stochastic variability of specific state variables in the model with their variability observed in the flight simulator. To test this type of validity, several stochastic simulation runs were made with the model and the variability of selected state variables, e.g., glide path interception altitude, glide path deviation, and course deviation, were determined and used for testing this validity type.

To check event validity of the model, occurrence times of 17 approach relevant events were used as performance measures. Such events may be either task events which the pilot causes when performing a task, e.g., 'activating autopilot lateral mode VOR' (LM.VOR), 'activating autopilot vertical mode VERTICAL SPEED HOLD' (VM.VSH), or they are system state events which occur when state variables reach specified thresholds, e.g., 'capturing the VOR radial 150°' (CPT.150), 'reaching indicated air speed 160 kn' (IAS.160), 'reaching flight attitude level' (FA.LEV). For all event occurrences, mean elapsed times into the approach were determined both from the 10 pilot flight simulator ILS-approaches and from 30 runs of the digital computer simulation. The times were compared statistically by means of the t-test. Existing deviations were eliminated by changing task duration distributions until no significant differences were obtained between flight simulator and computer simulation model data. Fig.10 illustrates the elapsed time intervals in which approach events would fall with 99 % probability. With model generated events the sample size is larger and the time variance is smaller than with pilot generated events. Therefore, confidence intervals of model generated events are also much smaller.

3.5. Analysis of simulation outputs

After model validation, the final step in our simulation study was the experimental application of the model to generate and analyse simulation output data. One goal of the study was to determine information flow requirements for the pilot-cockpit interface. Such requirements include all those variables about which information is transmitted to the pilot or which are subsequently affected by control outputs of the pilot. They can be characterized by the mission-required value range, time points, time interval, and their use sequence, frequency, and duration. In modern MMS those specifications are necessary to design display and control units [12],[13] which are based on interactive electronic concepts. The following question has to be answered, e.g., to design electronic display formats with quickly changing values: Which value of which state variable at which time point or during which time interval does the pilot need to perform a specific task? Information that the pilot requires at the same time should be combined in the same format. To arrange formats correctly, the information importance and its frequency of use have to be known. Important and frequently used information/formats have to be arranged in upper priority levels. Most requirement specifications can be obtained by using digital computer simulation. An exception is the importance of information which can be determined, e.g., by questionnaires.

In our study, output data of 30 simulation runs were analysed to obtain information flow requirements. The analysis was done by using SLAM elements for data collection and

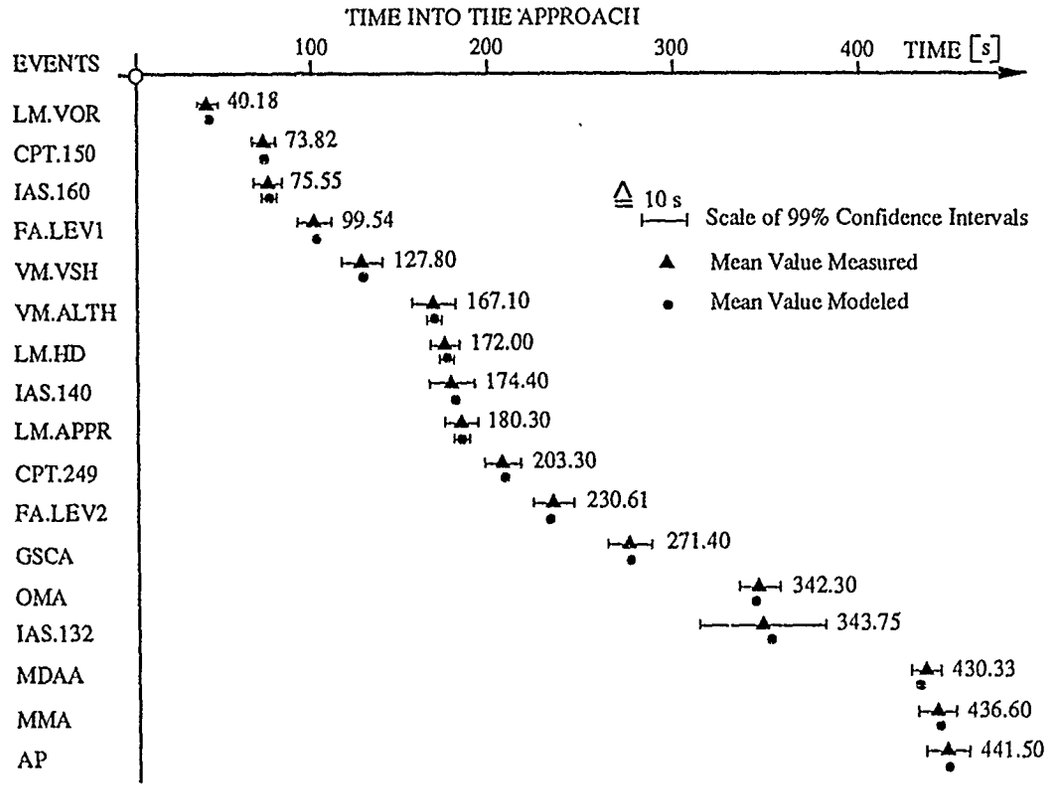


Fig. 10: Comparison of approach events both measured and modeled

statistical calculation. Analysis results related to state variables altitude, heading, vertical speed, roll angle, course, indicated air speed, heading marker, glide path deviation, and course deviation are listed in Table 2. For those variables the range of used values, the time interval of use, the use frequency, and the relative use duration with its mean and standard variation are listed. For explanation purposes, only one of the variables, altitude, will be interpreted in detail.

It can be seen that altitude is utilized in the value range between 365 and 3000 feet and between the 12th and 449th second during the considered approach. It was used an average of 55 times with a relative use duration and standard deviation of 27.1 % and 1.2 %, respectively. For the average approach duration of 443 seconds, the absolute use duration

Table 2: Analysis results for some selected variables

Relevant Variables	Abbr.	Dimen.	Range of Used Values		Time Intervall of Use		Use Frequency	Rel. Use Duration M. Val. [%]	Rel. Use Duration St.Dev. [%]
			from	to	from [sec]	to [sec]			
Altitude	ALT	ft	364,8	3000,0	11,7	449,3	55	27,1	1,2
Heading	HD	degree	89,8	263,0	0,0	240,5	35	16,3	0,9
Vertical Speed	VS	ft/min	-983,5	0,0	13,4	449,8	27	13,9	1,1
Roll Angle	RA	degree	-25,0	25,0	9,9	448,6	29	13,9	0,7
Course	CO	degree	150,0	263,0	2,3	238,4	31	12,5	1,4
Ind.Air Speed	IAS	kn	131,4	180,0	8,4	446,5	27	9,8	0,9
Head.Marker	HDM	degree	90,0	190,0	0,0	192,9	8	5,8	0,4
Glide Slope Dev.	GSD	dots	0,6	2,2	232,5	445,5	13	5,5	0,8
Course Dev.	COD	dots	-1,4	0,4	207,2	438,0	13	4,6	0,7

HISTOGRAM NUMBER 2

TIME HEADING USE

OBSV FREQ	RELA FREQ	CUML FREQ	UPPER CELL LIMIT
0	0.000	0.000	0.0000E 00
55	0.052	0.052	0.1000E 02
30	0.029	0.081	0.2000E 02
105	0.100	0.181	0.3000E 02
9	0.009	0.189	0.4000E 02
66	0.063	0.252	0.5000E 02
59	0.056	0.308	0.6000E 02
76	0.072	0.381	0.7000E 02
93	0.088	0.469	0.8000E 02
55	0.052	0.521	0.9000E 02
58	0.055	0.577	0.1000E 03
58	0.055	0.632	0.1100E 03
0	0.000	0.632	0.1200E 03
2	0.002	0.634	0.1300E 03
26	0.025	0.658	0.1400E 03
9	0.009	0.667	0.1500E 03
6	0.006	0.673	0.1600E 03
17	0.016	0.689	0.1700E 03
33	0.031	0.720	0.1800E 03
45	0.043	0.763	0.1900E 03
59	0.056	0.819	0.2000E 03
54	0.051	0.871	0.2100E 03
63	0.060	0.931	0.2200E 03
27	0.026	0.956	0.2300E 03
43	0.041	0.997	0.2400E 03
3	0.003	1.000	0.2500E 03
0	0.000	1.000	0.2600E 03
0	0.000	1.000	0.2700E 03
0	0.000	1.000	0.2800E 03
0	0.000	1.000	0.2900E 03
0	0.000	1.000	0.3000E 03
0	0.000	1.000	0.3100E 03
0	0.000	1.000	0.3200E 03
0	0.000	1.000	0.3300E 03
0	0.000	1.000	0.3400E 03
0	0.000	1.000	0.3500E 03
0	0.000	1.000	0.3600E 03
0	0.000	1.000	0.3700E 03
0	0.000	1.000	0.3800E 03
0	0.000	1.000	0.3900E 03
0	0.000	1.000	0.4000E 03
0	0.000	1.000	0.4100E 03
0	0.000	1.000	0.4200E 03
0	0.000	1.000	0.4300E 03
0	0.000	1.000	0.4400E 03
0	0.000	1.000	0.4500E 03
0	0.000	1.000	0.4600E 03
0	0.000	1.000	0.4700E 03
0	0.000	1.000	0.4800E 03
0	0.000	1.000	0.4900E 03
0	0.000	1.000	0.5000E 03
0	0.000	1.000	INF
1051			

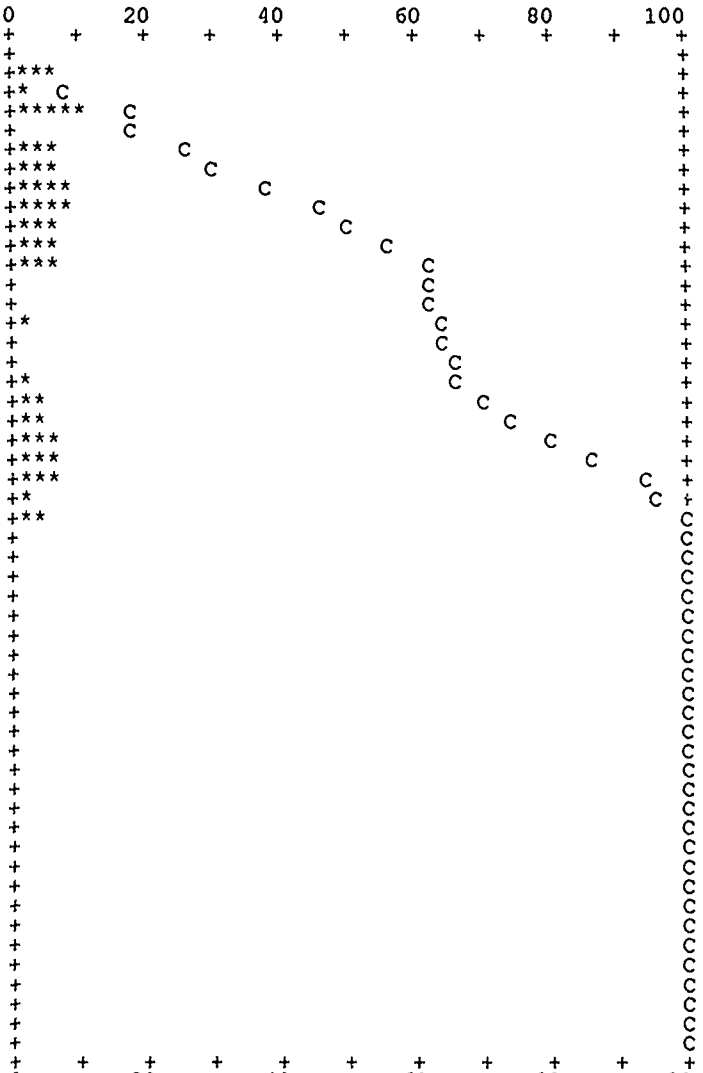


Fig. 11: Absolute and relative frequencies of heading use during the approach

and standard deviation are 120 s and 5.3 s.

Additional analysis details were obtained with SLAM by recording time points at which certain variables are used during the simulated approach. To analyse recorded data SLAM generates e.g., histograms, using approach time as one axis and dividing it into intervals of equal duration. The absolute, relative, and cumulative frequency of use were demonstrated in histograms for distinct variables for each time interval. By using this method it could be ascertained, for instance, that heading values were used mainly in the first approach phase (Fig.11). However, during the final approach the course deviation is used instead of the heading.

The sequence of information use is another important feature for determining the arrangement of information in a format. Sequences of tasks performed (timelines) and of state variables used during their performance can be obtained by plotting tasks and state variables over time in the same diagram (see, e.g., Fig.9).

The utilization time of a format determines, among other things, its position in the display priority. To demonstrate the possibility of determining this feature with the digital computer simulation, the average utilization time in percent of total approach time of display and control components of the considered cockpit are determined by analysing output data of the simulated approaches (Fig.12). The most often used display components in this study are the flight director indicator, variometer, course indicator, and altimeter. Because of the highly automated ILS-approach, control components are seldom used. The control

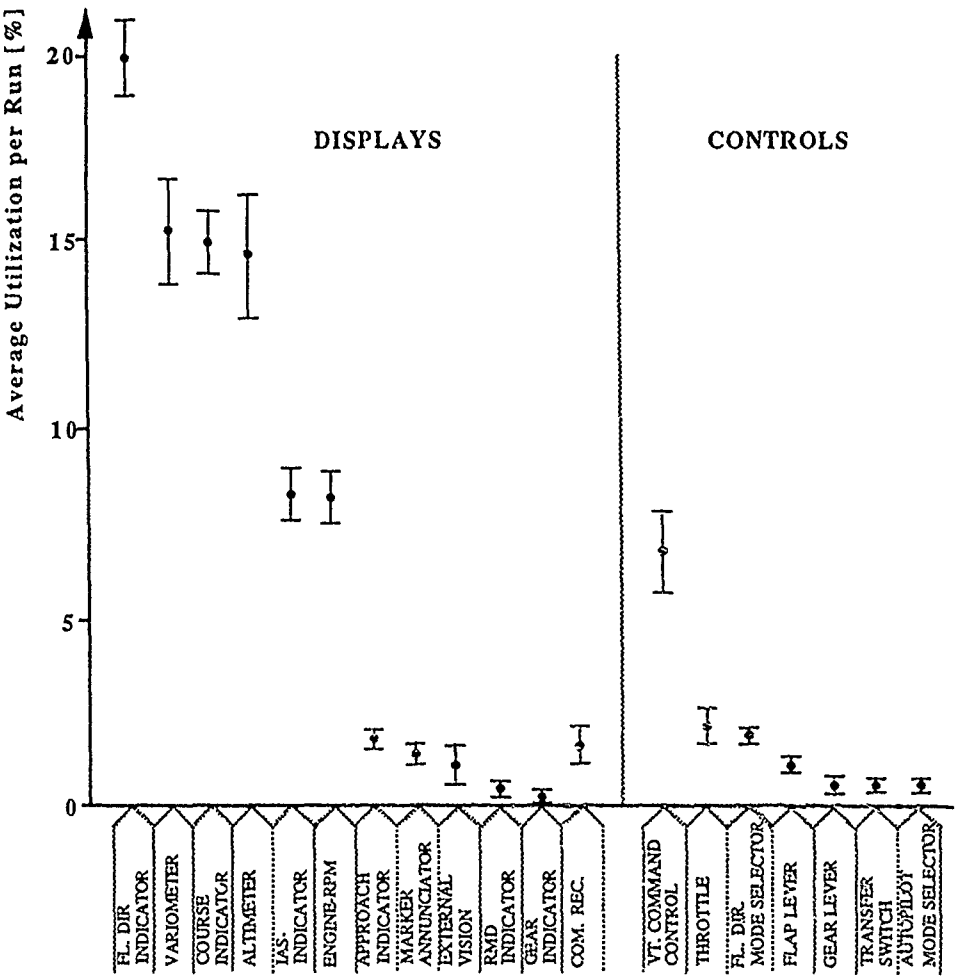


Fig. 12: Average utilization of displays and controls per simulated landing approach

component used most often is the vertical command control with which the vertical speed is adjusted when changing altitude to the required value for intercepting the glide slope. Further details and descriptions of the simulation study and its results can be found in [14].

4. CONCLUSIONS

The rule-based behavior of a pilot during a highly automated landing approach could be described in terms of situation-action rules of a production system. This was done by identifying tasks, their priorities, their inputs and outputs, and randomizing task performance durations. Using a production system to describe the pilot's knowledge for a successful landing, situation-oriented knowledge for monitoring, supervisory and control tasks, and procedure-oriented knowledge for cross check tasks could be combined in one model. Prerequisite for determination of production rules is a comprehensive analysis of tasks that must be performed and the description of system processes affecting those tasks. Advantages of that method are: the model can be easily established in a relatively short time; it is open to easy modification; and because of its characteristics it can easily be transformed into a simulation program. This was demonstrated by using the high level simulation language SLAM to implement the model and to exercise it dynamically on a digital computer.

A general advantage of the computer simulation method is its iterative nature. As soon as the flight processes have been described in enough detail, the model and the simulation method can be used in a top-down manner for further identifying and analysing pilot tasks until the required level of detail has been reached. The modularity of production rules as well as of SLAM network elements has proven to be very useful in that modeling process.

Simulation output data are trajectories of state variables and task timelines. Task timelines are generated dynamically because tasks are not preprogrammed but depend on flight segments, approach events, system states, etc. By analysing established production rules and simulation output data, task specific knowledge which the pilot needs for a successful approach and dynamic information flow requirements necessary for cockpit interface

design and evaluation can be determined. Although this method was applied to an existing MMS, it can be used to evaluate system concepts in early development phases, e.g. for determining the required task knowledge and information flow requirements for event and procedure oriented pilot tasks.

5. REFERENCES

[1] Rasmussen, J.: Skills, Rules, and Knowledge; Signals, Signs, and Symbols, and Other Distinctions in Human Performance Models. IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-13, NO. 3, May/June 1983, p.257-266.

[2] Rasmussen, J.: Information Processing and Human-Machine Interaction. New York, Amsterdam, London: North-Holland, 1986.

[3] Döring, B., Berheide, W.: A Review of Simulation Languages and their Application to Manned Systems Design. In: Moraal, J., Kraiss, K.F. (Eds.): Manned Systems Design - Methods, Equipment, and Applications. New York, London: Plenum Press, 1981, pp. 91-120.

[4] Pritsker, A.A.B.: Introduction to Simulation and SLAM II. Third Edition. New York: A Halsted Press Book, John Wiley & Sons; West Lafayette, Indiana: System Publishing Corporation, 1986.

[5] Rouse, W.B.: Systems Engineering Models of Human-Machine Interaction. New York, Oxford: North Holland, 1980.

[6] Schlesinger, S., R.E. Crosbie, R.E. Gagne, G.S. Innis, C.S. Lahvani, J. Loch, R.J. Sylvester, R.D. Wright, N. Kheir and D. Bartos: Terminology for Model Credibility. Simulation, March 1979, pp. 103-104.

[7] Miller, K.H.: Timeline analysis programm (TLA-1), Final report. Seattle, WA: Boeing Commercial Airplane Company, NASA CR-14494, 1976.

[8] Nilsson, N.J.: Principles of Artificial Intelligence. Palo Alto, CA: Tioga, 1980.

[9] Barr, A. and E.A. Feigenbaum (Eds.): The Handbook of Artificial Intelligence, Volume I. Stanford, CA: Heuris Tech Press; Los Altos, CA.: William Kaufmann, 1981.

[10] Pasteur, P.: Instrumentenflug, 2. Auflage. Zürich: Aerolit-Verlag Hans G. Auer, 1971.

[11] Sargent, R.: Validation of Simulation Models. Proceedings of 1979 Winter Simulation Conference, San Diego, CA, 1979, pp. 496-503.

[12] Helander, M.G.: Design of Visual Displays. In: Salvendy , G. (Ed.): Handbook of Human Factors. New York: John Wiley & Sons, 1987, pp. 507-548.

[13] Bullinger, H.J., Kern, P., Muntzinger, W.F.: Design of Controls. In: Salvendy , G. (ed.): Handbook of Human Factors. New York: John Wiley & Sons, 1987, pp. 577-600.

[14] Döring, B.: Analyse des Arbeitsprozesses bei der Fahrzeugführung am Beispiel eines Landeanflugs; Eine systemergonomische Simulationsstudie. Wachtberg-Werthhoven: Forschungsinstitut für Anthropotechnik, Bericht Nr. 59, 1983.

SYMBOLIC GENERATION OF AIRCRAFT SIMULATION PROGRAMMES

by

P. Maes and P.Y. Willems

Université Catholique de Louvain

Louvain-la-Neuve, Belgium.

ABSTRACT

The aim of this contribution is to present the main features of a multi-purpose computer programme which provides the equations of motion of aircraft in symbolic form and can be used in various testing and simulation procedures.

The entries of the programme are compatible with ISO standards. Various possibilities are given to the users and, when appropriate, standard choices are suggested. Both kinematical and dynamical equations are derived. These equations permit to determine the motion of a reference point fixed to the aircraft as well as the orientation of the system; they relate the variables which describe the motion to the controls and the interactions and perturbations acting on the system. The standard variables of the problem are the actual geographical position of the reference point (longitude, latitude and altitude) or cartesian coordinates (for flat earth problems), the velocity with respect to the atmosphere represented by its norm and its orientation with respect to the ground (airpath bank, climb and track angles) and to the aircraft (angle of attack and sideslip angle) and the body angular velocity (rates of pitch, roll and yaw).The parameters of the system, which appear in literal form, are the aircraft parameters (pertinent dimensions, mass distribution and aerodynamical characteristics) and the various geographical and meteorological constants (reference position, gravity constant, temperature, wind velocity and direction). The control variables (pitch, roll and yaw motivators and propulsive interactions) can be described in terms of actuators position or can be given as controller outputs. Meteorological perturbations (wind variations and athmospheric pressure and temperature changes) can also be considered as specific entries. According to the purpose of the application, various simplifications concerning the model are avaible, they concern the effects of the earth and the possibility of controls The kinematical model may include the effect of the earth rotation and, for long and middle range motion, the earth shape can be considered. On the other hand, if some of the variables can be "perfectly" controlled or measured, they can be considered as known quantities. The number of variables and accordingly the order and the complexity of the corresponding model are then reduced.

The programme is written in C-language, but its output is a standard FORTRAN subroutine which can be used as such by the user. Among other things, this programme can be used for simulation and design purposes for the vehicle and its control and navigation systems. It can also be used for Air Traffic Control simulation and trajectory optimization; coupled with a numerical linearization subroutine, it also proves useful for stability analysis.

INTRODUCTION

For the purpose of this paper the aircraft is considered as a quasi-rigid body. The mass of the system is allowed to vary and the mass time derivative is assumed to be a known function of time (possibly via the value of the trust control variable); the corresponding change of momentum is then described by equivalent propulsive interactions. The inertia matrix will also be considered as a known function of time (for instance depending on internal fuel distribution monotoring) It is further assumed that the various controls (trust and aerodynamics motivators motions) do not affect the mass distribution of the system.

The motion is described by the mechanical equations which relate the kinetic quantities (which depend on the mass distribution and the system kinematics) to the interactions (dynamic terms). In aerospace mechanics, the kinematical description is not always very simple and would, by itself, justify the use of symbolic manipulators. On the other hand, the interactions are generally described by tabulated experimental data which also need appropriate computer treatment

The following developments are based on the results of a previous AGARD publication [1] and are compatible with ISO standards [2].

EQUATIONS OF MOTION

For quasi-rigid systems, the equations of motion are the modified Newton-Euler equations. When the system reference point is its mass centre, the translational equations can be written as:

$$m\ddot{\underline{R}} = \underline{F} + \underline{F}_j$$

where: m is the actuel mass of the system described as a known function (of time);
$\underline{R}$ is the position vector of the mass centre;
$\underline{F}$ is the resultant of external forces and includes aerodynamical and gravity terms;
$\underline{F}_j$ is the "propulsive force" and corresponds to the momentum flow through the exchange surfaces.

In case of jet propulsion, the propulsive force is predominated by the effect of mass ejection through the nozzle ,see for

instance [3]) and the translational equations can be written as:

$$m\ddot{\underset{\sim}{R}} = m\underset{\sim}{g} + \underset{\sim}{F}_a + \underset{\sim}{F}_j^* \tag{1}$$

where: $m\underset{\sim}{g}$ is the weight;
$\underset{\sim}{F}_a$ is the resultant of aerodynamical forces but does not include the effects on the exchange surfaces;
$\underset{\sim}{F}_j^*$ is the "effective propulsive force" expressed as $\underset{\sim}{F}_j^* = \dot{m}\underset{\sim}{V}^*$,
where $\dot{m}$ is the time derivative of the mass flow through the nozzle;
$\underset{\sim}{V}^*$ is the effective ejection velocity with respect to the nozzle.

It should be noted that the effective ejection velocity is generally given as an experimental datum and includes aerodynamical effects on the exchange surfaces.

For systems with propellers, a similar form of equation is obtained by separating the aerodynamical interactions into conventional terms and "propulsive forces".

The rotational equations (with respect to the mass centre) have the following form:

$$\dot{\underset{\sim}{H}} = \underset{\sim}{L}_a + \underset{\sim}{L}_j^*, \tag{2}$$

where: $\underset{\sim}{H}$ is the moment of momentum with respect to the mass centre;
$\underset{\sim}{L}_a$ is the resultant of aerodynamical moments;
$\underset{\sim}{L}_j^*$ is the effective propulsive torque with $\underset{\sim}{L}_j^* = \underset{\sim}{p}_N \times \dot{m}\underset{\sim}{V}^*$
where $\underset{\sim}{p}_N$ is the effective position of the nozzle with respect to the reference point, here the mass centre.

The moment of momentum (with respect to the mass centre) is a linear function of the body rotational velocity vector $\underset{\sim}{\omega}$ and depends on the (central) inertia matrix of the system, $[I]$, and on the components of the internal angular momentum vector (with depends on the relative angular velocities of the engines), $[h]$; these parameters have to be provided as data (possibly under the form of time functions associated with their time derivatives).

If one takes as reference point of the system a point $\mathcal{P}$, fixed with respect to the structure and different from the mass centre, the position of the mass centre can be written as:

$$\underset{\sim}{R} = \underset{\sim}{P} + \underset{\sim}{p}_0$$

where: $\underset{\sim}{P}$ is the position vector of the reference point;
$\underset{\sim}{p}_0$ is the position vector of the centre of mass with respect to $\mathcal{P}$.

The vector $\underset{\sim}{p}_0$ is generally constant with respect to the body axis and should be provide as a datum. If the position of the centre of mass is allowed to move (with respect to the structure and consequently with respect to the fixed reference point) the first and second time derivatives of the components of $\underset{\sim}{p}_0$ have to be provided as time functions or computed.

In this case, the mechanical equations take the form:

$$m\ddot{\underset{\sim}{P}} = m\underset{\sim}{g} + \underset{\sim}{F}_a + \underset{\sim}{F}_j^* - m\ddot{\underset{\sim}{p}}_0 \tag{3}$$

and

$$\dot{\underset{\sim}{H}}^P = \underset{\sim}{L}_a^P + \underset{\sim}{L}_j^{*P} + \underset{\sim}{p}_0 \times m(\underset{\sim}{g} - \ddot{\underset{\sim}{P}}). \tag{4}$$

It is seen that the complexity of the equations depends on the choice of the reference point and on the possibility to have time varying parameters (choice between rigid and quasi-rigid systems).

A priori, the mechanical equations can be expressed in any reference axis system, but, in general, the rotational equations are expressed in a body fixed axis system and the translational equations in an axis system related to the the air path of the vehicle (see [4]). These choices will be taken here as they have some advantages for the various classes of problems which can be treated in system theory (simulation, control, trajectory optimization, system and environmental parameter estimation...).

KINEMATICS

The main problems, here, are the description of the relative orientations of the various axis systems, the determination of the corresponding angular velocities and the implementation of the various relations between these quantities.

The following axis systems are used:

- the *inertial axis system* - $\{x_0\, y_0\, z_0\}$: this system is assumed to be fixed with respect to inertial space and is used to define the various absolute velocities and accelerations which appear in the mechanical equations;

- the *aircraft carried (local) earth axis system* - $\{x_g\, y_g\, z_g\}$. this system follows the motion of the system reference point, its z_g-axis is aligned with the "apparent" local vertical, the x_g and y_g-axes being respectively aligned with the local north and east directions,

  - for flat earth approximations, this frame is assumed to be inertial, i.e. coincides with the inertial axis system and the position of the system is then given by the corresponding cartesian coordinates, X, Y, Z,
  - for more accurate models, the orientation of the local earth axis system with respect to an earth associated inertial axis system (as well as the position of the reference point of the system) is described by the earth rotation angle and the geographic coordinates of the reference point, i.e. the longitude, μ, the latitude, λ and the altitude, h - in this case the earth model (rotating or non rotationing, spherical or standard) has to be specified and the corresponding parameters have to be provided;

- the *body fixed axis sytem* - $\{x\, y\, z\}$: this axis system, whose x-axis is aligned with the longitudinal axis of the system and z-axis is located in the plane of symmetry, follows the rigid (or quasi rigid) body motion - its rotational velocity vector is described by its components (rates of pitch, p, roll, q, and yaw, r);

- the *air-path axis system* - $\{x_a\, y_a\, z_a\}$: the x_a-axis of this axis system is aligned with the velocity with respect to the atmosphere ($\underset{\sim}{V} = V\, \hat{\underset{\sim}{x}}_a$), the z_a-axis being located in the plane of symmetry - its orientation with respect to the earth axis system is described by the air-path attitude angles (air-path bank, μ_a, track, χ_a, and climb, γ_a, angles);

- the *experimental axis system* - $\{x_e\, y_e\, z_e\}$. the aerodynamical interactions are easily described in this system whose orientations with respect to the body fixed and the air-path axis systems are respectively described by the angle of attack, α, and the sideslip angle, β.

The variables μ, λ, h, (alternatively X, Y, Z) V, α, β, μ_a, χ_a, γ_a, p, q and r will be considered as the system variables; auxiliary variables are used in the programme and can possibly be used as outputs.

The necessary kinematical (and kinetic) quantities (including position, velocities, accelerations, momentum and moment of momentum) are then automatically computed in alphanumerical form (according to the provided assumptions); a detailed description of these procedures in given in [1].

In particular, the velocity of the system is written as the sum of the velocity due to earth rotation (equal to zero in the flat earth approximation), the velocity of the atmosphere with respect to the earth (the wind velocity, $\underset{\sim}{W}$) and the above-defined airspeed velocity, $\underset{\sim}{V}$ or:

$$\dot{\underset{\sim}{R}}^P = \underset{\sim}{\Omega} \times \underset{\sim}{R}^P + \underset{\sim}{W} + \underset{\sim}{V},$$

where $\underset{\sim}{\omega}$ is the earth angular velocity vector.

In this programme, the wind velocity vector is expressed in the local earth axis system and its components are considered as functions of the geographical coordinates and time explicitly, the time derivative of the wind vector also appears in the equations and the corresponding components have to be given or computed from the expressions of the wind components.

DYNAMICS

The various (external) forces and torques have to be described. For each type of interaction, these quantities will be represented by the resultants of forces together with the location of their corresponding "effective" application points and, when necessary, by additional "pure torques".

GRAVITY.- The gravitational interaction is be represented by the global weight applied at the centre of gravity (considered to coincide with the centre of mass). As already stated, the corresponding gravity force density vector,$\underset{\sim}{g}$, is assumed to be aligned with the $\hat{\underset{\sim}{z}}_g$ -axis with $\underset{\sim}{g} = g\, \hat{\underset{\sim}{z}}_g$, the gravitational parameter g can appear as such in the equation, can be replaced by a standard value or can be expressed as a function of the geographical coordinates (a prososed model is included).

PROPULSIVE INTERACTIONS. - The *trust vector* can be described by its components in the body frame (as a function of time). In general this vector belongs to the plane of symmetry of the vehicle and can, alternatively, be given by its norm, F_j^*, and an effective nozzle tilt angle, α_j. The trust can appear as such, be replaced by a nominal value, can be expressed as a function (to be provided) of the system variables such as the velocity and the altitude and other parameters such trottle position and temperature or can be referred to an appropriate file (of experimental data).

As already mentioned, the effective application point is described by the components of the effective nozzle position vector (with respect to the reference point), $\underset{\sim}{p}_N$, this, in general, permits to compute the propulsive torque. Possible jet

pure torques (such as jet damping torques) can be added, but their default value is zero.

AERODYNAMICAL INTERACTIONS. - In principle, the aerodynamical interactions can be obtained by integrating the applied aerodynamical surface force on the complete system.

There resultants, generally described by phenomenological relations, are represented by:
the aerodynamical force vector, $\underset{\sim}{\mathbf{F}}_a$, expressed in the air-path axis system as:

$$\underset{\sim}{\mathbf{F}}_a = X_a^A \hat{\underset{\sim}{\mathbf{x}}}_a + Y_a^A \hat{\underset{\sim}{\mathbf{y}}}_a + Z_a^A \hat{\underset{\sim}{\mathbf{z}}}_a,$$

where: $-X_a^A$ is the drag ($\mathcal{D}$),
Y_a^A is the cross-stream or lateral force,
$-Z_a^A$ is the lift (L),
and the aerodynamical moment (with respect to the mass centre), $\underset{\sim}{\mathbf{L}}_a$, expressed in the body fixed axis system as:

$$\underset{\sim}{\mathbf{L}}_a = L^A \hat{\underset{\sim}{\mathbf{x}}} + M^A \hat{\underset{\sim}{\mathbf{y}}} + N^A \hat{\underset{\sim}{\mathbf{z}}},$$

where: L^A is the rolling aerodynamical moment
M^A is the pitching aerodynamical moment
N^A is the yawing aerodynamical moment
} with respect to the mass centre.

The force and torque components are often normalized. If S is a reference area (generally an equivalent wing surface, S_L), ℓ a reference length (generally the overall length of the aircraft, ℓ_R), ρ the air density (possibly given as a function of altitude and temperature), the aerodynamical interactions can be normalized as:

$$X_a^A = \frac{1}{2}\rho V^2 S C_{Xa}^A, \qquad Y_a^A = \frac{1}{2}\rho V^2 S C_{Ya}^A, \qquad Z_a^A = \frac{1}{2}\rho V^2 S C_{Za}^A$$

and

$$L^A = \frac{1}{2}\rho V^2 S\ell C_\ell^A, \qquad M^A = \frac{1}{2}\rho V^2 S\ell C_m^A, \qquad N^A = \frac{1}{2}\rho V^2 S\ell C_n^A.$$

The corresponding aerodynamical coefficients are, generally, functions of the following quantities:

$$C_+^A = C_+^A(\alpha, \dot{\alpha}, \beta, p, q, r, \xi, \eta, \zeta, M)$$

where: M is the Mach number;
ξ, η, ζ are, respectively the deflections of the roll, pitch and yaw motivators, with in general:
$\xi = \delta_\ell = \frac{1}{2}(\delta_{a_L} - \delta_{a_R})$ where δ_a are the aileron rotation angles (positive for the ailerons down);
$\eta = \delta_m = -\delta_e$ where δ_e is the elevator rotation angle (positive for the elevator down);
$\zeta = \delta_n = -\delta_r$ where δ_r is the rudder rotation angle (positive for the rudder left).

These coefficients may appear as such, be expressed as linear function of the pertinent variables (for which the coefficients have to be provided) or can refer to appropriate data files.

In particular, the most important linear terms of these function can be written as:

$$C_{Xa}^A = C_{Xa_0}^A + C_{Xa_\alpha}^A \Delta\alpha + C_{Xa_\xi}^A \xi + C_{Xa_\eta}^A \eta + C_{Xa_\zeta}^A \zeta + C_{Xa_F}^A \Delta F_j^*$$

$$C_{Ya}^A = C_{Ya_\beta}^A \beta + C_{Ya_p}^A p + C_{Ya_r}^A r + C_{Ya_\zeta}^A \zeta$$

$$C_{Za}^A = C_{Za_0}^A + C_{Za_\alpha}^A \Delta\alpha + C_{Za_{\dot\alpha}}^A \dot{\alpha} + C_{Za_q}^A q + C_{Za_\eta}^A \eta + C_{Za_F}^A \Delta F_j^*$$

$$C_\ell^A = C_{\ell_\beta}^A \beta + C_{\ell_p}^A p + C_{\ell_r}^A r + C_{\ell_\xi}^A \xi + C_{\ell_\zeta}^A \zeta$$

$$C_m^A = C_{m_\alpha}^A \Delta\alpha + C_{m_{\dot\alpha}}^A \dot{\alpha} + C_{m_q}^A q + C_{m_\eta}^A \eta + C_{m_F}^A \Delta F_j^*$$

$$C_n^A = C_{n_\beta}^A \beta + C_{n_p}^A p + C_{n_r}^A r + C_{n_\xi}^A \xi + C_{n_\zeta}^A \zeta$$

where the various coefficients are estimated for a nominal value of the state variables, in particular α_0, V_0, F_{j0}^* and can vary with other parameters (such as altitude, Mach number...) with $\Delta\alpha = \alpha - \alpha_0$ and ΔF_j^*, the trust variation.

Aerodynamical torques can be given with respect to other points (for instance with respect to the aerodynamical centre $\mathcal{A}$, which is a point around which the pitching moment does not vary with the angle of attack). The torque transferts between points can then, automatically, be handled (when the corresponding position vectors with respect to the reference point are given).

PROGRAMME STRUCTURE

The main purpose of this programme itself is to provide a literal expression (character strings) of the dynamical equations of the system.

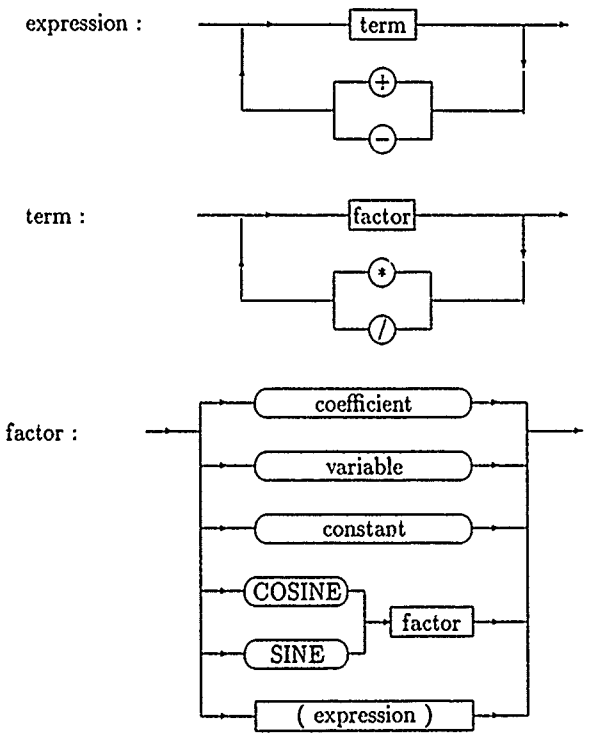
The process which writes mathematical expressions in a symbolic form can be divided into four parts :

- the identification of an expression according to a predefined syntax,
- the translation of such an expression into a programming language,
- the manipulation (+, -, *, /) of the expressions,
- the reduction.

SYNTAX

Each mathematical expression contains one or several terms linked by minus or plus signs, each term being the product of several factors. A factor can be an coefficient, a variable, a constant, a trigonometrical function whose argument is itself a factor or a mathematical expression between brackets.

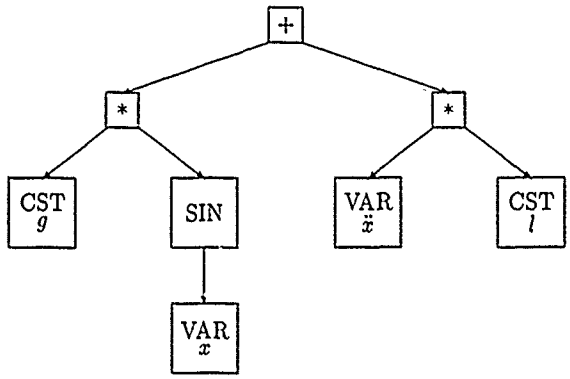
This can be schematically represented as follows [5]:



So defined, an expression can be considered as a tree structure whose nodes represent the *nature* of the expressions, for instance:

- a coefficient whose value is given as a floating point constant;
- a variable or a constant which is to be identified by an appropriate string of characters,
- a trigonometric function whose argument (also an expression) must be given,
- plus, minus, times operators whose two operands are given as expressions.

As an example, the expression $g * \sin(x) + \ddot{x} * l$ could be represented by the following tree:



The increase in size of the expressions during the computation of the equations and the above defined tree structure lead us to use dynamical variables (pointers) and an appropriate programming language for this kind of manipulations, the present choice is the C- language.

The use of dynamical variables reduces the DATA size of the programme and then permits its use with personal computers; nevertheless, it is clear that the number of pointers created during the computation may not exceed the maximum allowed by the memory.

MANIPULATIONS AND SIMPLIFICATIONS

The following operations have to be performed for the various expressions :

- creation of pointers and assignment of their nature,
- reading, initialisation or assignment of the value and/or identification for the coefficients, variables and constants,
- assignment of the arguments and operands to the appropriate pointers for the trigonometrical functions and the arithmetical operations,
- disregard of redundant calculations (adding 0, multiplying by 0 or 1),
- generation of auxiliary variables for long expressions.

For instance, if we know the expressions "a" and "b", the operation "a+b" consists in creating a new pointer whose nature is PLUS, whose left operand points to "a" and right one points to "b".

In order to avoid expressions like :

$$x = a + b - a$$

a relation of order is introduced. This order is given first by the *natures* and then by a lexicographical order on the strings of characters for instance, $0 < \cos(x) < \sin(x) < m * l < x+y$. After each operation, the resulting expression is rewritten according to the prescribed order and consecutive equivalent terms with opposite signs are cancelled and withdrawn. This way, we can ensure that the expressions cannot be further reduced without using specific trigonometric formulae, this kind of reductions can be performed by several programmes (MAPLE, MATHEMATICA, MACSYMA, REDUCE, ...) which can subsequently be used.

PROGRAMME IMPLEMENTATION

In a first step, the user is asked to choose between a certain number of fundamental options:

- *Earth shape*

  flat : In this case the position of the system is given by the cartesian coordinates X, Y, Z.
  The angular velocity of the earth (ω_0) and the angular velocity of the aircraft carried local axis system (ω_{g_i}) are equal to zero.
  The gravity (g) is considered as a function of the altitude h only ($g \equiv g(h)$).

  spherical : In this case, the earth axes a and b are equal to the radius of earth and the gravity will also be considered as a function of the altitude h only - the position is given by the actual geographical variables.

ellipsoidal : All the earth parameters have to be specified and auxiliaty geographical variables ε.

- *Reference point*

  The reference point $\mathcal{P}$ could be chosen as:

  - the mass centre - this choice implies that the vector $\underset{\sim}{p}_0$ and its time derivatives are identically equal to zero and that the inertia matrix $[I^P]$ is the central inertia matrix $[I]$;
  - a given point of the structure - in this case the vector $\underset{\sim}{p}_0$ and its time derivatives have to be specified by the user.

- *Aircraft configuration*

  - the mass of the aircraft (m) can considered as constant ($\Rightarrow \dot{m} = 0$) or as a time function to be specified later;
  - the inertia matrix can be considered as a constant or as a function of time - the choice of constant configuration also implies that the internal angular momentum $[h]$ and its time derivative $[\dot{h}]$ are equal to zero, as well as, when appropriate, the time derivatives of the relative position of the centre of mass;
  - Further, the inertia matrix can be chosen as a diagonal or semi-diagonal matrix, these assumptions leading to considerable simplifications in the final form of the equations.

- *Wind*

  The user could choose between the four following possibilities :

  No wind - that is $\underset{\sim}{W}=0$ and $\dot{\underset{\sim}{W}}= 0$,

  Constant - the components of $\underset{\sim}{W}$ (in the local axis system) depend only the location and are expressed, according to the previous choice, in terms of geographical or cartesian coordinates, i.e. $\underset{\sim}{W} = \underset{\sim}{W}(\lambda, \mu, h)$ or $\underset{\sim}{W} = \underset{\sim}{W}(X, Y, Z)$;

  Uniform - the components of $\underset{\sim}{W}$ in local axis system do not depend on geographical coordinates and time;

  Variable - the vectors $\underset{\sim}{W}$ and $\dot{\underset{\sim}{W}}$ explicitly depend on the position of the aircraft and time.

The equations of motion can be further simplified when additional assumptions are used. In some cases, it may be considered that some of the variables of the system are perfectly controlled (by the pilot or an automatic control device) or measured during the motion. These variables can then be considered as known functions of time; the corresponding dynamical equations are then discarded and the order of the system is reduced accordingly. In particular, one can consider trajectories for which only certain variables are involved, this is the case for planar trajectories (in route motion, constant climbing and descent).

In a second step, the user is asked to choose the variables he wants to deal with. According to this procedure, a variable is selected as a state variable (is considered in the state vector) or as a parameter (in this case, the corresponding state equation is discarded and the corresponding "parameter" has to be specified - possibly as a function of the remaining state and time).

As an example, for a flat earth choice, the equations of planar trajectories are obtained by the following selection of variables:

Name of the variable	representation	VARIABLE	PARAMETER
cartesian coordinates	X	X	
	Y		X
	Z	X	
airspeed	V	X	
angle of attack	alpha	X	
sideslip angle	beta		X
bank angle	Mu_a		X
track angle	Xi_a		X
climb angle	Ga_a	X	
rate of roll	p		X
rate of pitch	q	X	
rate of yaw	r		X

The third step allows the user to define all the parameters (including the discarded variables) as a numerical value, as a string of character (the proposed name, any new name, the name of a function of other parameters or variables). Clearly,

if a parameter is define as zero (0), it will not appear in the equations. For the above considered planar trajectories, the discarded variables can be considered, in this step, to be equal to zero or:

Name of the parameter	representation	VALUE	NEW NAME
cartesian coordinates	Y	0	
sideslip angle	beta	0	
	⋮		
gravity	g		g(h)
	⋮		

After all the variables and parameters have been specified, the programme generate the equations for the described problem.

RESULTING EQUATION SYSTEMS AND APPLICATION

In general, the final form of the equation takes the form of a system of differential equations associated with algebraic output relations, i.e. in matrix notation:

$$[M][\dot{q}] = [F(q, \delta, t)] \qquad [x] = [x(q, t)]$$

where $[q]$ is the state variable vector,
$[M]$ is a positive matrix,
$[F]$ is a vector with depend on the variables, the controls and time,
$[x]$ is the output vector.

As already mentioned, the output of the system may include other unknowns such as position with respect to a particular point of the earth surface (for which various approximations can be given), trajectory parameters with respect to the ground (rather than with respect to the atmosphere).

In general, the matrix $[M]$ is a diagonal matrix (or at least a block diagonal matrix with small dimensional blocks) and can be easily inverted (possibly in alphanumerical form). The system has then the following standard form :

$$\dot{q} = f(q, \delta, t) \qquad x = x(q, t), \tag{5}$$

and can immediately be used for system analysis applications (simulation, stabilization, control, optimization...).

The particular form of the syntax used in the programme also permits to classify the various terms with respect to a certain number of parameters, say $[\pi]$, (for instance parameters whose values are not perfectly known). The equation are generally linear with respect to these parameters and an identification model can then be obtained under the form.

$$[\phi][\pi] = [\Phi]$$

where the matrices $[\phi]$ and $[\Phi]$ are functions of the state of the system (state variables and their time derivative), the controls and, possibly, time explicitly. If the matrix $[\phi]$is a full rank matrix (this implies that the parameters are linearly independent and that the considered trajectory is well selected - permanently exciting), the selected parameters can be identified in a rather straightforward manner.

As an other example of problems that can be treated, let us consider one particular optimal trajectory problem for which additional information on the system is required. Let us assume that, for a fixed terminal time, we have to minimize the following cost function:

$$J = \varphi[q(t_f), t_f] + \int_{t_0}^{t_f} L[q(t), \delta(t), t]\, dt$$

for a system which satisfies the equations 5 with the initial conditions $x(t_0) = x_0$.

It is known (see for instance [6]), the control function $\delta(t)$ which solves this problem is obtained as the solution of the following system:

$$\dot{q} = f(q, \delta, t) \qquad \dot{\lambda} = -\left(\frac{\partial f}{\partial q}\right)^T \lambda - \left(\frac{\partial L}{\partial q}\right)^T \tag{6}$$

with:

$$\left(\frac{\partial f}{\partial \delta}\right)^T \lambda + \left(\frac{\partial L}{\partial \delta}\right)^T = 0 \tag{7}$$

and the boundary conditions:

$$x(t_0) = x_0 \quad \text{and} \quad \lambda(t_f) = \left(\frac{\partial \varphi}{\partial q}\right)^T$$

In is seen that the solution of this problem implies the knowledge of the gradients (with respect to the state variables and the controls) of the various terms of the cost function and of the right-hand sides of the equations of motion. This means that these gradient have to exist or to be approximated in a continuous manner; this generally implies further additional treatment of the data files.

Theses procedures are rather lengthy. Nevertheless, the structure of the alphanumerical programme permits to obtain these gradients and to construct the appropriate equations in a rather simple manner. The obtained system, being in general nonlinear, has not, necessarilly, explicit closed form solution and has to be solved numerically. The obtained alphanumerical form of the equations permits to reduce the number of numerical operations that have to be performed at each step of the numerical procedure and further the investigation of the influence of the variation of the parameters is greatly simplified.

EXAMPLE

As an example of the data input, we have considered the simple problem of planar trajectory determination with the flat earth approximation. The output gives the right-hand sides of the mechanical equations (first equation 6) and, when requested, the corresponding gradients appearing in the second equation 6 and in the relation 7 of the above-considered optimization problem. The output can then be integrated in a numerical programme; this permits to obtain the solution of the considered optimal control for a given cost function.

CONCLUSIONS

The presented programme permits to automatically develop the equations of motion of aerospace systems in various forms according to the options provided by the user. The obtained relations can be presented under the various forms which are appropriate for different applications in system dynamics (such as simulation, parameter identification or trajectory optimization). In the actual presentation, the various parameters can appear under the formof a desired name, a desired numerical value, a given function of the system variables or a reference to a data file. Known (perfectally controllled or measured) variables can be selected, the corresponding equations are then discarded and the variables considered as time functions (as new parameters).

REFERENCES

[1] Willems, P.Y., Aircraft dynamics for air traffic control, in: *Air Traffic Control* (A. Benoît, ed.), Agardograph (to be published)

[2] International Standard, *Flight dynamics - Concepts, quantities and symbols*, ISO 1151, 3d edition, 1985.

[3] Decuyper, M., J. Deutsch and P.Y. Willems, *Bases physique de la mécanique*, Masson éd., Paris,1981.

[4] Etkin, B., *Dynamics of atmosheric flight*, Wiley, 1972.

[5] Tremblay, J.-P. and P.G. Sorenson, *An introduction to data structures with applications*, McGraw-Hill, New York, 1984.

[6] Bryson, A.E. and Y.C. Ho, *Applied optimal control*, Blaisdell publishing Co., Waltham, 1969.

ACKNOWLEDGMENT

The results presented in this paper have been obtained within the framework of the Belgian Programme on Concerted Research Actions and on Interuniversity Attraction Poles initiated by the Belgian State, Prime Minister's Office, Science Policy Programming. The scientific responsability rests with the authors.

OUTILS FORMELS ET OUTILS DE SIMULATION

UN ATELIER COHERENT

-:-:-:-:-:-:-

Monsieur Patrick SCHIRLE
Responsable méthodes à la division systèmes avioniques

AVIONS MARCEL DASSAULT
78, Quai Marcel Dassault
92 SAINT-CLOUD - FRANCE

1 - RESUME

Les systèmes avioniques atteignent aujourd'hui un très haut niveau de complexité et représentent une part de plus en plus importante du coût des aéronefs. Leur évolution au cours de la dernière décade peut s'analyser sur les plans opérationnel, technologique et méthodologique.

La recherche d'efficacité opérationnelle et de polyvalence entraîne une intégration de plus en plus serrée des fonctions, qui se traduit par l'optimisation des ressources physiques et humaines (fusion capteur, ergonomie, systèmes experts).

L'évolution technologique se traduit d'une part par des architectures fonctionnelles complexes, intégrant des systèmes historiquement indépendants (navigation, commandes de vol, moteur, carburant, etc...) ainsi que par l'introduction massive du logiciel dans les équipements.

L'évolution méthodologique enfin, est la conséquence directe des évolutions opérationnelles et technologiques dans le but de conserver la maîtrise du développement de tels systèmes. L'assurance qualité système repose aujourd'hui sur la mise en oeuvre et le respect de méthodologies qui décrivent l'enchaînement des activités de conception des systèmes et de leur logiciels. De plus, en fixant précisément les tâches de tous les intervenants, elles permettent d'utiliser au mieux les compétences de nombreux partenaires au sein d'organisations industrielles de plus en plus vastes.

Pour améliorer leur efficacité, ces méthodologies s'appuient sur de nombreux outils informatiques rassemblés en ateliers cohérents. Selon leur place dans le cycle de développement, ils représentent des aides aux activités de conception, de vérification, de validation ou de contrôle qualité. Certains sont des outils de formalisation basés sur les techniques de traitement de l'information (aide à l'analyse fonctionnelle, à la spécification ou à la gestion), d'autres sont des outils de fond basés sur les techniques de simulation (aide à la conception, à la vérification/validation).

Lors des nombreuses étapes de conception, la simulation permet une vérification continue du contenu technique des différentes spécifications.

Les caractéristiques des simulations sont différentes d'une étape à l'autre.

Les simulations les plus amont permettent de choisir ou de confirmer des hypothèses de développement en démontrant leur opérabilité.

Plus tard dans le cycle, la simulation représente une image vivante du cahier des charges (modélisation de "ce que le système doit faire").

Enfin, une simulation de comportement du système est réalisée à partir des spécifications fonctionnelles des constituants. Possédant à ce niveau une architecture fonctionnelle identique à celle du futur système, cette maquette sera progressivement enrichie par simulation de caractéristiques fonctionnelles et temporelles de chaque équipement et rendue hybridable par l'adjonction d'interfaces lui permettant de dialoguer avec des équipements réels.

Ce dernier outil de simulation hybridable sera utilisé lors des étapes de validation des équipements (tests de recette) et du système global.

La communication décrira les méthodes et moyens mis en place par la société AMD-BA pour le développement des systèmes avioniques dont elle est l'architecte industriel. L'accent sera mis particulièrement sur les différentes techniques et outils de simulation et leur intégration dans un atelier système complet et cohérent.

2 - HISTORIQUE ET CONTEXTE

L'activité "système" représente aujourd'hui une branche maîtresse de l'industrie aéronautique. Son essor au cours des 20 dernières années a été prodigieux et modifie le paysage aéronautique, créant de nouveaux métiers et utilisant de nouvelles compétences.

L'évolution du besoin opérationnel a entraîné l'accroissement de la polyvalence des systèmes, se traduisant par une intégration de plus en plus serrée des fonctions au sein d'un même système et entre systèmes avion ou sol (le Système est alors la somme de plusieurs systèmes embarqués, ou non).

L'intégration des différentes fonctions est réalisée dans le but d'obtenir une efficacité opérationnelle maximale par :

- l'optimisation des ressources physiques (capteurs, actionneurs, organes de traitement de l'information) grâce à la fusion de données et aux réseaux d'échanges d'informations entre avions et/ou infrastructures terrestres et maritimes.
- l'optimisation des ressources humaines, grâce à une ergonomie particulièrement soignée de l'interface homme/machine, assurant un dialogue de haut niveau avec les pilotes, le système sélectionnant lui-même les informations utiles à chaque phase de la mission et les présentant sous la forme synthétique la plus appropriée.

Le caractère hautement évolutif des systèmes s'affirme de plus en plus. L'enveloppe opérationnelle doit pouvoir évoluer facilement : intégrer de nouvelles fonctions sans modifier la mise en oeuvre des précédentes ou améliorer les fonctions pré-existantes au travers des évolutions technologiques.

Sur le plan technologique les systèmes avioniques se caractérisent principalement :

- par l'utilisation de nouveaux capteurs utilisant des techniques telles que laser, Infrarouge, CCD etc...
- par une électronique de plus en plus compacte à base de circuits VLSI ou ASIC
- par des architectures fonctionnelles et matérielles de plus en plus complexes intégrant des systèmes historiquement indépendants tels que moteurs, commandes de vol, carburant ou freinage et systématisant l'emploi de liaisons numériques multiplexées entre équipements.
- par l'introduction massive du logiciel, apportant une souplesse et une ouverture considérables mais induisant des problèmes spécifiques dont la maîtrise s'avère encore aujourd'hui difficile.

Schématiquement, on peut noter trois stades dans l'évolution :

- des systèmes décentralisés, pas ou peu intégrés :

Dans les années 60, les systèmes avioniques sont constitués d'équipements spécifiquement aéronautiques tels les radars, les centrales anémométriques ou inertielles, ou les équipements de planche de bord, faiblement interfacés au moyen de liaisons analogiques dédiées. Chaque équipement représente alors le support d'une fonction opérationnelle bien identifiée (pilotage, navigation, interception, etc...), l'analyse fonctionnelle du système se limitant à l'affectation naturelle de ces fonctions aux boîtes noires correspondantes et en la définition des interfaces de servitude entre ces boîtes.

Au début des années 70, le logiciel prend pied dans ces équipements aeronautiques, améliorant notablement leurs capacités propres. La conception du logiciel est alors abordée dans un cadre restreint à chaque équipement, indépendamment de la conception du système lui-même.

- des systèmes centralisés et intégrés :

Quelques années plus tard, apparaît dans les architectures de systèmes une nouvelle race d'équipements : les calculateurs "purs". Ces équipements, non attachés à une fonction opérationnelle particulière ou à une ressource liée à la physique, supportent les traitements numériques de l'information, donc du logiciel. Ils assurent, de façon centralisée, l'intégration et l'affectation de toutes les ressources du système en vue d'une plus grande efficacité opérationnelle. Leur logiciel, baptisé logiciel système, représente une couche fonctionnelle amont par rapport au logiciel spécifique des équipements. Il est également le fruit d'une démarche de conception différente, sa définition est issue de l'analyse du système dans sa globalité et non plus de l'analyse particulière d'un équipement ou de la fonction qui lui est attachée.

- des systèmes décentralisés et intégrés :

La tendance actuelle pour le développement des systèmes est mixte : on assiste à une décentralisation des traitements, le logiciel système étant notamment réparti entre plusieurs équipements spécifiques ou non, mais également à une intégration puisque chaque fonction opérationnelle n'est satisfaite qu'au travers d'une collection de modules fonctionnels implantés dans de nombreux équipements. Leur développement implique une analyse fonctionnelle particulière, aboutissant à une architecture fonctionnelle distincte de l'architecture matérielle, le rôle opérationnel particulier de chaque boîte noire ne se dégageant plus de façon évidente.

Les systèmes avioniques embarqués actuellement développés par les Avions Marcel Dassault comportent plus d'une centaine d'équipements, dont la moitié sont fortement numérisés et dont la majeure partie sont fonctionnellement dépendants de logiciels. Le volume de logiciel système se chiffre en mega-octets, le nombre d'informations échangées entre équipements et/ou modules fonctionnels dépasse 30.000 et le débit d'informations sur les bus numériques est de plusieurs méga-bits par seconde.

La révolution méthodologique de ces dernières années est la conséquence nécessaire des évolutions opérationnelles et technologiques dans le but de conserver la maîtrise du développement de ces grands systèmes.

Parmi les facteurs les plus marquants, on peut citer :
- le rôle prépondérant de l'assurance de la Qualité en conception
- la modification des organisations industrielles
- les problèmes spécifiques du logiciel
- la généralisation de l'outillage informatique d'aide au développement

- Assurance de la qualité en conception : l'importance et le volume des travaux de conception attachés au développement des systèmes, la difficulté accrue de mesurer la qualité (particulièrement la sûreté de fonctionnement) du produit livré font remonter les activités d'assurance de la qualité du niveau de la fabrication au niveau de la conception. La qualité du produit est de plus en plus démontrée par la qualification des méthodes utilisées pour son développement, et non plus au travers du produit lui-même. Des documents d'assurance de la qualité des systèmes ou des logiciels sont les témoins de cette évolution.

- Organisation industrielle : la taille des systèmes actuels implique la mise en commun des ressources et des compétences réparties dans de nombreuses sociétés industrielles. (Le nombre d'intervenants dans le développement d'un système avionique de MIRAGE 2000 dépasse 25.000...). Seule une méthodologie rigoureuse peut servir de support aux nouvelles organisations industrielles en permettant :

  * de définir les tâches et responsabilités de tous les intervenants
  * d'assurer un développement harmonieux en renforçant la visibilité et la traçabilité

- Les problèmes spécifiques du logiciel : la spécification d'un logiciel consiste, à partir d'un besoin exprimé en terme de "service à rendre", à affiner au cours d'étapes successives et selon un processus itératif, l'expression écrite de ce besoin jusqu'à lui donner une forme directement interprétable par une machine informatique : le code. La production d'un logiciel ne concerne que sa compilation et sa reproduction. Les travaux de conception du logiciel sont de même nature que les travaux de conception du système, l'ensemble devant donc être le fruit d'une démarche méthodologique continue. En conséquence, les méthodologies de développement des logiciels seront cohérentes de la méthodologie de développement du système. Il faut enfin noter que le travail de spécification inhérent à un composant logiciel du système représente la somme des travaux de spécification à toutes les étapes de développement (système puis logiciel), relativement à ce composant. Cette constatation illustre la difficulté du problème de propriété du logiciel.

- L'outillage informatique d'aide au développement : les méthodologies de développement du logiciel dans une première phase puis du système aujourd'hui sont supportées par des outils informatiques de plus en plus nombreux et de plus en plus sophistiqués. Ces outils sont des aides à la conception, à la vérification, ou à la validation et sont regroupées en ateliers. L'apparition du logiciel a conduit à définir et à mettre en place des ateliers logiciels aujourd'hui nombreux et variés mais supportant des méthodes très voisines dans leurs principes. L'émergence de l'approche système, plus récente, a créé le besoin d'ateliers système, chapeautant les ateliers logiciels et supportant les travaux de conception des systèmes en amont de la réalisation des équipements.

3 - L'APPROCHE SYSTEME

Dès les étapes les plus amonts de la conception, les systèmes anciens pouvaient être découpés a priori en entités autonomes telles que radio-communication, radio-navigation, commandes de vol, contrôle moteurs ou telles que radar, centrale aérodynamique, etc... chacune de ces entités faisant l'objet d'une conception séparée. L'intégration, la décentralisation et la complexité ont modifié les règles du jeu.

Aujourd'hui, une approche globale du système, chapeautant toutes les approches "individuelles" de ses composants (grands ensembles, équipements, logiciels...) est devenue strictement nécessaire pour garantir les performances et la qualité.

Cette approche système, pour être efficace, passe par la mise en place d'une méthodologie système, définissant de nombreuses étapes de développement, amont des étapes de conception des équipements. Elle permet de "raffiner" et de spécialiser peu à peu les travaux de développement sans perdre de vue les objectifs initiaux exprimés au niveau du système global.

Elle implique le mariage de concepteurs de compétences différents, depuis le généraliste chargé du premier niveau de découpage du système en grandes fonctions, jusqu'au spécialiste logiciel ou matériel, en passant d'une vue générale et synthétique du système à des vues progressivement plus limitées et détaillées mais nécessairement cohérentes.

4 - METHODOLOGIE DE DEVELOPPEMENT DES SYSTEMES, ATELIER SYSTEME ET ATELIERS LOGICIELS

Dans ce contexte particulièrement évolutif, la société des Avions Marcel Dassault Bréguet Aviation a dû, à partir de son expérience des systèmes et en consentant de lourds investissements s'adapter rapidement pour conserver sa maîtrise des systèmes avioniques complexes :

- En définissant une méthodologie originale de développement de systèmes,
- En définissant et en réalisant l'atelier système correspondant
- En participant à la définition et à l'évaluation des ateliers logiciel.

La méthodologie système définit précisément les étapes du développement avec les tâches, produits et moyens qui leur sont associés. Y sont décrites les étapes de conception (branche descendante du "V" représentant le cycle de développement) et les étapes symétriques de validation (branche remontante du "V") intégrant ainsi les méthodologies propres au développement des équipements (matériel et logiciel).

L'application pratique de cette méthodologie pose un certain nombre de problèmes techniques, organisationnels et humains. En effet, elle implique une parcellisation des activités de conception avec répartition des tâches vers des concepteurs nombreux et de compétences diverses. De plus, les produits issus des étapes de conception ne sont en fait que des documents, réutilisés en aval dans le cycle.

Pour tirer le maximum d'efficacité de cette méthodologie basée sur l'effet collectif sans entamer la motivation de chacun des acteurs, il faut tenter de ramener chaque travail individuel à une notion de "produit fini" et donc donner à chaque ingénieur, non seulement les informations qui lui sont nécessaires, mais également et surtout les moyens d'analyse, de formalisation et de vérification de sa part de travaux : c'est le rôle de l'atelier système.

Cet atelier est composé de nombreux outils informatiques que l'on peut regrouper en deux familles :

- les outils formels
- les outils de simulation

Tous les outils possèdent des caractéristiques communes, nécessaires à leur fonctionnement en atelier : interconnectabilité, multi-utilisations, multi-versions, environnement informatique compatible. Les différents outils s'insèrent dans une structure d'accueil unique qui assure les fonctions générales suivantes :

- initialisation de projet
- gestion du plan de développement
- suivi des évolutions
- gestion de configuration au niveau système
- intégration des outils spécifiques de l'atelier (par des échanges standard d'informations)
- administrateur gérant les droits d'accès.

Plus particulièrement, elle permet :

- la production de l'ensemble de la documentation de définition et de spécification du système par composition des produits fournis par les différents outils.

- la gestion du suivi des évolutions et la garantie de cohérence de l'ensemble des produits par la le contrôle de l'application des mécanismes d'évolution intégrés dans les différents outils de l'atelier. Ceci permet l'instruction des modifications sous outil (en conservant l'état "avant"), la rédaction des fiches d'évolution et la mise à jour automatique de l'ensemble des documents.

Outils formels :

L'ensemble des outils formels représente une aide à l'application rigoureuse de la méthodologie par l'amélioration des communications, la formalisation des échanges et la gestion de la documentation de spécification. Certains outils sont d'un emploi général tels l'outil de traitement de texte SCRIBE ou l'outil de spécification graphique MITIA.

D'autres outils sont dédiés à des techniques particulières et regroupés en "sous-ateliers". C'est le cas en particulier des outils "sûreté de fonctionnement" rassemblés dans le sous-atelier AXE et qui permettent de supporter les analyses de sécurité, de construire des arbres de défaillance et d'établir les équations de probabilité des évènements.

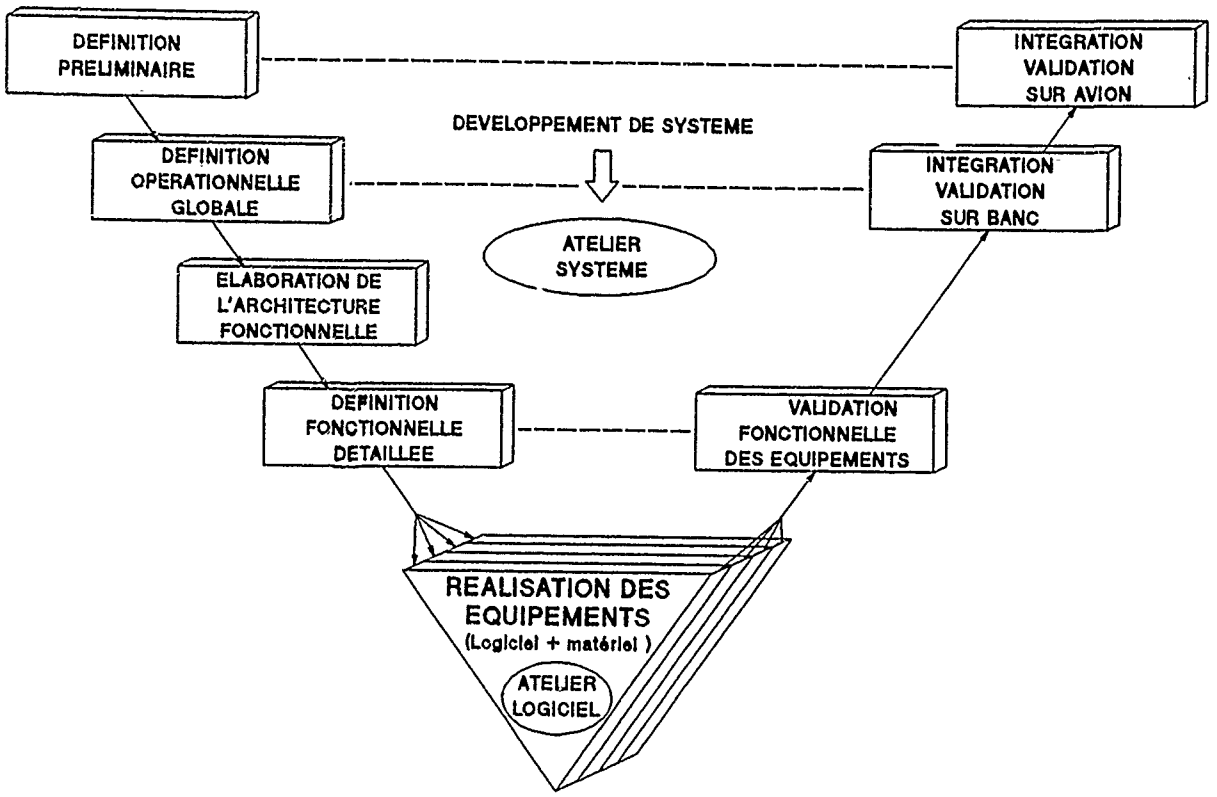
Enfin certains outils sont spécifiques des tâches d'une étape de développement donnée ; leur rôle sera précisé au chapitre suivant.

Outils de simulation :

Le rôle principal des outils de simulation est de supporter les activités de vérification au niveau des différentes étapes de définition et de spécification. Ils sont également utilisés en tant qu'aide à la conception en permettant à l'ingénieur un retour "vivant" et rapide du fruit de ses travaux, permettant ainsi une approche conception/simulation/conception itérative.

Leur principale caractéristique commune est la portabilité multi-sites permettant, au prix d'un effort de normalisation, de disposer d'éléments de simulation assemblables en fonction de la nature de l'application recherchée. On trouvera une description des différentes applications au chapitre suivant.

5 - ETAPES DE LA METHODOLOGIE DE DEVELOPPEMENT



5.1 - Définition préliminaire

Le rôle de cette étape est de définir à partir des objectifs contractuels et de l'expérience, le cadre dans lequel le système sera construit grâce à une analyse des missions demandées et une première étude de l'organisation du système.

L'analyse des missions s'appuie sur diverses études de faisabilité et pré-études de performance qui permettront de s'assurer de la faisabilité globale du système en termes de délais et de performances opérationnelles. Elle se concrétise par une liste des fonctions opérationnelles nécessaires à la réalisation de l'enveloppe des missions, des documents décrivant les concepts d'utilisation et fait ressortir un ensemble de besoins concernant particulièrement les modes de fonctionnement et les performances des principaux capteurs.

L'étude d'organisation du système conduit à une première définition de l'architecture matérielle et géographique du système (liste d'équipements et d'emports, organisation matérielle du poste de pilotage, aménagement des soutes, etc...).

Outils formels :

- outils généraux
- outil de C.A.O. (Catia)

Outils de simulation :

- L'outil employé, SAMOS, permet d'illustrer sous une forme vivante, les concepts établis et donc d'assister les concepteurs dans leur tâche de création en évitant le "vertige de la feuille blanche". Composant une bibliothèque de modèles (avion, capteur, dispositifs sol), il permet d'évaluer des scénarii opérationnels particuliers. Chaque "cas de simulation" est défini et réalisé par une équipe intégrée concepteur/simulateur.

5.2 - Définition globale du système

L'étape de définition globale consiste à définir précisément les services que le système doit rendre (et non pas comment le système sera construit).

Le résultat de l'étape est une description, en terme de scénario opérationnel (donc vu de l'utilisateur), de la mise en oeuvre et du fonctionnement nominal du système pour toutes les fonctions qu'il assure. Cette description se traduit par deux types de documents.

- Documents de règles générales

Ces documents décrivent de façon unique, les règles et philosophies d'utilisation applicables à toutes les fonctions opérationnelles dont est et sera doté le système. Ils garantissent ainsi la mise en oeuvre cohérente du système, relativement à chaque fonction opérationnelle. Véritable structure d'accueil opérationnelle pour les fonctions que le système recevra, l'ensemble de ces documents servira de cadre à l'écriture des spécifications globales et de base à l'analyse fonctionnelle du système.

Exemples :
. Règles générales d'utilisation des commandes et visualisations : critères de multiplexage des commandes, règles d'affectation des visualisations, menus des postes de commande banalisés, etc...
. Règles générales de superposition des fonctions opérationnelles : sélection, superposition, exclusion, mémorisation, etc...
. Règles générales de signalisation des pannes et anomalies : niveaux et principes de signalisation, cohérence des messages.
. Règles générales de maintenance
. Etc...

- Documents de spécification globale.

Chaque document de spécification globale décrit le scénario d'utilisation du système relativement à une fonction opérationnelle donnée. Chaque fonction opérationnelle fait donc l'objet d'une spécification qui est écrite dans le respect des règles générales. Ces spécifications étant indépendantes, elles peuvent être élaborées de façon autonome et asynchrone. La cohérence et l'indépendance des spécifications g obales sont assurées par les règles générales.

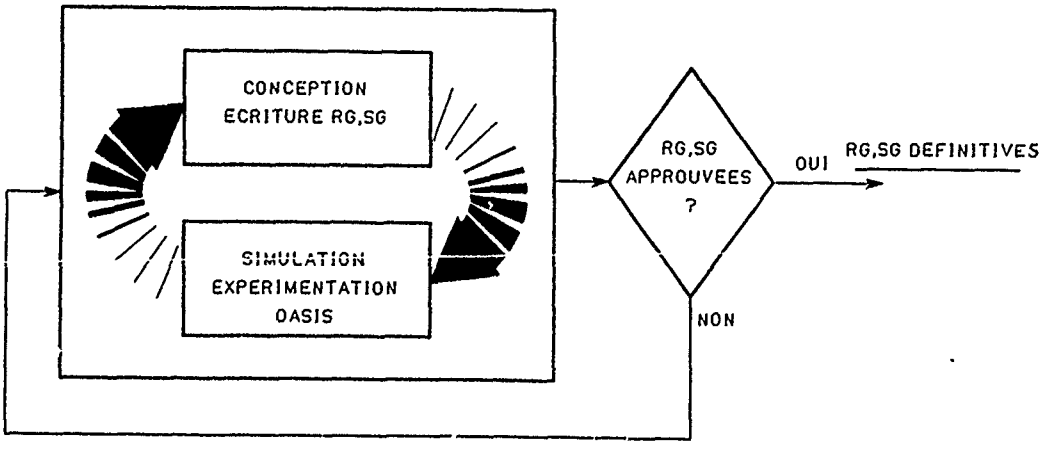
Outils formels :

- outils généraux

Outils de simulation :

- Outil SAMOS
- L'outil OASIS est utilisé comme outil d'aide à la conception et d'aide à la vérification. Construit autour d'une maquette représentative du poste de pilotage, il permet une simulation dynamique en temps réel des commandes, séquences et visualisations, permettant ainsi une vérification efficace des scénarii décrits dans les documents de règles générales et de spécifications globales, directement par les utilisateurs.

A cette étape les outils SAMOS et OASIS sont utilisés selon le principe suivant :



5.3 - Elaboration de l'architecture fonctionnelle

Le rôle de cette étape est de procéder à l'analyse fonctionnelle du système et d'en déduire son architecture fonctionnelle. Le dossier d'architecture fonctionnelle (produit de l'étape) décrit la solution apportée aux besoins exprimés à l'étape de définition globale.

L'étape se décompose en deux phases :
- la construction du graphe d'architecture fonctionnelle du système
- la projection de l'architecture fonctionnelle sur l'architecture matérielle.

a) Construction du graphe d'architecture fonctionnelle

La méthode utilisée consiste en une décomposition hiérarchique progressive du système en éléments fonctionnels, selon des critères précis. Chaque niveau successif de décomposition entraîne :

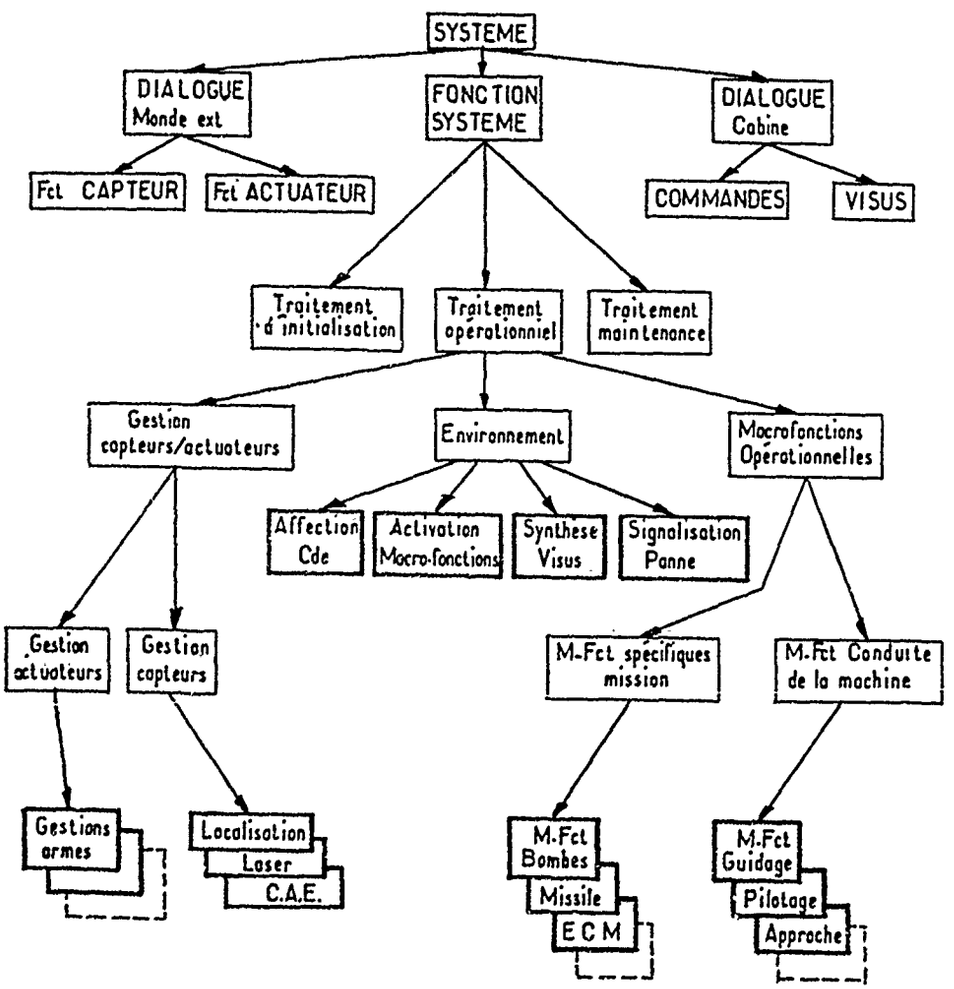
- une justification du découpage réalisé permettant la compréhension et l'approbation des choix et décrivant les contraintes prises en compte.
- une collection systématique des interfaces induites entre éléments de l'architecture.
- un affinage progressif de la définition de ces interfaces par rapport au niveau de décomposition immédiatement supérieur.

Le graphe fonctionnel décrit le système selon une arborescence cohérente. L'ensemble des éléments terminaux de la décomposition, appelés modules fonctionnels, et leurs interfaces représentent l'architecture fonctionnelle du système. Ces modules seront spécifiés puis réalisés matériellement ou logiciellement.

Les contraintes prises en compte pour l'établissement du graphe peuvent être :

- des contraintes de qualité, particulièrement l'évolutivité ; ces contraintes imposent des règles d'indépendance entre modules fonctionnels et des règles de regroupement de tâches dans les modules.
- des contraintes opérationnelles exprimées dans les documents de règles générales. leur analyse permet de dégager la structure du "coeur fonctionnel" du système, ensemble de modules particuliers dont le rôle sera d'assurer la gestion des autres modules de l'architecture.

EXEMPLE D'ARCHITECTURE FONCTIONNELLE



b) Projection de l'architecture fonctionnelle sur l'architecture matérielle

Cette phase consiste à intégrer l'architecture fonctionnelle préalablement définie et l'architecture matérielle proposée au cours de la définition préliminaire. les modules fonctionnels sont alors distribués dans les équipements et identifiés (matériel ou logiciel).

Cette projection définit le compromis de réalisation tenant compte de plusieurs facteurs :

- des facteurs technologiques : selon la nature des traitements à effectuer, choix de l'équipement le plus adapté.
- l'optimisation de la connectique : on recherche une distribution des modules minimisant le flux d'échange entre équipements et privilégiant "le plus court chemin" pour les informations critiques d'un point de vue temps réel.
- des facteurs de qualité particuliers : par exemple, les modules supportant des fonctions critiques d'un point de vue sécurité seront regroupés dans un même équipement et/ou redondés dans plusieurs équipements.
- des facteurs déterministes de savoir-faire ou d'organisation industrielle.

En conséquence de l'affectation des modules aux équipements, les données fonctionnelles sont éclatées en trois catégories :

- les données inter-équipements numériques
- les données inter-équipements analogiques
- les données échangées entre modules d'un même équipement.

Outils formels :

- Outils généraux

- La tâche de définition du graphe fonctionnel est supportée par un outil de conception système : OCS. Cet outil permet la construction graphique assistée, la collection cohérente des interfaces à tous les niveaux de décomposition et l'identification des chaînes fonctionnelles (dépendance des informations).

- La tâche de projection de l'architecture fonctionnelle est supportée par un outil d'aide à l'architecture : OEA, qui permet, à partir des données fonctionnelles recensées sous l'outil OCS de déterminer automatiquement la charge des bus numérique reliant les équipements, relativement à une projection donnée, et donc d'optimiser la répartition des modules fonctionnels dans les équipements.

- La gestion des données inter-équipements numériques est supportée par l'outil GIN qui permet, à partir des données fonctionnelles et de caractéristiques des bus numériques de spécifier les informations, messages et trames.

- La gestion des données inter-équipements analogiques est supportée par l'outil SINOPTIX qui permet, à partir des données fonctionnelles et des caractéristiques physiques et électriques des liaisons analogiques, d'élaborer les schémas synoptiques de câblage.

Outils de simulation

- Sans objet.

5.4 - Définition détaillée

L'étape de définition détaillée consiste à établir le cahier des charges contractuel de chaque équipement pour les aspects tant matériel que logiciel.

Les constituants de ce cahier des charges sont :

- le document de spécification technique d'intégration (STI)
- les documents de spécification détaillée des fonctions (SDF)
- les fiches d'interface numériques relatives à l'équipement
- les fiches d'interfaces analogiques relatives à l'équipement.

a) Spécifications techniques d'intégration

Ces documents décrivent les caractéristiques physiques, mécaniques et électriques des équipements, ainsi que les fonctions autonomes, assurée par le matériel ainsi que par le logiciel équipement (à l'exclusion du logiciel système).

b) Spécification détaillée des fonctions

La spécification détaillée consiste en la spécification des fonctions de transfert de chaque module fonctionnel identifié dans l'architecture.

Idéalement, ces documents doivent contenir "tout ce qui est nécessaire et seulement ce qui est nécessaire à la réalisation des logiciels et des matériels" et représentent donc une définition vis à vis des réalisateurs. La modularité elle-même de cette documentation (un document autonome par module fonctionnel) induit une contrainte de réalisation aigüe, l'architecture logicielle de chaque équipement devant respecter le découpage imposé par l'analyse fonctionnelle du système.

Les SDF représentent la charnière entre les activités systèmes, de responsabilité AMD-BA et les activités de réalisation des logiciels, confiés dans la majeure partie des cas à des industriels coopérants. Elles représentent une documentation contractuelle très volumineuse (plus de 10000 pages pour un système de taille moyenne) dont la qualité est primordiale.

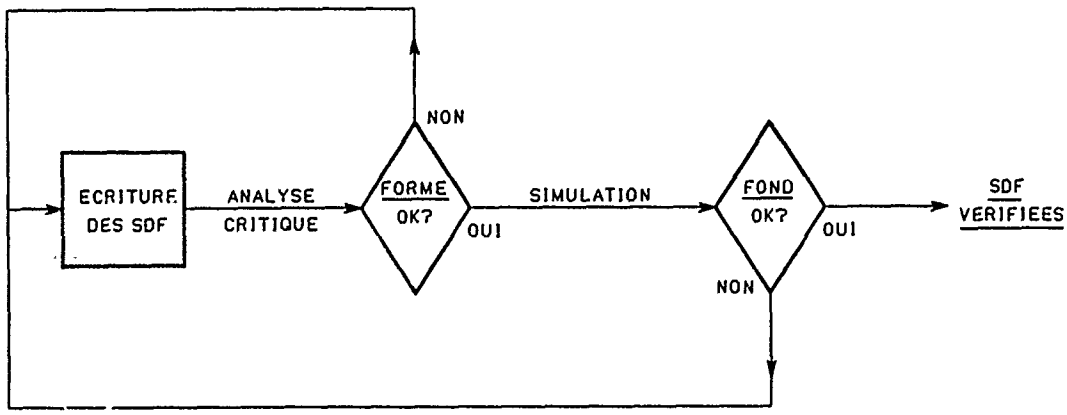
Ces SDF font donc l'objet d'une vérification très poussée, utilisant des techniques de maquettage (rapid-prototyping).

Maquettage des SDF :

- Le maquettage des spécifications détaillées comporte trois étapes :

  . L'analyse critique des spécifications
  . la réalisation de la simulation
  . l'exploitation de la simulation

  Ces étapes s'enchaînent selon le principe suivant :



- L'analyse critique a pour but d'améliorer la qualité des spécifications pour l'aspect formel, c'est-à-dire s'assurer que ces spécifications sont :

  . lisibles
  . complètes
  . cohérentes
  . sans ambiguité
  . réalisables d'un point de vue informatique

  Un spécificateur étant naturellement satisfait de son document grâce à sa connaissance du contexte opérationnel, pour que la technique soit rentable, il faut isoler les lecteurs critiques de ce même contexte en limitant les explications fournies sur les spécifications. Le mot d'ordre est : ne pas juger de ce que doit faire la spécification (fonctionnel) mais juger uniquement la manière dont elle est écrite et sa faisabilité.

  L'équipe de relecture est donc distincte de l'équipe de spécification et n'a pas connaissance du besoin opérationnel exprimé à travers les spécifications détaillées (spécifications globales non fournies).

Les erreurs détectées pendant l'étape d'analyse critique se répartissent en trois types principaux :

. erreurs de rigueur
. erreurs de généralité
. inadaptation de la spécification à une réalisation informatique.

- La réalisation de la simulation comporte deux phases :

. l'élaboration de l'architecture de simulation
. le codage des traitements décrits dans chaque SDF

Pour assurer une représentatibilité maximale, l'architecture de simulation est conforme à l'architecture fonctionnelle du système. En conséquence, toutes les interfaces recensées lors de l'analyse fonctionnelle sont transformées selon une chaîne automatique en variables logicielles (déclaratives FORTRAN en l'occurrence).

Cette base de donnée tient lieu de référentiel de variables pour la simulation des différentes SDF, modélisées à partir des fonctions de transfert spécifiées. On obtient ainsi l'équivalence suivante : un module fonctionnel = un document de SDF = un module de logiciel de simulation = un module de logiciel embarqué.

- Après mise au point, la maquette est mise à disposition des spécificateurs, pour exploitation. Cette exploitation est réalisée au travers de scénarii de tests effectués au niveau de chaque module puis au niveau de l'ensemble des modules, par les spécificateurs eux-mêmes et par d'autres intervenants dans le programme. Un conversationnel adapté permet la stimulation des interfaces fonctionnelles et le relevé de leur variation au cours de la phase de simulation, par des tests de type scénarii, représentatifs d'une utilisation opérationnelle ainsi que par des tests combinatoires (recherche de changement d'état d'une sortie particulière) ou aléatoires (tests de robustesse).

- Apports du maquettage

Le maquettage des spécifications détaillées permet :

. L'amélioration de la qualité des spécifications : le contrôle de forme permet de corriger et de compléter la spécification pour l'amener à un niveau autorisant son codage direct par des informaticiens ; il agit comme un puissant révélateur d'erreurs et un contrôle qualité efficace. L'exploitation de la maquette permet une réelle vérification de la spécification détaillée au niveau module, équipement, sous-système ou système de façon "horizontale" (tests exhaustifs d'un élément) ou "verticale" (test d'une chaîne fonctionnelle donnée).

. La constitution d'une bibliothèque de tests statiques et dynamique qui sera réutilisée dans les étapes de validation (branche remontante du "V").

. La mise à disposition, avant réalisation des équipements, d'une référence fonctionnelle vivante, utilisable à des fins pédagogiques, didactiques ou trivialement à des fins d'analyse et de contrôle des modes de fonctionnement du système ou d'analyse a priori de modifications.

Outils formels

- outils généraux
- outil de conception système OCS
- outils de spécification (DLAO, MITIA)

Outils de simulation

- outil de maquettage OCCAM

5.5 - Réalisation des équipements

Le développement et la réalisation des équipements (matériel + logiciel) sont des tâches avales des étapes de développement du système résumées plus haut, qui n'entrent pas dans le cadre de cette communication. Pour le logiciel en particulier, elles concernent les étapes de développement suivantes :

- conception globale du logiciel
- conception détaillée du logiciel
- codage
- tests unitaires du logiciel
- tests d'intégration du logiciel
- tests fonctionnels du logiciel

Remarque : les méthodologies de développement de logiciel à partir des spécifications système ainsi que les ateliers logiciels correspondants peuvent être sensiblement différents selon les réalisateurs.
Néanmoins, l'architecte industriel du système doit s'assurer de leur adéquation à la méthodologie de développement du système par des audits ou à travers les normes appliquées par chaque réalisateur.

5.6 - Validation fonctionnelle des équipements

Le but de cette étape est de procéder à une évaluation fonctionnelle séparée de chaque équipement développé afin d'aboutir à un niveau de confiance suffisant avant intégration au système.

L'étape se décompose en deux phases :

- une phase de validation chez l'équipementier
- une phase de validation chez l'avionneur

1ère phase (chez l'équipementier)

Le principe consiste à dérouler des scénarios fonctionnels au niveau de l'équipement de façon à :

- valider les fonctions implémentées dans l'équipement en regard des documents de spécification (STI + SDF)
- mesurer diverses marges de fonctionnement caractéristiques (temps de tracé pour un équipement de visualisation, charge de calcul et taux d'occupation mémoire pour un calculateur, etc...).

Les scénarii de test utilisés proviennent de plusieurs sources.

- des scénarii fonctionnels issus du maquettage des spécifications détaillées (cf paragraphe 5.4)
- des scénarii générés au moyen de banc d'intégration système lorsque celui-ci est opérationnel (cf paragraphe 5.7).
- des scénarii enregistrés en vol par l'intermédiaire des installations d'essais (cf paragraphe 5.8).

Tous ces scénarii, après formattage et concaténation, sont appliqués aux équipements sous test par l'intermédiaire d'un moyen standard : MAGE qui permet l'interfaçage électrique et numérique des différents équipements.

2ème phase (chez l'avionneur)

- Cas des équipements à forte valeur ajoutée système.

Pour les équipements supportant la plus grande partie du logiciel système ("coeur système") tels que calculateurs de missions ou calculateurs de gestion de l'interface homme/système, la représentativité et la couverture des tests réalisés par le moyen MAGE est insuffisante.

Ces équipements sont testés dans un environnement particulier représentatif de la totalité du système : le CIH.

D'un point de vue matériel, le CIH est constitué :
. d'un système informatique temps réel (construit autour de moyens informatiques non spécialisés et de forte puissance) supportant une modélisation du système avionique
. d'un poste de pilotage composé d'une maquette de cabine programmable
. d'un système de simulation de l'environnement opérationnel (sphère d'imagerie synthétique)
. de consoles d'exploitation permettant la mise en oeuvre de scénarii de tests.

D'un point de vue fonctionnel, la modélisation est obtenue par la mise en commun de plusieurs simulations :
. une simulation globale du système avionique obtenue à partir de la maquette des spécifications détaillées (cf paragraphe 5.4), enrichie des caractéristiques fonctionnelles et temporelles lui permettant un dialogue "en direct" avec des équipements réels.
. une simulation de l'environnement extérieur du système, composée de modèles représentatifs des capteurs de l'avion, du dispositif ou du théâtre d'opération.

Outre l'aide à la validation des équipements, le CIH permet la vérification globale des étapes de conception, les utilisateurs étant à même de comparer, d'un point de vue opérationnel, le modèle détaillé du système disponible sur le CIH au modèle disponible sur l'outil OASIS à l'étape de définition globale (cf paragraphe 5.2).

- Cas des capteurs

La validation des équipements capteurs nécessite l'utilisation de bancs partiels d'intégration, permettant de tester leurs fonctionnalités et performances en reponse à des stimuli spécifiques de la physique du capteur. On trouve particulièrement des bancs d'essais pour les radars, les dispositifs inertiels, ou les capteurs anémo-baro-clinométriques.

Ces bancs partiels sont couplables tant au CIH qu'au banc d'intégration système (cf paragraphe 5.7).

Outils formels
- Outils généraux

Outils de simulation
- MAGE : moyen autonome de génération d'environnement
- C.I.H. : Centre d'Intégration Hybridable

5.7 - Intégration et validation sur banc d'intégration système

Le but de cette étape est de s'assurer, avant intégration sur avion, de la conformité du système à sa définition globale, de la cohérence des configurations successives, et d'évaluer le comportement du système dans tout son domaine d'utilisation (domaine avion, cas de pannes, tolérance aux pannes, etc...).

Les bancs d'intégration permettent la mise en oeuvre aisée des équipements constituant le système. Ils possèdent :

- l'ensemble des équipements réels du système intégré
- des terrasses déportées pour l'installation des capteurs
- un câblage du même type que celui de l'avion
- une distribution électrique identique à celle de l'avion
- des moyens de surveillance analogiques et numériques
- des simulateurs d'interfaces numériques et analogiques
- des panneaux d'accès aux informations système
- des interfaces avec un complexe informatique temps réel
- une implantation réaliste des équipements de dialogue homme/machine

L'environnement du système avionique embarqué est restitué sur le banc grâce à un complexe informatique permettant les fonctions de stimulation et de simulation. Ces deux fonctions ont pour but de générer un environnement réaliste évoluant avec la même dynamique de paramètres que sur avion.

- La stimulation consiste à rejouer sur le banc un scénario enregistré sur avion de façon à mettre le système dans un état identique à celui rencontré en vol.
- La simulation consiste à générer un scénario de façon interactive, permettant ainsi de "piloter" le système et d'étudier ses réponses dans tout son domaine d'utilisation.

Les bancs d'intégration permettent aujourd'hui de procéder à la plupart des tâches de validation et de qualification des systèmes, réservant les tests en vol aux quelques essais critiques pour lesquels le banc n'est pas suffisamment représentatif.

Leurs avantages sont leurs coût et souplesse d'utilisation (comparés à ceux d'un avion), leur disponibilité, leur facilité d'évolution, leur actuelle représentativité et enfin leur puissance d'analyse sans cesse croissante.

5.7 - Intégration et validation sur avion

Le but de cette étape est de vérifier et de garantir le fonctionnement opérationnel du système embarqué. Les tâches se composent d'essais au sol et d'essais en vol.

Au sol sont conduits des essais d'intégration nécessitant l'avion complet, des essais de maintenance, ou des essais de comportement électromagnétique réalisés dans une chambre anéchoïde.

En vol sont conduits des essais d'ouverture de domaines, des essais de séparation d'emports, les essais fonctionnels complémentaires des essais réalisés sur banc d'intégration et enfin des évaluations opérationnelles particulières à la demande de l'avionneur lui-même ou de ses clients.

Les moyens utilisés sont nombreux et variés, soit non exhaustivement : les avions prototypes, les installations d'essais, les dispositifs de télémesure, les salles d'écoute, les complexes informatiques temps réel, les moyens de préparation et restitution de mission et les différents matériels de mise en oeuvre en piste.

6 - CONCLUSION

Les systèmes avioniques complexes sont aujourd'hui soumis à des exigences contraignantes en matière de qualité et de performances.

La prise en compte et la réalisation des objectifs assignés aux systèmes passe par une maîtrise du processus de développement.

Cette maîtrise est obtenue par le respect d'une méthodologie rigoureuse, recouvrant de façon continue la totalité du cycle de développement et qui est à la base de toutes les actions Qualité.

Cette méthodologie est supportée par une panoplie d'outils constituant l'atelier système et les ateliers logiciels.

La méthodologie résumée dans cette communication a été mise en place progressivement dans les bureaux d'études des AVIONS MARCEL DASSAULT BREGUET AVIATION depuis 1982.

La mise en place opérationnelle de l'atelier système remonte à 1987 pour le développement du système MIRAGE 2000-5, l'atelier comportant à cette date la quasi totalité des outils formels ainsi que les outils de simulation correspondant aux étapes de conception.

Aujourd'hui, dans le cadre du développement de l'avion Rafale D, l'atelier complet décrit dans cette communication, est utilisé dans un contexte multi-industriel, le système étant conçu et réalisé en coopération avec plusieurs industries aéronautiques.

AIRCRAFT CONTROL SYSTEM DESIGN. SYNTHESIS, ANALYSIS AND SIMULATION TOOLS AT AERMACCHI

L.Mangiacasale, L.V.Cioffi, C.A.Bonatti
Air Vehicle Technology
AERMACCHI S.p.A.
Via Sanvito, 80
21100 - Varese
ITALY

SUMMARY

Three phases of an airplane control system design are presented and discussed. From the preliminary synthetic design to the non-linear simulation the various steps proceed through the computational methods currently exploited at Aermacchi. Optimal and Sub-optimal methods are used in the first phase in order to get information about control strategies; accurate linear analysis is then performed with complex linear models for the continuos and sampled-data design. The design is completed with three and six degree of freedom non-linear simulations in which the complete airplane is simulated with an even more complex modelization.

1. INTRODUCTION

The design of modern aircraft has become a tremendous task, featuring very important interdisciplinary activities.
In this field there is a strong interaction between aerodynamics, flight mechanics and control systems, thus a closely integrated approach is needed from the beginning of the conceptual design phase until the prototype flight.
On the other hand, the increased complexity of the design phase requires the use of automated and routine procedures, which, in turn, mean the development of full Computer Aided Design Tools.
Automated computation, however, should be developed and applied without forgetting the real aircraft and its requirements (for instance Flying Qualities).
This means that the CAD tool should be developed in such a way that F.Q. can be considered during each phase of design.

2. THE PHASE OF DESIGN

The design of control systems for a modern aircraft requires that the following three phases be considered:

1) Synthesis Phase
2) Analysis Phase
3) Simulation Phase

These three phases are characterized by the increasing complexity of the models.

3. THE SYNTHESIS PHASE

The synthesis phase mainly concerns the control strategies, therefore, in the preliminary stage of design, the control philosophy and architecture should be searched for among a large number of possible alternatives in regard to measurements for feedback, employed control surfaces, cross-feeds between the control lines.
The aircraft target performance must however be borne in mind constantly.
Modern design methods allow engineers to develop a large number of tradeoffs in the preliminary phase of design, when the aircraft configuration is still under discussion.
Well known methods as:

- Linear-Quadratic Regulator (Deterministic or Stochastic)
- Eigenstructures Assignment
- Numerical Minimization

and others, are used in the preliminary design of control systems.
The synthesis phase for the generation of the control law, Fig. 1, is based on two fundamental pillars:
- simple design models

- fast interactive computer programs.

Simple design models means that the aircraft is described by using the classical linearized equations for the separed longitudinal and laterodirectional modes. This linear model is augmented with the transfer functions describing the dynamics of the control actuation system (i.e. electrohydraulic actuators).

In the Aermacchi designs, second order transfer functions are taken into account to model the actuators. Even if a first order model is a quite good approximation for synthetic purposes, a second order one features enhanced model fidelity in the medium--high frequency range, and often increases the design flexibility. This obviously entails an increase in the model dimensions (number of state variables). No sensor transfer function is considered. The basic model is then expanded by adding some forms of Model Following, Command Generators, and integrators (as currently used in aircraft control systems). Turbulence models are also added, which are used to assess the performance and requirement for flight in turbulence.

Moreover, using some form of physical reformulation of the command input vector (control surface rate input in lieu of control surface deflection input), it is possible to add, at the input of each actuator, low pass filters, whose characteristic (cross-over frequency) is designed by the procedure itself. The (jet) engine transfer function can be further added to the model if autothrottle strategies have to be evaluated.

Control laws (feedback and forward gains) are synthesized by using the Linear-Quadratic -Regulator together with the Model Following techniques. The following step are made in the design development using the computer package to Ref. (1):

- regulator design (Matrix Riccati Equation solution)
- time response to step, pulse, doublet input
- frequency response to the pilot input (closed control loops)
- Root-mean-square evaluations for flight in turbulence
- Root-mean-square evaluation for (white) sensor noise
- multivariable robustness analysis by using a modern technique called "Singular Value Analysis"

In the optimization process (based on the output and input weighting matrices) the following Flying Qualities parameters are evaluated:

Pilot command performance
- Time Response Parameter (TRP, ref.(2))
- Equivalent Control Anticipation Parameter (CAP, ref.(3))
- Dropback (DB, ref.4)

Flight-in-turbulence performance
- Discomfort index (DI ref.(5))
- Half-g bumps per minute (Ride Bumpiness) ($N\frac{1}{2}g$ ref.(6))
- Crew-Mission-Performance Limitation (CMPL ref.(7))

During the optimization process many trade-offs can be developed in order to understand the effects of the different feed-backs on the control system behavior/performance. Gains which do not affect aircraft flying qualities can be nulled with the aim of generating a not too complex feed-back architecture as early as the beginning of design.

The optimal design is currently developed both in the continuous domain and in the sampled data domain; analog design is used as "reference" for the subsequent digital design.

Problems encountered in the synthesis phase

One of the most known drawbacks of the Linear Quadratic Regulator is the need to feed-back each of the state variables included in the formulation of the design model.

Unless the corresponding gains can be zeroed for the aforementioned reasons, when a state variable cannot be measured, the problem can be solved by reconstructing the state (using observers).

Another problem arises when it is desired to feedback a washed-out variable (see for instance the washout yaw rate in the lateral-directional control systems), since, in general, the LQ method gives a feedback to both the "normal" and washed-outed variable.

This situation is handled with some difficulty and might require a resort to "design tricks" when using LQR full-state feedbacks.

An alternative currently exploited is to develop control systems by Numerical Optimization.

In this method, once the design model has been defined (like in the LQR case) the

designer is free to establish the feedback architecture in terms of measurements and dynamic compensation (or filter), then to calculate the feedback gains around the loops he/she is willing to use in the control scheme.
The computer program used at Aermacchi (ref. (8)) works on state variable models and can design full state feedback regulators (with a numerical optimization instead of the Matrix Riccati Equation) or limited output regulators with and without compensation networks; it can implement Explicit or Implicit Model Following and Sensitivity Reduction controllers. The general formalism looks like the LQR procedure, with a performance index (J) still built using weighting matrices.

Drawbacks of Suboptimal Design

It is well known that the solution of the sub-optimal design depends on:

- the weighting matrices
- the feedbacks used to start the computation
- the number of iterations
- the convergence criteria used to stop the computation

As a consequence, the designer's workload for the control of the procedure and the achievement of a good solution is very high, and increases along with the dimensions of the design model. For very large systems with many parameters to optimize, Sub-optimal Design may become very burdensome.
Finally, the main features of these two methodologies can be summarized as follows:

LQR: no convergency problem; model following requirement easily satisfied; capability to cope with strongly interconnected MIMO systems; low human workload

SUBOPTIMAL: capability to treat medium size systems without defining a feedback for each variable (filters, sensors, high dynamics and not measurable states may be included in the plant within the design phase); design of dynamic controllers; high human workload.

Suggestions for Synthetic Design

Following the experience gained in this field, the following steps are recommended:

- to build a linear state variable model with dimensions (or complication) matching the designer's experience and the physical understanding of the problem
- to generate an Optimal Design by using LQR or other design tools available to the designer
- to develop trade-offs in order to minimize the number of required feedbacks (maintain good control system performance and aircraft Flying Qualities)
- to establish and redefine model for Suboptimal Design
- to use the Numerical Minimization Design by starting, when possible, from the optimal gains, and to optimize only a reduced number of feedbacks.

4. THE ANALYSIS PHASE

The Analysis phase is different from the Synthesis phase because of two main reasons:

- computer models become complicated
- the computer programs which handle the models should be very powerful and flexible.

The analysis models complexity can be significantly increased with respect to that of the Synthesis models by:

- adding sensors transfer functions (1st, 2nd or higher order, based on the available literature or data from suppliers)
- adding shaping filters to generate a more realistic coloured noise
- adding prefilters and postfilters (in sampled data systems)
- redefining the actuators models adding higher dynamics
- completing the scheme with state reconstructors or observers (if needed).

There exists a number of computer programs which are able to handle complex linear systems, continuous or sampled data; the computer program to Ref. (9) is presently used at Aermacchi for this purpose. The program accepts system description in "mixed form" so that the designer can use a matrix description for the aicraft, and transfer functions for the control system blocks (in the s-plane or in the z-plane); then the

connections defined in an "ad hoc" routine, allow the computer program to build the full model for the computation of:

- frequency responses
- time histories
- power spectral density (for continuous systems)
- root locus and root contours
- transfer functions (Bode plots)

How to develop the Analysis phase

The Analysis phase is developed in the following way

- the optimal design scheme from the LQR procedure is implemented and analyzed with the above software in order to assess the design quality, and to discover errors in the general modeling (cross-check)
- the full-state control scheme is completed with all the previously detailed elements, and the computations are accomplished
- the results of the full system are compared with those of the synthetic design, opportune action is taken to restore gain and phase margins of the synthetic design, if possible by using compensation networks and/or by modifying the feedback gains. At the end of the analysis phase the designer should be able to provide a control system architecture which, coupled to the aircraft, guarantees the required Flying Qualities.
  It is worth pointing out that the design of compensation networks, which is developed by using the analysis model, is a complicated step for complex control systems with cross-feeds between control lines; in fact, since an analytical tool to design multi-loop compensators is not available at present, a trial-and-error method is used, based on (SISO) Bode plot.
  A flow diagram of the analysis scheme is shown in fig. 2, while fig. 3 shows a block diagram of the flaperon control system.

5. THE SIMULATION PHASE

As well known to people involved in the synthesis and analysis of complex control systems for aircraft, the models exploited in the previous phases of the design are linear. But in an aircraft non linearities are present for several reasons:

- aerodynamics is strongly non-linear, and this non-linearity is a function of angle of attack, control surface deflection, Mach number and so on
- equations of motion are non linear
- the control actuation system can feature significant non-linearities in terms of control surface rate saturation and deflection limitations
- dead-zone and hysteresis can be present in the sensors and also in the electro-hydraulic valves.

It comes therefrom that a large number of non-linear simulations are needed before clearing a control system, in order to:

- verify the quality of the control law within the flight envelope (Mach number, altitude, angle of attack)
- evaluate the impact of the non linear elements on the Flying Qualities of the aicraft, mainly at low angles of attack and during precision maneuvers
- complete the control law parts not designed during Optimal Synthesis (such as the control coupling between longitudinal and lateral-directional dynamics at high angle of attack)
- assess the control system sensitivity to aerodynamic coefficients variation and to other uncertain parameters
- clear the control system and generate performance which will be cross-checked in the Flight Simulator or during flight tests.

Although many powerful computer programs exist for simulation purposes, a Simulation Language (Ref. (10)) is used on UNIVAC/1100 Machine at Aermacchi.
At present, a complete non-linear model has been built. It contains:

- non-linear longitudinal aerodynamics for angles of attack $-10° < \alpha < +35°$, at low Mach number
- longitudinal control law with tables for gain scheduling, filters and ancillary computations

- sensor dynamics and actuation models with rate saturation and stops
- turbulence model of the Dryden type
- thrust model with generalized tables and simple transient dynamics simulation.

A simple scheme is shown in fig. 4, and a linear and non-linear simulation (of an approach phase) are given in fig. 5.
A complete six-degree of freedom simulation model is being developed.

CONCLUSION

A wide experience in control system synthesis, analysis and simulation for aicraft has been gained at Aermacchi starting from the early eighties.
Design models for synthetic design have been developed, tested and exploited for optimal (LQR) and Sub-optimal (Numerical Optimization) design.
The synthetic schemes have been expanded for analysis purposes, and compensation networks have been successfully defined by using conventional techniques (Bode plots/ /root-locus).
It is suggested that, during the synthetic design, the engineer tries to understand to which aspect of the optimization he/she should give special care in order to minimize the work required for full control law tuning during the analysis phase.
The quality of the designed control law has been tested with extensive 3 DOF simulations with non-linear aerodynamics.
The many steps of the design have demonstrated that the procedure is viable for a real development.

REFERENCES

(1) LQG ALPHA. Control System Design Software. COmputer Program for Use in Linear System Studies ALPHATEC Inc.
(2) D.J.Moorhouse, R.J.Woodkock, Background Information and User Guide for MIL-F-8785C, AFWAL TR81-3109, July 1982.
(3) D.E.Bischoff, The Definition of Short Period Flying Qualities Characteristics via Equivalent Systems, J. of Aircraft, Vol. 2, n. 6, June 1983.
(4) J.C.Gibson, Piloted handling Qualities Design Criteria for high Order Flight Control Systems, AGARD CP333 (Criteria for H.Q. of Military Aircraft).
(5) Background Information and User's Guide for MIL-F-9490, Table I.
(6) J.Becker, Gust Load Prediction and Alleviation on a Fighter Aircraft, AGARD Report 728, September 1985.
(7) M.Hacklinger, Design Problems of Military Aircraft as Affected by Turbulence, AGARD CP-140 (Flight in Turbulence), 1973.
(8) P.J.Fleming, The Suboptimal Linear Regulator CAD Program, University College of North-Wales, Bangor, Gwynedd.
(9) J.W.Edwards, A FORTRAN Program for the Analysis of Linear Continuous and Sampled-Data System, NASA TM X-56038
(10) MARSYAS: Marshall System for Aerospace Simulation, Marshall Space Flight Center, Alabama.

CONTROL LAW SYNTHESIS

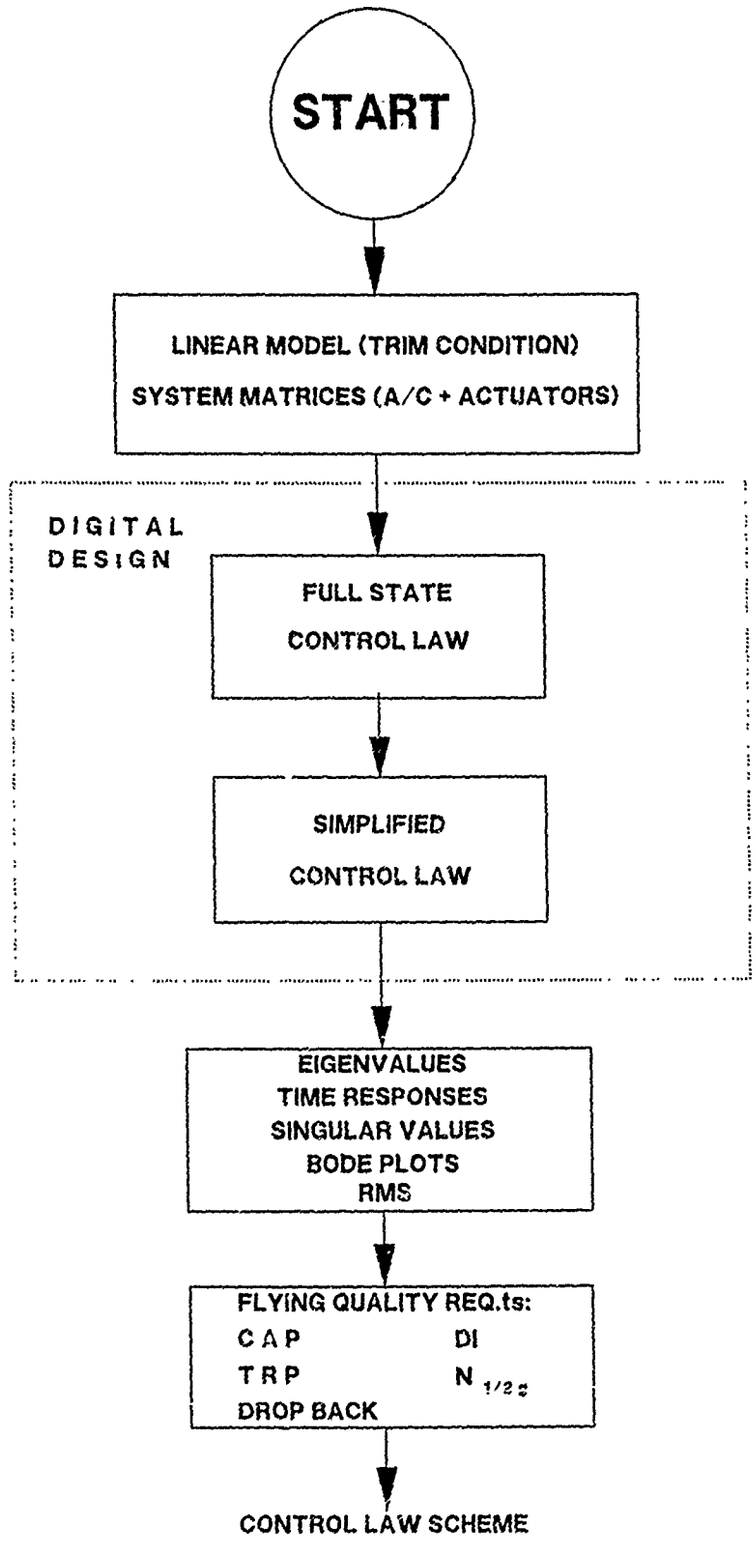


FIG. 1 - THE SYNTHESIS PHASE

CONTROL SYSTEM ANALYSIS

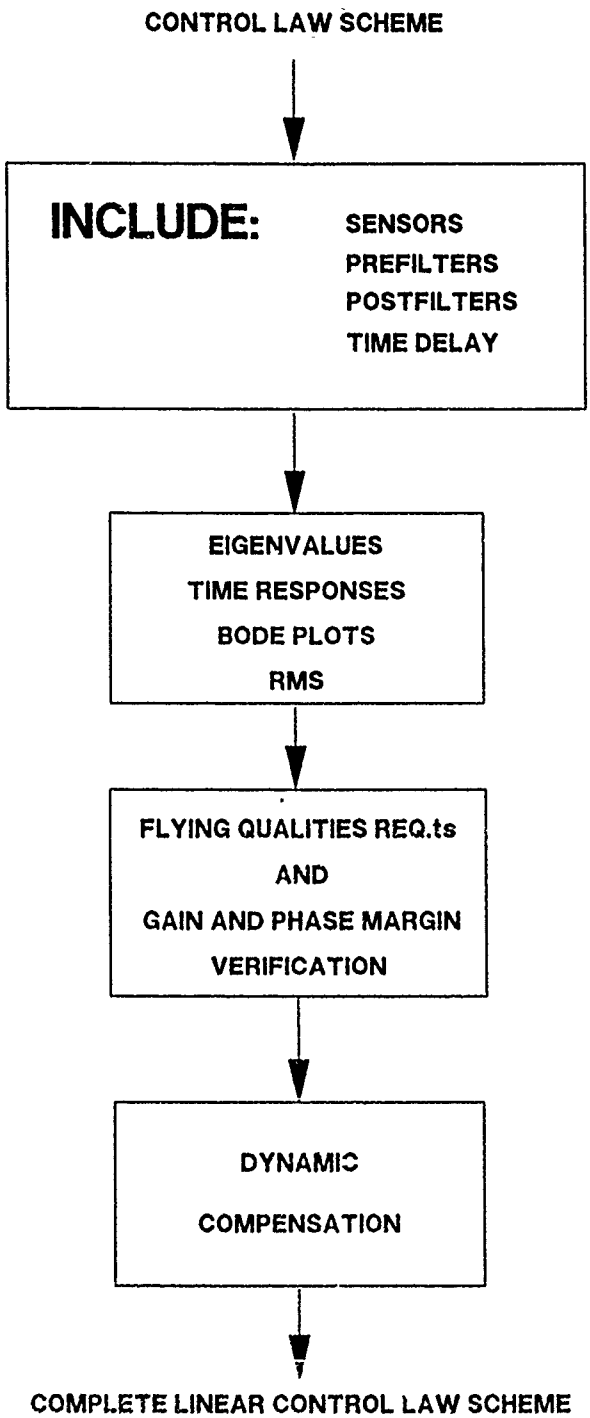


FIG. 2 - THE ANALYSIS PHASE

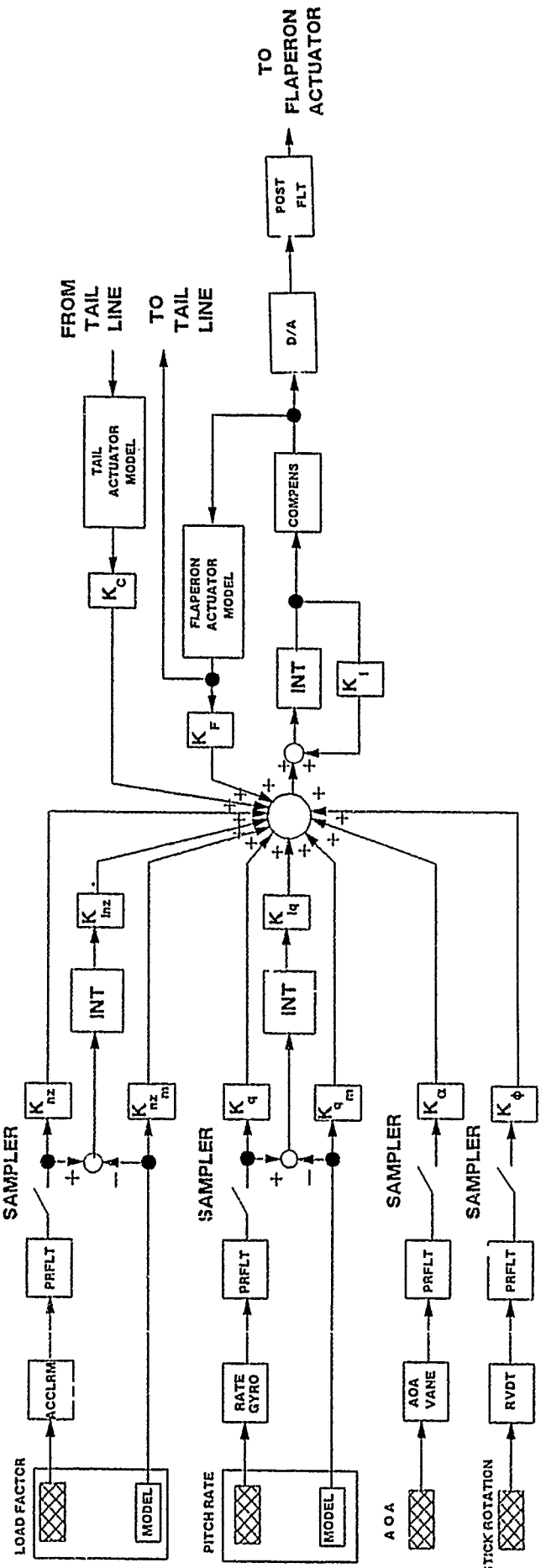


FIG. 3 - BLOCK DIAGRAM OF THE FLAPERON CONTROL SYSTEM

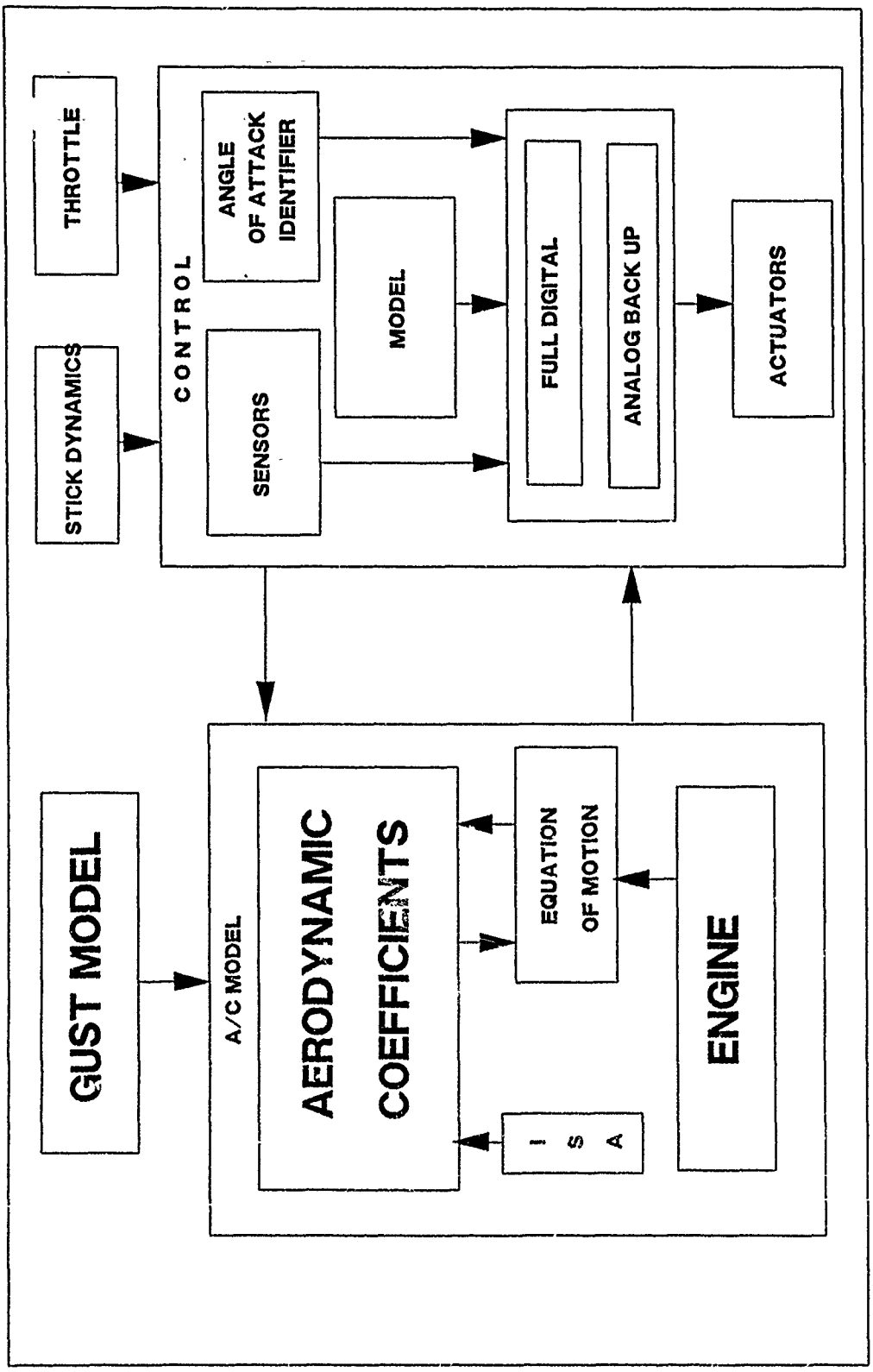


FIG. 4 - THE COMPLETE LONGITUDINAL NON-LINEAR MODEL

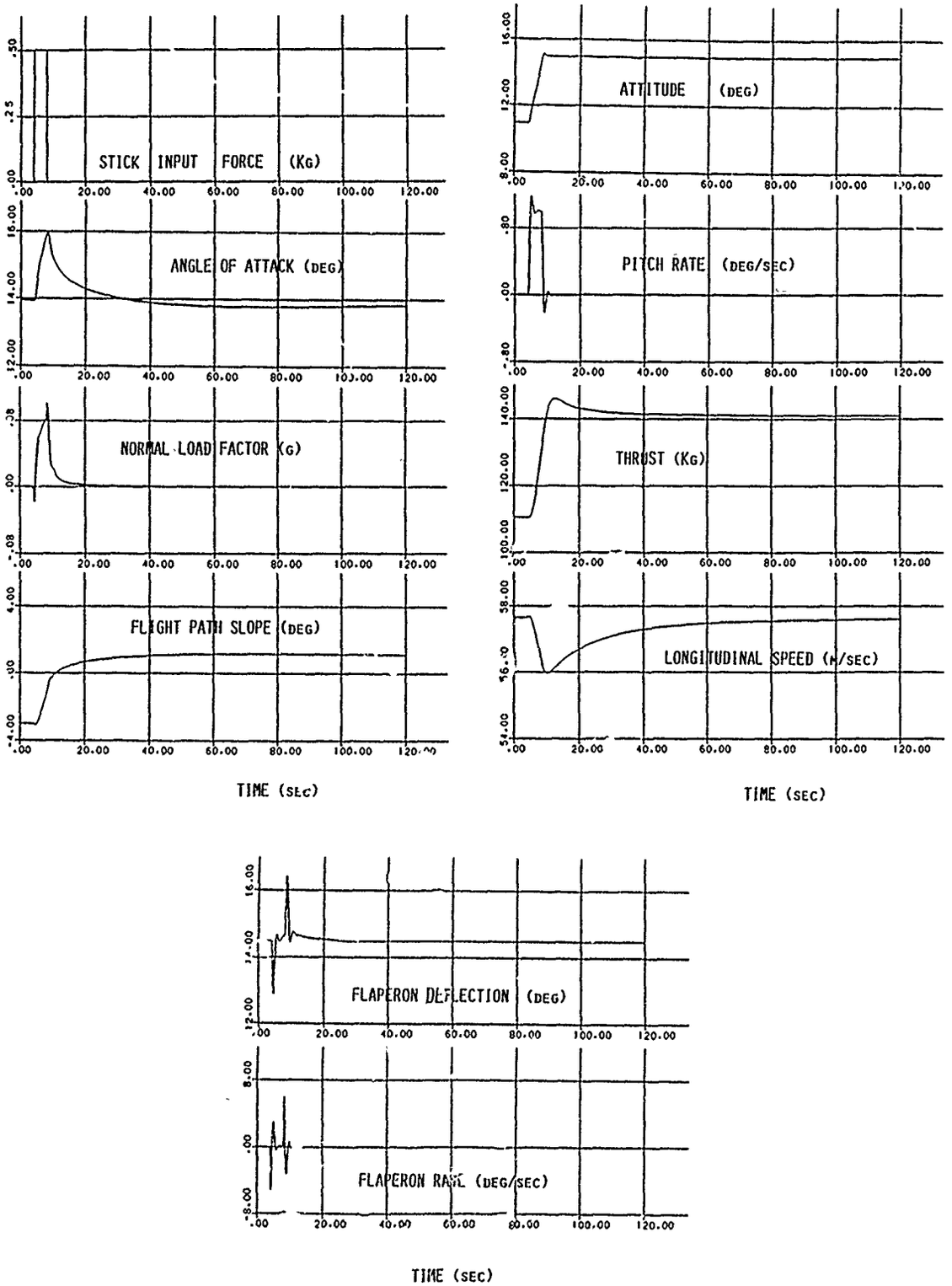


FIG. 5 - NON LINEAR SIMULATION: RESPONSE TO A STICK COMMAND WITH AUTOTHROTTLE ON

COCKPIT MOCK UP CMU
A DESIGN AND DEVELOPMENT TOOL

by

Dipl.-Math. Christoph Weber
Elektronik-System-GmbH
Vogelweideplatz 9
8000 München 80
Germany

Summary

Designing a modern helicopter cockpit, ergonomics, operational and technical aspects have to be considered. For ensuring a low cost development schedule the CMU is a flexible, inexpensive design and development tool for optimization of the Man Machine Interface (MMI). The ESG CMU, realized in close cooperation with the user, is a full-size model cockpit of future helicopters such as NH 90 and PAH-2. The future user is integrated in the experimental closed-loop simulation with our CMU.

Problem situation

The design and development of new helicopters and thus of new and modern cockpits always requires the integration of latest technologies and of the human being in the latter. In the past the primary emphasis in this connection was placed on the technology in general, although, however, the crew is directly involved in the operation, i.e. by the tasks of helicopter command and control and mission performance. A further aspect is that both the physical characteristics, including the mind and the human intellect have hardly changed in contrary to the technology, so that it appears that man becomes the weakest link in the man-machine system. The primary objective today must be to relieve the man in the cockpit. Thus the crew-system interface is an essential key for not exceeding the limits of psychical and physical crew stress also under extreme conditions, such as

- Lowest flight
- Night missions
- Operations under extremely bad visibility conditions.

The problem becomes even more obvious by the fact that an enormous increase in data volumes as the result of new, additional sensors, and inclusion in command and control systems leads to a steady data quantity and workload increase, thus exceeding the limits of errorfree data processing by the crew.

The design and layout of an advanced cockpit, e.g. for the NH90 and PAH-2 is influenced by ergonomic, operational and technical aspects, whereby to optimum layout of the MMI's plays an important part.

In order to consider the latter, i.e.

- Optimization of the MMI -

in connection with the future NH90 and PAH-2 helicopters, ESG performs the following experimental programmes in parallel with the development efforts:

CMU (S) Cockpit Mock Up (Side by Side)

CMU (T) Cockpit Mock Up (Tandem)

AVT Equipment test facility

Both the CMU (S) and CMU (T) are a flexible, costfavourable cockpit design tool for a side-by-side cockpit (NH90) and a tandem cockpit (PAH-2).

The AVT is an equipment test facility enabling the analysis of advanced equipment components in addition to the MMI aspects of a modern cockpit in the course of flight trials. The AVT, however, is not the subject of this paper.

Tasks and Objectives

As the task and objective in connection with the CMU (S) and CMU (T) are identical, the only difference being the version of the respective cockpit design tools due to the different functional requirements of the NH90 and PAH-2, only the CMU (T), briefly addressed as CMU is described in the following.

The essential task of the CMU design and development tool is the conversion of theoretical conceptual designs to experimental hardware for analysing the MMI for the future PAH-2 under quasi-real operational conditions.

Thus the CMU covers the following spectrum:

- Specification activities for the PAH-2 cockpit on the basis of knowledge gained by experiments

- Checking of existing specifications and developments of the PAH-2 cockpit

- Verification of theoretical proposals on the following:
  - o cockpit layout
  - o control and display
  - o crew communication in a tandem cockpit
  - o crew workload relief in a tandem cockpit

The purpose is as follows:

- gaining experimental information on the man-machine performance capability

- optimizing the development result, the development sequence with regard to time and cost aspects

- limiting the development risk

In addition to ergonomic questions and requirements regarding the PAH-2 cockpit, crew-supporting functions are tested experimentally, taking new control and display technology into consideration.

Thus the CMU is a tool for providing specifications via experiments in parallel to the development effort.

In order to enable a CMU to fulfill this task and objective, the test setup must be flexible with regard to hardware and software, in order to provide for a rapid investigation of alternative configurations and/or functions.

Procedure

In order not to face the crew with additional demands as a result of the new technologies, but to enable the latter to place its demands on the future technologies involved, ESG practises a cockpit development approach wit also considers the experience of the future users, the test pilots and flying instructors.

The basis of the experimental tasks for verifying and modifying the MMI are the preliminary specifications and/or open questions from the actual PAH-2 cockpit development. These basic elements are thus the 1st iteration of the process of iterative optimization/specification of the MMI with the help of the CMU. The results obtained experimentally in cooperation with the future user flow back to the actual development.

Thus the CMU is an effective aid for checking both in theory and practice important fields of the MMI in the future jointly by the users and system engineers long before the existence of a prototype of the new helicopter (Figure 1).

The CMU, of course, cannot and is not intended for replacing flight trials completely. It remains a fact that the degree of confidence in the results obtained by flight trials is higher than that obtained by results derived from a crew station such as the CMU, although flight trials are very time-consuming and expensive. With regard to the latter aspects, the obvious advantage is offered by a support tool like the CMU, as it enables the performance of the tests

- without flying authority
- irrespective of the weather conditions
- with a large number of trial subjects
- with non-flight-qualified commercial equipment

For these reasons the CMU, as a link between the developing source and the user, is an important effective and cost-favourable design and development tool enabling the limitation of the risks associated with a complex cockpit development.

Description of the CMU for the PAH-2

The Cockpit-Mock-Up (CMU) is a stationary, duplicated full-scale PAH-2 cockpit, i.e. a tandem arrangement of the crew work stations including the appropriate control and display facilities, such as:

Multi Function Displays
Helmet Mounted Sight/Display
Central control units

Furthermore, the CMU includes:

- a simulation facility for the digital map not yet available
- a sensor vision equipment
- a test engineer station
- efficient simulation computers, interface computers, symbol generators

Both cockpits in the CMU are largely independent from each other and equipped whit a complete instrumentation each. The control and display instruments are positioned in the instrument panel and on both side consoles. Different instruments only exist as a result of the different tasks of the pilot and the commander, and the slightly different space conditions in the rear cockpit.

Figure 2/3 shows the front/rear cockpit layout. The MMI in the CMU, without the simulation and test equipment consists of the primary control and display facilities, such as:

Flight Controls
Multi Function Displays (MFD)
Helmut Mounted Sight/Displays (HMS/D)
Radio Frequency Indicator (RFI)

and the secondary control and display facilities, such as:

Control Display Units
Line Selector Keys/Function Keys on the MFD
Incremental Potentiometers
Roller ball/Joystick

and the optical/accoustic warning facilities, such as:

Master Caution Light
Master Caution Panel
Engine Fire Light/Fire extinguishing system

including the standby instruments and the weapon control elements.

In order to provide a largely realistic cockpit MMI the additional functioning control elements have been realized:

Power supply panel
Hydraulic panel
Fuel panel
Engine start panel
HMS/D-control panel
Lightning panel
Intercom system etc.

The essential control and display elements are briefly described in the following:

Both the primary control facilities cyclic stick, collective stick and pedals, and the seats can be individually adjusted to the respective test crew. Furthermore the control facilities between the front and rear cockpit are coupled electrically and not mechanically as usually, so that the control can be influenced externally by test engineer.

Thus it is possible, e.g. to investigate questions being part of the sensitive subject of the crew communication in a tandem cockpit, such as "when does the commander in the rear cockpit realize that the pilot still assumes the flight control?"

The multi-function displays include flight control, navigation, tactical situation, weapon and system information, whereby it is possible to display this information as a function of the task/mission

- continuously
- as required
- or automatically

The MFD's are arranged in the middle of the instrument panel and are thus within the central reach and visibility of the crew.

Due to the special mission conditions of the PAH-2 the simulation of target acquisition, target engagement plays an essential part. In this phase the sensor information is displayed to the crew overlayed with synthetic symbology for flight control and weapon employment on the HMD.

The central Control and Display Unit (CDU) is located in the left front console area (front and rear cockpit).

It is the main input unit and is primarily used for controlling the following:

Flight Management
Communication
Navigation
Digital input of system data

The Line Selector Keys and Funtion Keys are used for adding/removing information on the MFD's as a function of the respective situation.

The incremental potentiometers are used for the analog input of the system data, such as speed command, heading command etc.

The roller ball and joystick are used alternatively. They serve as control element for cursor control on the MFD's for flight path and tactical situation handling and for the control of the sensor and thus of the commander's line-of-sight.

In addition to the actual cockpit, the CMU includes the following:

- the map display facility:
This facility (Figure 4) enables the simulation of the future digital PAH-2 map. For area navigation and representation of the tactical situation the map is shown on the displays.

This takes place as a function of the helicopter location and heading. The map can be shown in the following scales.

1 : 50 000, 1 : 100 000, 1 : 250 000

Flight path and/or tactical information can be synthetically superpositioned on the map representation on the MFD.

-the sensor display unit
The sensor display unit (Figure 5) consists of a landscape generator and a target generator.
The landscape generator generates reality-close, synthetic computer images on the basis of digitized landscape photographs, in such a way as if the landscape would be shown to a pilot/commander through a daylight camera.

In accordance with the cockpit geometry of the PAH-2 two eyepoints are realized at an altitude difference of approx. 3 m. The viewing directions of the pilot and commander are independent from each other and are each controlled by the respective line-of-sight of the HMS.

The target generator is used for the visual representation of targets, enabling the realistic indication of the battle area to the crew, including ground and air targets. The targets are controlled by a test engineer, so that a high flexibility is reached, in order to generate a practice-oriented field-experienced workload during the activities for target acquisition and fighting. With the help of a suitable video switching unit the crew can switch the "sensor image" of the respective crew member to the own MFD.

-the test engineer station
The test engineer station is equipped in such a way that mainly two functions can be controlled from the latter:

- simulation from a ground radio station/Tower
- control and supervision of the tests

Two MFD's and one RFI are available to the test engineer, so that he can always indicate one of the images presently shown in the cockpit (MFD1, MFD2 and HMD Pilot/MFD3, MFD4 and commander HMD), and that he is informed on the respective radio frequencies.

CMU Functions
A number of funtions were realized (Figure 6) for the dynamic operation of the CMU. The functions are divided as follows:

Display functions
Display and symbol generator function
Sensor vision
Visual warnings

Control functions
Control display unit functions
Mode control function
Flight control function

Simulation functions
Helicopter dynamics
Engine functions
Weapon functions
Warnings
Flight management function
Navigation function

Basic functions
Test-control function
Stimuli function
Recording/analysis function

The requisite simulation functions are stimulated by the crew inputs, and the respective results such as flight guidance or status information are shown on the cockpit display units, so that the test crew is part of the CMU closed loop simulation (Figure 7).

As a whole the CMU is so efficient as to enable the performance of special tasks and combined tasks up to the largely complete mission in the "man in the loop operation" under quasi-real operational conditions. Furthermore the computer configuration and its efficiency, including the modular functional setup provide sufficient flexibility and extensions. Thus the CMU is a closed-loop simulator, which permits the finding of advanced solutions concerning the man-machine complex in close cooperation with the future user.

Test performance
The CMU tests will start in summer 1990. For the first test phase the main testing aspects are split up according to:

- Mission planning/changing
- Tasks during Target acquisition/fighting
- Tasks during cruising

Mission planning/changing
This mainly involves investigations on mission planning and its changing on board. In this connection activities regarding flight path planning and flight management and activities for the representation and changing of the tactical situation are primary factors.

Tasks during Target acquisition/fighting
The emphasis of the cockpit management investigations planned are the activities after approaching the firing position. In this operational condition the operator's orientation towards the outside, including the associated information representation, is of particular importance. To that effect the facility for sensor vision was provided, which permits an image representation with overlayed symbology on the Helmet Mounted Display and Multi-function Display (Head-Down Display) as a function of the line-of-sight and other command and control inputs.

Cruise flight

In this connection the emphasis is on the following investigations:

- assessment of the symbology overlayed on a map representation
- operational sequencies in the associated modes

In detail, particular attention is paid on the following subjects:

- Special tasks, such as:
  Take-off preparation
  Operation Communication
  Situation processing
  Situation briefing/Takeover Guidance
  Flight path processing
  Emergency situations
  Target acquisition/weapon delivery
  Helmet symbology

- Combined tasks/Mission Phases, such as:
  Enroute flight with route/situation changing
  Enroute flight with route/situation change and system failures
  Monitoring of battle area (Target acquisition, fighting) with system failures

The individual investigations are assessed as follows:

- Subjectively - by the collection of comments
- Objectively, by data recording and determination of handling times and fault frequencies therefrom.

The basis of the first trial phase is a test description consisting of a scenario and a catalogue of questions. The tasks specified in the latter have been agreed with experts (test plots/flying instructors).

It is intended to employ 6-7 crews for the tests, with a two-week trial duration each involving two crews. This approach was already a success in connection with the NH90 and provides for the effective performance of work.

Conclusion
The CMU Design and Development Tool permits the specification of the requirements of a modern Tandem Cockpit such as the PAH-2 by way of experiments in cooperation with the user, and the optimization of work reduction measures so that the following features can be guaranteed in connection with the cockpit development:

- Adaption of the technology to man
- Reduction of the handling steps
- Integrated information display in accordance with the respective situation
- Automation of routine tasks

Thus the CMU is an important feature with regard to a cockpit simulation accompanying the development.

Furthermore, the CMU Tool provides for the timely detection of development risks for the purpose of saving time and cost with regard to the actual development effort.

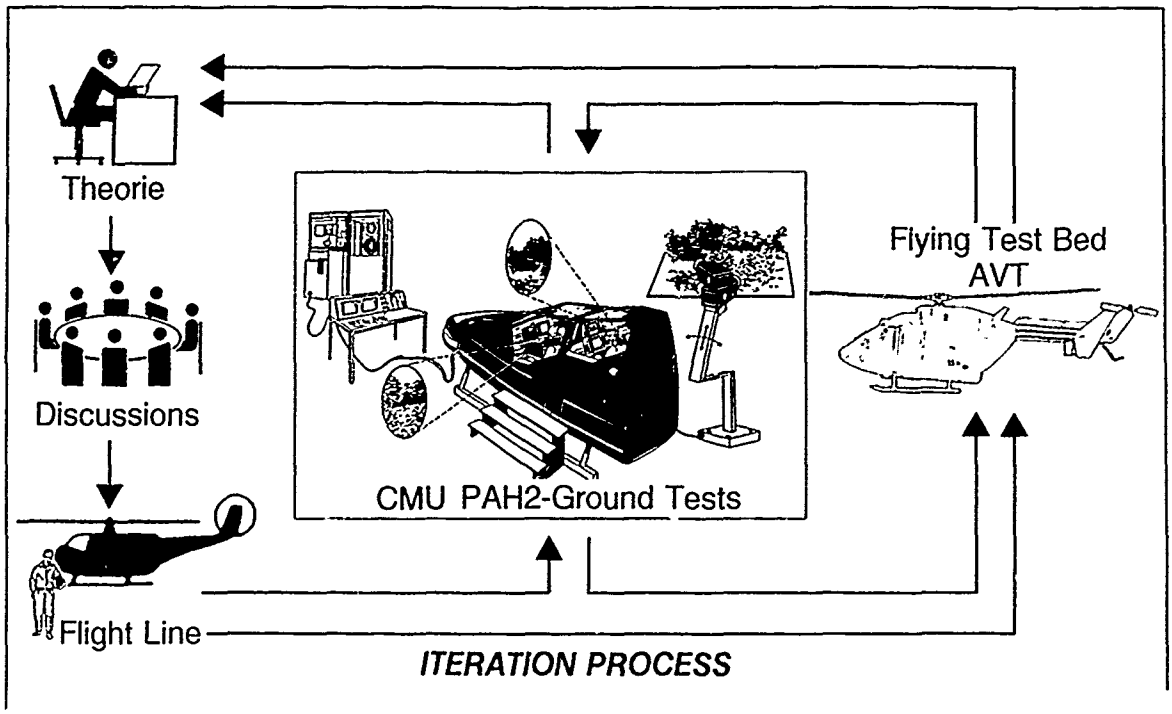


Figure 1: Procedure

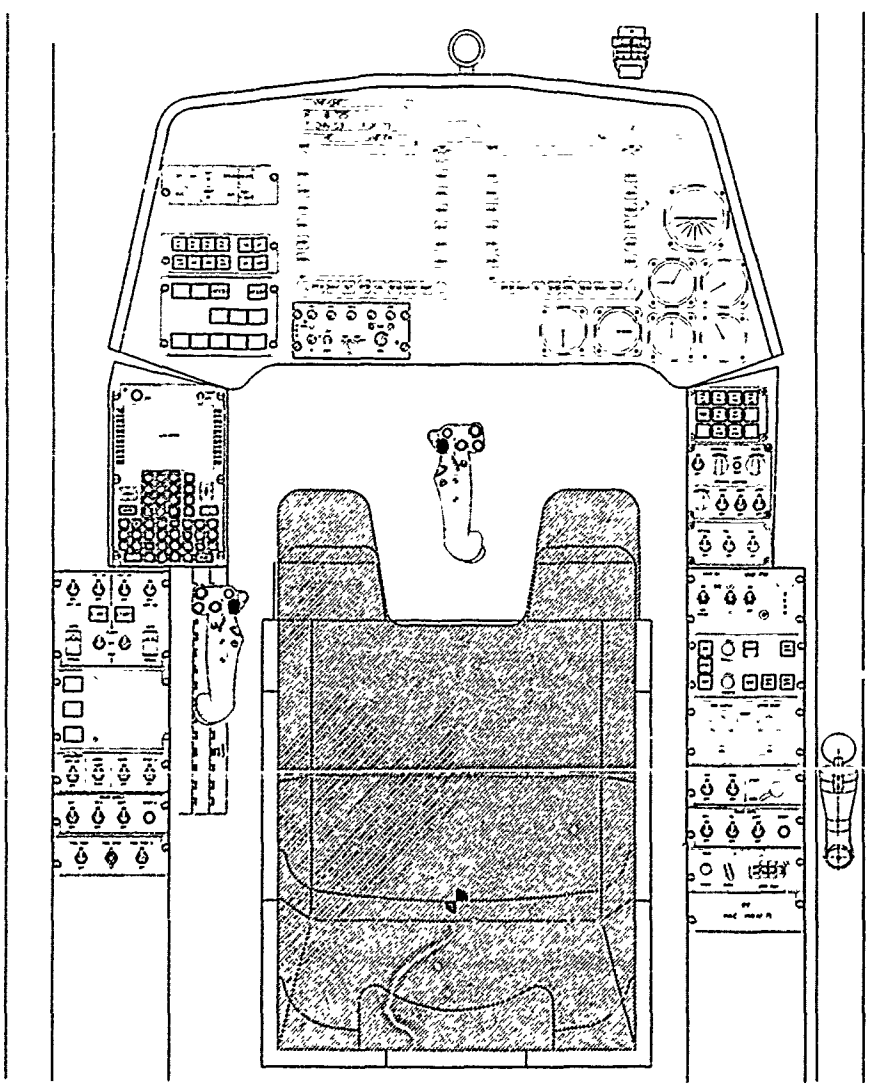


Figure 2: Front Cockpit

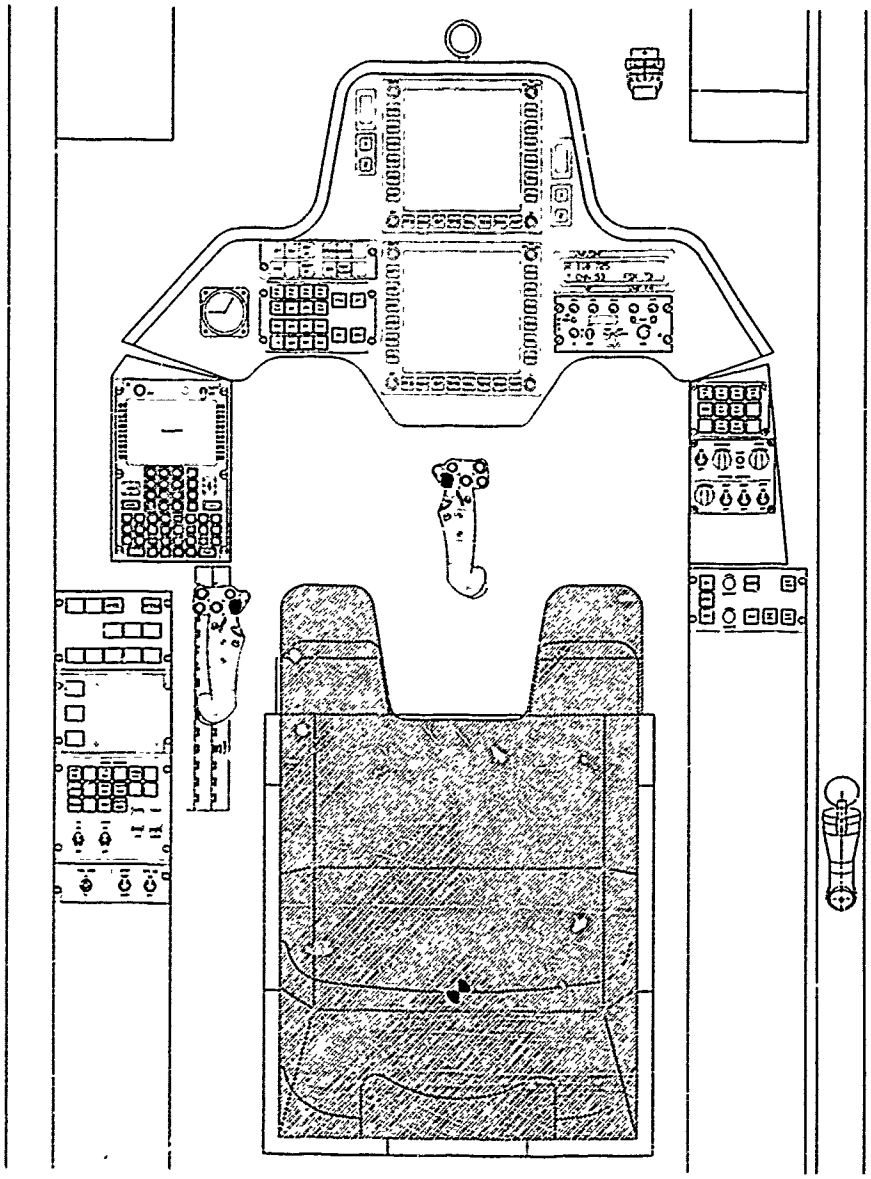


Figure 3: Rear Cockpit

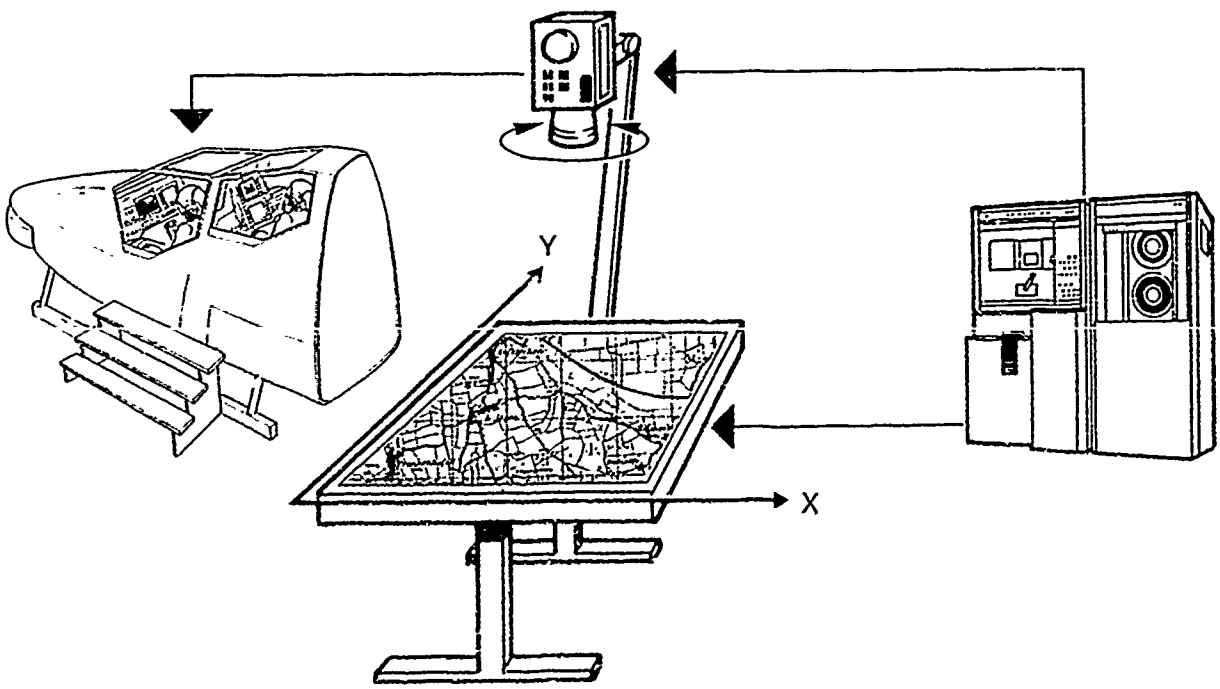


Figure 4: Simulation of digital map

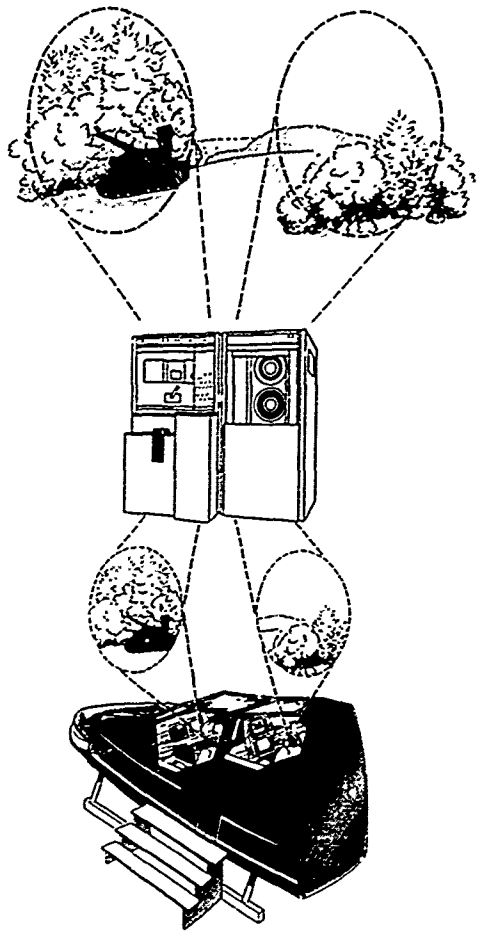


Figure 5: Realization of sensor sight

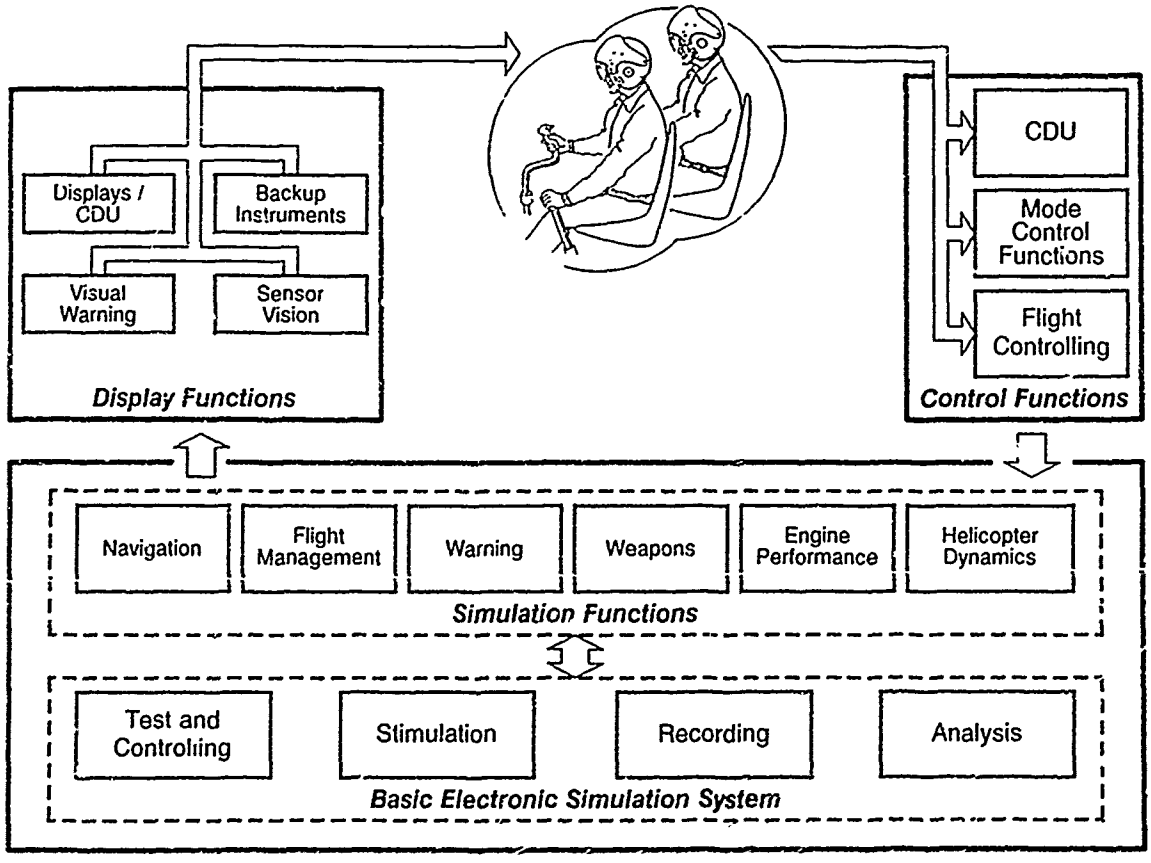


Figure 6: Diagram of CMÜ-Functions

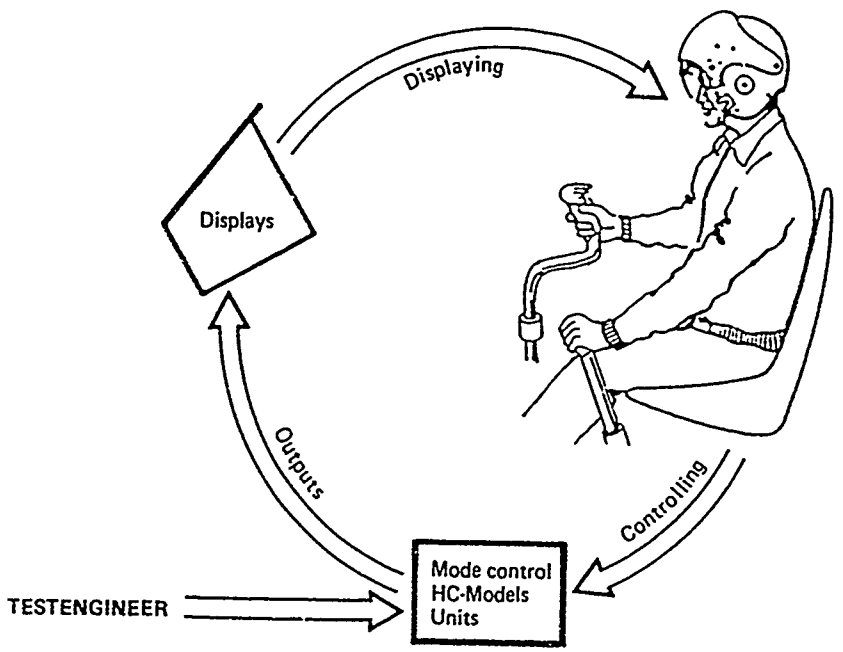


Figure 7: Man in the Loop

COMPUTER-AIDED CONTROL LAW RESEARCH - FROM CONCEPT TO FLIGHT TEST

by

B.N. Tomlinson, G.D. Padfield, P.R. Smith
Flight Management Department
Royal Aerospace Establishment, Bedford, UK

SUMMARY

Active control technology has brought a dramatic change to the way aircraft can be designed and flown. The challenge for flight control research is, given the potential of ACT, to define what is required. The problem is that to answer this question a 'flyable' implementation is needed, whether for a piloted flight simulator or for full-scale flight. The need for an implementation introduces issues of software design and management and a possible 'conflict' with the needs of research. This paper describes an environment for flight control law research being developed in Flight Management Department at RAE Bedford, to provide a rigorous yet flexible framework. A comprehensive life cycle is defined for the evolution of flight control laws from concept via piloted simulation to flight test which, in its current form, has four major phases: conceptual design, engineering design, flight clearance and flight test. Conceptual design covers off-line simulation - modelling, design and analysis - and some pilot-in-the-loop simulation using RAE's Advanced Flight Simulator. The outputs from this stage include information on the response types and system characteristics required. Engineering design is the process of full control law design and includes further refinement of control system architectures via modelling and more detailed piloted simulation. Flight clearance consolidates results from earlier stages and achieves a verified implementation for the target flight control computer. Flight test evaluates the control system in full scale flight and appropriate operational conditions, with further comprehensive data collection.

A structured description of all these phases is presented, with particular application to the ACT Lynx, and some examples of the control law life cycle are given. The role and value of the conceptual phase is particularly emphasized.

1 INTRODUCTION

Active control technology has brought a dramatic change to the way aircraft can be designed and flown. The constituent elements of ACT - sensors, actuators, computers and inceptors - can nowadays be integrated to confer flight behaviour of almost infinite variety within the dynamic envelope of the host airframe.

The challenge for research is, given the potential of ACT, to define what is required. The problem in flight control research is that to answer this question a 'flyable' implementation is needed, whether for a piloted flight simulator or for full-scale flight. The need for an implementation introduces issues of software design and management and a possible 'conflict' with the needs of research. This paper aims to describe an environment for flight control law research being developed in Flight Management Department at RAE Bedford, the centre for flight control research within the Royal Aerospace Establishment.

Existing production, development and research aircraft utilising active control technology have demonstrated startling qualities, both good and bad (Refs 1,2), in their search for optimum flying characteristics. While sensors, actuators and pilot's controller characteristics will all feature in the overall performance, it is the control law, in the heart of the computer system, that determines the flying qualities. A control law in an ACT system aims to assist the pilot by conferring Level 1 handling (Ref 3) throughout the operational envelope and provides stabilisation much more effectively and efficiently than the human pilot. Features such as manoeuvre limiting and carefree handling, implemented via a control law, can open up areas of the flight envelope which might otherwise be inaccessible or potentially dangerous. With these potential benefits, arising from the harmonising of previously conflicting performance and safety requirements, it is hardly surprising that fixed wing and, more recently, rotorcraft manufacturers are striving to capitalise on ACT.

RAE Bedford has been conducting flight control research for many years and has created a number of special facilities for this purpose, both on the ground and in the air. The objectives of RAE control law research are to define requirements - what modes of flight control are feasible and desirable - and how they should be expressed, and to create design tools and a design environment to enable concepts to be explored. A current activity is aimed at establishing a comprehensive life cycle definition for the evolution of flight control laws from initial concept via piloted simulation to flight test, using computer-based aids and tools as much as possible.

Two flight test vehicles serve as the focus for flight control research at RAE: the specially adapted two-seat Harrier known as VAAC (Vectored Thrust Aircraft for Advanced Flight Control - Ref 4) and the RAE ACT Lynx project (Ref 5). The latter

project, still at the proposal stage, is the main driver for this paper and poses particular challenges as its proposed low level, high speed flight envelope requires that its experimental flight control system be designed to be flight critical.

With the ultimate objective being flight test evaluation, there is a need for a framework for research, development and documentation which would follow a control system from concept to clearance for flight. It should be systematic and as procedural as possible, without being in conflict with the needs of a research environment where a high degree of versatility and the ability to foster new ideas is essential. Yet the environment must still aim to produce a result which can be cleared to fly. The life-cycle process should allow for the ability to iterate sub-stages while the overall requirement and design evolve incrementally to final definition. It should also recognise the need to document unsuccessful systems on the way and should carry with it, through all stages, the design criteria that have been employed or are being developed.

It is important that the differences between a research environment and a manufacturing production environment are fully appreciated. The task of research is to determine what is required and produce a clear and comprehensive requirements definition whereas the manufacturing task is to make and deliver in quantity a reliable and effective product.

If research only proceeds on paper, or perhaps in a piloted flight simulator, the standard of implementation may not need to be over rigorous, and 'ad hoc' procedures may be adequate. If, however, the research concept is to be evaluated in full-scale flight, albeit in a research aircraft, then a design and implementation capable of being approved for flight still has to be achieved. This is the challenge. There is a conflict between the objectives of research, with the associated qualities of inspiration and expertise, and the implications of flight, with the equally important qualities of discipline and integrity. Procedures and working practices are needed that reconcile these conflicting qualities without inhibiting creativity.

Some significant issues in research, relevant to the resolution of this conflict, include the following:

the need for freedom to create and try out new ideas, while applying discipline, if verification and validation (V&V) for flight are to be achieved;

control laws are the central theme, not the hardware implementation;

the 'product' of research is design knowledge and criteria, and methods of design, not the final form of the hardware;

there is a need to capture knowledge about a control law as design proceeds;

the specification and expression of a control law should (ideally) be separate from its implementation in software;

systematic and usable procedures and working practices are required to permit evolution of control laws with discipline.

RAE is formulating a Control Law Life Cycle model and a set of procedures appropriate to a research environment. This model has four main phases:

conceptual	- deals with basic concepts but includes creative design with simple models; maps out the requirements;
engineering design	- involves design of a complete control law to match the aircraft model and satisfy requirements;
clearance	- implements the control law on target hardware for flight;
flight test	- evaluates the control law in full scale flight.

This paper highlights the development of this life cycle model, with particular reference to the conceptual and engineering design phases. The model is described in more detail in Section 2 and some examples are given in Section 3.

2 CONTROL LAW LIFE CYCLE

2.1 BACKGROUND TO THE LIFE CYCLE

One of the most significant attributes of the helicopter is its operational versatility; it can hover efficiently, manoeuvre relatively freely in all directions at low speed and accelerate to moderately high forward speed (150-200 kn), all of this in a variety of environmental conditions. Current generation rotorcraft achieve auto-pilot and stabilisation functions through limited performance systems with the result that, for flight in other than close to trim or gentle manoeuvres, the piloting workload increases markedly due to poor handling qualities and the need to monitor carefully any tendency to reach or transgress vehicle limitations.

Significant improvements are possible through the use of active control technology[6], tailoring the response characteristics to the flight condition. This increased versatility presents a new challenge to the control law designer. Response types that confer Level 1 handling (acceptable with low workload) vary with flight conditions, mission task element (MTE) and useable cue environment (UCE). The UCE (Ref 7) is classified as 1, 2 or 3 depending on how difficult it is, with the prevailing visual cues, to control attitude and velocity precisely (1 being the easiest).

Figure 1 illustrates how the required (pitch/roll) response types change according to the currently proposed requirements[7]. In the low speed regime below 45 kn, to achieve Level 1 handling as UCE degrades from 1 to 2 and 3, the preferred response type would need to change from rate command to attitude command/attitude hold and finally to translational rate command with position hold. In forward flight, rate command is preferred for all UCEs. Also indicated in Fig 1 are generic soft and hard flight envelope limits corresponding, for example, to fatigue and static strength loadings. An active control system (ACS) needs to be quite complex to take account of these varied requirements, with automatic mode blending or pilot-operated mode switching together with appropriate limiting at the envelope boundaries.

For a production application, such an ACS would evolve from a requirement specification through design and implementation to flight test and certification. Through this life cycle, design freedom and knowledge typically vary as shown in Fig 2 (Ref 8). The greatest freedom coincides with the least knowledge at the beginning of the process. When the system is ready for flight test, the knowledge has increased but the design freedom is very limited. This interchange of freedom and knowledge is inevitable but it does highlight the need to accumulate as much correct knowledge as possible while the design freedom is high. This requirement is further emphasised by the curve showing typical committed risk and cost during the evolution. Typically 80% of the commitment is made by the end of engineering design. All the functional details, including errors, frozen at this stage will have considerable impact on risk and cost in future phases. With a sound and valid requirement, good quality design tools and coherent verification and validation (V&V) procedures, the process should deliver a successful control system with a high probability of error-free software.

In a research context, several aspects of the above process need modification. In most cases, the requirement is poorly understood or even becomes the objective of the research itself. The engineering design tools may be immature, because of the novel application and again, development of these (eg vehicle simulation model, control law optimisation method) may be an objective of the research. Implementation and the ensuing verification will, in principle, be similar to the production application although the demand for rapid change at this stage may place more emphasis on tools and automated aids than in the production environment.

Flight tests are the ultimate validation, provided relevant operating conditions can be found, but the need to explore a range of configurations, categorising good and bad features, requires considerable flexibility. To emphasise the need to raise both design freedom and knowledge in order to confer this flexibility, the life cycle has to expand to include an iterative mechanism feeding knowledge back to the requirements capture and design phases. In research, it has been found to be appropriate and productive to introduce a conceptual design phase as a requirement capture activity. The introduction of this phase highlights the need for design during the phase of greatest leverage on the final research results. Introducing conceptual design also acknowledges the iterative nature of research, as illustrated in Fig 3. Features of this iteration are:

a. at all stages, the discovery of a fault, design error or uncertainty will require the return to a previous stage.

b. for safety, changes to control laws made in the flight phase should cover incrementally only those regimes already mapped in ground simulation throughout the previous stages.

c. the iterative cycle accumulates knowledge and this has to be documented in a consistent and coherent manner.

d. passage from one phase to another should only be allowed following a satisfactory outcome of procedural tests.

Considering on the one hand the highly interactive nature of these development phases and on the other the safety issues associated with flight critical software, the need for a management support environment is paramount. Important attributes of such an environment are:

i each development phase should consist of a defined set of activities with distinct documented outputs.

ii capture and retrieval of design knowledge should be emphasised, together with audit trail data.

iii clearly defined tests, consistent throughout the phases, are required.

iv procedures should enhance, not inhibit, creativity.

v management support and design tools should form an integrated, computer-based environment.

2.2 OUTLINE OF LIFE CYCLE

In its current form, the RAE Control Law Life Cycle has four major phases (Fig 4): conceptual, engineering design, flight clearance and flight test.

The conceptual phase evaluates the basic concept in a form that can encapsulate the operational requirements. It includes off-line simulation - modelling, design and analysis using control system design and analysis packages such as TSIM (Ref 9) - and some pilot-in-the-loop simulation using RAE's Advanced Flight Simulator. The outputs from this stage include information on the response types and system characteristics required.

Engineering design is the process of full control law design in conjunction with a representative vehicle model, and includes further refinement of control system architectures via detailed modelling and extensive piloted simulation.

Flight clearance consolidates results from earlier stages and achieves a verified implementation for the target flight control computer. Validation of the design to include a loads and stability analysis will also form part of this phase.

Flight test evaluates the control system in full scale flight and appropriate operational conditions, with further comprehensive data collection.

Each phase is described with the aid of structure diagrams, as used in the Jackson JSP and JSD methods for software and system design (Ref 10). The diagrams are complemented by a process dictionary containing text describing the sub-processes and activities. A simple example of a structure diagram is shown in Fig 5 and contains the basic elements of sequence, iteration (*) and selection (o). Such a diagram is read as

> "Generic structure consists of Start sequence followed by Middle followed by End sequence. Middle consists of an iteration (zero, 1 or many repetitions) of Select A or Select B."

Thus processes evolve from left to right with activities at the lowest boxes (leaves) of each branch. One of many possible sequences of activities in an actual life history of 'generic structure' is

> Start, A, A, B, A, End.

Formalised structure diagrams of this nature possess many benefits

- they clarify the overall process;
- they identify the sequence of possible and necessary activities;
- they require the basis of decisions (whether of selection or iteration) to be defined;
- they provide a helpful means of communication and a focus for discussion among team members.

A computer-based tool is available to generate them (Ref 11).

The whole life cycle model and method is currently in a prototype stage at RAE. Activities in the first two design phases are now discussed briefly, together with aspects of the design knowledge that needs to be captured. Some examples are presented later. The structure of activities within the clearance and flight test phases are still evolving and will be reported on at a later date.

2.3 CONCEPTUAL PHASE

As noted above, the emphasis in the conceptual phase is to establish the design requirements and criteria for the engineering design phase. This phase is creative, the principal stages being problem expression, design (consisting of modelling and evaluation) and review; the full process structure is illustrated in Fig 6, emphasising the specific activities of each sub-phase, including documentation.

Within problem expression, a simple but non-trivial sub-phase, is a choice of activities that have considerable influence on the value of the research. Problem expression can take the form of text and diagrams; the entry in the process dictionary may look like

> 1.1.2 *Express: create new expression of the problem - using text and diagrams express the high level statement of the problem in sufficient detail to initiate the conceptual phase.*

An example might be -

> design a full authority active control system for a Lynx helicopter to achieve Level 1 handling qualities in air combat.

Necessary subsidiary problem expressions would be -

> a. Determine location of (pitch/roll/yaw) Level 1/Level 2 handling boundary on the bandwidth/time delay diagram (Ref 7) for rate-response type rotorcraft in tracking phase of air-to-air combat MTE.
>
> b. Determine location of Level 1/Level 2 handling boundaries on the attack-parameter diagram for acquisition phase of air-to-air combat MTE (Ref 7).
>
> c. Determine minimum level of various pitch/roll/yaw cross-couplings necessary to guarantee Level 1 in air-to-air combat MTE.

Design criteria can be expressed in various ways. In the frequency domain, bandwidth and phase delay metrics are defined in Ref 7, indicating the nature of the system response. These two metrics can be plotted against one another as shown in Fig 7, which identifies boundaries between the three handling qualities levels. Hence, at the design stage, a control law can be tuned for good closed loop characteristics. Desirable locations for the system roots are also specified in the 's' plane for different vehicle axes and tasks. To cater for large amplitude motions, time domain criteria are specified as shown in Fig 8. Ratios of peak angular rate over peak attitude change (the attack-parameter) are used to characterise handling qualities levels in the time domain.

Eventually this part of the research is complete when the criteria are validated and their range of application established but, to begin with, many gaps may exist. For example, whereas the basic format for (a) and (b), as illustrated in Figs 7 and 8, may be established, that for cross-coupling may not be and another level of problem statement needs to appear. In the conceptual phase, problem expression is the key to 'starting on the right track', but it is the modelling and evaluation sub-phases that give substance to the overall phase and generate new knowledge. Activity 1.2.1 may be described in the process dictionary as

> 1.2.1 *modify existing model: an appropriate conceptual model for developing the required criteria already exists, so select and modify as required.*

The modelling requirement must clearly be traceable to the problem expressions. For example, expression (a) above will require models that allow bandwidth and time delay effects to be evaluated independently and in isolation from other interferences.

For a rate command system in roll, such a model might take the transfer function form

$$\frac{p}{\eta_{lc}} = \frac{Ke^{-\tau s}}{\frac{1}{\omega_n} s + 1}$$

where p is the roll rate, η_{lc} is the pilot's lateral cyclic, ω_n and τ the natural bandwidth and effective time delay respectively and K the control power. The roll axis bandwidth is a function of both ω_n and τ and, like τ, is an equivalent or conceptual system parameter that encapsulates a range of higher order effects that occur in real aircraft, eg actuation, rotor dynamics, stick dynamics. This equation is a low order transfer function model, the coefficients of which can be modified to vary the 'aircraft' response in a direct and explicit manner. In this way, representative tasks can be explored experimentally and required system bandwidth, damping etc can be determined.

Examples of conceptual modelling and simulation studies[6] conducted at RAE will be discussed further in Section 3 of this paper. Simulation activities occur in the second of the design sub-phases, as part of evaluation and are composed of non-real-time TSIM and real-time piloted simulation activities. The piloted experiments must be designed to the same level of detail as a more comprehensive engineering simulation. In principle, the tests here should be identical to those conducted later, in Phase 2, in terms of tasks, UCE, pilots, etc. Full documentation is crucial at this stage; the dictionary entry might be as follows -

> 1.2.9 *document results of piloted evaluation - should contain a complete description of tasks, simulation environment (eg cues) plus supporting validation documentation, pilot details, together with the simulation results, eg pilot comments plus ratings, results of data analysis presented in format established in Phase 1.1.*

A key feature of the conceptual design phase (1.2) is the iteration, allowing several passes through the modelling-evaluation sequence if required, to derive the required knowledge. This may, for example, be necessary to establish a suitable format for pitch/roll or roll/pitch cross-coupling criteria.

A review sub-phase is included in the conceptual phase, as in all four major phases, acknowledging the need to make a decision at each point as to whether the results are satisfactory, and sufficiently promising to proceed further, and if so, to deliver a specification for the next phase.

2.4 ENGINEERING DESIGN

As in the conceptual phase, problem expression, design and review cover activities in the engineering design phase - Fig 9; however, the level of detail will generally be considerably greater and elapsed activity times considerably longer. The problem expression sub-phase takes as a starting point the specification output from Phase 1, representing, in part, the design criteria for the control system.

Greater detail will be required, however, to reflect the depth to which the problem is tackled in this phase. Environmental constraints and robustness criteria will form part of the expression, as will requirements on uncontrolled modes eg structural. Internal control system loop performance requirements may also be defined in terms of gain, admissible interactions, structure etc. The design sub-phase contains more substantial activities within modelling, synthesis and evaluation. Fig 9 is expanded as far as the leaves only for the synthesis activities: these include method selection, control law structure and parameter optimisation and documentation. Control law design method selection is emphasized as an activity; the approach taken here, whether time or frequency domain, classical single-input/single-output (SISO) or multiple-input/multiple-output (MIMO), will depend on a number of factors. Experience of the engineer is important but a method that matches the way the problem is expressed will always have clear advantages. The optimisation activity involves craft-like skills, trading off performance against robustness, to achieve the best controller. On-line documentation during the activity is crucial to avoid the perpetration of the 'black-box syndrome', ie a unit whose internal functions are not known in detail. A working practice that emphasises rationale choice and decision making and the associated recording is favoured.

2.5 CLEARANCE AND FLIGHT TEST PHASES

Activities within these later phases are the subject of current research at RAE. The clearance activities will include software verification and a degree of validation using more comprehensive vehicle mathematical models than in earlier phases. Flight test represents the ultimate research evaluation although, ironically, this phase offers no scope for design innovation and creativity; flight test is essentially a knowledge gathering exercise but there is considerable scope for innovation in experimental design. Such experimental activities will be emphasized in this phase as the experiments will, above all else, determine the success of the research. The iterative nature of the whole life cycle (Fig 3) is again emphasized. Most concepts are expected to have several iterations before yielding mature knowledge, fit for use in a procurement requirement specification or in definitive handling criteria.

Examples from previous RAE experience will now be reviewed and related to the life cycle.

3 CONTROL LAW DESIGN AND EXAMPLES

3.1 CONCEPTUAL STUDIES

Early studies of helicopter agility (eg Ref 12) used a relatively complex, non-linear mathematical model of the helicopter. This model allowed realistic flight over the complete speed range from hover to cruise, and could be adapted, via parameters, to represent the dynamics of various rotor types. Such variations and their effects on helicopter agility were explored in these early experiments. Study of a broad range of desirable response types and characteristics using such a model required the addition of a full control law. Design of a control law to match the model and enable a variety of characteristics to be defined and assessed proved to be a major task, and too inflexible for initial research. It was for such reasons that the 'conceptual' model was created[13] to provide a suitable vehicle for the exploration of desirable control characteristics.

Fundamental piloted simulation studies of helicopter control using the conceptual approach have evaluated, for example, attitude versus rate response types and the benefits of additional augmentation functions to provide turn coordination with height control (Refs, 6, 14).

To complement the requirements of Ref 7, some work at RAE Bedford has focussed upon metrics for the measurement of helicopter agility (Ref 15). Fig 10 illustrates the form of the agility factor that has been derived and shows that, as pilot aggression increases, so agility factor increases, but the handling qualities ratings can rapidly deteriorate. It is vital for future battlefield helicopters that maximum use can be

made of the inherent agility of the aircraft and that the vehicle handling qualities are Level 1 at all times. Conceptual simulation can, in principle, identify the preferred response types and handling parameters to confer these attributes.

Most recently, the potential of helicopter carefree handling systems has been assessed[16] in the Bedford Advanced Flight Simulator. Several configurations were evaluated, including visual warning, tactile warnings via collective and pedal shakers, and direct intervention systems. Dramatic improvements were observed (Fig 11) particularly with the highest level of augmentation. These experiments demonstrated the power of direct intervention control and were only possible because it was relatively straightforward to add carefree handling features to the conceptual model. Devising such techniques for a full model of a helicopter and its advanced control system is currently in progress but is expected to take several man-years of effort.

The examples cited above were conducted without the benefit of a management environment. They did, however, highlight the fundamental value and importance of conceptual studies and modelling and the need for a cohesive method that connected such activities with engineering design.

3.2 ENGINEERING DESIGN PHASE

This phase consists of mapping the required characteristics from the conceptual phase onto the target aircraft. From the control law designer's point of view, this involves using the design freedoms, available in the method employed, to converge on a solution which meets the design requirements in terms of performance and which also provides adequate levels of stability and performance robustness. A more comprehensive model of the flight behaviour will be required exhibiting the real-world non-linear and time-dependent effects.

To compensate for variation in the aircraft's behaviour due to changes in forward speed, weight etc; the control law design will typically be optimised at a number of aircraft operating points, and some means of gain scheduling or controller switching employed to recognise the non-linear nature of the control problem.

The design method that will be highlighted in this paper is that of H-Infinity, which enables the designer, as part of the controller synthesis phase (Fig 9), to specify frequency-dependent weighting terms that characterise performance and robustness requirements. The detail of the method will not be elaborated here. Refs 17 and 18 describe the use of this design method in a helicopter application, which resulted in a successful piloted simulation trial on the RAE Advanced Flight Simulator.

As an example, Fig 12 shows that several points in the design space of phase delay versus system bandwidth can be obtained from use of the method. The handling qualities at each point can then be evaluated in piloted simulation to check compliance with the criteria.

Control design methods are usually based on linear systems theory, hence the maximum performance that can ultimately be obtained depends upon the extent to which non-linear effects cause the closed-loop response to be degraded. This can arise from non-linearity either in the vehicle model employed in the simulator or in the actual aircraft aerodynamics experienced in real flight, as well as in the actuators and sensors. In particular the actuators are prone to authority, rate and acceleration limits which must be accommodated.

Introduction of the controller into the more realistic non-linear model may therefore introduce difficulties in meeting the required performance, which will need to be resolved. It may be that the initial specification was too optimistic, or poses unacceptable demands on the aircraft actuation, engine, or structure. If so, this would require an iteration back to the conceptual phase to quantify the impact of a reduction in system performance.

The engineering design phase is completed when 'acceptable' closed-loop performance is obtained from the combination of control law and vehicle model. The output of this phase is a definition of a non-linear control law, in the form of a set of state-space matrices and block diagrams that specify the controllers, to enable the required performance to be achieved over the full flight envelope of the aircraft. Discrete switching between the controllers evolved for each design point may be necessary, using so-called 'bumpless transfer methods'.

4 TOOLS AND FACILITIES FOR THE LIFE CYCLE

4.1 TOOLS AND FACILITIES

In progressing through the phases of the control law life cycle, a control law designer will employ a number of tools and facilities. Tools are generally computer-based software packages. Facilities are major resources such as aircraft and flight simulators. Those that exist within the RAE environment at present are described briefly here.

During the conceptual phase, a modelling and prototyping tool, such as the TSIM (Ref 9) or MATLAB (Ref 19) software packages, will be employed to express and synthesize

the initial concept and provide a first level of analysis. TSIM, for example, enables a model to be defined and analysed by classical frequency response and root locus methods. Time response behaviour can also be generated and examined, with and without disturbances such as turbulence.

A fully engineered control law model can also be analysed in TSIM, in conjunction with an aircraft mathematical model, using the same software implementation that is used in the simulator. Portable control law and aircraft models are created to work both under TSIM (on VAX/VMS computers) and on the AFS (on Encore Concept-32 computers).

Software design is aided by tools which support the Jackson design method (JSD), eg Speedbuilder and Program Development Facility (PDF). The latter has the ability to generate compilable Fortran, Coral66 or Ada source from the design description expressed as process structure diagrams. Static analysis tools are also available.

Control of developed software is also an important task, to ensure that what is used is approved, and to provide a mechanism to manage change. The specific tool in use for this purpose is Lifespan (Ref 20).

The major facilities are the Advanced Flight Simulator (AFS), illustrated schematically in Fig 13, and, for flight test, the research aircraft - VAAC Harrier (Ref 4) and Lynx (Ref 5).

The progress of all simulations using the AFS is monitored using software (Ref 21) built on a database management package (Ingres). This enables definition of what is flown to be closely controlled, and the results of experiments to be captured, so that, after a sortie, the user can identify, through an automated sortie journal, all the main actions and the precise circumstances in which data were gathered. Data acquired from flight tests is also collated and managed by an Ingres-based system, but using a simpler approach (Ref 22).

4.2 COMPUTER IMPLEMENTATION

The control law life cycle outlined in this paper generates in each phase information and knowledge about the control law design and its implementation that need to be retained. A potential problem is that the total procedure could involve a new and significant overhead; in practice, the procedure must be a help to the designer. The computer can help here by automating the procedure and by engaging the designer in a dialogue at the end of each phase to ensure that all design knowledge is captured properly. Design information accumulates progressively: the state of the design, with descriptions at various levels of detail, both functional and structural; goals and goal structure; design decisions and the rationale for the design, eg related to sensor availability, choice of feedbacks, assumed motivator authority; and results of analysis. Initially, this would be assembled in a database as an information source. A second phase could involve creation of a 'design associate' to provide another level of interpretation or transformation of the knowledge. Such an associate, on which research is in progress elsewhere (eg Refs 23, 24), could, for example, observe that the designer has varied a parameter during the design activity and selected a final value, and could ask why. A design associate is relevant not only to the design process but also to future 're-design' within the system, in real-time (Ref 25).

5 CONCLUSIONS

Flight testing controllers for enhancing flight performance provides the ultimate test of the viability of an ACT system. In flight critical conditions, control laws have to be correct with a very low risk of failure or of the occurrence of a design fault. This emphasises the need to establish a coherent and consistent requirements capture and design cycle prior to test. The evolution of the control law through these early stages is likely to be considerable. In a research environment, when many ideas are developing in parallel, a disciplined working practice needs to be established. This paper has covered the topic from the perspective of research at RAE and has outlined the prototype of a suitable life cycle model. A number of observations can be made in conclusion:

1 Given the 'freedom' of a software-based control law, the challenge today is to define the requirement. Research is needed to identify -

what is required - ie the character of the response;

how the requirements might be achieved - ie effective methods to design and implement the requirements;

how well the requirements are actually achieved in an implementation.

The last of these constitutes validation of the initial concept and is the aim of taking a control law through to flight test.

2 Research into flight control laws poses particular problems because the emphasis is not on the technology of the hardware implementation or even on the software but more on the concepts being evaluated. For the concepts to reach the stage of being suitable for evaluation through flight test, an implementation

still has to be achieved to high standards. Control law design in a research environment thus requires a more rigorous approach than used in the past, and demands a coherent and consistent requirements capture and design cycle. How to impose such discipline without inhibiting creativity is the management challenge RAE is attempting to meet.

3 A control law life cycle has been defined for research at RAE consisting of four main phases:

Conceptual
Engineering design
Flight clearance
Flight test

4 The conceptual phase includes conceptual simulation. This kind of simulation has an important part to play in the identification of appropriate response types and design criteria for control laws. In effect, it delivers the requirement and should be performed before any detailed design begins. This is a special feature of the RAE approach.

5 Requirement capture and expression should be separated from any consideration of representation in software but requirement definition and conceptual design are interactive.

6 Throughout the control law life cycle, capture of design knowledge is vital, not just what but why. Some control law concepts will be abandoned during the overall process. In a research environment, it is important to record the details, otherwise, the same path may be followed again at a later date. Furthermore, unsuccessful systems are also a contribution to knowledge and may identify the need for further research.

7 Concepts that complete the life cycle through to flight evaluation will only do so on the basis of 'audit trail' information that keeps track of what decisions were taken, why and by whom.

8 Computer-based methods to capture this design knowledge and 'audit' information are essential, to reduce overheads and encourage designers to record this information.

9 Tools used on computers, (eg design packages) need to be implemented with 'quality' in order to be trusted; their implicit methods and algorithms must be visible and their range of validity must be declared. This does not do away with the condition that tools still need intelligent users.

The life cycle model under development at RAE utilises features of the Jackson software design method to formalise the multitude of activities and sub-phases within the process of control law research from concept to flight test. This paper has highlighted the conceptual and engineering design phases within the model and has provided examples of how previous research activities fit into the proposed structure. Work is continuing to detail the clearance and flight test phases in readiness for ACT flight research in flight-critical helicopter applications.

REFERENCES

1. AGARD FMP Symposium on "Active Control Systems - Review, Evaluation and Projections", Toronto AGARD CP 384 (1984).

2. "Fly by wire under fire - Saab Gripen prototype crashes", Flight International, 11 February 1989.

3. Cooper, G.E., Harper R.P., "The Use of pilot rating in the evaluation of aircraft handling qualities" NASA TN-D-5153, AGARD Report 567 (1969).

4. Nicholas, O.P., "The VAAC VSTOL flight control research project" SAE Paper 872331 Proc International Powered Lift Conference P-203, Santa Clara, CA (1987).

5. Padfield, G.D., Winter, J.S., "Proposed programme of ACT research on the RAE Bedford Lynx" RAE TM FS(B) 599 (1985).

6. Buckingham, S.L., Padfield, G.D., "Piloted simulations to explore helicopter advanced control systems" RAE TR 86022 (1986).

7. "Handling Qualities Requirements for Military Rotorcraft" United State Army Aviation Systems Command Aeronautical Design Standard, ADS-33C (1989).

8. Schrage, D.P., "The impact of total quality management (TQM) and concurrent engineering on the aircraft design process" AHS Conference on Vertical Lift Technology, San Fransisco (1989).

9. Winter, J.S. Corbin, M.J. Murphy, L.M., "Description of TSIM2: a software package for computer aided design of flight control systems" RAE TR 83007 (1983).

10. Cameron, J.R., "JSP & JSD: The Jackson approach to software development" IEEE Computer Society Press (1983).

11. "Program Development Facility" Michael Jackson Systems Ltd.

12. Tomlinson, B.N. Padfield, G.D., "Piloted simulation studies of helicopter agility" Vertica, Vol 4, pp 79-106 (1980).

13. Winter, J.S. Padfield, G.D. Buckingham, S.L. "The evolution of active control systems for helicopters - conceptual simulation to preliminary design" AGARD FMP Symposium on "Active Control Systems - Review, Evaluation and Projections", Toronto AGARD CP 384 (1984) Also RAE TM FS(B) 594 (1985).

14. Buckingham, S.L. "Exploratory piloted simulation studies of helicopter advanced control systems" AGARD GCP Symposium on "Helicopter Guidance and Control Systems for Battlefield Support" AGARD CP 359 (1984).

15. Charlton, M.T. Padfield, G.D. Horton, R.I., "Helicopter agility in low speed manoeuvres" RAE TM FM22 (1989).

16. Massey, C.P. Wells, P., "Helicopter carefree handling systems" Proceedings of Royal Aeronautical Society Conference on "Helicopter Handling Qualities and Control", London (1988).

17. Yue, A., H-Infinity Design and the Improvement of Helicopter Handling Qualities" PhD Thesis, Oxford University, (1988).

18. Yue, A. Postlethwaite, I, Padfield, G.D., "H-Infinity Design and the Improvement of Helicopter Handling Qualities" 13th European Rotorcraft Forum, Sept 8-11 1987 Vertica, Vol 13, No 2, pp 119-132 (1989).

19. Moler, C. et al, "PRO-MATLAB Users' Guide", The Maths Works, Inc (1986).

20. "Lifespan" YARD Software Systems Ltd (1984).

21. Tomlinson B.N. Bradley, R. Flower, C., "The use of a relational database in the management and operation of a research flight simulator" AIAA Paper 87-2573 CP (1987) Also RAE TM FM 1 (1988).

22. Foster G.W., "A system for analysing flight trials data" RAE TM FS(B) 672 (1987).

23. Myers, T.T. McRuer, D.T., "Advanced piloted aircraft flight control system design methodology - the FCX flight control expert system" Systems Technology Inc (1988).

24. Butz, B.P., "An autonomous associate to aid in control system design" IEEE Montech 87 Conference (1987).

25. Boulton, C.B. Bennett, M.E. Clare, J.N. "A study of the feasibility of using artifical intelligence in aircraft flight control systems" Cambridge Consultants Ltd (1988).

Copyright (c) Controller HMSO London 1990

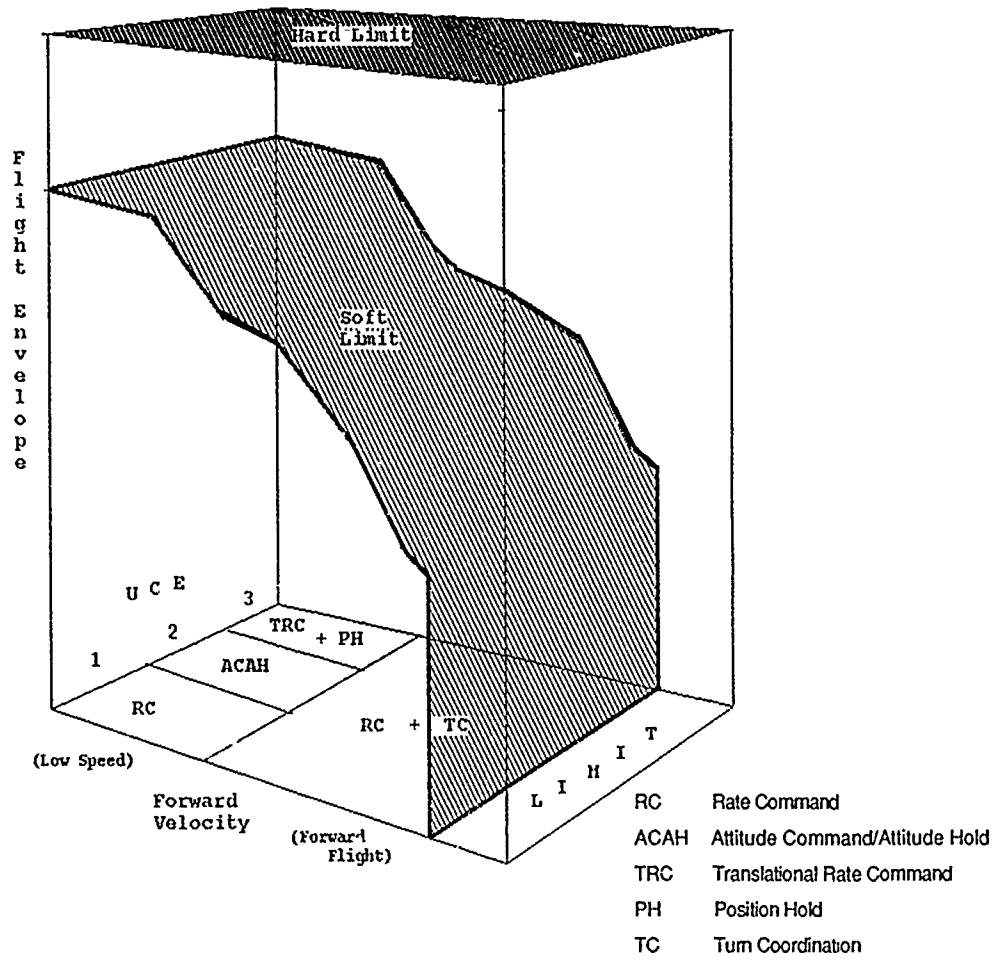


Fig 1 Rotorcraft (Roll/Pitch) Response Types for Level 1 Handling in Specialised Mission-Task-Elements eg Sidestep, Slalom, Air Combat

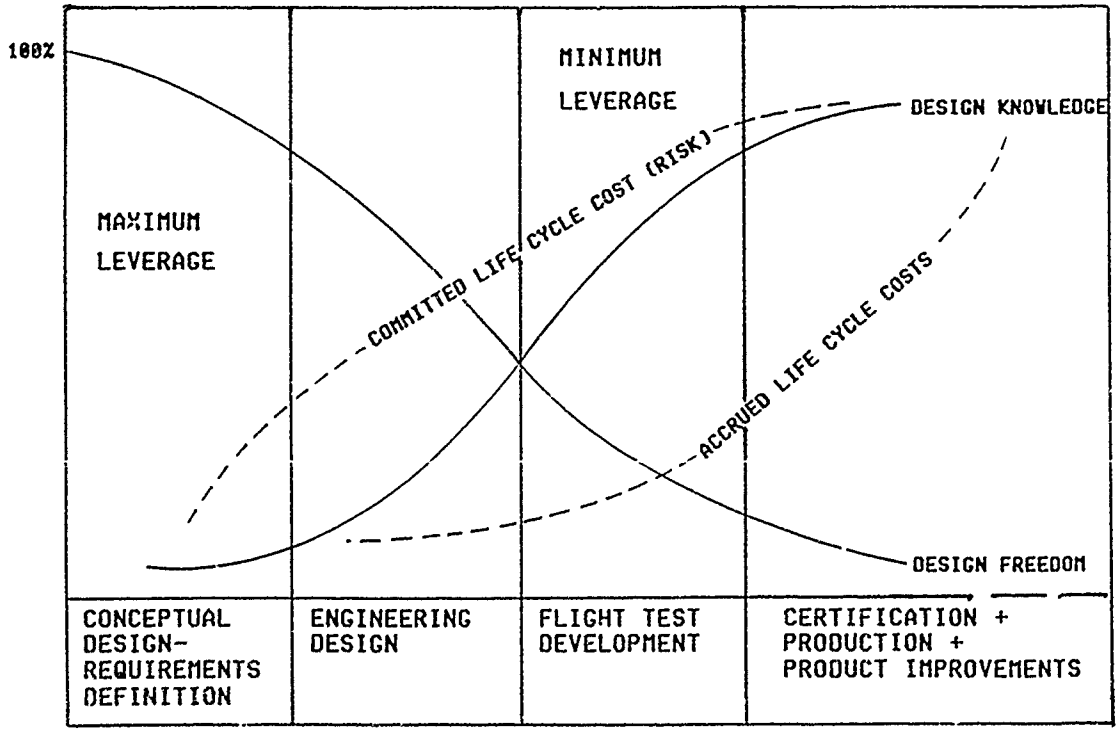


Fig 2 Trends During Procurement Cycle

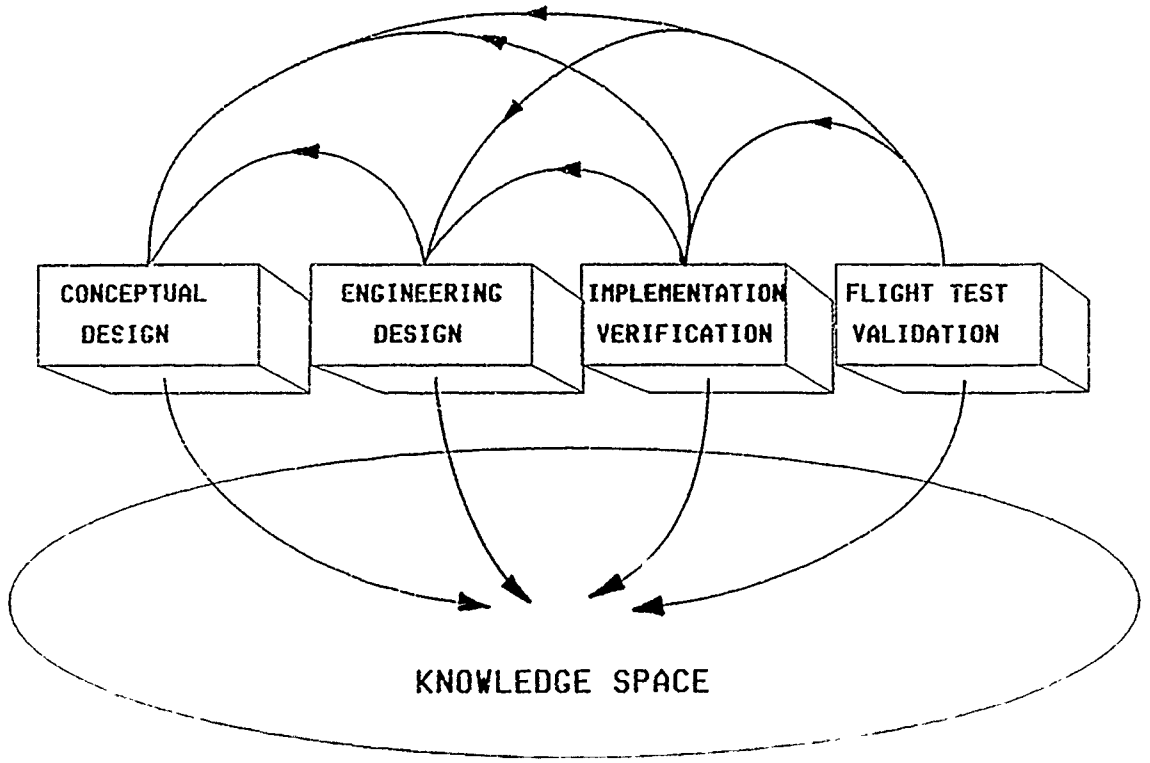


Fig 3 Sources of Knowledge

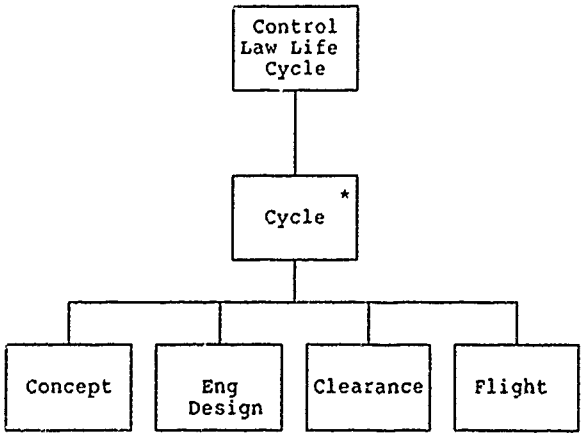


Fig 4 Control Law Life Cycle

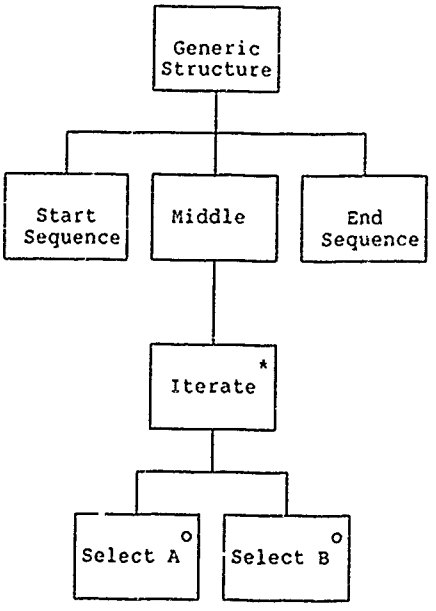


Fig 5 Generic Structure Diagram

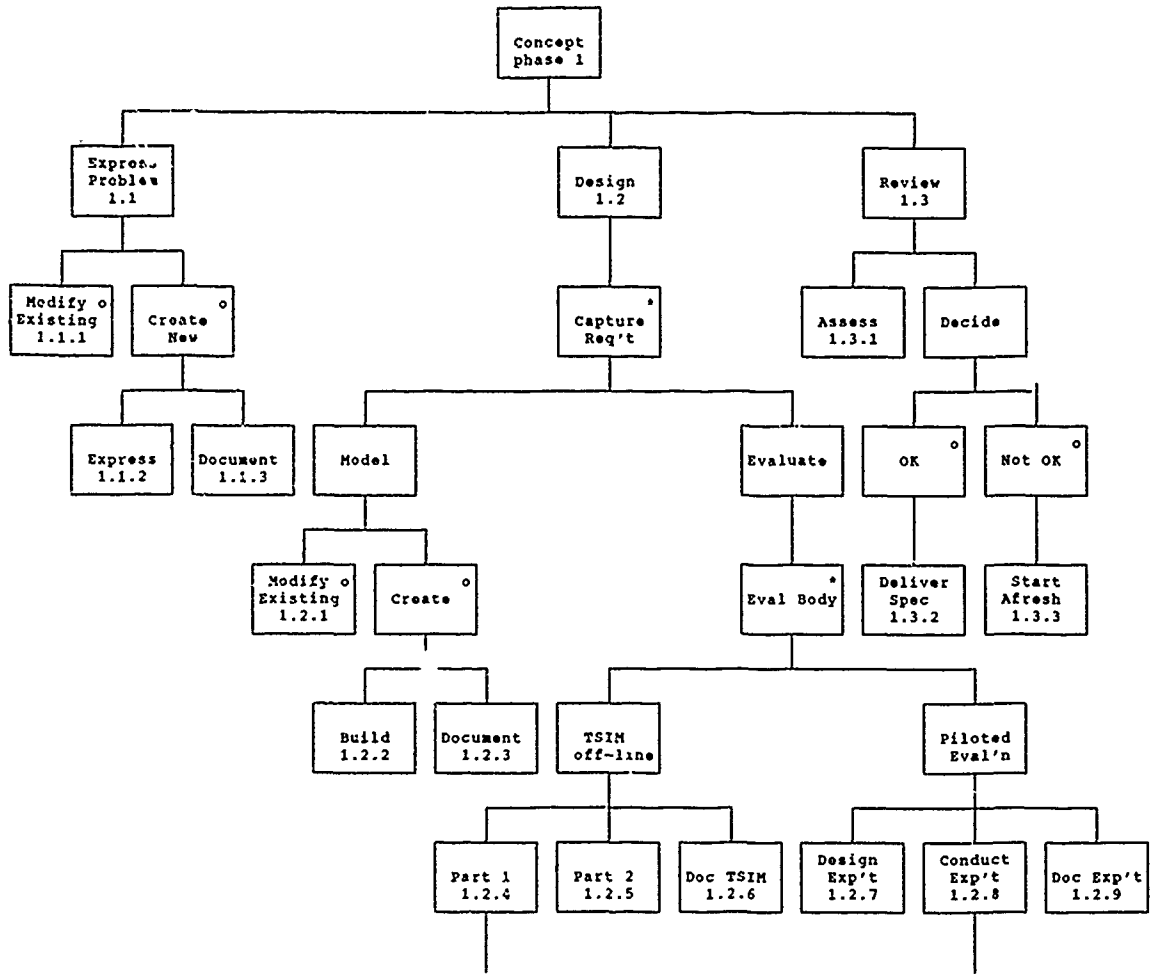
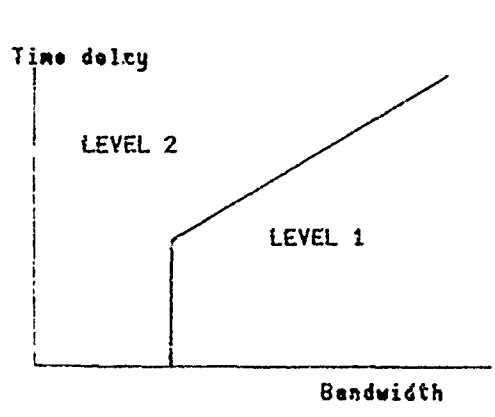
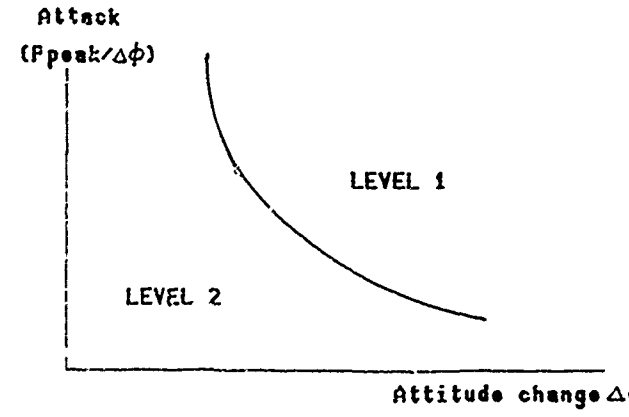


Fig 6 Conceptual Phase Structure Diagram



(a) Bandwidth/Time Delay

Fig 7 Bandwidth/Time Delay Handling Qualities Diagram



(b) Attack-parameter

Fig 8 Attack-Parameter Diagram

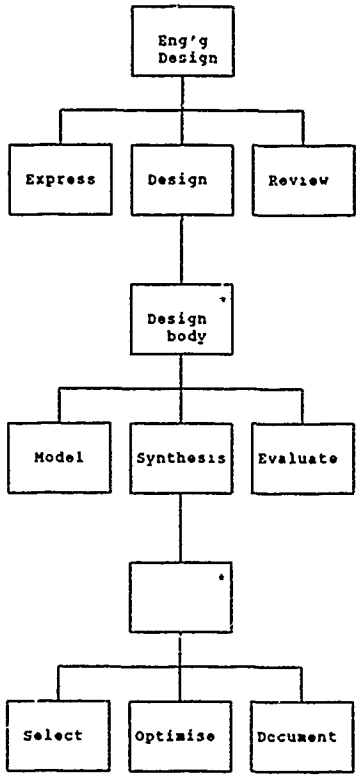
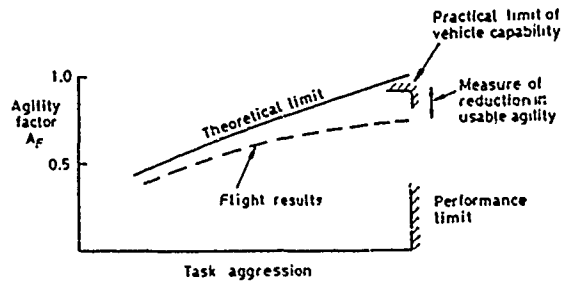
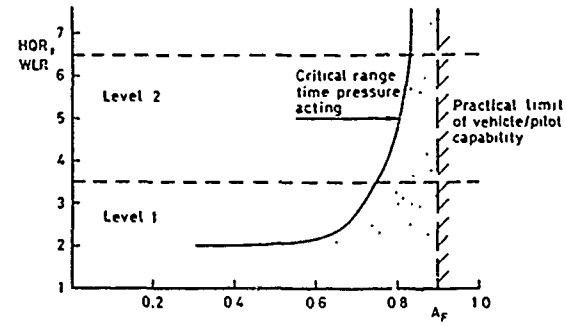


Fig 9 Engineering Design Phase Structure Diagram



a) Variation with task aggression



b) Handling and workload degradation

Fig 10 Agility Factor

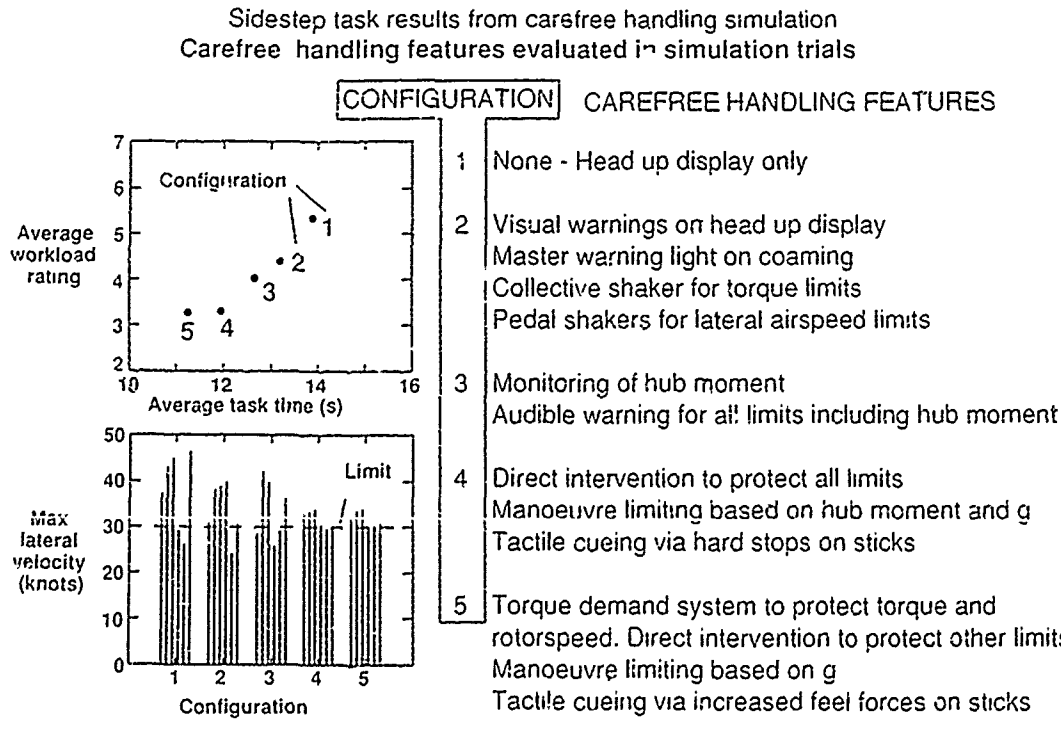


Fig 11 Sidestep Task Results from Carefree Handling Simulation

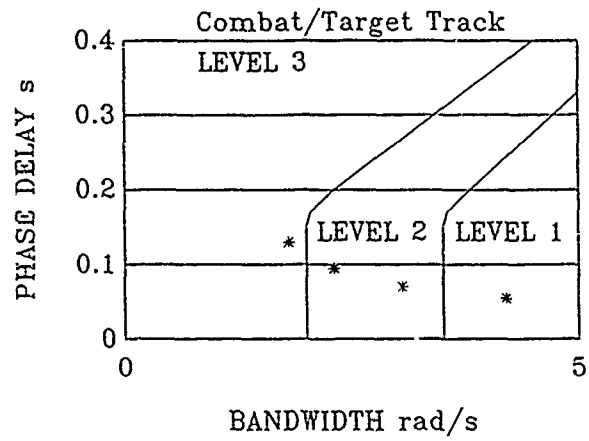


Fig 12 H-Infinity Design Results for a Lynx Helicopter

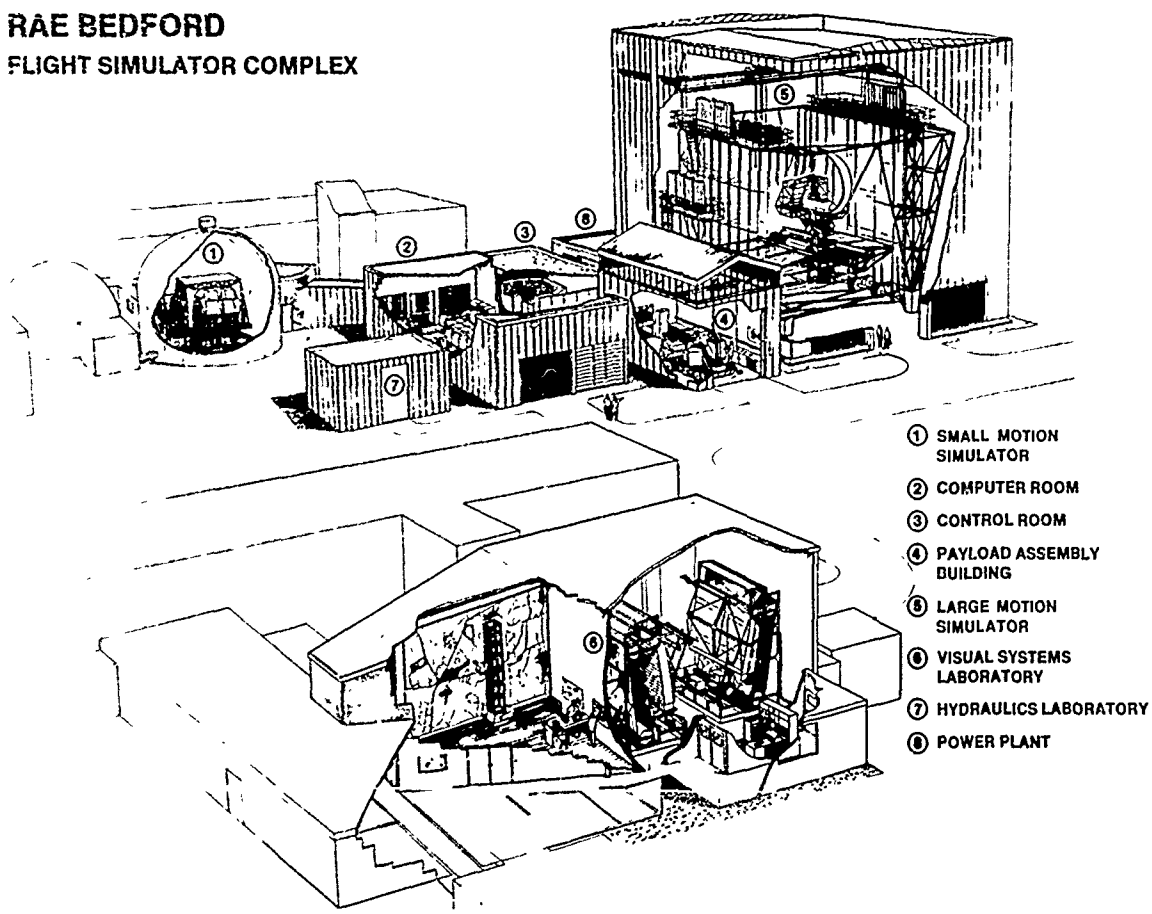


Fig 13 RAE Bedford Flight Simulator Complex

REAL-TIME HARDWARE-IN-THE-LOOP SIMULATION FOR 'ATTAS' AND 'ATTHES' ADVANCED TECHNOLOGY FLIGHT TEST VEHICLES

by

Peter Saager
DLR Braunschweig
Institut für Flugmechanik
Flughaf
D-3300 Braunschweig
Germany

SUMMARY

The paper dealing with the applications of the ground based real- time simulations used by the Institute for Flight Mechanics at DLR, Braunschweig, is followed by the presentation of the implemented hardware concept and some special aspects with regard to the simulation computers AD10 and AD100. This includes considerations about the analog and digital input/output handling with connected hardware in the loop (HIL).

The essential advantage of higher simulation languages (CSSL- based ADSIM, MPS10) as important software tools for the development, modification and implementation of complex and extensive software modules under real-time simulation aspects is another point of consideration. Based on this discussion is the description of problems with the correlation between the simulation frame-time and the actual integration stepsize. Suitable integration algorithms and other supporting methods used within real-time simulations to compute the dynamics of stiff systems are described.

The presented helicopter's mainrotor simulation model serves as an example for the complexity of software modules, incorporated into the real-time simulations.

Finally the actual method for the verification and validation of the simulation results and the principle diagnostic- and test- software application concept is explained. The conclusion then gives some informations about the usefulness of ground-based real-time simulation facilities with regard to the research support of flight test vehicles like: inflight simulators.

1. INTRODUCTION

The Institute for Flight Mechanics of DLR, Braunschweig developed and operates special equipped flight test vehicles which are mainly used as inflight simulators for research purposes (Ref. 1). One is the 'Advanced Technologies Testing Aircraft System' (ATTAS), based on a VFW 614 transport aircraft (Fig. 1).
The other one is called the 'Advanced Technologies Testing Helicopter System' (ATTHeS) and is based on a BO-105 helicopter (Fig. 2).
Both vehicles are equipped with fly-by-wire control, onboard computing capacities, avionics and special sensor systems.

The successful use of these flight test vehicles needs the capabilities of ground based real-time simulators for preflight testing of experimental hard- and software components in order to decrease costs and to increase functionality and safety of flight tests. This includes the training of test pilots to support the correct run of succeeding real flight experiments.

To accomplish these required tasks, the simulations have to process 'real world'-identical data streams with the same quantities, formats and transfer rates as produced within the actual flight test vehicles. Moreover special data processing hardware components as integral parts of the ATTAS and ATTHeS research aircrafts have to be connected with the real-time simulators, including 'Fly by Wire'- and 'Model Following Control'-computer systems for example, as well as fix based cockpits with input/output capabilities for the preflight training of pilots and the control of simulation runs.

Those described common objectives and demands determine the general structures of both simulations:

- The ATTAS ground based transport aircraft real-time simulation, completely installed and used since several years
- The ATTHeS ground based helicopter real-time simulation, installed at present with the exception of some peripheral hardware connections

2. HARDWARE STRUCTURES

Because the principle hardware arrangement of the real-time helicopter simulation is based on experiences with the ATTAS simulation during the last years, both simulator hardware structures look very similar to each other (Fig. 3). An important advantage produced by this concept is the possibility to minimize the number of different hardware components and that decreases the expenditure of hardware maintenance and expensive software development.

The 'host'-computer VAX-750 works as a supervisor and controls both simulations interactively. The total simulation software, consisting of real-time parts running with the AD100 (helicopter simulation) and AD10 (transport aircraft) and non-real-time parts running with the VAX itself is developed with this 'host'- machine. The real-time software is loaded into the program- and data-memories of the simulation computers AD100 and AD10 immediately before the simulation run and the control is given by the interactive simulation software packages, executed at the VAX computer. The normal, not time critical interactive data transfer between the 'host' and the simulation computers is performed by 16-Bit parallel interfaces (HIC, RIC). The same connection can be activated with the helicopter simulation for graphic 'quicklook' outputs during a simulation run, using the standard ADSIM (Ref. 2) graphics package.

The AD100 and the AD10 are multiprocessor systems, special designed to fulfil the continuous real-time simulation tasks of complex dynamic processes (Ref. 3).

Only a short view of the AD100 internal harware structure shall be given, because the operation modes of both machines are comparable, but the capacity of the AD100 as the more modern computer is 4 to 5 times higher than that of the AD10. The AD100 is a 65-Bit floating point machine which executes 20 Mega-Flops per second in a scalar operation mode. The data memory is optional expandable up to 16 Mega Bytes. This computer system consists of several processors, working in parallel and performing different jobs (figure 4). The base system configuration contains:

1. Supervisor (SUP)
   (for interactive communication with the 'host')
2. Arithmetic Logic Unit processor (ALU)
   (65-Bit floating-point arithmetic)
3. Multiplier (MUL)
   (53-Bit multiply-operation)
4. Storage processor (STO)
   (control of 64-Bit high-speed data memory)
5. Communication processor (COM)
   (control of all other processors during runtime)
6. Function Memory Unit (FMU), optional!
   (control of 16 Mbyte CMOS-memory)
7. Communication Link Processor (CLP), optional!
   (provides 4 ports, e.g.: fibre optic, Ethernet)

Two additional slots are kept free for future expansion boards.

Every processor has its own instruction set. Prepared program modules, called 'kernels', which are stored within the ADSIM library (trigonometric functions, square root, signum, coordinate transformations, function interpolations, etc.) are loaded into the different program memories. This 'kernels' are written in AD100-assembler language and take notice of the processors work distribution. The COM processor contains a sequence of 'calls', created by the compiler, dependant on the users program, which are processed one after another. Thereby large programs can be handled without wasting the program memories. All processors are connected by a common bus (PLUS-BUS) (Fig. 4).

In addition to the condition of short frametimes for a simulation run, it is of great importance to have an efficient input, output (I/O) system for fast data transfers between the simulation computers and different hardware in the loop (e.g.. data transfer between AD100, 68020 interface computer and the Model Following Computer System, 'MFCS'). This problem is solved with the connection of a separate I/O control processor (IOCP) to the COM- memory of the AD100. The IOCP works in parallel to the other processors and has its own program memory too. It is linked up to a special I/O rack, which can be supplied with a great number of different interfaces – e.g.. Sense Lines (SL), Control Lines (CL), Analog/Digital Converters (ADCs), Digital/Analog Converters (DACs), Dual Ported Memories (DPMs), etc. – .

The program code for the IOCP has to be written in the special ADRIO programming language and allows most variable data format conversions to provide compatibility with other connected data processing systems.

Both simulation computers (AD100, AD10) are linked up with standard VME-Bus devices based on 68020 CPUs (the AD100 via DPM and the AD10 via Digital to Digital Connection (DDC) (Fig. 3). The true data rate for this configurations is about 1.3 to 2.0 Mbytes/sec.

The VME-Bus computers work like intelligent interfaces between the asyncronous as fast as possible running real-time machines and the peripheral devices and have a typical given frametime of 20 milliseconds. This systems are expandable up to three CPUs, to get smaller frametimes, equal or less than 10 milliseconds.

The application of computers with the wide-spread VME-Bus standard also offers the oportunity to purchase a variety of special interfaces. Expensive and time consuming self- developments are avoidable. The ATTAS simulation uses such interfaces as MILBUS 1553B and ARINC 429 for example. In addition to the input/output-data distribution tasks, the VME-Bus computers have to execute some non-time-critical simulation program parts as. navigation, calibration and data calculation and transfer for the cockpit instrumentations.

3. SIMULATION SOFTWARE STRUCTURES

The ATTAS base aircraft, the VFW 614, is simulated with six degrees of freedom. The accelerations are calculated within the body-fixed coordinate axis system, Euler angles are determined by solving the quaternions differential equations (Fig. 5).

In total thirteen electro-hydraulic actuator control systems, including six separate controlled Direct Lift Control flaps (DLC- flaps) and additional the static and dynamic behaviours of the two jet engines have to be simulated. The aerodynamic coefficients are calculated by linear interpolation with function table look up methods. Also a MIL-standard wind and turbulence model and a standard atmosphere model are included.

Some statistic data may give an impression of the simulation size:

- state variables: 62
- algebraic variables: 413
- generated functions:
  - one variable: 130
  - two variables: 107
  - three variables: 33

All the above listed models are programmed with the MPS10 (Modular Programming System for the AD10) simulation language and are calculated with the AD10.

MPS10 is a special software package running only with an AD10 multiprocessor but it already contains a lot of simulation language elements, which are required by the Continuous System Simulation Language (CSSL) standards:

- modular programming structure
- extended continuous simulation function library
- interactive software package

All MPS10 modules are written in the AD10 macro assembler language in consideration of the special hardware structure of this machine. That accounts for the high computation speed of about 3.5 milliseconds for one frame, including input/output handling.

The dynamics of the helicopter BO105 are calculated by the AD100 exactly in the same way than that of the VFW 614: (Fig. 5)

- 6 DOF
- body fixed coordinate axis system
- quaternions differential equations

The difference between the transport aircraft and the helicopter simulation consists in the additional computation of forces and moments of helicopter subsystems, which exercise an influence on the center of gravity motion of the helicopter.

Subsystem models implemented within the BO-105 real-time simulation are (Fig. 6):

• Mainrotor model

The model describes the computation of the forces and moments at the rotor hub for the four-bladed mainrotor of the BO-105 helicopter with flapping and lagging degrees of freedom and coincident flapping and lagging hinges. Blade flexibility is neglected (Ref. 4). The computation is based on the Blade- Element-Method:

Each of the four blades is subdivided into 10 elements, covering equal annulii aereas during one rotation (Fig. 7). Therewith the number of elements increases at the more active aerodynamic parts of the rotor blade. The aerodynamic coefficients are calculated at each element's reference point as nonlinear functions of the local mach-number and the angle of attack. The total forces and moments at the rotor hub are then determined by the summation of all blade elements' aerodynamic forces and moments plus the blade inertia forces. The model is expanded by trapezoidal downwash effects and a tip loss factor influence.

• Tailrotor model

The tailrotor simulation is based on the tip path plane model for a teetering rotor (Ref. 5). The dynamics of the two blades are neglectable and therewith the forces and moments are computed in a quasi stable state, but the mainrotor downwash is included.

• Fuselage model

The nonlinear aerodynamic is calculated by function table look up methods. Mainrotor downwash effects are considered.

• Empennage model

The aerodynamic forces and moments are computed via determination of aerodynamic coefficients at the horizontal stabilizer by linear interpolation of functions dependent on the local angle of attack and the mach-number. Again downwash effects are considered

The helicopter simulation is completed by the computation of the kinematics and dynamics of control systems for longitudinal control, lateral control, collective pitch mainrotor, collective pitch tailrotor and a dynamic engine model including RPM-governer functions.

All those above described models are implemented within the BO- 105 simulation running in real-time on the AD100. The most complex and computation-time consuming part is the simulation of the mainrotor. For example: The extensive differential equations describing the accelerations of the flapping and lagging angles have to be solved for each of the four blades and the aerodynamic forces must be calculated forty times each frame if ten elements per blade are chosen. More than 75 percent of the actual total helicopter simulation frametime of 1.9 milliseconds is used for the mainrotor calculation. Some more informations about the size of the helicopter simulation are given by the following data, comparable to the VFW-614 simulation:

- state variables: 32
- algebraic variables: 1000
- generated functions:
  - one variable: 215
  - two variables: 87

The short computation time is not only the result of using a special designed fast computer, but also based on the application of the continuous simulation language ADSIM. ADSIM is used to implement and to control the real-time helicopter simulation and contains most of the CSSL standards. Like MPS10 it consists essentially of two parts, the programming language itself and an extended interactive software package. The language supports block structured programming and makes it possible to translate the mathematical description of a simulation model easily. Comprehensive function and model libraries containing typical continuous simulation elements as for example functions describing nonlinearities, modules for multivariable function interpolation and standard models for simulation subsystems are available.

User written FORTRAN code may be included for non-realtime program parts to execute pre-calculations, modify parameters and control the simulation runs. One application of this feature is the control and parameter modification during the iterative procedure of the helicopter's trim calculation.

Many of the ADSIM software elements are written as 'kernels', specially designed for the multiprocessor hardware to provide maximal computation speed.

The ADSIM interactive program package is running on the VAX computer and enables the user or programmer to control the simulation. The graphic online quicklook option is only one available tool. Parameter modification, display of simulation data, speedup changing to run faster or slower than real-time are other useful options. Moreover of great importance are the ADSIM supported twelve integration algorithms which can be changed interactively:

- ADAMS-BASHFORTH 1 to 4
- ADAMS-MOULTON 1 to 4
- RUNGE-KUTTA 2 and 4
- RUNGE-KUTTA-REALTIME 2 and 3

The AD100 standard integration algorithm is AB-2 ! (see: Numerical Integration)

The software modules running on the 68020 VME-Bus computers of the two seperate simulation facilities are identical in essential parts. Handling input/output data via different interfaces, calibration, navigation and optional data recording are the most important tasks of both computers. The standard programming language 'C' and the real-time PDOS operating system are applied system software tools.

Interactive program parts executed with the VAX-750 and the VMS operating system are written in FORTRAN.

4. NUMERICAL INTEGRATION

Every numerical integration algorithm applied to solve nonlinear common differential equations produces more or less accurate approximations of the real solutions. Special restrictions with the choice of suitable integration algorithms used by real-time simulations may cause additional accuracy problems and should be noticed unconditionally to avoid non-acceptable dynamic errors. Most important restriction consists in the correlation between the integration step size and the actual frametime (time, the computer needs for one program-run through the state equations). Normally the frametime equals the integration stepsize, running a simulation in real-time. Therefore only integration methods with fixed stepsizes are usable. On the other hand the dynamic error increases proportional to the increasing stepsize. This leads to serious problems, especial with large real-time simulations and/or stiff systems.

Changing the integration algorithm may sometimes help to solve the problems, but unfortunately that did not produce the wished results in the actual case of the above described VFW-614 simulation for example. Not the integration of the flight dynamic equations but the integration of the differential equations describing the dynamic behaviour of the electro-hydraulic actuators with its higher natural frequency showed non-acceptable results.

The successfully applied method to eliminate this errors is called the Multiple Framing Technique (Fig. 8).

The basic idea of this technique is using an integer multiple of the total frame rate to reduce the integration stepsize for fast subsystems.

with a total frametime h and a multiple frame rate n, the stepsize for a fast subsystem results in $h' = h/n$. Of course, the total frametime increases because of the multiple computation of the fast subsystems, but the actual stepsize for the subsystem becomes substantially smaller than without multiple framing. The VFW-614 simulation for example. with a frame rate $n = 5$ and a total frametime $h = 6.7$ milliseconds, the actuator subsystems can be integrated now with a stepsize of 1.34 milliseconds instead of 3.5 milliseconds as before and therewith satisfactory results are obtained.

There are no difficulties on principle with implementing this method, if the input data for the fast subsystem are generated by interpolation or extrapolation techniques, considering the correct timing. But the implementation becomes evidently more transparent, if using a 'single pass' integration algorithm (results by only one pass through the state equations) instead of a 'multiple pass' method.

The multiple framing technique is most effective, if the fast subsystem's frametime is considerably smaller than that of the slow system. This method is often an useful tool for the implementation of 'stiff systems' real-time simulations as demonstrated with the ATTAS ground based real-time simulation.

Unfortunately not only the correlation between stepsize and frametime has to be noticed if real-time suitable integration algorithms will be selected but also other criterions reduce the number of applicable methods.

First criterion. the algorithm has to be explicit, because implicit algorithms use the solution of the right hand side of the differential equations at a time, when this solution is not yet present in real-time (e.g.: trapezoidal integration (Fig. 9).

Second criterion: the algorithm has to be input compatible.It has to use only known past or present inputs (Ref. 6).

For instance: standard RUNGE-KUTTA 2nd order (Fig. 10) is not well suitable without modification (Fig. 11) or input extrapolation, similar to the input data generation necessary with the multiple framing technique.

An additional information is of interest too: multiple pass algorithms compared to single pass algorithms lose their true higher accuracy under real-time conditions, because they need multiple passes through the state equations to get the desired results and that leads to multiple total stepsize.

Summing up, it looks as if 'ADAMS-BASHFORTH' single pass predictor algorithms are the best suited numerical integration methods for large real-time simulations. But these methods have to be used carefully too, because of startup problems and, more important, numerical stability problems with higher order AB methods in conjunction with increasing stepsizes.

With regard to the above considerations about real-time numerical integration methods, low order AB algorithms for both, the helicopter BO-105 and the wing aircraft VFW 614 simulation are chosen.

5. SIMULATION VERIFICATION AND VALIDATION

The simulation implementations are started with database informations given by wind-tunnel experiments and other aircraft manufacturer specifications, supplemented by more or less reliable data modifications during the simulation installation phase. This first approach to simulate the actual dynamic behaviour of the research aircraft has to be improved and verified to guarantee maximal coincidence, in particular with respect to the ground tests of model following control software quality.

The verification concepts of the helicopter and transport aircraft simulations are identical in principle and are described in (Fig. 12).

Both aircrafts are equipped with onboard data recording devices (magtape, disk), identical or compatibel to ground based devices as parts of the simulators' hardware. Starting with stationary flight conditions, all actuator input commands and sensor output data are recorded during a test flight via the onboard computer system. The actuator input commands may be generated by the test pilot or by test software (predefined input signals, e.g.: 3-2-1- 1 sequences) executed with the onboard computers.

The simulation verification process is started with an equilibrium calculation to get the same or similar stationary flight conditions as the real aircraft. Then the simulation is started and the onboard recorded input data are used as input commands for the simulation run. The simulated sensor output data, corresponding to the aircraft sensors, are stored on a second storage device. The flight test data and the simulation results are then compared and analyzed after the simulation run has finished. This comparison is practicable with several simulation independent running software tools. Parameter and database modifications are then the next step, followed by another simulation run and comparison as explained above. This iterative procedure is finished and the simulation is validated, if predefined quality criterions are accomplished. An extension of the simulation with software modules supporting the automatic control and execution of the verification process is intended. There is no difficulty with imbedding such program parts into ADSIM.

The ADSIM interact routine will also be used for the internal verification to detect hard- and/or software errors, before the simulation is started for user experiments like preflight tests of onboard components. This is performed by the computer supported comparison of formerly generated and validated reference data with actual simulation data. Non-acceptable differences are detected and recorded.

The generation of simulation reference data is performed by starting the simulation with predefined trim conditions for the flight vehicle. Special defined and computer calculated signals (e.g.: 3-2-1-1 signals) are then used successively as input commands for all actuator control systems. All essential simulation data during one run (10 seconds) are recorded with a 20 milliseconds intervall time, creating a suitable database for the internal verification. Recording new reference data is only permitted if simulation models are modified.

The actual simulation data set, produced with the same input data and the same method as described above, is always calculated before using the simulation. The actual data are recorded too, and finally compared with the reference data. Differences are detected and registered. Subsequent hard- and software test programs are then available for the exact error identification.

This internal verification, based on the higher verification and validation process, is a very effective method to check the functionality of simulations including almost all integrated hardware components.

6. CONCLUSION

Though the verification and final validation processes have not been finalized yet, comparisonsr of the real-time simulations with real flight data already indicate sufficient agreement (Fig. 13). This is the result of using well suited real-time models and fast special designed hardware components corresponding with simulation specific software tools like ADSIM and MPS10. The ATTAS ground based real-time simulation is successfully used for the preparations of research experiments with the flight test vehicle since several years. Moreover the experiences with this simulation were especially useful with respect to the installation of the ATTHeS real-time simulation, which will be used too as an indispensable aid supporting the scientific work in the field of helicopter research, including the integration of peripheral hardware components.

7. REFERENCES

1. Hanke D., Bouwer G. 'DFVLR In-flight Simulators ATTAS and ATTHeS for Flying Qualities and Flight Control Research', AGARD-CP-408, FMP-Symposium Flight Simulation, Cambridge, UK, 1985
2. ADI 'ADSIM Reference Manual', Ann Arbor, Mi, USA
3. ADI 'AD100, AD10 Maintenance Manuals', Mi, USA
4. Gruenhagen von, W. 'Modellierung und Simulation von Hubschraubern - Mathematische Beschreibung', DLR Report IB 111 88/06, 1988

5. Talbot P.D., Tinling B.E., Decker W.A., Chen R.T.,
'A Mathematical Model of a Single Main Rotor Helicopter for Piloted Simulation'.
NASA Technical Memorandum 84281, 1982

6. Howe, R.M. 'Dynamics of real-time integration algorithms', 1.st European ADIUS, Braunschweig, Germany 1988

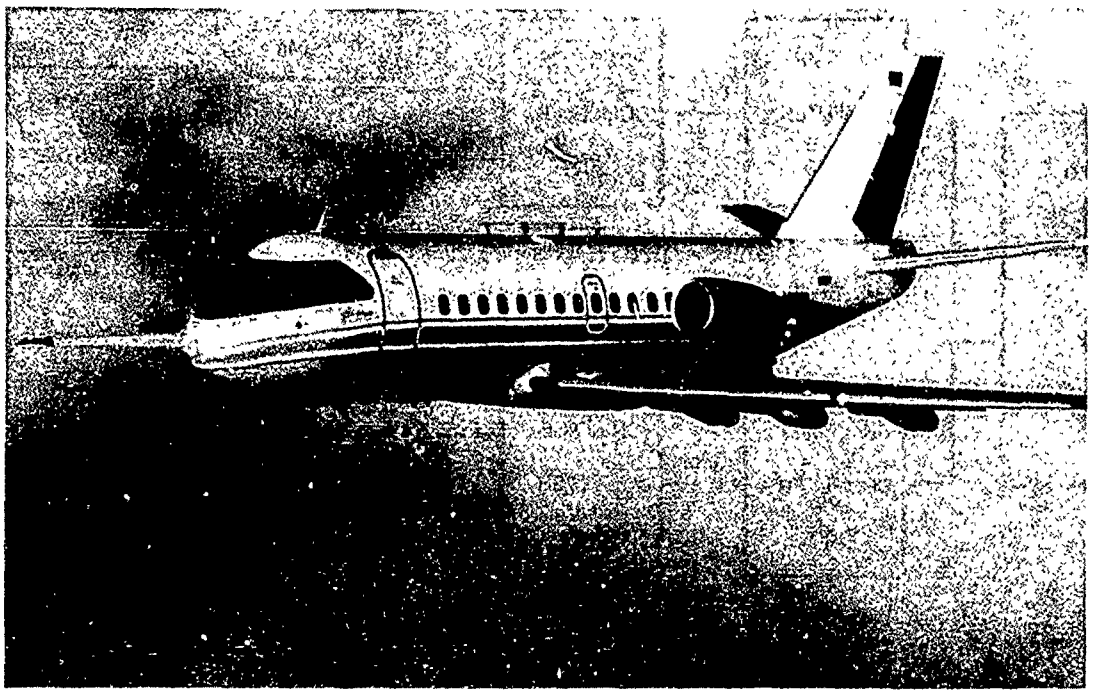


Fig. 1 Advanced Technologies Testing Aircraft System

Fig. 2 Advanced Technologies Testing Helicopter System

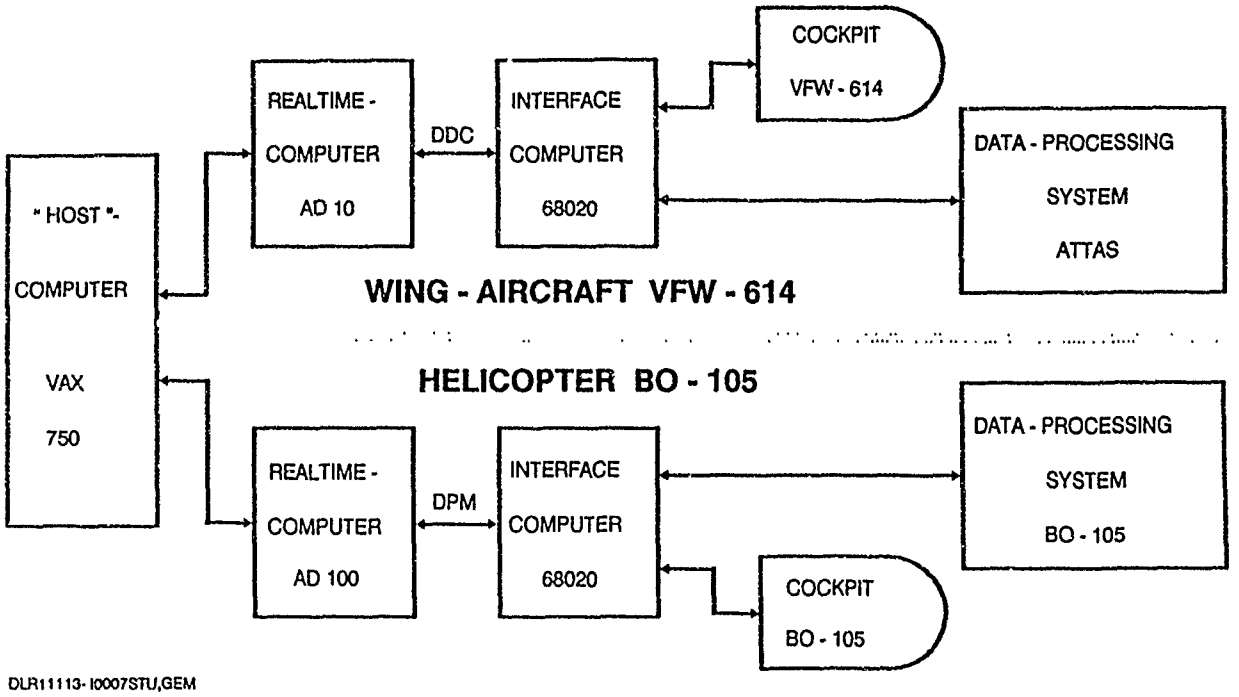


Fig. 3 Real-Time Simulations (Principle Structure)

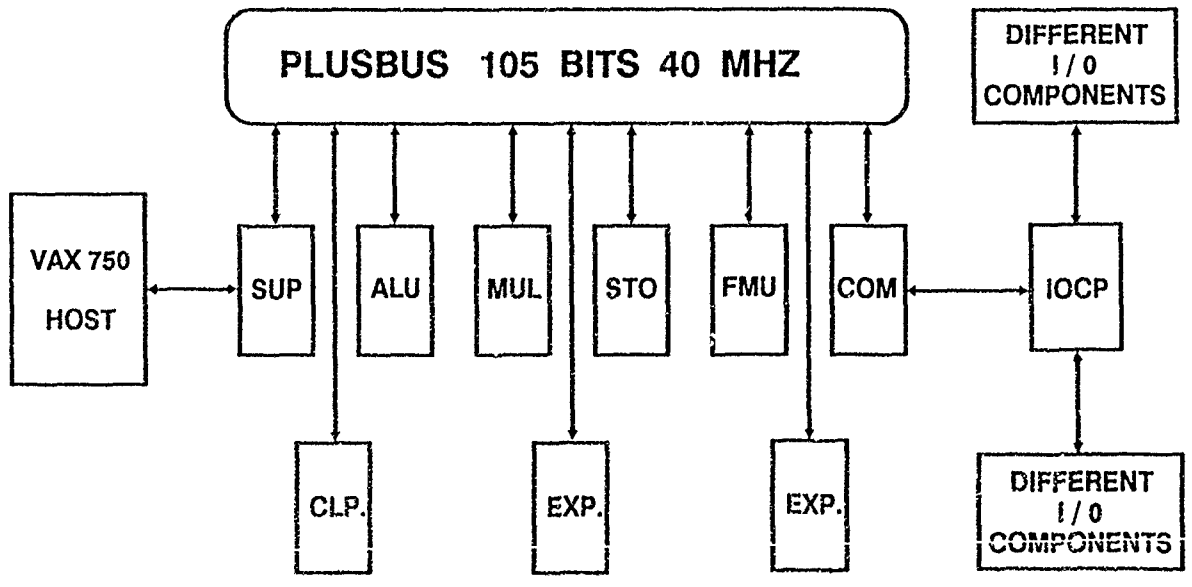
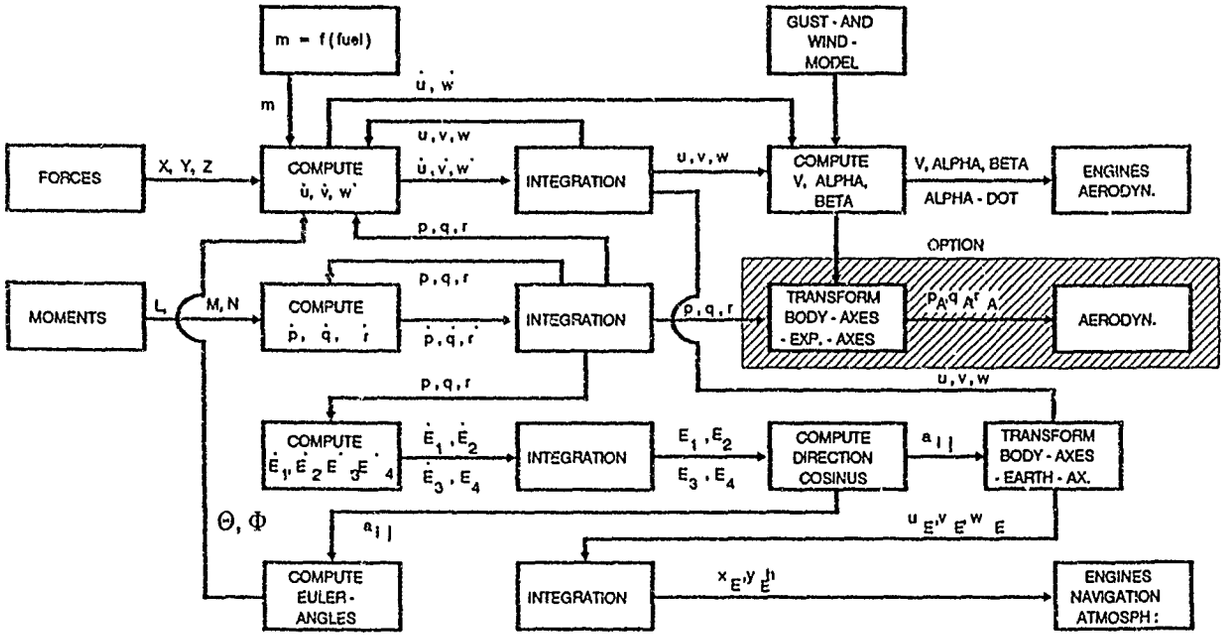
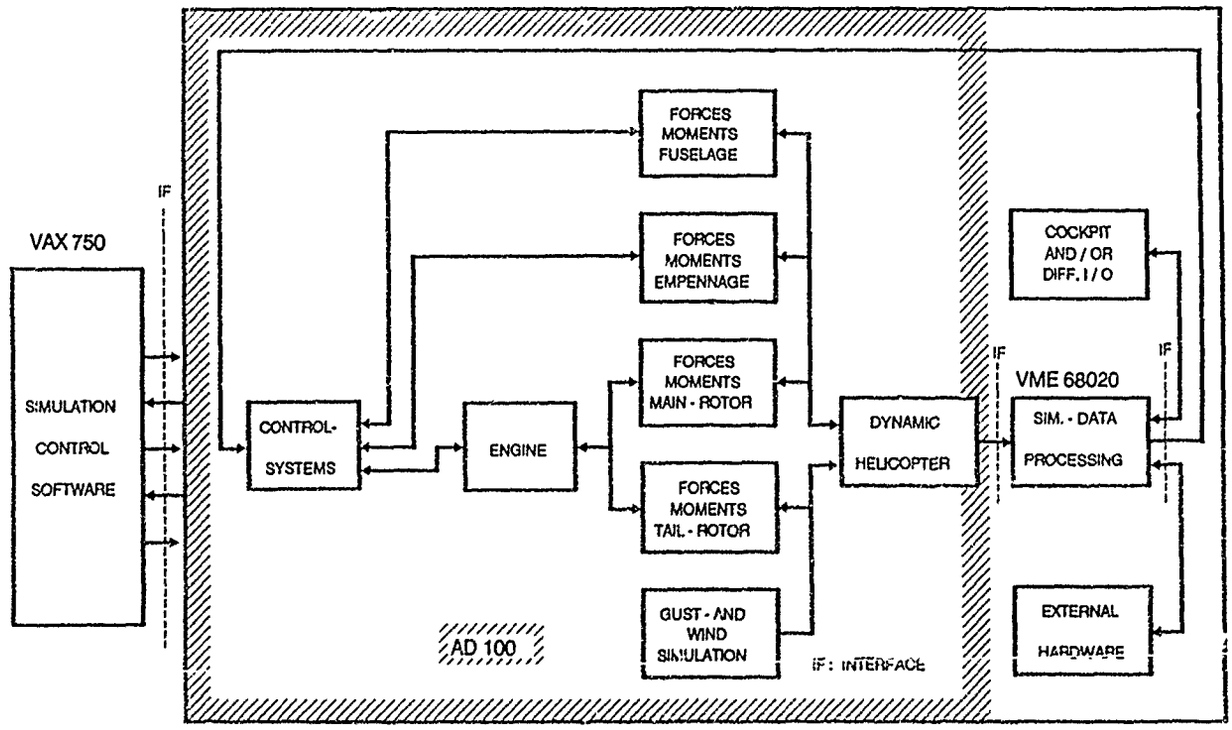


Fig. 4 AD100 Hardware-Structure



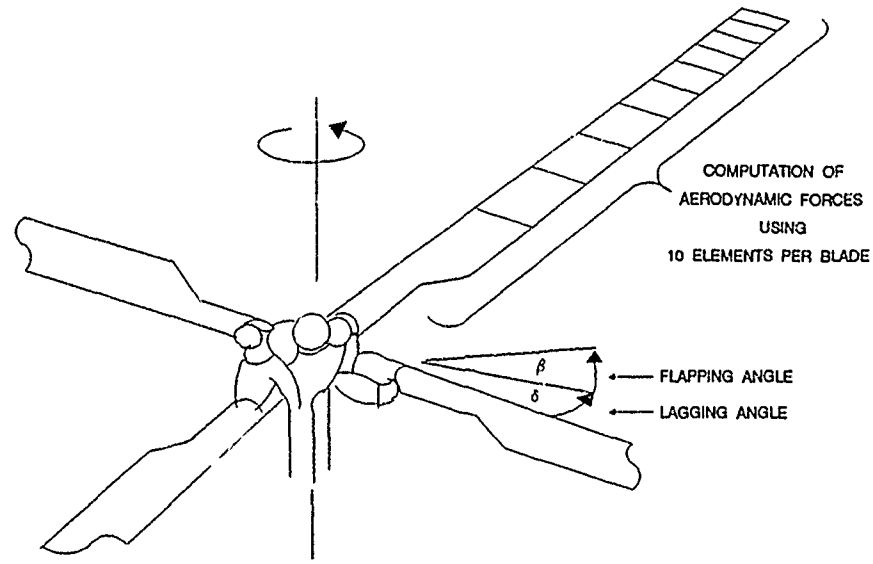
DLR 11113 - I0012STU,GEM

Fig. 5 Flight Mechanical Equations



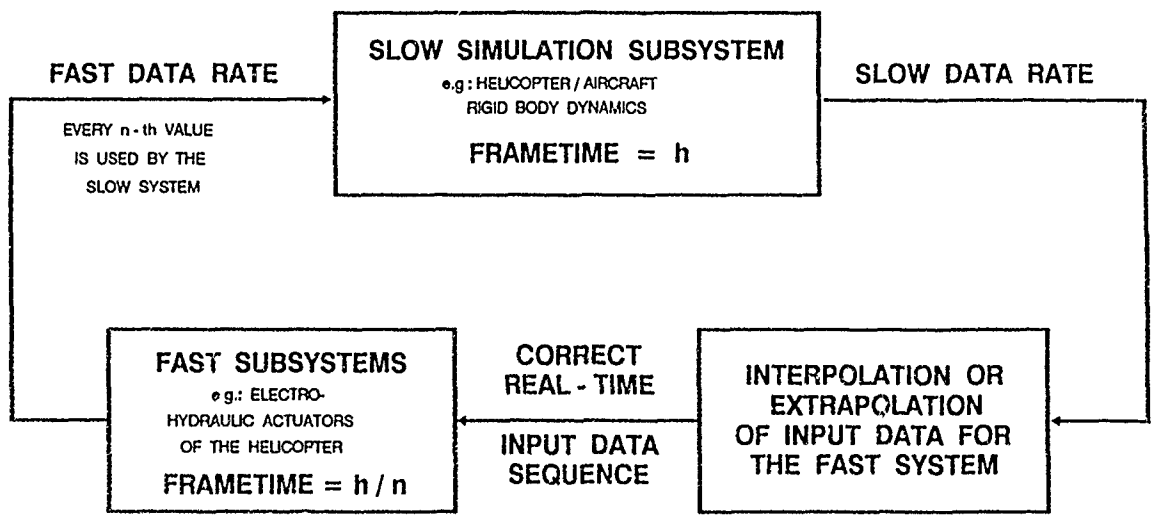
DLR11113-I90009STU,GEM

Fig. 6 Helicopter Simulation (Modular Structure)



DLR11113-I0015STU,GEM

Fig. 7 Mainrotor Blade Segmentation



DLR11113-I0006STU,GEM

Fig. 8 Multiple Framing

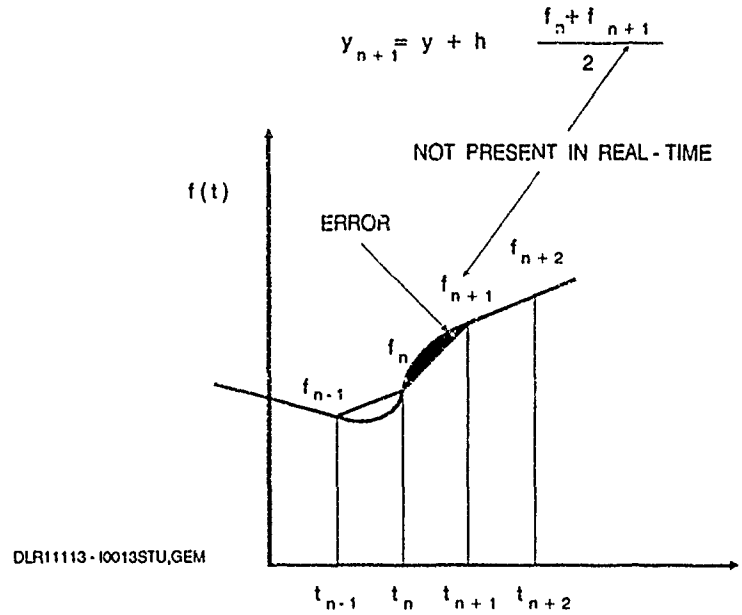


Fig.9 Trapezoidal Integration

DIFFERENTIAL EQUATION :

$$\dot{Y} = F(Y, X(t))$$

FIRST PASS (EULER) :

$$F_n = F(Y_n, X_n)$$

$$Y1_{n+1} = X_n + h F_n$$

(NORM. NOT PRESENT IN REAL - TIME)

SECOND PASS (TRAPEZOIDAL) :

$$F1_{n+1} = F(Y1_{n+1}, X_{n+1})$$

DLR1111-I0014STU,GEM

$$Y_{n+1} = Y_n + h \frac{F_n + F1_{n+1}}{2}$$

Fig. 10 Runge-Kutta 2nd Order Integration

DIFFERENTIAL EQUATION :

$$\dot{Y} = F(Y, X(t))$$

FIRST PASS :

$$F_n = F(Y_n, X)_n$$

$$Y1_{n+\frac{1}{2}} = Y_n + \frac{h}{2} F_n$$

(CORRECT REAL-TIME INPUT)

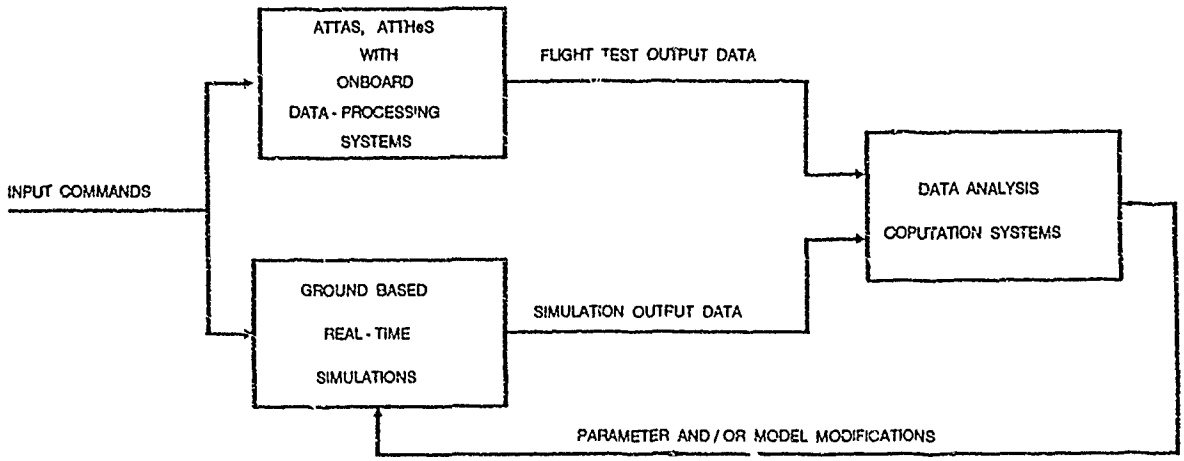
SECOND PASS :

$$F1_{n+\frac{1}{2}} = F(Y1_{n+\frac{1}{2}}, X_{n+\frac{1}{2}})$$

DLR-11113-I0014aSTU,GEM

$$Y_{n+1} = Y_n + hF1_{n+\frac{1}{2}}$$

Fig. 11 Modified Runge-Kutta 2nd Order Real-Time



DLR11113-I0016STU,GEM

Fig. 12 Principle Simulation Verification Concept

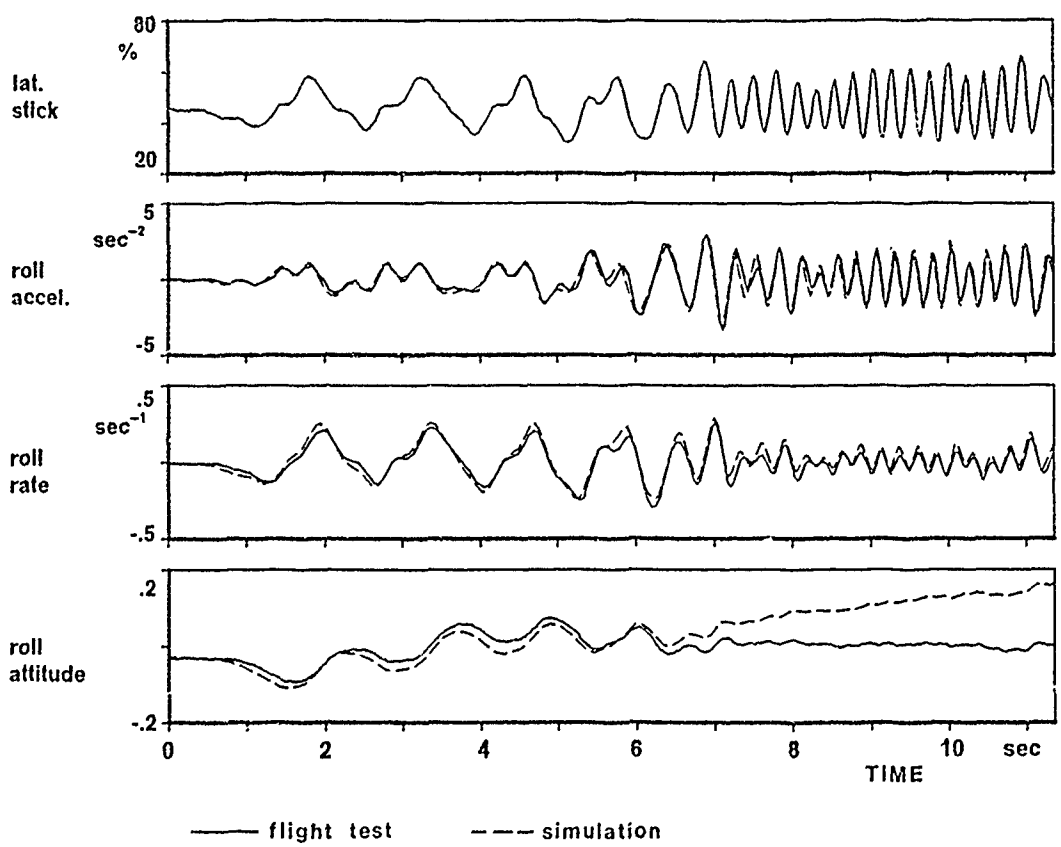


Fig. 13 ATTHeS Comparison of Simulation and Flight Test

HARDWARE-IN-THE-LOOP SIMULATION AT THE NAVAL WEAPONS CENTER

R. A. Licklider, Missile Simulation Branch (Code 3914)
A. B. Galloway, F. Schiavone, RF Missile Systems Branch (Code 3911)
E. J. Bevan, W. Williams, EO/IR Missile Systems Branch (Code 3912)
Naval Weapons Center, China Lake, California, 93555, U.S.A.

Summary

Hardware-in-the-Loop (HWIL) simulation as it is practiced at the Naval Weapons Center is described along with its use in tactical missile development. Computational aspects of HWIL are discussed along with the types of simulations that form system analysis efforts. Target generation techniques in the RF and IR domains are presented with some comments on utility and cost.

A Hardware-in-the-Loop (HWIL) facility is used at the Naval Weapons Center, China Lake, California (NAVWPNCEN) for the system integration and analysis of tactical missile guidance and control systems. The facility allows missile components from the seeker to the autopilot to be exercised, integrated and tested with target signals generated and transmitted across free space. Rotational motion is often impressed on the devices under test to stimulate gyros and examine seeker-body coupling effects. This paper provides an overview of the HWIL process and its utility in missile system analysis.

Missile simulations usually occur in a time-stepped manner, working through the block diagram shown in figure 1.

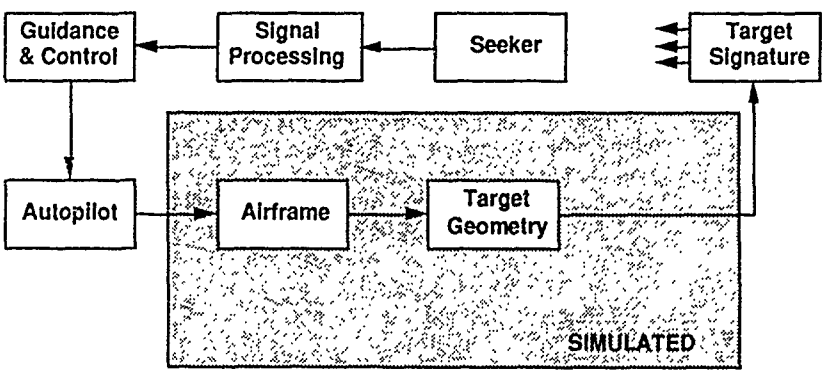


Figure 1

In an all-math simulation, these blocks exist only in the computer. in HWIL, the simulated blocks are contained in the shaded area of figure 1, while the remaining blocks represent actual missile hardware and a target image. A target signature is generated at the appropriate frequency and projected across free space to the seeker. The assembly containing the seeker, its signal processing components, the guidance and control unit and the autopilot is generally mounted in a flight table that responds with the rotational motion generated by the airframe simulation. The target geometry is updated in simulation and the signature modified for succeeding time steps.

The motivation for HWIL simulation stems from two major sources. First, it is the only way to integrate missile guidance components together and exercise them as a system in the laboratory. It provides a means both of understanding the performance of a missile seeker/guidance system and insuring that the system works. Mathematical models of the components may be developed and tested using HWIL. Unfortunately, one only models what one understands. The hardware may exhibit features that have not been accounted for in a mathematical model. HWIL allows the system analyst to detect and understand unexpected hardware performance in the model building process. The second reason for HWIL is a purely economic one. Table 1 shows the number of flight tests required to qualify three Sidewinder (AIM-9) missiles developed at NAVWPNCEN. The first missile, AIM-9D, was developed in an era where mathematical simulations were just beginning to make an impact. AIM-9L underwent extensive mathematical simulation, but only very limited HWIL simulation. AIM-9M underwent extensive mathematical and HWIL simulation. The reduction in test firings of almost 50% from AIM-9L to AIM-9M is attributable in large part to HWIL testing. The use of HWIL led to a deeper understanding of system operation and caught several design and implementation flaws that would have led to flight failures had firing occurred.

	AIM-9D (1960-1964)	AIM-9L (1972-1975)	AIM-9M (1978-1981)
Research & Development	49		
Engineering Test		11	2
Development Test		10	8
Technical Evaluation	32	20	8
Operational Evaluation/ Initial Operational Test and Evaluation	48	28	17
Total	**129**	**69**	**35**

Table 1 - Sidewinder Firings

HWIL is complementary to flight tests, but in no way should replace them. There will always be a requirement to fire a certain number of missiles, but HWIL can leverage those firings by reducing the degree of risk inherent in them. Firings are also extremely useful for validation of mathematical and HWIL simulations, a point that cannot be expressed strongly enough. The increasing complexity of missile systems and combat scenarios, and the repeatability of HWIL simulations work together to provide strong motivation for a HWIL effort in understanding missile performance during and after development.

The computational requirements of HWIL may not seem severe at first, since only a small part of the actual missile is simulated. However, the equations of motion of the missile must be solved in real time. If, for example, the equations governing flight are solved in half real time (one second of flight time corresponds to two seconds of wall clock time) with hardware in the loop, the effective seeker bandwidth is doubled. This is not a realistic test of the hardware.

The NAVWPNCEN Simulation Laboratory (SIMLAB) is built around the concept of a workstation (figure 2). The workstation allows the system analyst to operate a simulation from a single location, with all the computational and interface tools required at that one location. The computational engine of HWIL is an Applied Dynamics AD-100. This 20 Megaflop machine has several positive attributes for real-time HWIL simulation. First, it is fast, with integration and table lookup algorithms built into hardware. Second, the ADSIM programming language used on the AD-100 allows the differential equations of flight motion to be written down explicitly. This speeds the simulation process and makes for code that is more easily dealt with by the analyst. Finally, the input/output (I/O) subsystem contains a rich set of software commands as well as A/D's, D/A's, digital control and sense lines and other devices useful for reading from and writing to hardware. An important feature is the ability to do I/O operations in groups. I/O in HWIL is susceptible to the "time skew" problem. If one is reading from a number of devices in a system and the reads are done sequentially (as is the case with most computers), the data read from the first device will be somewhat older than the data read from the last. With group operations, all of the devices may be read from or written to at one instant. Then, the data itself is collected sequentially from the I/O subsystem and processed.

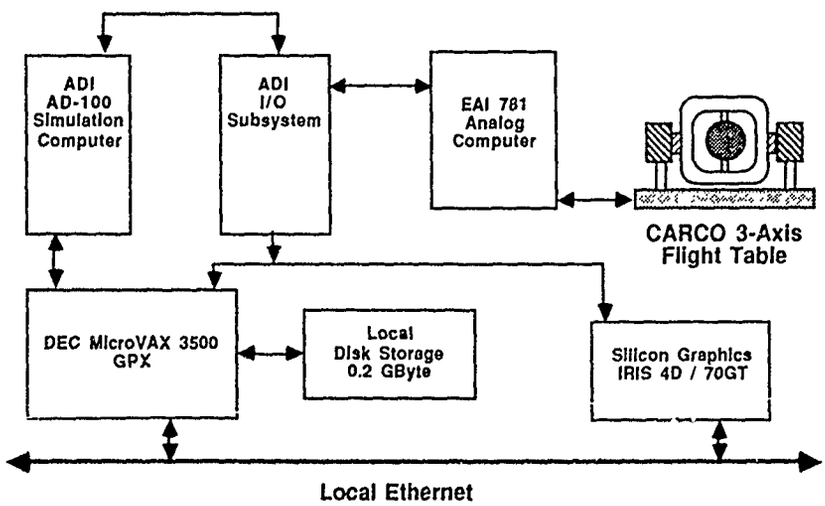


Figure 2

Figure 2 does not show the missile interfaces, nor does it include any of the target generation information flow. In reality, the I/O subsystem and analog computer interact with both the missile system under test and the target generation devices.

An Applied Dynamics AD-10 is sometimes used in place of the AD-100 for HWIL computations This machine is a somewhat older version of the AD-100 and uses fixed point arithmetic computations verses the AD-100's floating point. The programming language of the AD-10 is more difficult to master than the AD-100's ADSIM. As a result, these machines are declining in user popularity. They are, however, quite comparable in speed to the AD-100.

The analog computers used are over 25 years old. They still find extensive application in the SIMLAB, even though most of the computational load has been taken over by the AD-100s. The analog computers drive the flight tables, although we are beginning to procure tables with digital controls. They are also quite useful as signal conditioners. The analog computers can still perform some computations faster than their digital counterparts, particularly in the area of nonlinear actuator modelling. The use of analog computers is more or less one of individual preference in the SIMLAB, since one can always build custom analog circuits to do their job when necessary. While a user base exists, we will continue to maintain the analog computers. The future of these devices is uncertain, considering the availability of digitally controlled flight tables and the ever increasing speed of digital computers. Eventually, we will probably move to a smaller series of analog computers for the one task of analog signal scaling that will probably always be with us.

A Digital Equipment Corporation MicroVAX 3500 is used for data reduction, general purpose computations, and as a host for the AD-100. The Silicon Graphics IRIS 4D/70GT is used to provide a real-time display of information during HWIL simulation. The workstations are linked via ethernet to printers, tape drives and larger mass storage devices. The ethernet link also provides a means of communications between workstations.

Computers employing the UNIX operating system such as the Silicon Graphics IRIS are beginning to make a much more significant impact on mathematical and HWIL simulation. Originally, these machines were procured for use in simulation visualization. Their substantial computing speed make them quite attractive in mathematical simulations, but the unique architecture of the AD-100 is unmatched in its HWIL capability by any existing UNIX machine. Holden[1] details the use of these computers for both simulation visualization and target generation for Tracker-in-the-Loop simulation (to be covered later in this paper) It would appear that UNIX-based workstations will make an increasing impact on HWIL simulation in the future.

A three axis flight table provides a means of impressing rotational degrees of freedom on hardware under test. This rotational motion exercises autopilot gyros and allows one to study seeker-body coupling The hydraulically driven tables are capable of reproducing the rotational motion encountered in that most demanding of scenarios, an air-to-air missile encounter. Typically, an air-to-air table can move a 57 kg cylinder with a load inertia of 28 kg-m^2 200 degrees/sec in pitch and 400 degrees/sec in yaw. The pitch number is lower since that axis must carry the weight of all the other axes. For an approximately symmetric six degree of freedom (6-DOF) simulation, the lower number represents the true rotational capability of the table.

HWIL simulation is really a part of the simulation and system analysis process in missile development Normally, the analysis of a missile system involves the following simulations:

(1) Point Mass (3-DOF)
(2) Digital Reference Simulation (6-DOF)
(3) Digital Seeker Simulation
(4) Tracker-in-the-Loop Simulation
(5) Seeker HWIL Simulation

The first three simulations are purely mathematical. The point mass (3-DOF) simulation utilizes trim aerodynamics but does not include missile body dynamics. It is useful for quick, simple calculations of gross missile performance. Often, this simulation is implemented on a personal computer. The digital reference simulation (6-DOF) and a detailed digital seeker simulation are two of the end products of HWIL, yet their development starts in the earliest stages of system analysis. The 6-DOF is a airframe simulation that models missile body motion in detail. It is useful for defining the edge of the missile performance envelope in a kinematic sense. The seeker and signal processing in this simulation are fairly simple. The digital seeker simulation models the blocks of figure 1 that will be tested in HWIL. The seeker and signal processing in particular are modelled to a very detailed degree. The goal of this simulation is to evaluate system software at a high level and examine counter-measures. None of the first three simulations need run in real time.

The Tracker-in-the-Loop simulation is used when the target presentation requirements are too rigorous for a seeker HWIL simulation. This occurs, for example, when an IR imaging seeker is used. The seeker signal processing must be tested against extended targets (e.g., those that subtend more than a pixel on the imaging plane), backgrounds and countermeasures. Since IR scene projectors are now only in their very expensive infancy (see Holden[1]), another method must be used. Figure 3 shows a block diagram of Tracker-in-the-Loop.

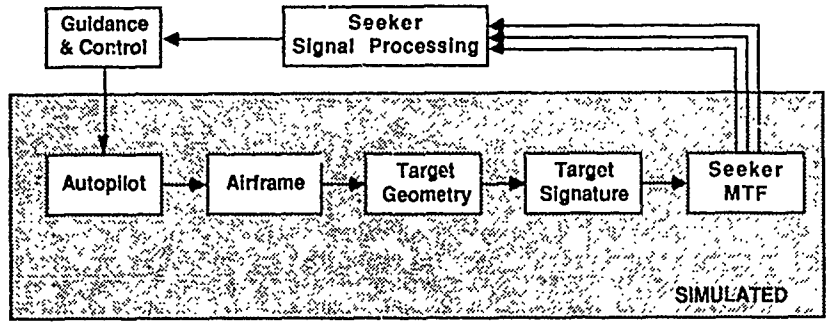


Figure 3

Here, the target signature and seeker modulation transfer function (MTF) are modelled in the computer. What emerges from simulation is a digital video signal that represents the processing that the analog components of the seeker have performed on the target image. One pays a price for this. The seeker is left out of the loop, and rotational flight motion is not impressed on the guidance components. Thus, seeker-body coupling studies must be done under more simple (i.e., point source) target generation conditions.

The value of Tracker-in-the-Loop lies in the ability to exercise the seeker signal processing block. Virtually all imaging systems are software intensive, and the signal processing of the image constitutes a large share of the amount and complexity of processing in the system. The requirement for target signature display is avoided, but the target generation requirement still exists. See Holden[1] for more details.

The seeker HWIL simulation tests the missile components as a system, as in figure 1. Even in seeker HWIL, a subset of simulations is used. This is normally the first opportunity in the development process to move the missile guidance components as a system on a three axis rate table. Normally, a simple pinned stability simulation is used first to determine the angular stability of the missile hardware. The simulations become more complex, finally cumulating in a 6-DOF that reproduces all of the rotational motion that the missile would experience in flight.

Target generation is the last subject that will be considered here, but in many ways it is the most complex. In seeker HWIL, a realistic target in the frequency band of interest must be shown to the seeker. This target should move, since a finite line of sight rate produces the most interesting phenomena in guidance systems. The target should appear to increase in intensity as the simulation moves towards the endgame Backgrounds and countermeasures should be available to tax the signal processing systems. Unfortunately, the ideal target generation system does not exist. Various simplifications are made to balance reality with cost. The first of these is concerned with target motion.

For economic reasons, one is often faced with a fixed target source. The use of phased arrays has ameliorated this problem somewhat in the RF domain, but this is an expensive solution that is not always available In order to present the missile with a line of sight rate so important to the navigational equations, a technique called Synthetic Line of Sight (SLOS) is often used. To understand SLOS, we first define missile line of sight angles in figure 4. The missile line of sight to the target σ and the platform angle ϕ are both measured from a reference, generally the horizon for elevation and a compass direction for azimuth. For simplicity, consider only one component, azimuth or elevation.

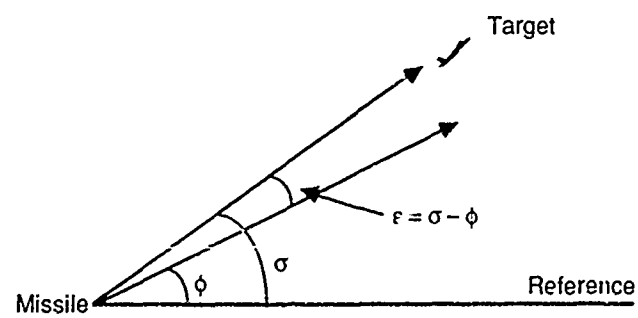


Figure 4

The error angle ε is generally much smaller than ϕ or σ, and is typically on the order of 0.1 degrees.

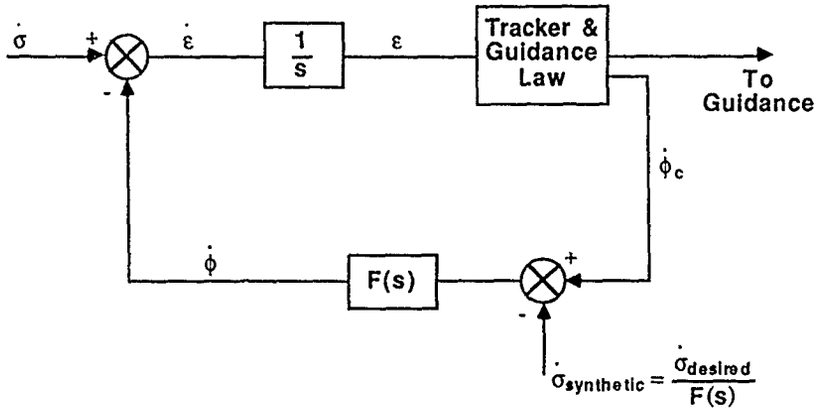


Figure 5

Now, consider the simplified block diagram of a gimbaled missile tracker control system shown in figure 5. The tracker must keep the seeker platform axis (measured by ϕ) aligned with the line-of-sight σ. The integration of

$(\dot{\sigma} - \dot{\phi})$ is accomplished within the seeker hardware to produce a measure of ε. Further processing produces a guidance signal and a seeker command $\dot{\phi}_c$ to close the tracking loop. When it is not practical to present a moving target to the seeker, an electrical signal $\dot{\sigma}_{synthetic}$ is summed into the tracking loop as shown in figure 5, while the seeker continues to track a stationary target ($\dot{\sigma} = 0$).

The guidance signal is used to drive flight table motion and a mathematical model of the control actuator, airframe, and the relative geometry between missile and target. This model also calculates $\dot{\sigma}_{synthetic}$. Disturbance torques caused by mass unbalance or acceleration induced gimbal friction are easily introduced by summing them with $\dot{\sigma}$. The flight table is controlled so that simulated gimbal angles (between the seeker platform and the airframe) are equal to their in-flight values.

The SLOS method may be used to accurately simulate error angles, coupling torques and gimbal excursions without costly target motion equipment. However, SLOS has several disadvantages. It is difficult to apply when there are multiple targets in the field of view, or with missiles using airframe mounted gyros and/or dual mode seekers.

In IR systems (particularly imaging systems), the approximation of a static target source or a rudimentary extended source is often made. Two systems are available in the SIMLAB for testing such systems. The first utilizes a static source that is reflected through a mirror assembly (figure 6).

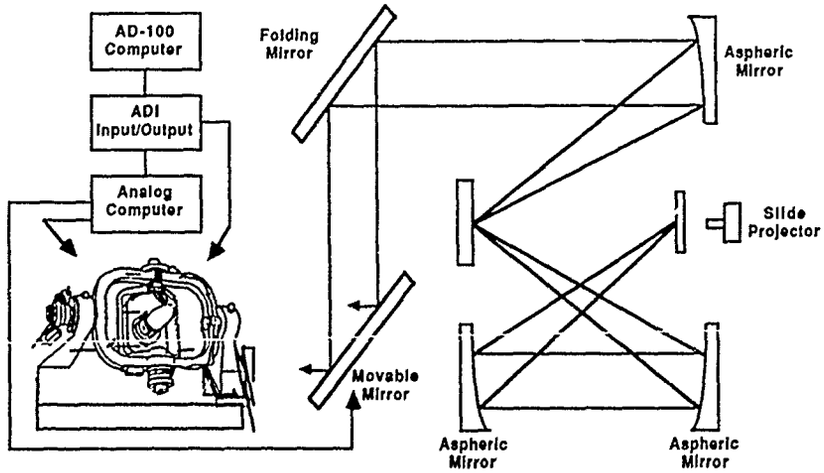


Figure 6

The digitally controlled mirror was recently installed to avoid reliance on SLOS. This provides the capability to move a static image around the field of view of a seeker. The static image does not increase in extent as the simulation progresses, and the system is limited to a single color target.

A more advanced system is shown in figure 7.

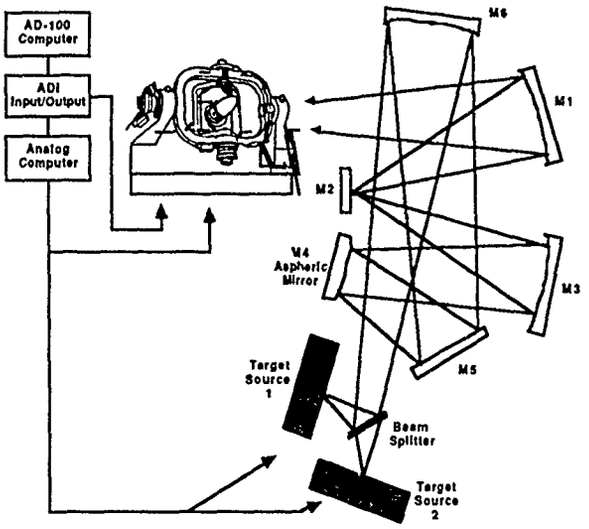


Figure 7

This system is designed for HWIL testing against targets in the far IR spectral region with a slowly varying line of sight rate. The target generation system is unique in two ways. First, a beam splitter is used to allow two IR sources at different temperatures to contribute to the composite target signal. Second, each target source is capable of varying in target extent. The sources are rectangular black bodies that are partially obscured by sliding plates. Figure 8 shows the function of each target source.

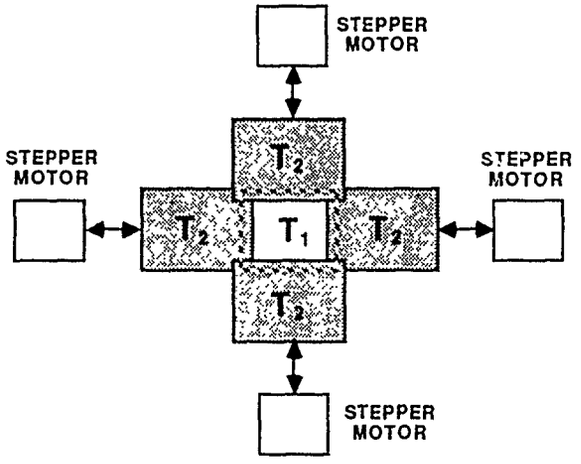


Figure 8

The net effect is a hotter target at T_1 surrounded by a colder background at T_2. The stepper motors provide a small line of sight rate as well as a change in target extent during the course of simulation.

The preceding two examples show simple IR target generators for seeker HWIL either in use or under construction at NAVWPNCEN. An active research program in IR scene generation is underway as well. A number of technologies are under consideration, including heated resistor networks and liquid crystal light valves. As this paper is concerned with the present state of HWIL simulation at NAVWPNCEN, these areas will not be treated. As a final comment, the presentation of a complex target with backgrounds and countermeasures requires both a scene projection system and a computer image generation capability. The latter is a matter of algorithm and code development matched with (rapidly increasing) computer speed. The former is an area of applied physics where considerable challenges remain.

In the RF domain, synthetic line of sight has been used with a single point target for a number of years. Recently, the introduction of multiple targets, countermeasures, and a desire for multi-spectrum simulation has motivated the search for target motion systems. Table 2 shows the relative merits of some systems under consideration.

Point Source	Motion Dimension	Glint Simulation	Complex Target H,V Pos	Wide Angle Target	Cost OK for mm wave RF	Relative Cost 8-18 GHz	Relative Cost 18-40 GHz
Simple Track	1-D	Yes*	Yes*	No	Yes	1	1
Robotic Arm	2-D	Yes*	Yes*	No	Yes	50	50
X-Y Translator	2-D	Yes*	Yes*	No	Yes	50	50
Electronic Steered Array	2-D	Yes	Yes	Yes	No	100	400
Reflector (Theoretical)	2-D	Yes	Yes	Yes	Yes	200	200

* with triad array

Table 2

The simple track system offers the ability to have multiple targets with one dimensional motion, and is now in use at NAVWPNCEN. An electronically steered array (phased array) is under construction, and plans for either a robotic arm or an X-Y translator are in progress. The latter two have an advantage over phased arrays in that they may be used for multi-spectral target presentation. A small triad array allows RF extended target effects to be generated using a target translator. It should be noted that, to avoid correlation between targets, independent target generation devices are required for multiple targets.

An active research program is also underway for RF target generation systems. In particular, reflector systems are being studied as a possible solution to many of the problems inherent in other designs.

This paper has presented a brief overview of Hardware-in-the-Loop as it is practiced at NAVWPNCEN. A few caveats are in order in conclusion. First, the laboratory is dedicated to tactical missile HWIL simulation. This drives the computational, interface and target generation efforts. Second, system analysis is the product of HWIL simulation. Our goal is to understand missile hardware performance at the system level, supporting weapon designers in the research and development process. The laboratory tends to be quite generic, with multiple projects sharing resources. Finally, the laboratory is continually changing, with active research in simulation technology, computer systems, image processing, IR scene and RF target generation. This research is crucial to maintaining a state of the art facility.

References

[1] Holden, B.J., "Computer Graphics in Hardware-in-the-Loop Missile Simulation", NATO-AGARD Guidance and Control Panel 50th Symposium, Izmir/Cesme, Turkey, May, 1990.

Acknowledgements

The lead author profited considerably from discussions with Robert L. Pheysey of the EO Missile Systems Branch, Naval Weapons Center on the topic of synthetic line of sight.

SIMULATION OF MULTIPATH FOR SEMIACTIVE MISSILES

by
Richard M. Smith
Joe Y. Yee
Chong S. An
Allen L. Haun
Naval Weapons Center
China Lake, California, U.S.A.

1. SUMMARY

High fidelity modeling of the multipath environment is required to properly test the performance of software tracking algorithms in a semiactive missile. Examples of multipath models are presented. Simulation data is compared to flight test data and the underlying signal mechanisms are explained.

2. INTRODUCTION

This paper describes the development and testing of sea surface multipath models. These models represent the various forward reflective signal paths encountered in a typical missile engagement. They are used to create the simulated environment necessary to evaluate tracking algorithms in the guidance computer of a semiactive missile. The modeling effort relied heavily on flight test data for insight and verification. As a consequence, simulated test data corresponds well with flight test data for test flights at low target altitudes.

The multipath models are used in an all-digital five-degree-of-freedom engagement simulation. The simulation contains extensive radio frequency environment and signal processing models and forms a closed loop around the missile's embedded computer. These simulations are supported by the computer-in-the-loop (CIL) facility.

The CIL facility complements the six-degree-of-freedom (6-DOF) and hardware-in-the-loop simulations at companion facilities. The CIL facility provides more missile signal processor and environmental detail than the typical 6-DOF simulation. It can provide environmental detail difficult to implement in hardware-in-the-loop simulations and has the potential to evaluate a system design before the hardware is available. It also makes use of the actual missile software to check out missile system performance. Because of its all-digital nature, the simulations are repeatable and serve as a valuable tool in evaluating or developing the embedded computer software. These features come at a cost of longer simulation run times compared to the two other types of simulations.

The challenge is to develop good practical models while balancing modeling difficulty, modeling fidelity, and the need for short simulation run times.

There are three pathways for multipath forward scattering. (1) the transmit paths from illuminator to missile rear receiver, (2) the transmit paths from illuminator to target, and (3) the receive paths from target to missile front receiver. Within each of these three pathways, there are three types of forward scatter multipath components: direct path, specular path, and diffuse path (see Figures 1a and 1b).

The modeling treatment for the direct and specular signal components are straightforward and well established. The modeling treatment for the diffuse signal component is much more difficult, and practical treatments for simulation are still evolving. Because of modeling difficulties, the diffuse components are often neglected or implemented with very simple models.

Regardless of the modeling difficulty, the diffuse signal component must be given a careful treatment. The diffuse signal, not the specular, is the principal contributor to multipath generated phase and angle noise that leads to missile tracking performance degradation. The reflective surface region supporting the generation of the diffuse signal generates amplitude, phase and angle noise. Our modeling approach is to subdivide the diffuse region into many surface slices, sum the individual power returns, convert to voltage, and calculate phase. This has to be done for the various reflective paths between the illuminator and the missile's rear and front receivers. These models are calculation-intensive and time consuming and place a computational burden on the simulation's host computer. Practical limits on computing capability impose a compromise between modeling fidelity and computer run time. Because the simulation is all digital, it can run non-real time. This allows some trade-off between fidelity and length of simulation run time.

The embedded computer utilizes measurements derived from the signal strength, Doppler signals, and angle signals. These measurements include various signal statistics. Algorithms compare these signals in association logic and in track logic. The embedded computer reconfigures the front receiver to optimize signal processing, enabling it to better measure its environment and track its target. High fidelity simulation modeling of the radio frequency environment and signal processing is necessary to provide realistic representative signals. These signals are required to exercise the many software signal processing paths necessary for testing the embedded computer software effects on missile performance.

Our model implementations represent a reasonable compromise between the conflicting requirements of high fidelity modeling and reasonably short simulation run times.

"Approved for public release; distribution is unlimited."

3. REAR MULTIPATH EFFECTS

Our early developmental effort was focused on rear multipath rather than front multipath. The simpler geometries of rear multipath allowed multipath effects to be better isolated and provided a more direct comparison between simulation and test results. The rear multipath models were later used to develop the front multipath models by adding angle effects.

The rear multipath links are between the illuminator and the missile rear receiver. Because the illuminator antenna is tracking the target, the missile, after launch, can move into antenna nulls. This greatly reduces the direct path signal to the missile rear receiver. This makes the direct signal strength sensitive to engagement geometry. The specular and diffuse rear multipath reflection points also lie in the antenna's sidelobes and main beam. However, the specular and diffuse path illuminator to surface to missile signal strength changes slowly compared to that of the direct path illuminator to missile signal strength. This can result in the specular and diffuse signals providing a slowly changing level forming a floor under the direct signal. When the direct signal drops to near or below the specular and diffuse signal levels, relatively large phase and amplitude noise disturbances are generated in the missile rear receiver. This phase noise is propagated into the front receiver where it reduces track quality, distorts apparent target Doppler information, and affects the track algorithms. The amplitude noise is not propagated into the front receiver because of the amplitude leveling action of the rear receiver automatic gain control (AGC).

Flight test data for surface-to-air shots indicating rear multipath effects are presented in Figure 3. The curves indicate that rear multipath amplitude and phase noise increase as target altitude decreases because more of the surface is strongly illuminated by the main beam and the strong sidelobes near the main beam and because the geometry supports a larger diffuse area. The surface illumination increases the diffuse multipath signal component. This diffuse phase noise is propagated into the front receiver where it degrades Doppler measurements and track performance.

As shown in Figure 3, the rear AGC signal represents the amplitude of the received signal in the rear receiver. A higher value represents a stronger signal. The front track received detector gives an indication of the phase noise in the front receiver. A lower signal with high frequency components indicates a signal with poor coherency due to phase noise The front track receiver detector signal periodically drops to zero due to reset logic in the signal processor and is independent of phase noise from multipath effects.

Figure 3 compares amplitude and phase signal test data for surface-to-air engagements against a 3.05-km (10-kft), 1.5-km (5-kft) and 18.3-meter (60-feet) altitude targets. At 11 seconds in the 3.05 km altitude engagement the direct path between illuminator and missile rear receiver enters a strong illuminator antenna null. This causes a decrease in the direct signal and uncovers the diffuse signal floor mentioned earlier. This results in the observed increase in noise. At the lower target altitudes the relative strength of the diffuse signal increases causing a noisier structure in the amplitude signal. There is also a corresponding increase in phase noise propagated into the front receiver. In the 18.3-meter target altitude engagement, the front receiver phase noise is due to both front and rear multipath. However, the rear multipath noise is the major contributor to front receiver phase noise until the last few seconds of flight.

Although it is difficult to see in the figures, there is specular interference lobing on the rear AGC signal at the end of the flight. This occurs because the missile reenters the illuminator main beam, which strengthens the direct signal, and the low missile altitude geometry supports the generation of significant specular returns. Together, these two signals interfere with one another to form the lobing structure.

This test data illustrates that rear multipath effects can be the dominant factor in generating front receiver phase noise and must be carefully treated in any multipath modeling effort designed to test tracking algorithms.

4. REAR MULTIPATH MODEL

Figure 1 presents the direct, specular, and diffuse paths that form the multipath between illuminator and missile rear receiver. Signals in the models are represented as complex vectors containing phase and amplitude. The small platelet theory and diffuse reflection concepts described in Reference 1 are used as the theoretical basis for calculating the diffuse area, diffuse cross-section, and specular scattering coefficient. Reference 2 provides additional insight on relating multipath effects to radar systems.

Figures 4, 5, and 6 present the direct, specular, and diffuse mathematical models. After these rear multipath models were developed and tested against flight data, they were applied to the front multipath transmit and receive paths.

The expression for the direct path illuminator to missile voltage (Eim) is given in Figure 4. The terms generating the amplitude signal component are those typically encountered in the radar range equation. The phase components of the signal are contained in the exponential terms. The exponential term containing Rim represents phase shift due to propagation delay. The term containing theta-Aim accounts for the 180-degree phase shifts between lobes of the antenna voltage pattern.

The expression for the specular path illuminator to surface specular image point to missile voltage (Eism) is given in Figure 5. The expression for the amplitude component of the specular signal is essentially the standard treatment found in most texts on forward scattering. The one exception is the treatment of the smooth planar surface reflectivity (Ro). Ro is limited to a value of 0.3 to account for the antenna smearing the theoretical Ro curve in angle. The phase term includes both the phase of the smooth sea reflection coefficient and propagation phase delay. In addition, the phase of the illuminator to specular point antenna phase shift is included.

The expression for the diffuse path illuminator to diffuse area strip to missile voltage (Eidm) is given in Figure 6. The general approach in generating diffuse voltages is to first calculate the diffuse power return for small strips across the diffuse area and then sum the powers and convert to a voltage amplitude. The voltage noise characteristics are obtained by generating real and imaginary Gaussian samples filtered with a bandwidth determined by the Doppler spread across the diffuse area. The propagation delay to the center of the diffuse area is added to the noise vector phase. Currently, the area is being divided into 10 strips.

In calculating the diffuse power (Pidm), an incident (S1) and reflected (S2) shadowing function is used for each area strip. These shadowing functions are calculated with respect to the horizontal plane of the general diffuse area and not the tipped platelets. The shadowing function is given in Reference 3.

The diffuse scattering coefficient model is a hybrid treatment combining components of two models from Reference 1. The weighting across the diffuse area is appropriate for the Gaussian distribution of surface heights, but the boundary of the area is based on a uniform distribution and is truncated early. The intent is to expand the diffuse area borders appropriate for Gaussian distribution when more host computing power becomes available. As it now stands, this model yields less diffuse power than either of the reference models. The scattering coefficient presented is appropriate for rough surfaces. For extremely smooth surfaces (such as less than sea state 1), a smooth surface model is combined with the rough surface model using an appropriate transition.

The theta-Aid phase term in the expression for Eidm is due to 180-degree phase shifts between antenna lobes (refer to Figure 6). The theta-d phase term is generated from bandpass-filtered Gaussian noise. The bandwidth is determined by the Doppler spread across the diffuse patches.

5. REAR MULTIPATH MODEL VALIDATION

Figure 7 presents simulated data for the 3.05-km target altitude engagement. It illustrates the relative angular position of the missile to the antenna during flight as well as the simulated direct, diffuse, and specular signal components. Note the strong sidelobe peak between the deep nulls at 11 seconds and 13.8 seconds. The component signals indicate the diffuse noise floor and the relative strength of the direct path signal. It is easy to visualize the dips in the direct signal revealing the diffuse component. This signal structure is used in validating our models against flight test telemetry signal records. The geometry of this engagement did not support significant specular returns.

Figure 8a compares simulated results against the test data presented previously. Note than in the simulation, the direct signal decreases and the noise increases when the missile enters the antenna's deep sidelobe nulls at 11 and 13.5 seconds. When the missile enters the sidelobe peak between the nulls, the direct signal increases and the noise decreases. The signal structure between the simulated and test data is similar although the times do not match exactly. The times that the missile passes through illuminator peaks and nulls depend on individual missile gyro drift rates, accelerometer bias, thrust profile, and antenna pattern variation. The simulation explains and reproduces the interaction between the direct, diffuse, and specular signals observed in flight test data.

Figure 9 presents simulation data for the 18.3-meter target altitude engagement. It indicates the simulated missile angle position relative to the antenna pattern and the direct, specular, and diffuse path signals. Figure 8b indicates their sum as seen in the rear AGC plus the front track receiver detector. In this engagement against a 18.3-meter altitude target, the amplitude of the diffuse component is stronger compared to the direct path signal than that of the 3.28 km altitude target. This results in a relatively noisy AGC signal. During the last few seconds of flight, the phase noise is reduced as the missile enters the illuminator antenna main beam, and the direct signal increases. At the end of flight, the specular component increases, and interference lobing is observed on the AGC signal.

Figure 8b compares flight test data with simulated data for the 18.3-meter altitude target engagement. The flight test data and simulated data for the rear AGC signal have similar noise level and signal characteristics. The phase noise is reduced during the first few seconds of flight when the missile passes through the antenna main beam and strong sidelobes, which increases the direct signal strength while the diffuse signal is relatively weak. The phase noise increases as the diffuse signal increases and the missile moves into weak sidelobes, reducing the direct path signal. The phase noise is reduced at about 22 seconds as the missile enters the antenna main beam.

During the last few seconds of flight when the missile enters the illuminator antenna's main beam, the direct signal strength is increased and the effect of the diffuse noise is reduced. At the end of the flight there is a small amount of interference lobing on the rear AGC signal due to a strong specular component.

6. FRONT MULTIPATH MODEL

The treatment in modeling front multipath is different from that for rear multipath for the transmit path from illuminator to surface to target and for the return path from target to surface to missile (see Figure 2). The transmit path has no angle information and is treated similarly to rear multipath. Unlike the rear multipath, however, the transmit path signal model must also include modulation by the fluctuating target before reception by the missile via direct or reflected path. The transmit path diffuse area is currently divided into 10 strips for calculating diffuse voltage.

The return path has angle information. Ideally, the return diffuse area would be divided into many small strips, each at a separate angle. These individual strips would be processed and added by the monopulse receiver. Currently, however, the return diffuse area is divided into 20 strips. These 20 strips are partitioned into two groups of 10. The center of power for each 10-strip area is treated as a point target for angle calculations.

Figure 10 presents front multipath transmit path direct, specular, and diffuse component signal powers and their sum for the surface-to-air 18.3-meter target altitude engagement. The sum transmit signal has a shallow null at mid-flight due to interference between the direct and specular signal components. The high frequency noise riding on the top of the sum signal (front AGC) is partially due to computer calculation noise that has yet to be corrected. Any transmit signal noise is seen as noise riding on top of the front receiver AGC signal. This effect is more apparent if the diffuse signal is weak and the AGC is high, indicating that the return signal is strong and is little affected by system thermal noise or diffuse noise.

Figure 11 presents the front multipath receive path direct, specular, and diffuse signal powers without the application of target fluctuation modulation. Target fluctuation effects are included in the simulated front AGC signal of Figure 12. Notice that the receive path specular return shown in Figure 11 is significant only during the first few seconds of flight when the engagement geometry supports strong specular returns because of the low grazing angles. The "image-like" returns that cause so much trouble in low angle tracking can be derived through diffuse theory and appear as narrow band diffuse signals. The classical specular model treatment will not provide strong specular signals at the end of flight because the typical flight geometry does not produce small grazing angles in the return path. A lot of misguided effort has been devoted to massaging the specular model in attempts to produce the "image-like" returns.

7. FRONT MULTIPATH MODEL VALIDATION

Figure 12 compares flight test and simulated data for the engagement against the 18.3-meter target. The signals presented are the front receiver AGC, pitch boresight error, narrow band Doppler error, and wide band Doppler error. The front AGC measures front receiver signal level. It indicates target fluctuation on the direct return signal, and signal interference between the target signal, front multipath, and receiver noise. The boresight error signal indicates target angle relative to the missile antenna boresight angle. The narrow and wide band Doppler error signals are front track receiver outputs that indicate target Doppler tracking error.

At the beginning of flight, the low altitude missile geometry produces low grazing angles that support the generation of strong receive path front multipath specular signals. This produces low frequency interference signals in the front AGC for the first few seconds of flight. As the missile increases altitude, the specular component of the front multipath rapidly diminishes.

During the next portion of flight, the relative long range target provides a weak signal and thermal noise is high relative to target and front multipath signals. This results in noisy front AGC, boresight errors, and Doppler error signals.

The range from missile to target gradually decreases as the flight proceeds and front multipath signals increase relative to thermal noise. The influence of thermal noise decreases. The diffuse noise floor builds up under the direct path target to missile signal. When the fluctuating direct target signal dips and uncovers diffuse noise, the front AGC, pitch boresight error, and Doppler error signals become noisy. The eye can easily correlate dips in the AGC with increases in noise for the other signals. This is also the cause of some of the front receiver phase noise seen at the end of flight illustrated in Figure 8b.

When boresight error increases because the direct signal dips, the pitch boresight error usually develops some downward bias. During the early part of flight, the return signal is weak and receiver thermal noise generates significant boresight error noise. However, this noise is not biased and causes less of a guidance problem.

The simulated Doppler error is noisier than that of the test data. The simulated Doppler error noise is sensitive to the surface roughness model and will provide a better match to test data if a slightly smoother surface model is chosen. When the missile enters the illuminator main beam near the end of flight, the direct receive path signal increases relative to the diffuse signal and the Doppler error signals become less noisy. The Doppler error signal is strongly influenced by the diffuse signal component when the direct signal component decreases, indicated by a dip in the front AGC signal. During these dips, the Doppler error is biased toward a lower or higher Doppler error as the doppler tracker shifts toward the Doppler frequency of the diffuse noise signal. The Doppler error amplitude and bias is dependent on the relative missile to diffuse area velocity.

8. CONCLUSIONS

The flight test and simulated data have the same basic structure. This structure and the individual signal stream will vary somewhat from test flight to test flight and from simulation run to simulation run.

In general, our simulation yields results similar to that observed in flight test data. In addition, the modeling explains the mechanisms resulting in the observed signal characteristics.

The following are some conclusions derived from this work:

(1) Rear multipath must be modeled to adequately simulate multipath effects for some semiactive missiles.

(2) An effective compromise must be made between obtaining the necessary diffuse modeling fidelity and fast simulation run times.

(3) Diffuse reflections can generate significant front receiver phase, amplitude, and boresight error noise.

(4) Diffuse reflections can be more significant than specular reflections for low altitude engagements.

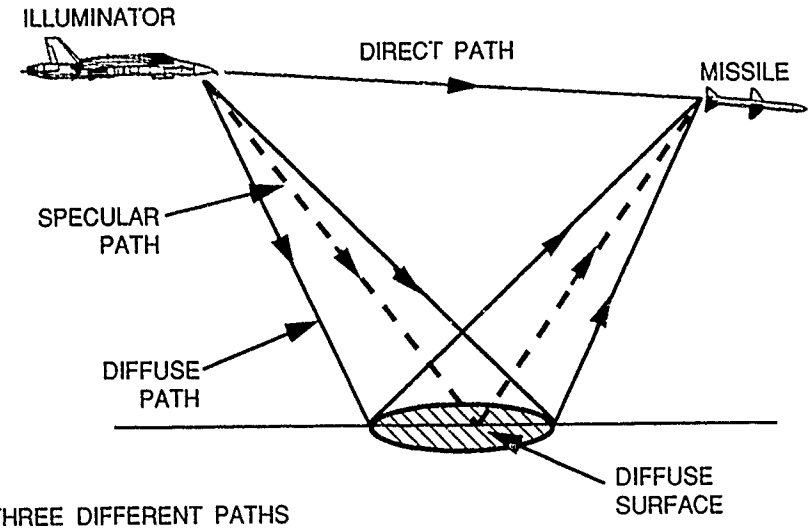
(5) Classical specular model theory does not lead to proper simulation treatment of the "image problem" seen in flight test data whereas diffuse theory does.

REFERENCES

1. P. Beckmann and A. Spizzichino. "The Scattering of EM Waves from Rough Seas," New York: McGraw-Hill, 1970.

2. D. K. Barton. "Low-Angle Radar Tracking," Proc. IEEE, Vol. 62, No. 6, pp. 687-704, June 1974.

3. B. G. Smith. "Geometrical Shadowing of a Random Rough Surface," IEEE Trans. Antennas and Propagation, Vol. AP-15, pp. 668-671, September 1967.



- THREE DIFFERENT PATHS
  - DIRECT PATH ——————
  - SPECULAR PATH — — —
  - DIFFUSE PATH ——————
- SIGNALS REPRESENTED AS COMPLEX VECTORS
- SMALL PLATELET THEORY USED TO DETERMINE DIFFUSE SURFACE AREA (REF. 1)
- SPECULAR SCATTERING COEFFICIENT IS FROM SAME SOURCE
- REAL AND IMAGINARY COMPONENTS OF DIFFUSE NOISE SIGNAL ARE GAUSSIAN WITH ENGAGEMENT GEOMETRY DEPENDENT BANDWIDTH

FIGURE 1. Rear Multipath.

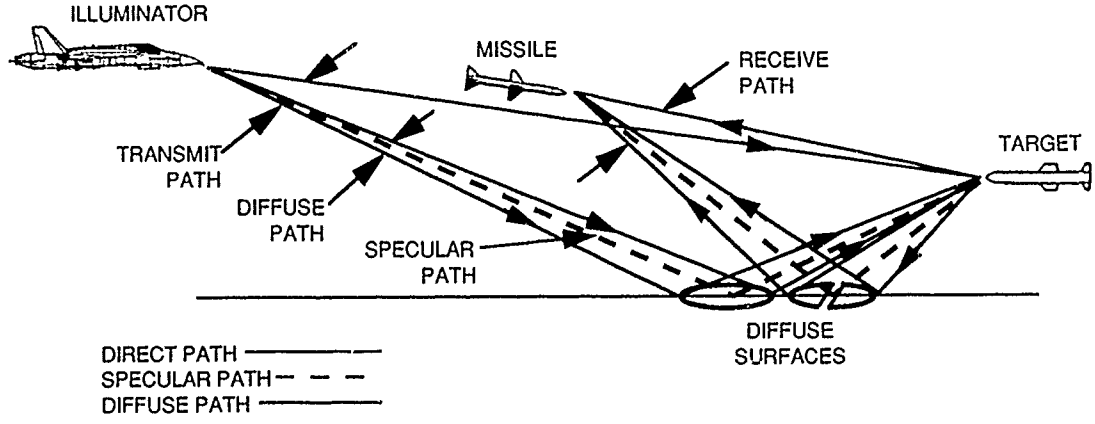


FIGURE 2. Front Multipath.

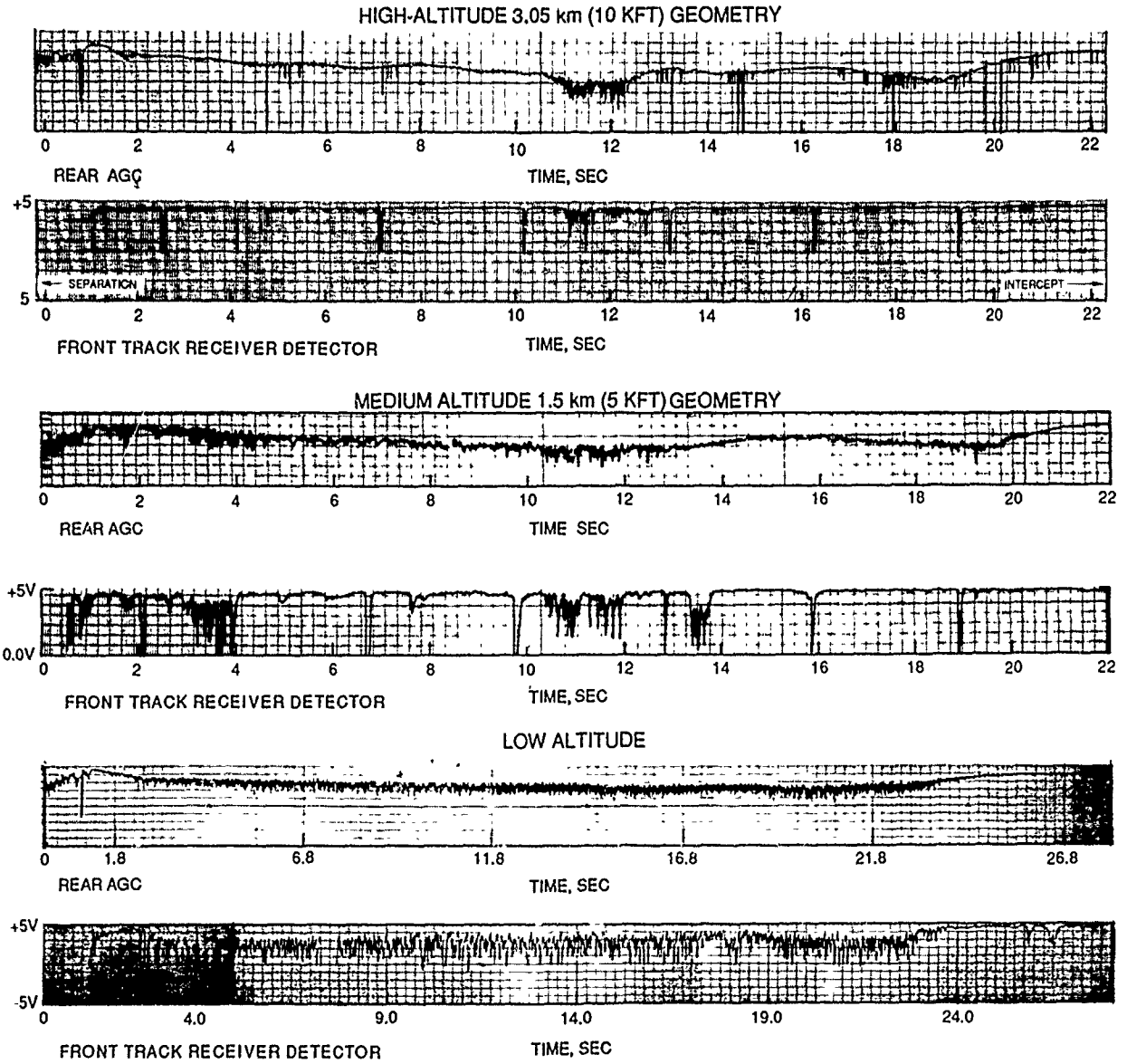
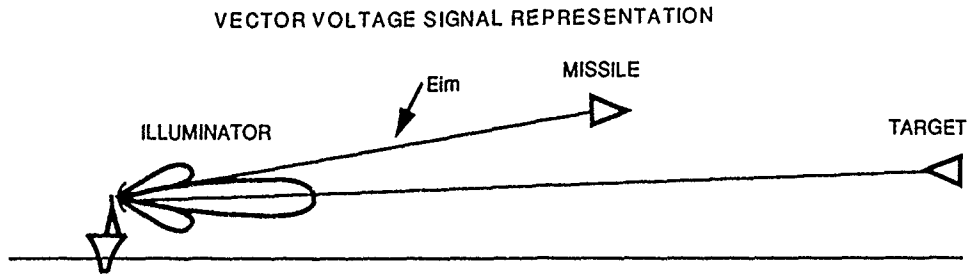


FIGURE 3. Effects of Rear Multipath Disturbs Front Receiver Signals.



Eim = $[100(PiGim/4\pi Rim^2)(Gri\lambda^2/4\pi)]^{1/2}$ EXP(-J2 π Rim/λ) EXP(JθAim)

Eim = ILLUMINATOR TO MISSILE VOLTAGE

Pi = rms ILLUMINATOR POWER

Gim = ILLUMINATOR TO MISSILE ANTENNA POWER GAIN

Rim = ILLUMINATOR TO MISSILE RANGE

Gri = MISSILE REAR ANTENNA GAIN

λ = ILLUMINATOR WAVELENGTH

θAim = ILLUMINATOR TO MISSILE VOLTAGE ANTENNA PHASE

FIGURE 4. Description of Rear Multipath Model - Direct Signal Path.

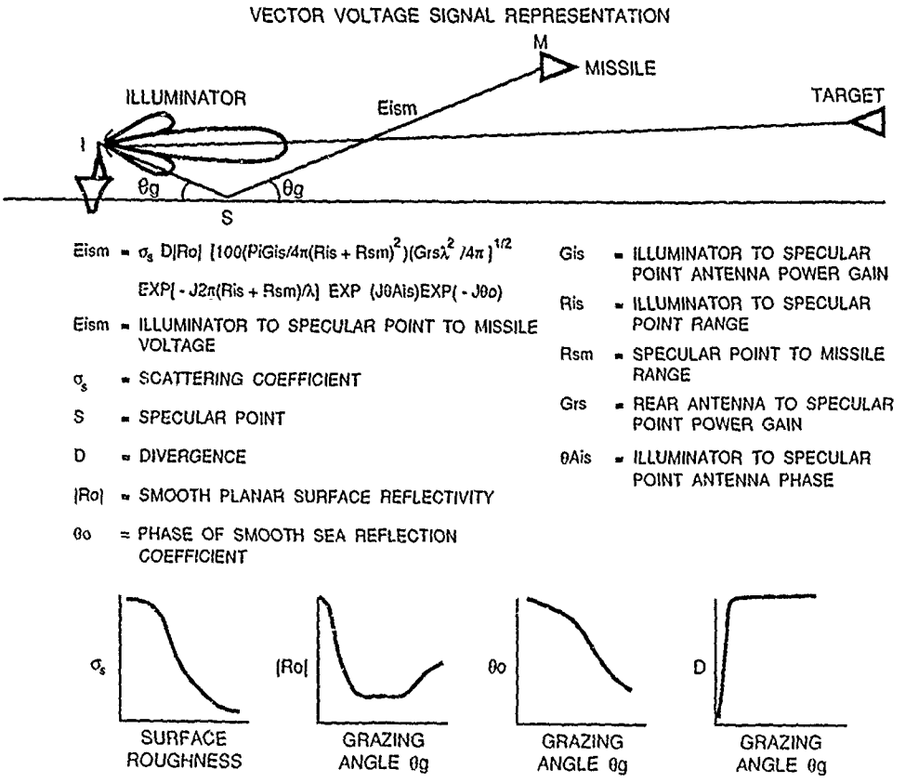


FIGURE 5. Description of Rear Multipath Model - Specular Signal Path.

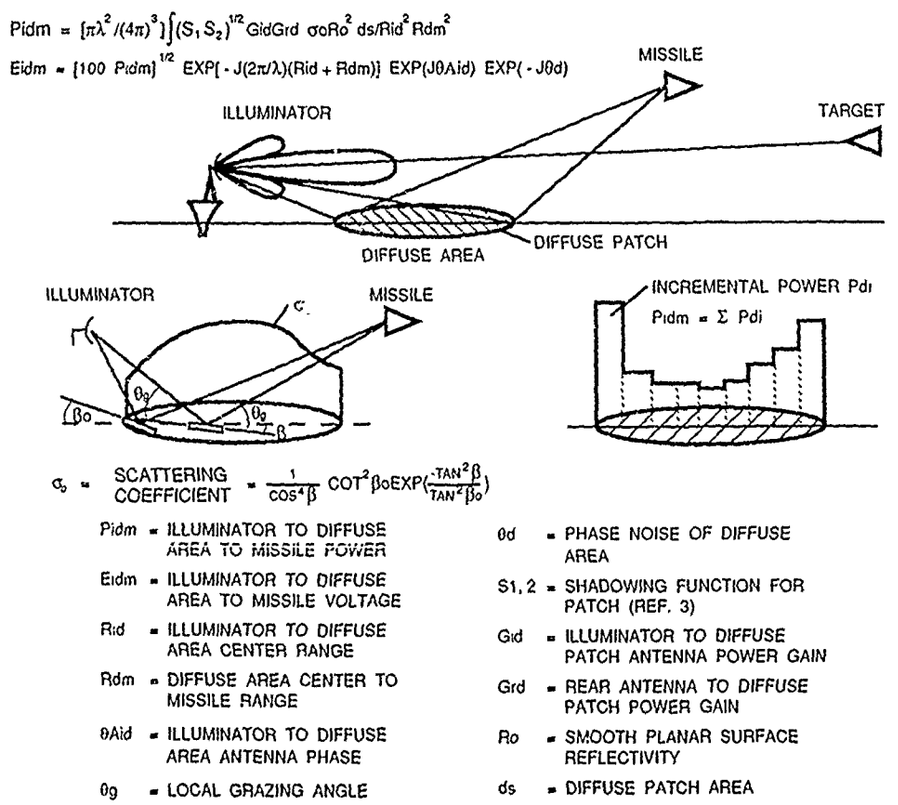


FIGURE 6. Description of Rear Multipath Model - Diffuse Signal Path.

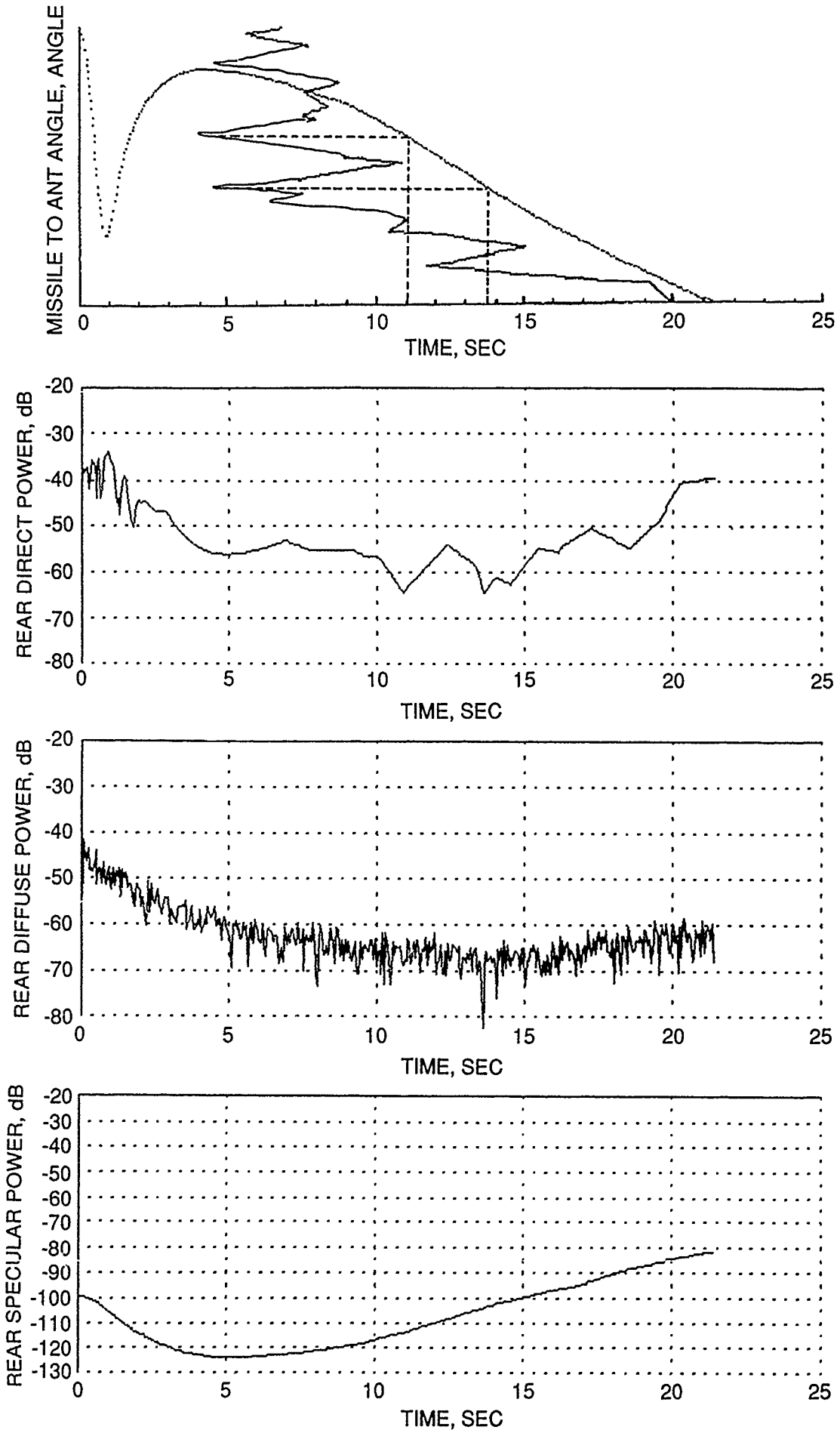
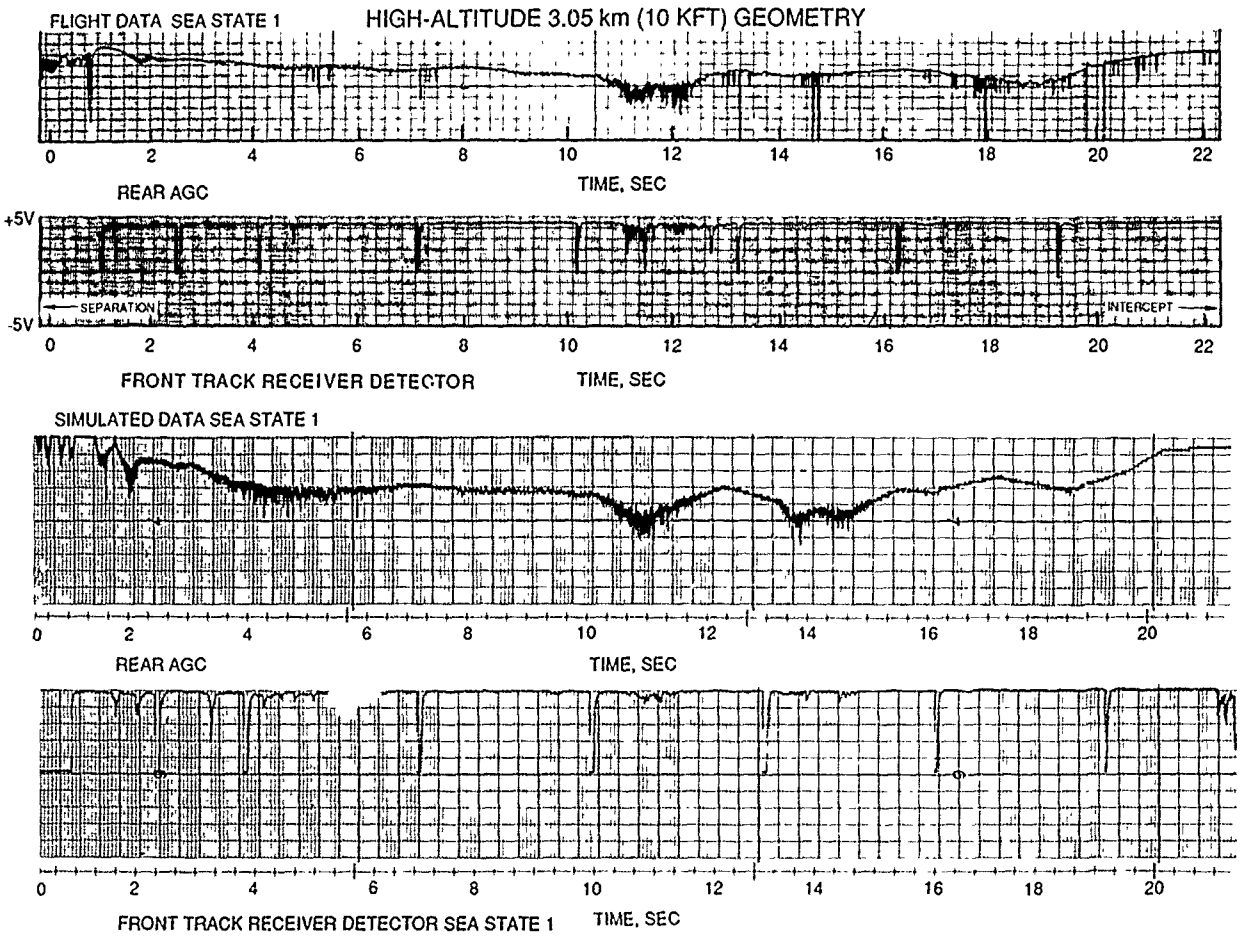
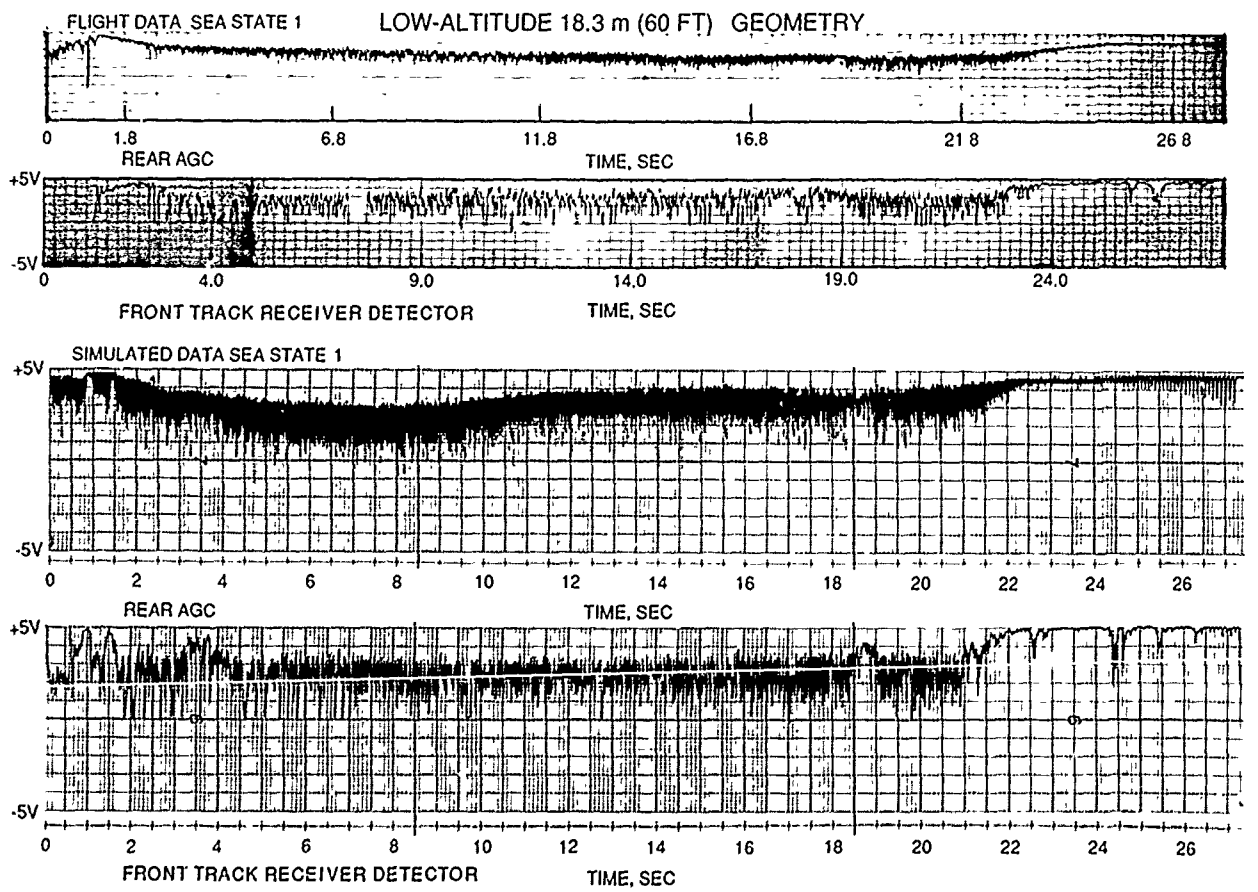


FIGURE 7. Rear Multipath Model - Signal Components Ship Launch - Target Altitude 3.05 km (10 kft).

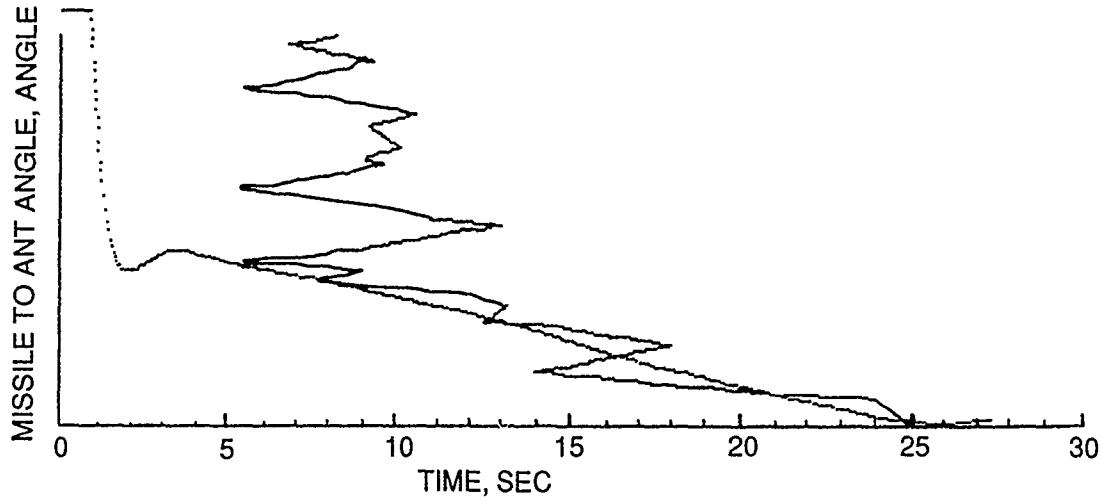


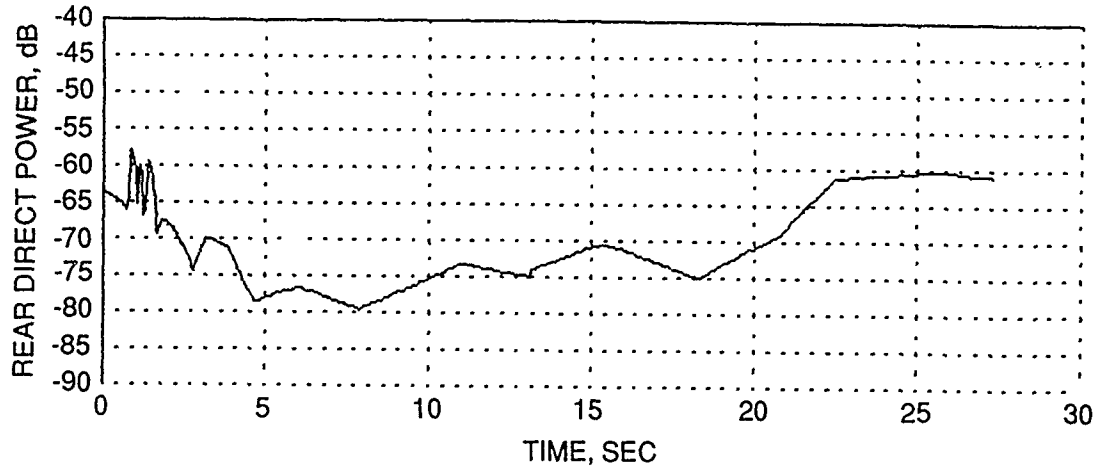
(a) High Target Altitude 3.05 km (10 kft).

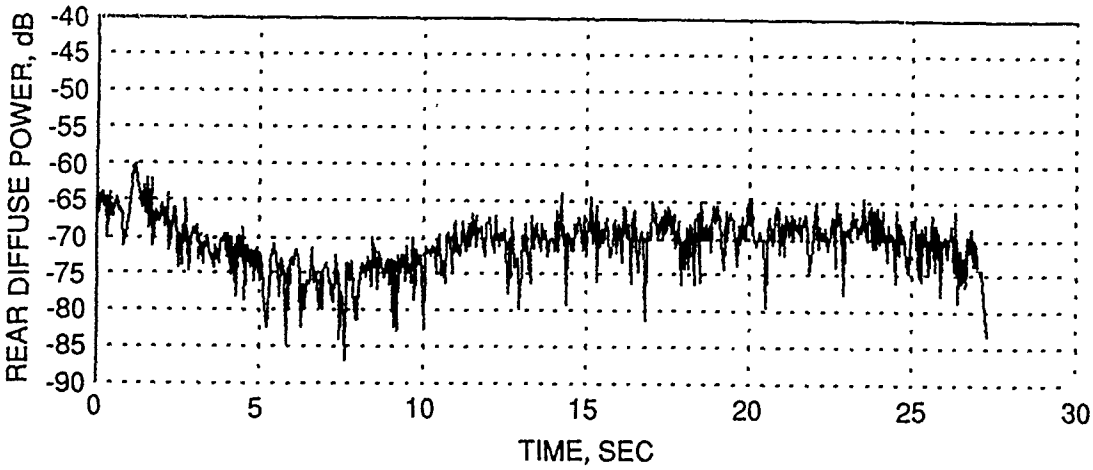


(b) Low Target Altitude 18.3 m (60 ft).

FIGURE 8. Rear Multipath Model Validation.







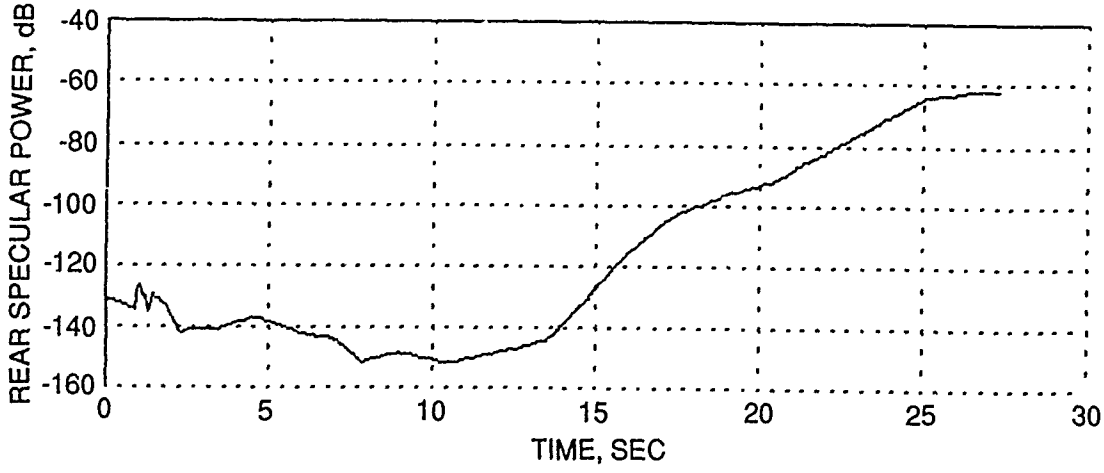


FIGURE 9. Rear Multipath Model - Signal Components Surface Launch - Target Altitude 18.3 m (60 ft).

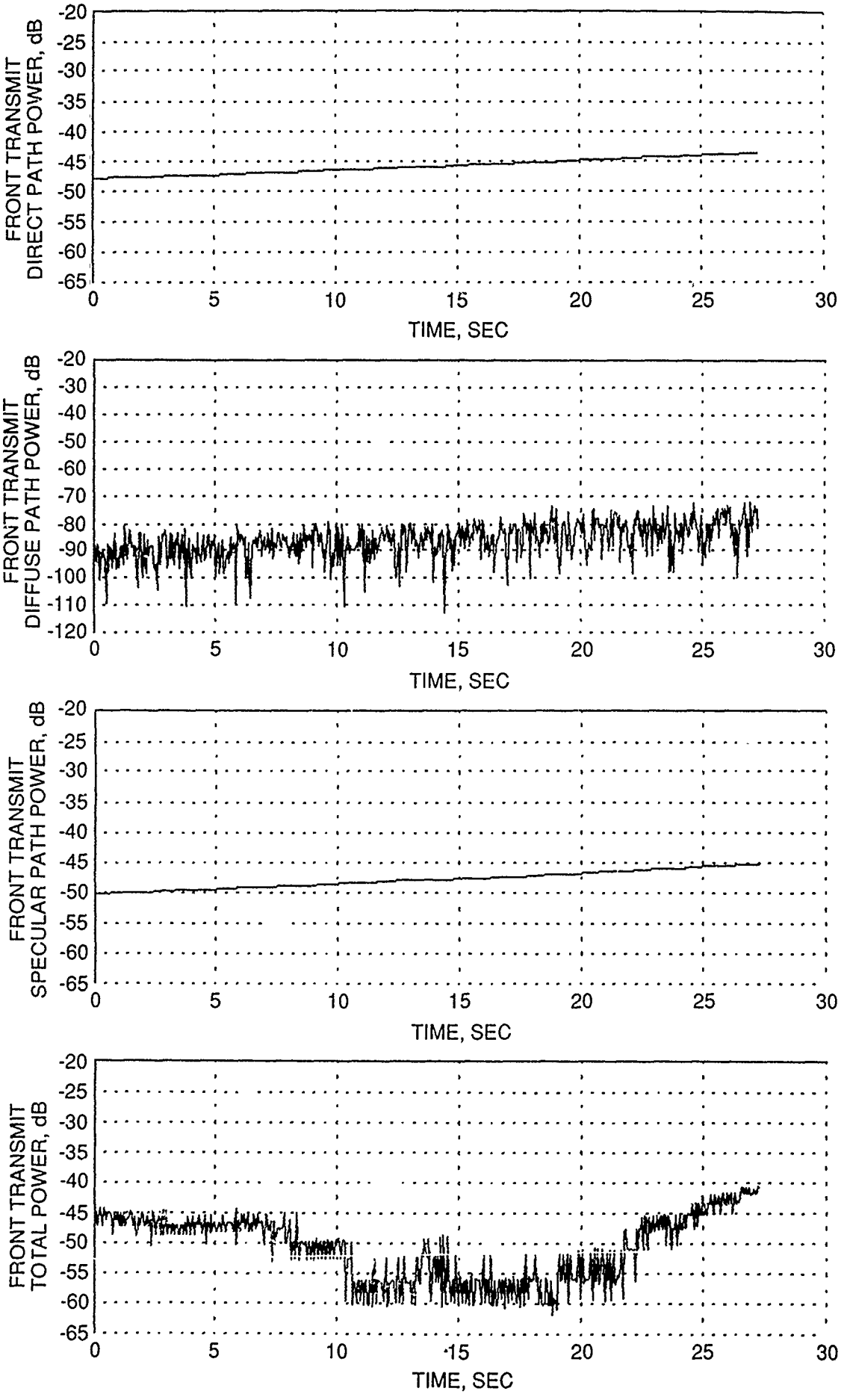


FIGURE 10. Front Multipath Transmit Path Power Components Surface Launch - Target Altitude 18.3 m (60 ft).

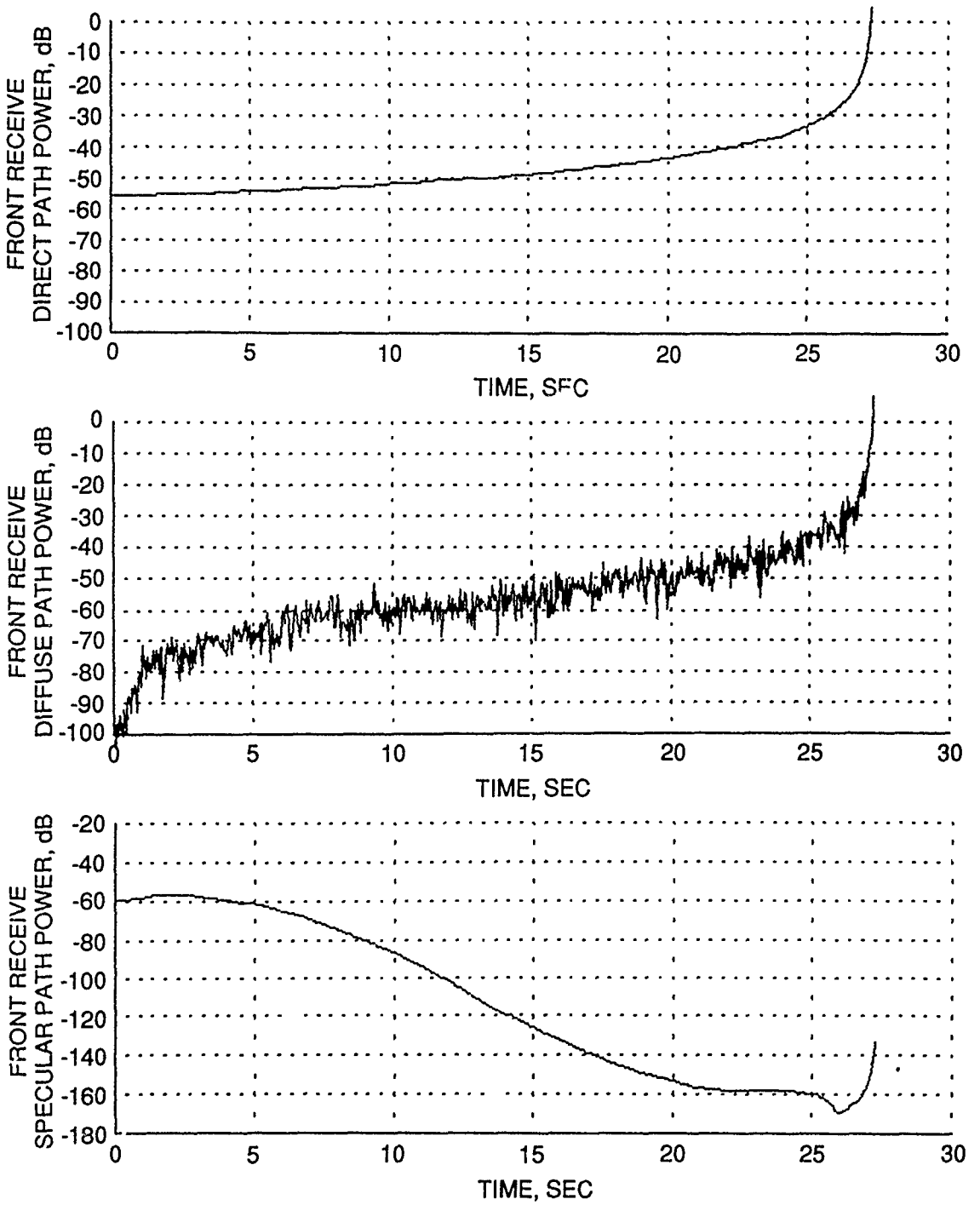


FIGURE 11. Front Multipath Receive Path Power Components Surface Launch - Target Altitude 18.3 m (60 ft).

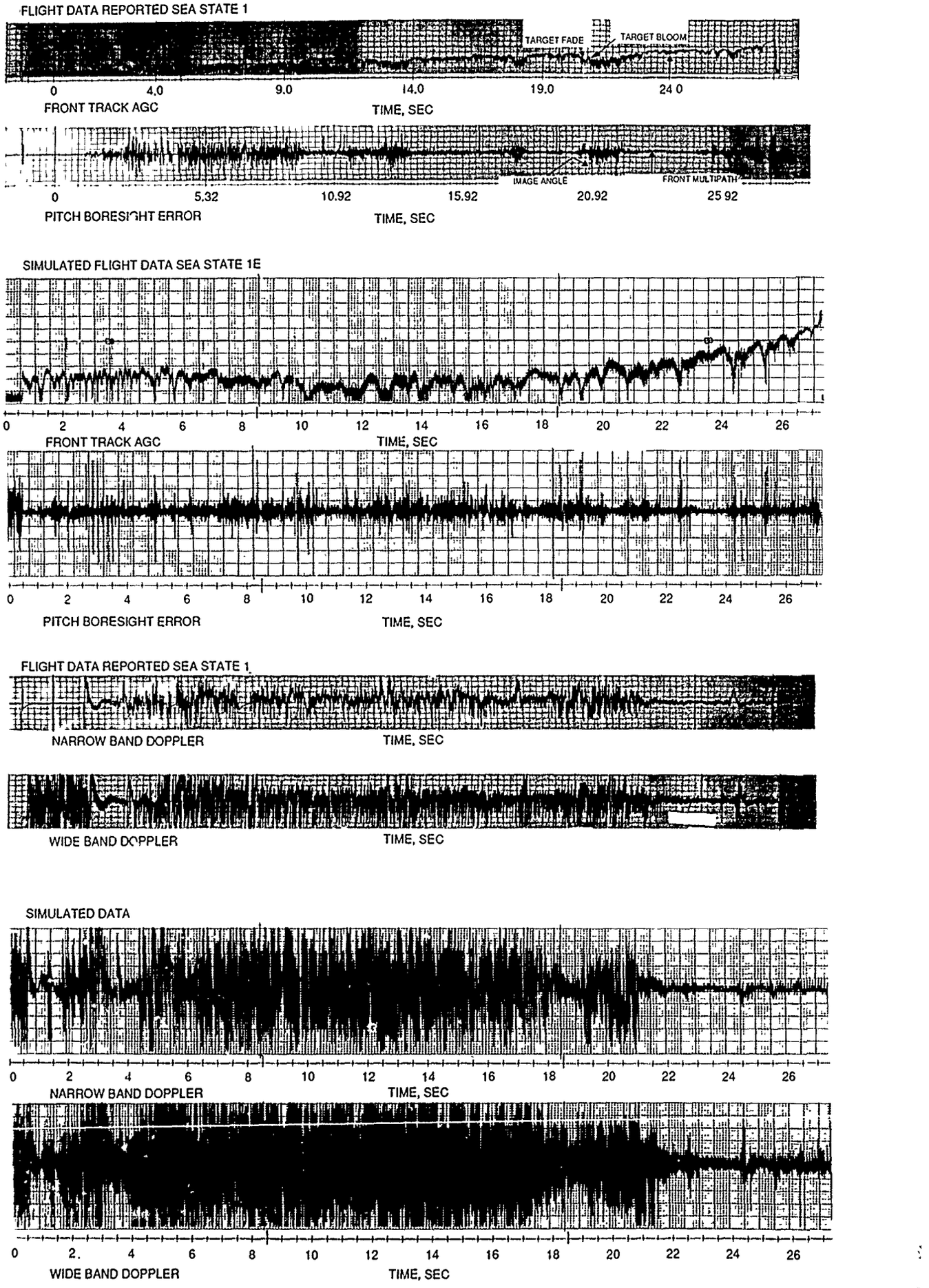


FIGURE 12. Front Multipath Validation - Low Altitude 18.3 m (60 ft).

A NEW APPROACH TO HARDWARE-IN-THE-LOOP SIMULATION (FALKE SHUTTLE)

by

C.-H.Oertel, K.Alvermann, R.Gandert and B.Gelhaar
DLR
Deutsche Forschungsanstalt für Luft- und Raumfahrt
Institut für Flugmechanik
D-3300 Braunschweig
Germany

1. Abstract

Nowadays system simulation is an important task in the development procedure of new and improved flight vehicles. In addition to typical off-line and non-real-time system simulations, special requirements for real-time computation speed exists for flight simulators. A further application of real-time simulation is the so called 'hardware-in-the-loop simulation' where real components like new closed loop controllers or complete on-board systems are tested under realistic conditions.

In the past a lot of companies have designed and built special purpose simulation computers which are very powerful but expensive. The progress in computer science shows a trend to distributed systems where multiple processors are running in parallel to improve the performance dramatically. At the DLR Institute for Flight Mechanics a computer system, based on the TANSPUTER was designed to achieve real-time simulation capabilities for the FALKE Shuttle. This flight vehicle is a reduced-size model of a reentry body which is used for a new aerodynamic flight test technique.

After an introduction to this FALKE flight test technique the paper presents the characteristics of the hardware-in-the-loop simulation. Then an introduction to TRANSPUTERs and the associated philosophy is given. This leads to a description of the needed hardware for the simulation including all the interfaces to the FALKE flight object. The next part of the paper presents the characteristics of the simulation model and its mathematical formulation. Then a three dimensional real-time representation of FALKE Shuttle is described. In the last part of the paper results of the executed hardware-in-the-loop simulation are presented.

2. FALKE flight test technique

Since more than a decade the US Space Shuttle and recently the USSR Buran Shuttle prototype have been developed for space transportation purposes. Other reusable space transportation systems are planned (e.g. NASP, Hermes, Sänger). Therefore the aerodynamic testing of spaceplanes is gaining importance. The validation of important aerodynamic and control parameters (e.g. derivatives) in the transsonic, supersonic and hypersonic flight region via wind tunnel testing is for various reasons critical:

- Aerodynamic parameters for sufficient high Reynolds and Mach numbers can be obtained only with limited accuracy.
- Real gas effects are difficult to simulate.
- Model suspension and wind tunnel wall interferences affect the measurements.

Therefore the analytical and experimental prediction of the dynamic behavior of such reentry flight vehicles is quite limited. Thus a new complementary flight test technique has been developed, which will contribute to an improved confidence level of estimated aerodynamic and control parameters and, hence, the flying qualities of such vehicles prior to their first flight. This flight test technique which is based on balloon assisted drop model testing is called FALKE (Fallkörpererprobung).

A US Space Shuttle replica with a length of about seven metres will be lifted to an altitude of about forty-five kilometres by a balloon (Fig. 1). From this altitude the model is dropped and accelerated in vertical free fall up to a flight Mach number far beyond Mach 1. In order to take full advantage of the potential energy, a steep dive with small angles of attack has to be followed. This is achieved by an on-board flight control system which computes the control surface deflections. Maneuvers can be flown from thirty kilometres down to six kilometres. The resulting altitude-Mach diagram (Fig. 2) corresponds to the final phase of the US Space Shuttle orbiter during landing. The size of the model guarantees sufficient high Reynolds numbers. The maneuvers will include computer generated input commands which were optimized for system identification

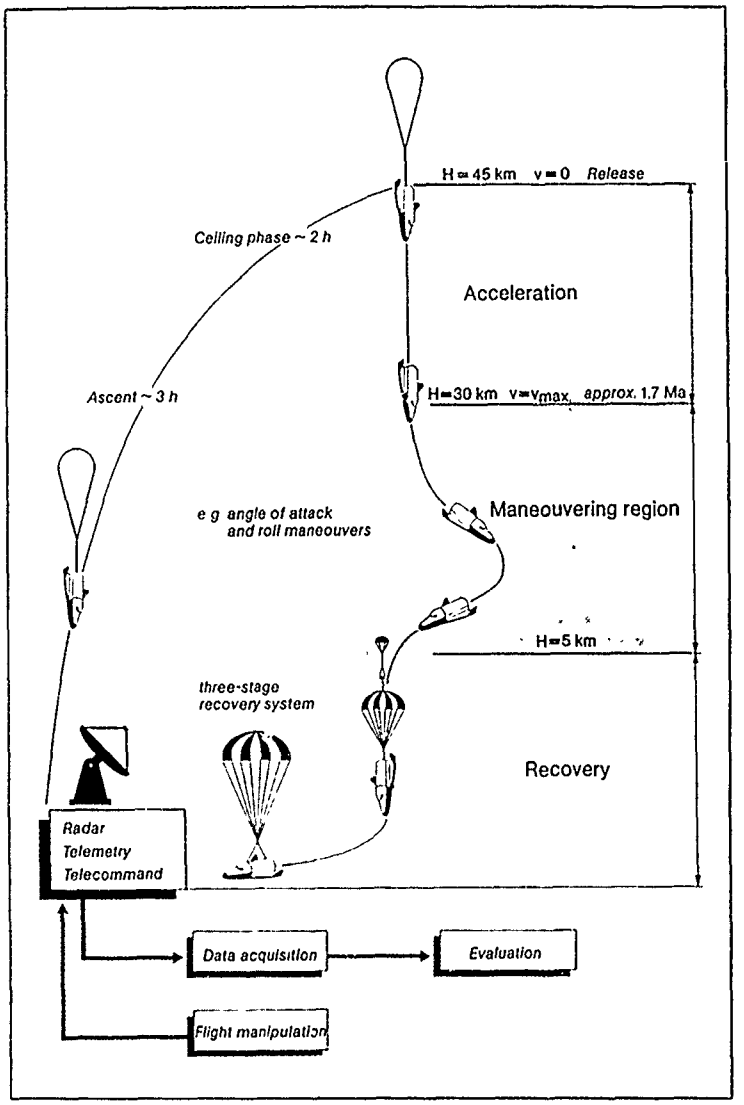


Fig. 1 Typical mission profile

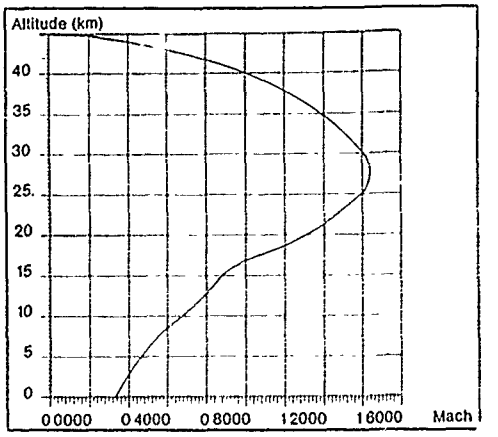


Fig. 2 FALKE Altitude-Mach envelope

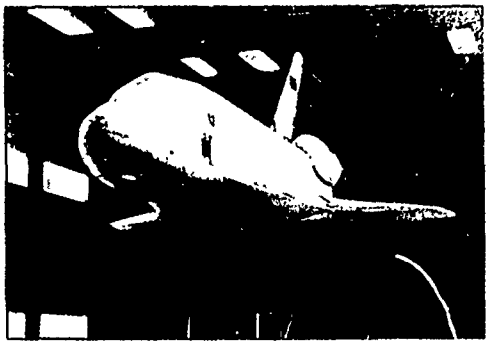


Fig. 3 FALKE Shuttle in the DNW

purposes. The FALKE Shuttle response is measured by suitable inertial and air data sensors. These values are stored on board. The data will be used for system identification purposes to estimate important aerodynamic and control properties [1]. At an altitude of six kilometres the model is recovered by a parachute system. After a gentle landing FALKE Shuttle can be reused for future missions. The maneuvers (e.g. input signals) must ensure that

- reliable parameter identifications are maid possible,
- maximum loads and turning rates will not be exceeded,
- during the entire mission, the model will stay within the maximum allowable horizontal test range of five kilometres.

FALKE is a research programme sponsored by the German Ministry for Research and Technology (BMFT) and the German Aerospace Research Establishment (DLR). The following companies and institutions are involved [2]:

OHB-System GmbH:	main contractor, definition of the flight vehicle, data acquisition, electrohydraulics, integration and test, ground support equipment, system safety;
Autoflug GmbH:	recovery system;
CNES-ASA:	mission responsibility, provision of a balloon;
DLR:	parameter identification, wind tunnel tests, data analysis, hardware-in-the-loop simulation;
Hoffman-Flugzeugbau:	construction of the flight vehicle;
ISRA:	software for the flight controller;
MBB ERNO:	on-board computer;
ZARM:	analysis of aerodynamic data.

The FALKE project has been organized in the following two steps:

(1) In order to verify the FALKE flight test technique, a flight vehicle of known flightmechanic and aerodynamic characteristics, thus providing excellent benchmark tests, i.e. the US Space Shuttle orbiter is selected as above discussed. The FALKE flight model is therefore an exact replica at scale 1:5.24 of the US orbiter. For this spaceplane a sufficient large aerodynamic data base exists [3]. Therefore the obtained flight determined derivatives can be correlated to this data after a successful mission, thus leading to a judgement of this new technique.

(2) Implementation of the FALKE flight test technique for the European space plane programme Hermes. In order to utilize the maximum acceptable model length of about seven meters the Hermes space plane configuration has to be scaled to 1:2.5.

The first FALKE mission is planned in 1990. The replica of the US Space Shuttle orbiter has already been tested in the DNW, the German Dutch wind tunnel [4] (Fig. 3). Some parameters of the FALKE flight vehicle are:

length	= 6.86 m	height	= 2.62 m
wing surface	= 8.49 m^2	wing span	= 4.38 m
wing depth	= 2.22 m	weight	= 640 kg

During the flight three maneuvers are planned:

1.) impulse on elevons:

altitude	= 20.2	kilometres
duration	= 2	seconds
amplitude	= -6.0	degrees

2.) doublette on elevons:

altitude	= 11.4	kilometres
duration	= 2	seconds
amplitude	= +-0.5	degrees

3.) 3211-signal on elevons:

altitude	= 9.2	kilometres
duration	= 7	seconds
amplitude	= +-0.2	degrees

3. Hardware-in-the-loop simulation

The FALKE Shuttle is equipped with the following systems:

- movable control surfaces,
- electrohydraulics for steering the control surfaces,
- on-board flight-control computer with sequencer, controller, maneuver program
- data recorder,
- telemetry and wireless command,
- Rosemount air data sensor for measuring the angle of attack and angle of sideslip as well as pitot and static pressure,
- temperature sensor,
- Litef LTR-81 attitude and heading reference system (AHRS) which measures the angular position, rate of turn, acceleration, etc.,
- recovery system with parachutes,
- battery as power supply.

Such a complex system has to be properly tested before operation. Particularly a test of the completely assembled FALKE Shuttle is sensible, after all components have been shown to function. This is done by hardware-in-the-loop simulation. Thereby all those parts of the system which cannot operate in their normal environment during the test, have to be simulated in real-time. In the case of FALKE Shuttle the following components have to be simulated (Fig. 4).

- aerodynamics and flight mechanics of the aircraft,
- air data and temperature sensors,
- attitude and heading reference system (AHRS).

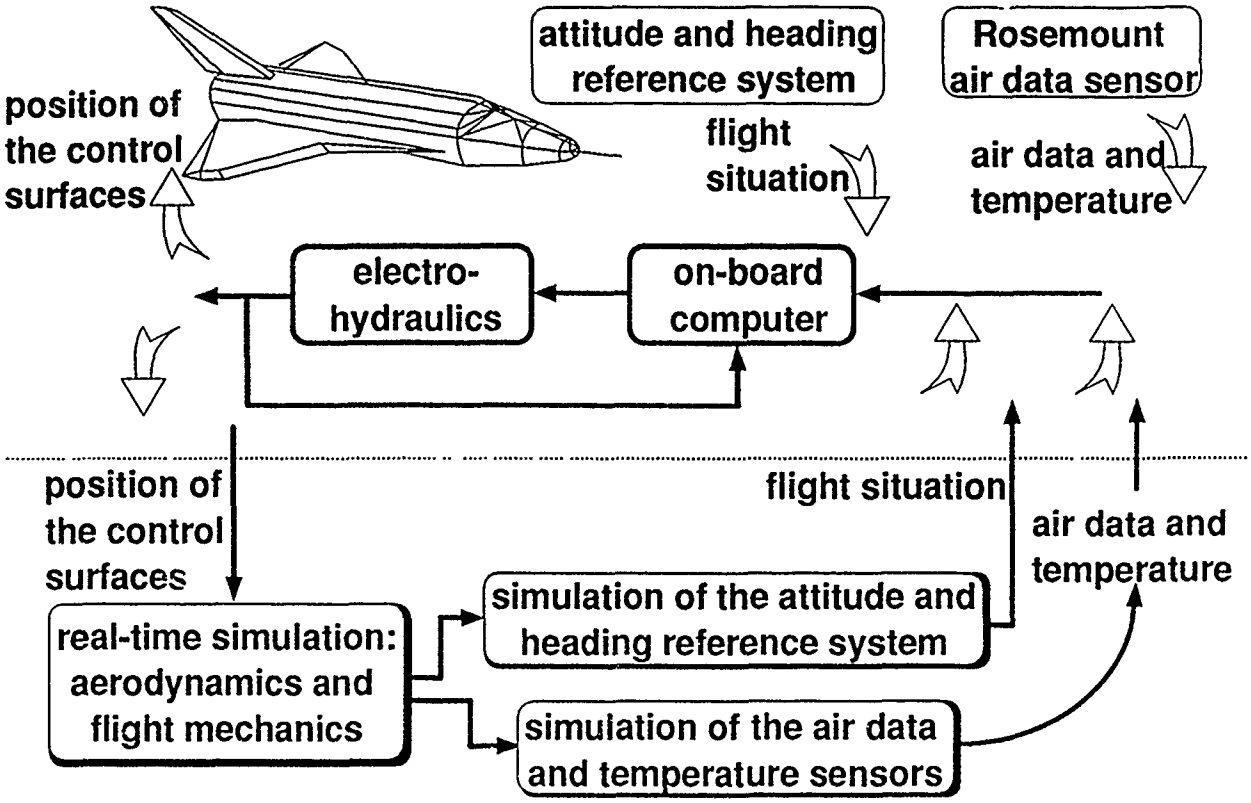


Fig. 4 Hardware-in-the-loop test

The simulation delivers the missing measured values and variables of state by repeatedly calculating a mathematical model. The quality of this model therefore determines the quality of the simulation. FALKE Shuttle output and simulation input are the positions of the different control surfaces. Based on this information and on the knowledge of the variables of state at a certain time t, the simulation determines the variables of state at the time $t + \Delta T$. Another transformation of these values delivers the corresponding measured values of the sensory units. These are transmitted to the FALKE Shuttle flight-control computer by suitable interfaces. This computer analyses the arriving data the same way as in a real flight and commands the positions of the control surfaces. This closes the control loop. With a hardware-in-the-loop simulation the following parts can be tested under realistic conditions:

A) Operation of the assembled system (hardware):

- electrohydraulics and control surfaces,
- on-board computer and data recording,
- electronics and interfaces,
- power efficiency of the on-board systems.

B) Performance of the flight-control computer (software):

- flight maneuver program,
- robustness of the controller,
- usage of the telemetry,
- release of the recovery system.

In particular the test "robustness of the controller" needs separate attention. For gaining information about the controller, it is necessary to make the environment of the tests harder. This can be done by

- variation of the initial conditions (when FALKE Shuttle is released from the balloon), in particular tests with different rates of turn and angular positions,
- variation of the derivatives,
- variation of the moments of inertia and the mass,
- generation of additional stimulations of the flight dynamics (vehement blasts of wind),
- reduction in the efficiency of the control surfaces,
- raising of the delay time of the sensory units (AHRS),

- displacement of the centre of gravity,
- flight with a higher angle of attack,
- generation of additional systematical and statistical errors in the simulation of the sensory units,
- generation of failures of parts of the measuring instruments.

After a few tests with these additional burdens, i.e. tests with realistic up to worst case conditions, a judgement can be given about the quality of the flight-controller.

The simulation computer for the hardware-in-the-loop tests has to meet all the following requirements:

- The calculation of the mathematical model has to be achieved in real-time, i.e. during the existing time span ΔT all variables of state and measuring values which are valid at the next mesh point have to be estimated.
- The simulation has to be linked to the FALKE Shuttle via suitable interfaces.
- The computer system must be mobile; an in-the-field-test must be possible a short time before flight.
- Certain processes must be started by a clock.
- For the judgement of the simulation results, possibilities should exist
  - to analyse important flight data online graphically,
  - to save data on a file,
  - to output data on a printer.
- The computer system must have an expandable and modular structure, that means
  - if a specific component of the software has to be modified, all other parts may remain unchanged,
  - clearly defined and simple interfaces should connect different tasks of the software (this is especially important if several people work at the same project),
  - a fixed performance of the simulation computer is not wanted; the system has to be expandable for further similar problems.
- Any change of the software, as for example a variation of the initial conditions or derivatives, must be possible in a short time and without great effort during the tests.
- The price for the computer system should be modest.

4. OCCAM and TRANSPUTER

4.1 Philosophy

A system in the 'real world' can be described as a set of processes which work in parallel and exchange information between them. These processes are local and exchange their information only with neighbouring processes. A good approach for numerical simulation would be, to have the same set of information exchanging processes on a computer too. This kind of mapping of communicating parallel processes in the real world to communicating parallel processes in the computer would provide a consistent relationship between the real world model and the realization in the numerical world inside the computer. Unfortunately traditional computers do not match the basic requirements of this approach i.e.

- parallelism / quasi-parallelism
- communication
- locality

in a sufficient manner. In addition, most of the traditional languages are not developed to do this job. Although most computers allow parallel communicating processes on one machine or even two, their operating systems, schedulers, semaphore-techniques etc. are the bottleneck for real-time simulation. They need a lot of code and a lot of time, because these computers have been optimized to do one job at a time. Parallelism has been introduced by software overhead to these machines, but the basic concept of the machines has not really been influenced by realtime simulation requirements.
What we need for fast real-time simulation of complex systems is:

a) significant improvement of performance by having many of physical computing units,
b) moderate price of the units because we need a lot of them,
c) a good balance between performance, memory capacity and communication capability of each computing unit,
d) configuration capability of the units in order to have a consistent model,
e) a language allowing a simple and clear description of the configuration and communication of such multi-processor systems.

The requirements a) - d) are fulfilled by the TRANSPUTER, a new processor which has been developed by INMOS, funded by the European ESPRIT project. The high level language OCCAM, which has been designed by the same company, focuses on the requirement e).

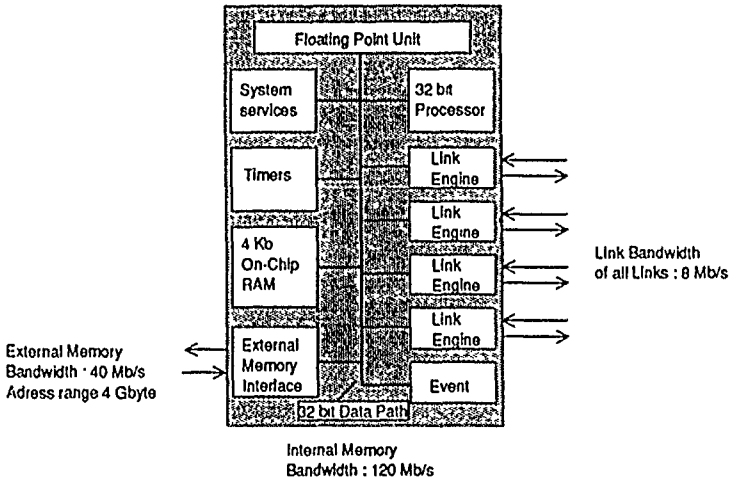


Fig. 5 Internal structure of Transputer T800

4.2 TRANSPUTER

The TRANSPUTER (Fig. 5) is a Von Neumann machine with a built-in floating point unit and 4kB on-chip RAM. It has been optimized to run OCCAM programs. Two special features make it an ideal machine for parallel processing:

- Hardware scheduling:

  All scheduling between processes is done in hardware. No operating system to schedule processes is needed. All this is done inside the TRANSPUTER and is complete transparent to the programmer. Scheduling takes only some microseconds if, e.g. a process becomes ready to run.

- Communication links:

  Each TRANSPUTER has four bidirectional serial 20 Mbit/s links, each working with a separate link engine using DMA. To send, e.g. 1000, words via a link to another TRANSPUTER, the user only has to start the transfer and nothing else. If the receiving TRANSPUTER is not ready to receive the message, the sending process is descheduled automatically until the message has been received. Even if the processor is sending and receiving on all four links simultaneously with full speed, the internal bandwidth guarantees, that the CPU's work goes on. As we will see later, the hardware links correspond to logical channels between OCCAM processes.

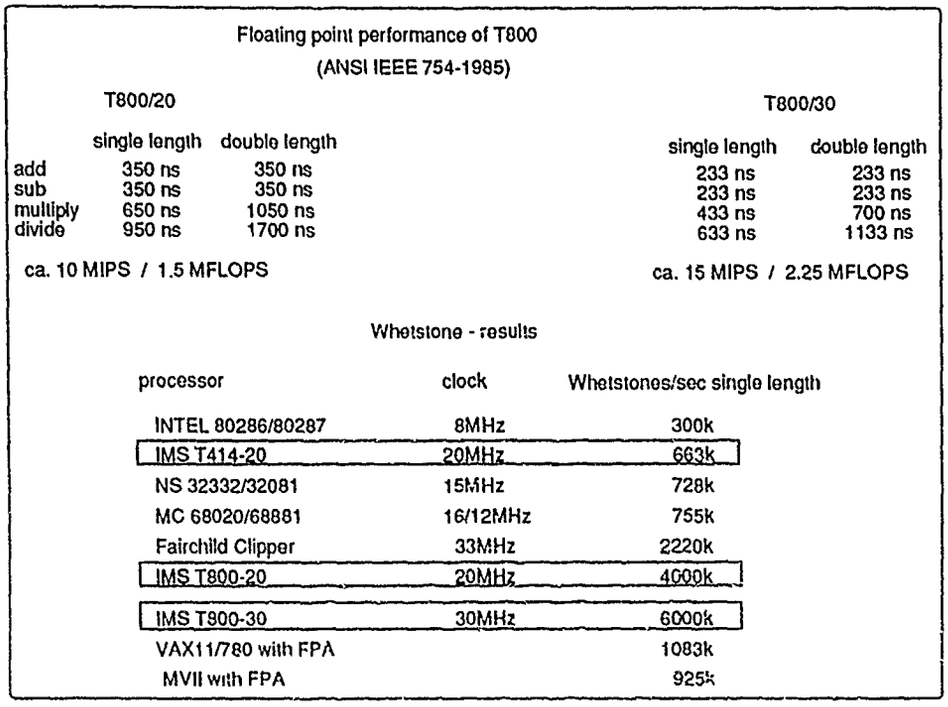
- Performance:

  Fig. 6 shows some benchmarks of the TRANSPUTER and its competitors. The Whetstone benchmark is a typical sequential program. So the TRANSPUTER doesn't work in the parallel world it has been designed and optimized for.

4.3 OCCAM

The basic concepts of OCCAM have been proposed by Hoare, [5]:

- at runtime, a process is loaded and fixed on its processor,
- at runtime, processes can not be created, i. e. all processes of the system exist when the program starts, whether they do anything or not,
- processes only work with local memory,
- a process communicates to another via any kind of communication network.



Floating point performance of T800

(ANSI IEEE 754-1985)

	T800/20		T800/30	
	single length	double length	single length	double length
add	350 ns	350 ns	233 ns	233 ns
sub	350 ns	350 ns	233 ns	233 ns
multiply	650 ns	1050 ns	433 ns	700 ns
divide	950 ns	1700 ns	633 ns	1133 ns
	ca. 10 MIPS / 1.5 MFLOPS		ca. 15 MIPS / 2.25 MFLOPS	

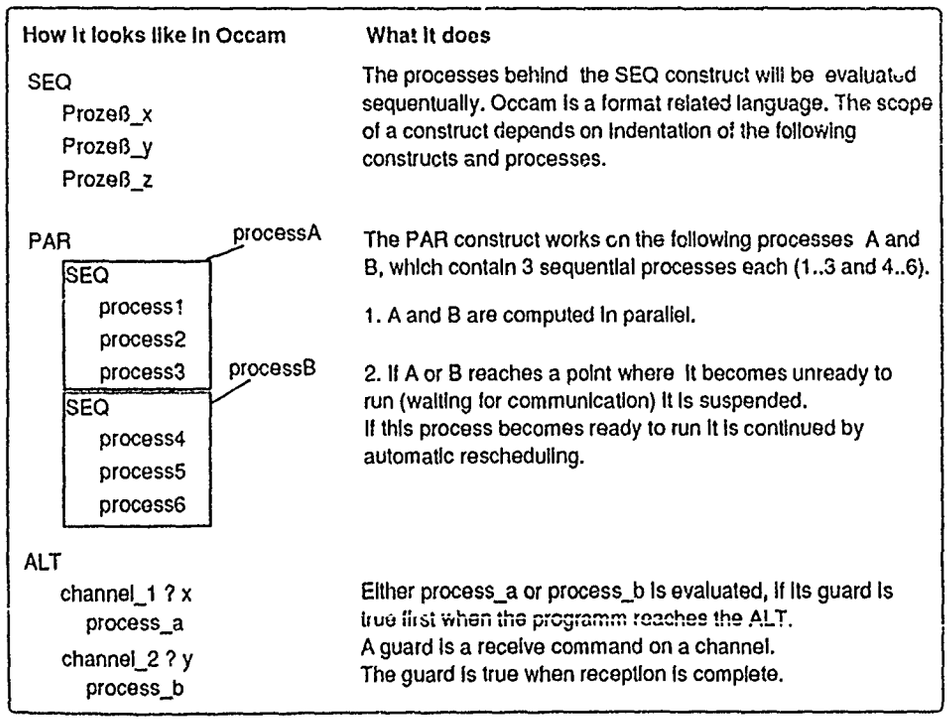
Whetstone - results

processor	clock	Whetstones/sec single length
INTEL 80286/80287	8MHz	300k
IMS T414-20	20MHz	663k
NS 32332/32081	15MHz	728k
MC 68020/68881	16/12MHz	755k
Fairchild Clipper	33MHz	2220k
IMS T800-20	20MHz	4000k
IMS T800-30	30MHz	6000k
VAX11/780 with FPA		1083k
MVII with FPA		925k

Fig. 6 Performance of T800 and other processors [6]

OCCAM provides the following language elements to achieve the above rules:

- processes
- constructs
- procedures
- communication.



How it looks like in Occam	What it does
SEQ Prozeß_x Prozeß_y Prozeß_z	The processes behind the SEQ construct will be evaluated sequentually. Occam is a format related language. The scope of a construct depends on indentation of the following constructs and processes.
PAR processA SEQ process1 process2 process3 processB SEQ process4 process5 process6	The PAR construct works on the following processes A and B, which contain 3 sequential processes each (1..3 and 4..6). 1. A and B are computed in parallel. 2. If A or B reaches a point where it becomes unready to run (waiting for communication) it is suspended. If this process becomes ready to run it is continued by automatic rescheduling.
ALT channel_1 ? x process_a channel_2 ? y process_b	Either process_a or process_b is evaluated, if its guard is true first when the programm reaches the ALT. A guard is a receive command on a channel. The guard is true when reception is complete.

Fig. 7 OCCAM Language elements

Processes may consist of other processes. The atomic process is SKIP which does nothing. The assignment (f.e. a:=b) is a process too. The constructs WHILE,FOR,IF etc. are well known from other languages. But there are some additional new constructs like SEQ,PAR and ALT. Fig. 7 shows what constructs in OCCAM look like and what they do. These are the important basic language elements to formulate sequential, parallel, or alternative processes. OCCAM is a format

related language, i.e. a construct works on all following processes which are indented more than the construct itself.

4.4 Communication

The most important feature of OCCAM is the communication between processes. Two processes communicate if one process sends a message on a channel and another receives a message on the same channel.

First process:	channelname ! message.to.send
Second process:	channelname ? message.to.receive

If both processes above work in parallel, they communicate if both reach their send respective their receive point. If one reaches this point first, it waits until the other one reaches its respective point. Of course both processes are de-scheduled automatically by the processor if another process is ready to run. This is done in microseconds. The channel 'channelname' is a logical channel, connecting two processes. Fig. 8 gives the rules of the process-channel concept of OCCAM.

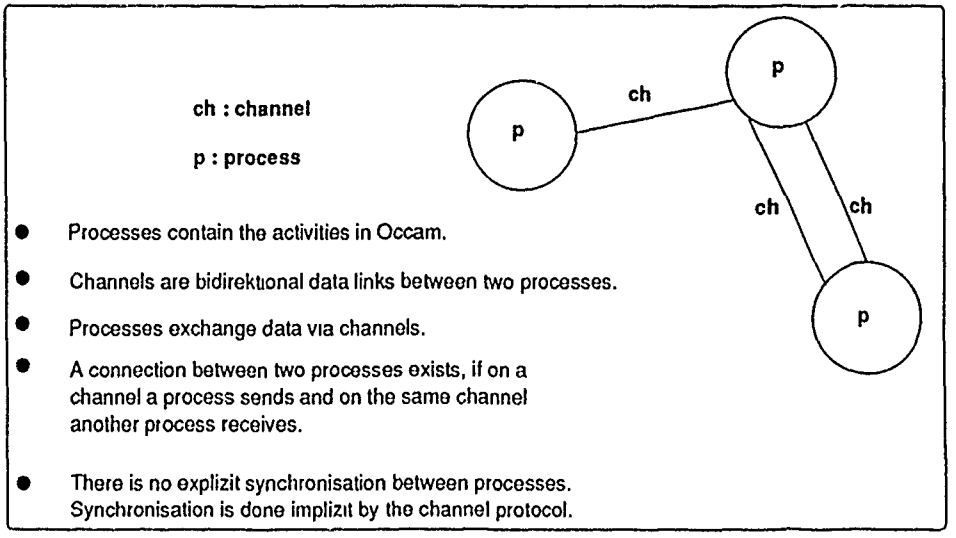


Fig. 8 OCCAM's process-channel-concept

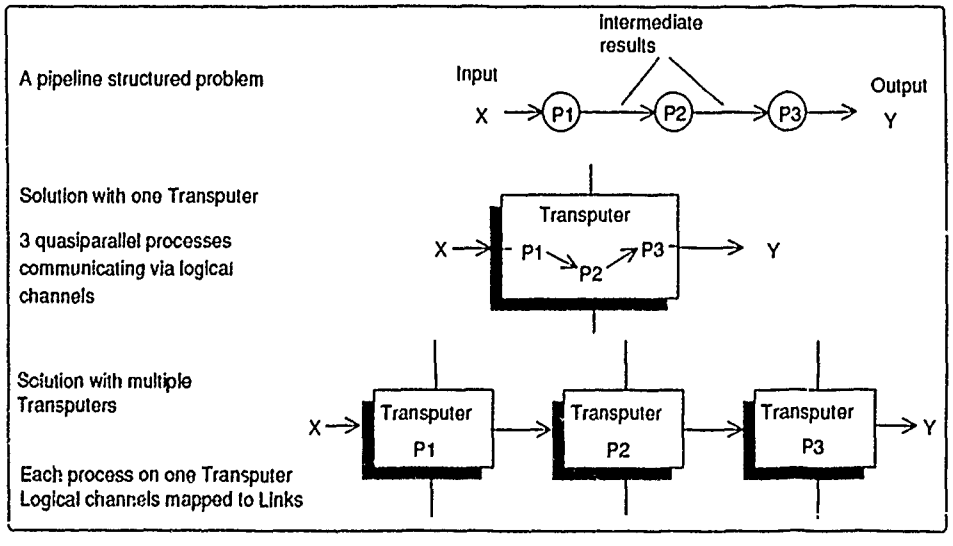


Fig. 9 Placing of channel and process to hardware

4.5 TRANSPUTER-Networks

If several processes run in parallel on one machine, of course they can only run quasi-parallel. If it is nessesary to run these processes faster, they can be placed on several TRANSPUTERs. Without modifying the source code of the program the same processes can run, each on a different TRANSPUTER, by connecting the hardware links of the TRANSPUTER and placing the logical channels on physical links. All placing is done outside the source code. Therefore a system of hundreds of processes can be tested numerically on one machine and then be placed on a lot of

TRANSPUTERs in order to improve the performance. Fig. 9 gives an example for placing channels and processes.It should be noted that, if all communication between the processes has been defined and it has been shown that the program runs without deadlock, the processes can be modified and developed independently. The programmer modifying the process only has to be aware, not to change the interface (e.g. channels and channel protocols) to the rest of the program. The first step in developing an OCCAM program is therefore to evaluate a communication model which represents the problem's structure. After this, the differer processes are evaluated. TRANSPUTERs can be connected in different network structures. Because each problem has an optimal network structure, one has to decide, which structure provides the best 'fit' to the dedicated requirements. Basic structures are, for example:

data flow processing	--> pipeline
2-D-Euler	--> grid
minimum distance between 2 processors	--> hypercube
problems with feedback	--> tree

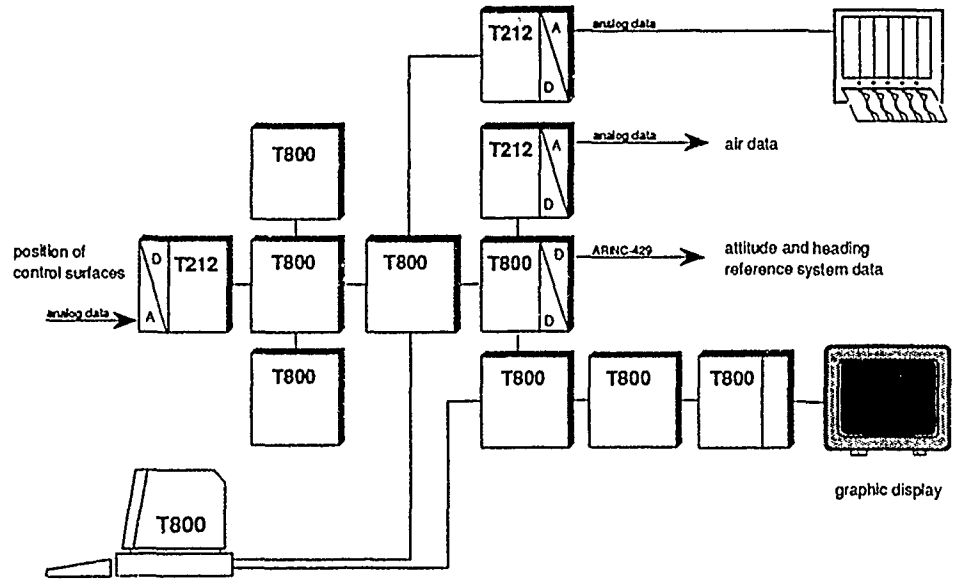


Fig. 10 Hardware structure

5. Hardware

Fig. 10 gives an overview of the system for the Hardware-in-the-loop simulation of the FALKE Shuttle. Due to the use of TRANSPUTERs the structure of the system is highly modular. Every module contains at least one TRANSPUTER whose four serial links provide communication with the other modules of the system. Thanks to this busless system interconnection scheme an unlimitedly large system can be assembled whose overall performance linearily increases with the number of modules used. This stands in contrast to usual bus-coupled multiprocessorsystems whose performance decreases after reaching a maximum due to the "bottleneck" of the common bus. In our system one can distinguish between two kinds of modules. First, there are pure computation modules, which besides the TRANSPUTER contain only a large read/write memory. On the other hand there are modules with a relatively small memory but which are equipped with additional hardware for communicating with the real world around them. These include, for example, modules for input/output of analog or digital signals. The local TRANSPUTER of such a module not only controls the specific I/O section but naturally can be used for a large variety of other computational tasks.

For the presented system only one device, the ARINC-429 module, has to be developed. All other modules shown here came from applications we formerly worked on and could be adopted without any modification.

All boards are of EURO-LONG size and use a DIN-96 connector with a common pinout. The modules fit into a system backplane for primarily supplying power to the boards. Additionally every module slot has a field of eight connectors. Each connector joins the signal of one bidirectional link and reset I/O. All signals are distributed differentially according to RS-422 standard and therefore allow distances of up to 20 meters between two modules. Thus, a widely distributed sy: m can be constructed without degradation of communication performance. But normally a module link node is connected via a short flat cable to another node. In the following we give a short description of our modules.

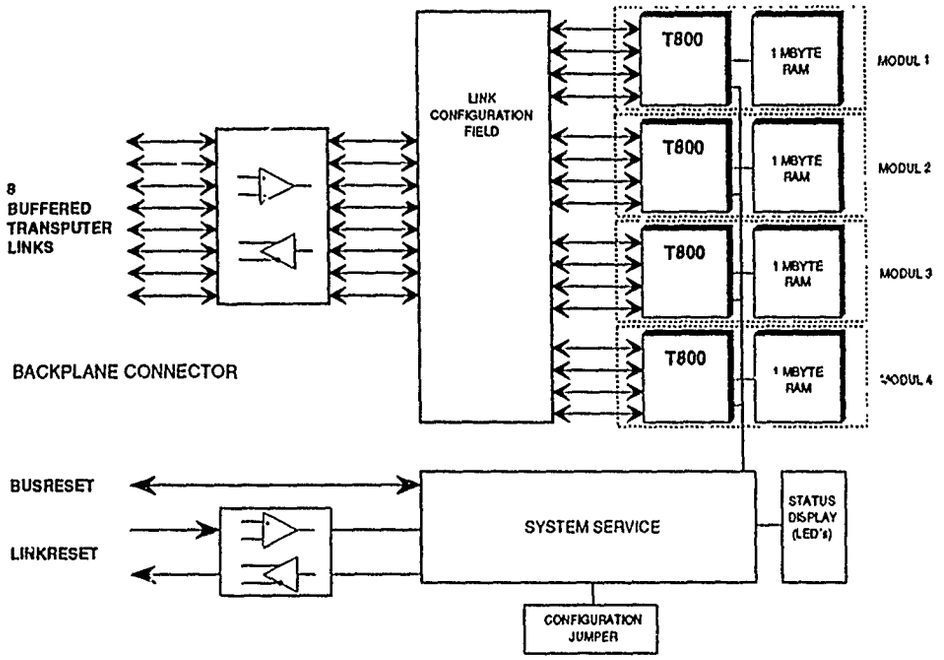


Fig. 11 Quad TRANSPUTER-Board

5.1 Quad TRANSPUTER Board

For purely computational tasks we have developed a board which carries up to four TRANSPUTERs as Piggy-Packs (Fig. 11). Every computer module has a TRANSPUTER T800 (25MHz) and a local memory of 1 Megabyte. Besides a 32 bit-integer-ALU, the T800 includes a floating-point unit and, like all other TRANSPUTERs, a hardware scheduler for concurrent programs. The overall performance of a fully populated Quadboard is 7.5 Mflops and 50 millions of instructions per second.

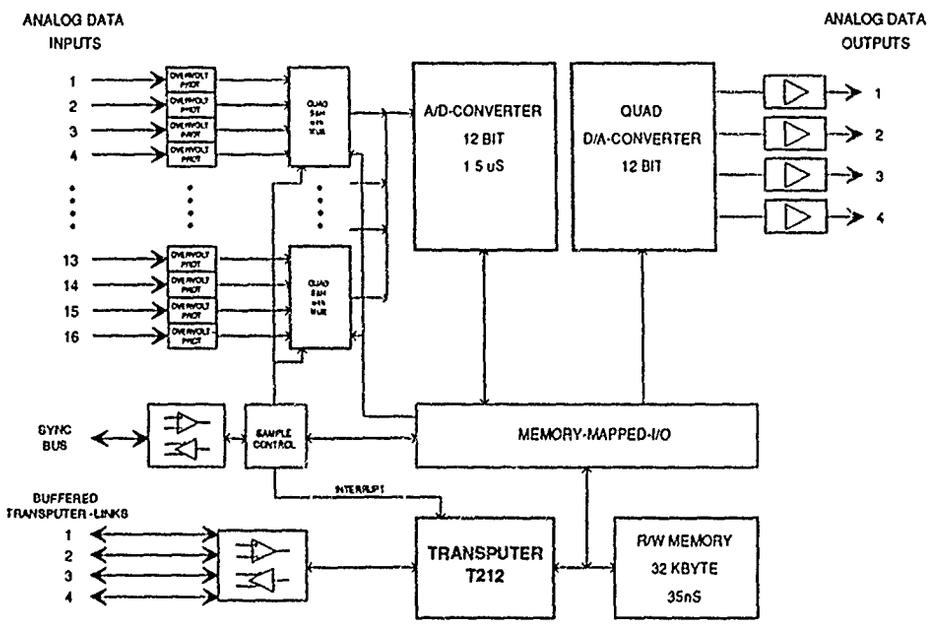


Fig. 12 Analog data input/output module

5.2 Analog Data Input Module

Fig. 12 shows a blockdiagram of our module for capturing up to 16 analog signals. All signals are sampled simultaneously and then sequentially digitized at a resolution of 12 bits. The conversion time is 1.5 microseconds per signal. The module can also generate four analog signals at 12-bit resolution.

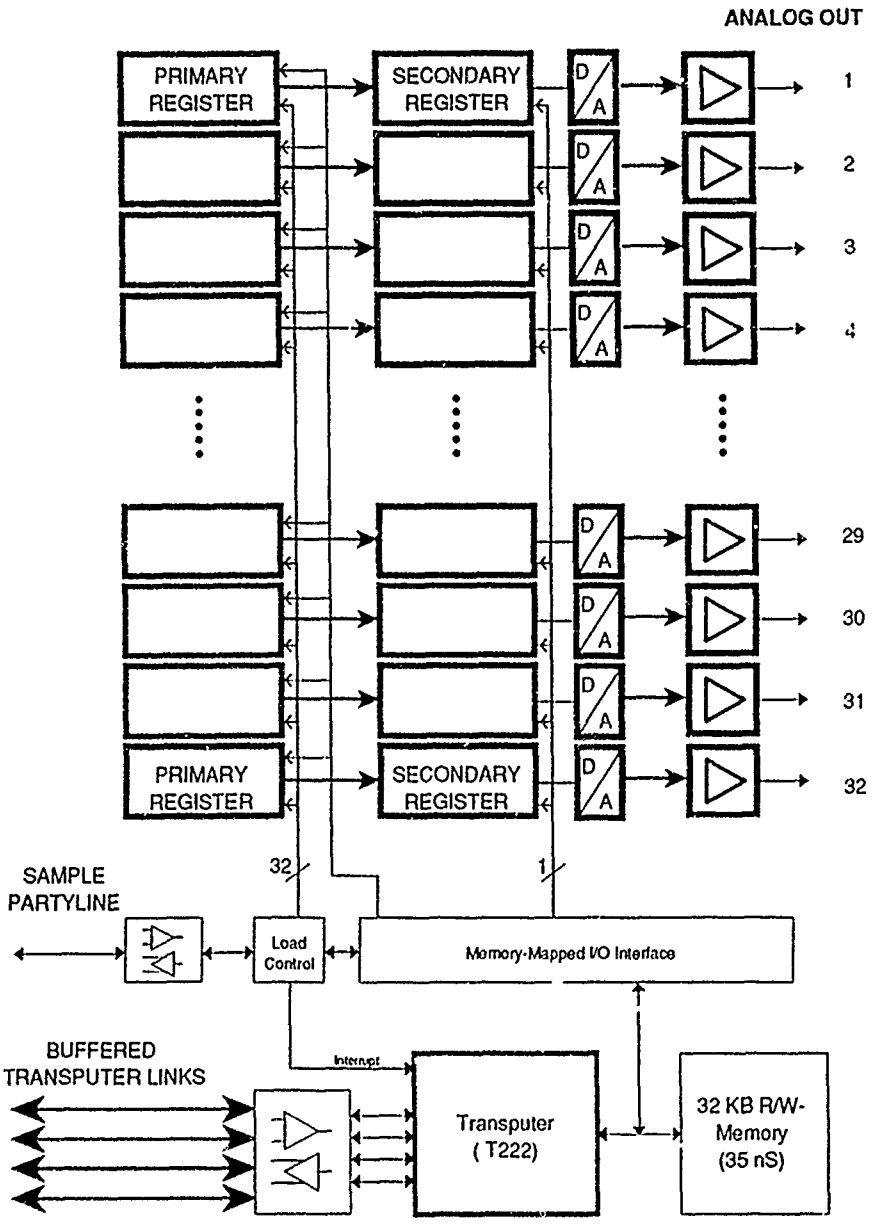


Fig. 13 Analog data output module

5.3 Analog Data Output Module

We also have developed a board wich generates up to 32 analog signals from 12-bit resolution signals (Fig. 13). The binary representations of the signals to be outputted are first loaded sequentially into their primary registers. Afterwards the contents of these primary registers are copied by a common loadpulse into the secondary registers which feed directly into the D/A-converters. Thus all outputs change their values at the same time.

5.4 ARINC-429 Data Module

Our ARINC-429 interface module contains four independent receivers and two transmitters (Fig. 14). The module supports data rates of 100 and 12.5 kilobits per second and can be directly connected to the ARINC-bus. The module controls all receivers and transmitters in parallel without significantly decreasing the data rate.

5.5 Graphic Display Module

The Graphic Display Module which comes from "Parsytec" in Germany has 1 Mbyte of Video-RAM, 1 Mbyte dynamic RAM and a 32-bit TRANSPUTER (T800 or T414) as graphic generator. The Video-RAM is organized in 1024 x 1024 pixels, one byte per pixel. A colour look-up table selects 256 out of 262144 possible colours. The board has an integrated video timing generator and can be programmed to match the timing requirements of a wide range of displays.

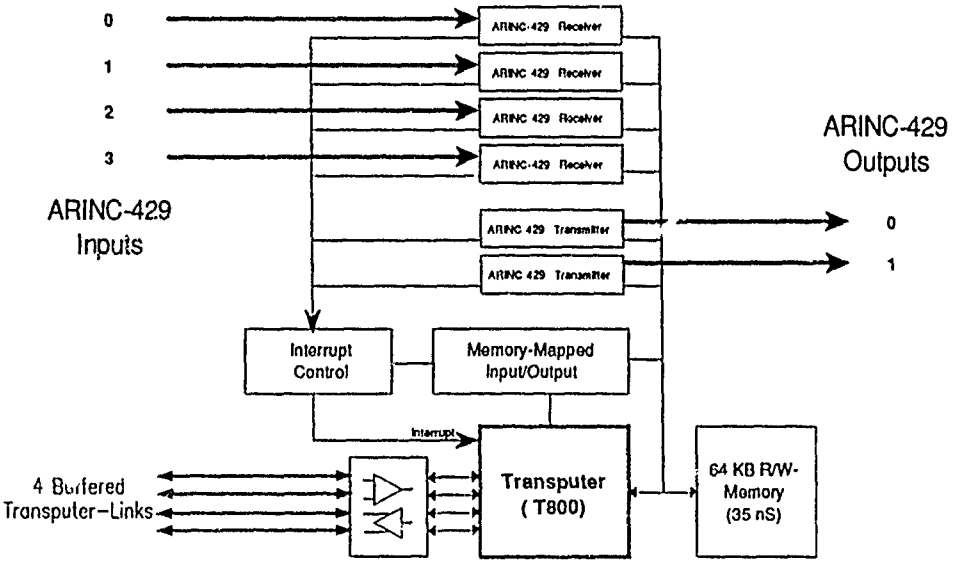


Fig. 14 ARINC-429 module

6. Flight mechanics and sensors

The mathematical model of a technical system is defined, in general, by a set of partial differential equations. These can be formulated with difference equations and solved with the help of a suitable numerical integration algorithm. This delivers the variables of state x at the mesh point $t + \Delta T$:

$$x(t + \Delta T) = x(t) + f[x(t), u(t), t]$$

Hereby u are external inputs, i.e. in the case of FALKE Shuttle the positions of the control surfaces. The measured values y can be evaluated by another transformation g:

$$y(t + \Delta T) = g[x(t), x(t + \Delta T)]$$

The stepsize of the numerical integration has to fulfill the following demands concerning a real-time simulation:

- The stepsize has to be small enough to prevent numerical errors caused by the integration algorithm.
- In the case of FALKE Shuttle, the stepsize has to be smaller than the cycle time of the on-board computer (the FALKE Shuttle flight computer repeats its algorithm every forty milliseconds).
- The required calculation time for one integration step has to be smaller than the fixed integration stepsize.

The integration stepsize is chosen to be fifteen milliseconds. This simulation is a relatively slow mechanical system for a TRANSPUTER network, compared to a helicopter rotor model [7]. Thus it is not necessary to search for a special real-time integration algorithm. The simulation of the flight mechanics and aerodynamics is taken over from an existing FORTRAN program on a high-capacity computer [8]. This includes the time-tested numerical integration method from Runge-Kutta (fourth order). This ensures that the simulation results on the TRANSPUTER network are equivalent to the results produced by the high-capacity computer. The description of the aerodynamics is based on the Aerodynamic Design Data Book (ADDB) [3] from Rockwell. This delivers the aerodynamic derivatives for the longitudinal and lateral motion of the US Space Shuttle orbiter: The coefficients for lift, drag, side force, pitching moment, rolling moment and yawing moment are composed out of several functions. Each of these functions must be interpolated out of a three dimensional table with nonlinear input values at unequal distances in accordance with the Aerodynamic Design Data Book. Input values of these tables are

- the angle of attack α in the range $-10^0 <= \alpha <= 25^0$,
- the angle of sideslip β in the range $0^0 <= \beta <= 20^0$,
- the Mach number Ma in the range $0.25 <= Ma <= 3$,
- the positions of the control surfaces, i.e. rudder, elevons, allerons and body flap.

For a complete computation of all derivatives, thirty-eight tables have to be evaluated. This requires the most calculation time in the simulation. Therefore two TRANSPUTERs are brought

into action to deliver the lift, rolling moment, yawing moment and side force coefficient to a third TRANSPUTER. This processor evaluates the drag and pitching moment coefficient. Additionally the model of the atmosphere and the equations of motion are computed as well as the numerical integration algorithm. Subsequently the variables of state for the centre of gravity are available (Fig. 15).

Another TRANSPUTER is connected in front of the three processors just discussed. It is responsible for the analogue to digital conversion of the arriving positions of the control surfaces. If required, this processor is also capable of simulating the electrohydraulics (dead time and different regulating speed for the control surfaces). This is useful when merely the on-board computer is tested. The real-time requirement is fulfilled from this TRANSPUTER as well: Its internal timer sends the new positions of the control surfaces to the next TRANSPUTER via link every fifteen milliseconds, being the impulse for another calculation. The analogue to digital conversion can be repeated several times during these fifteen milliseconds. This allows the implementation of a digital filter.

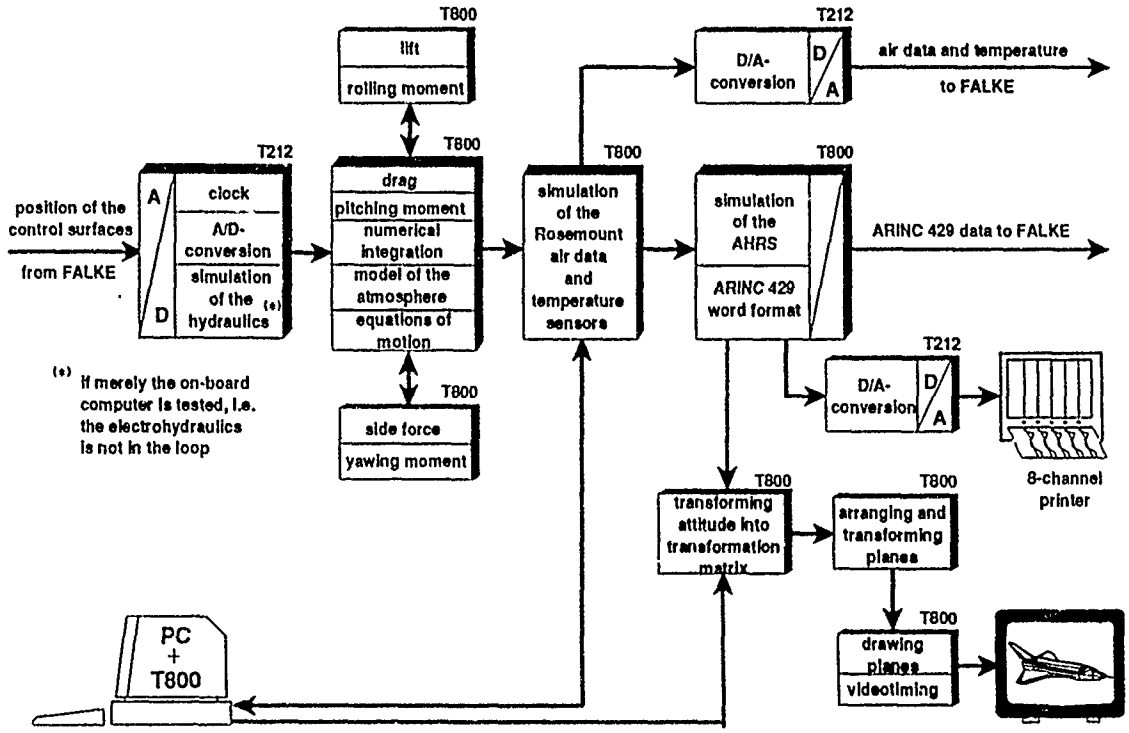


Fig. 15 System structure (Task distribution)

The variables of state are sent to another TRANSPUTER which calculates the pressures in the static tube, which will be measured by the Rosemount air data sensor during the flight. Additionally a temperature sensor is simulated measuring the temperature on the outer wall of the FALKE Shuttle. The following calculations are carried out:

- Evaluation of the approach speed in the pressure tube, in consideration of the translatory velocity and those parts of the velocity which are produced by the rates of turn.
- Transformation of the angle of attack and the angle of sideslip from the centre of gravity into the position of the air data sensor.
- Consideration of wind tunnel calibrations of the Rosemount air data sensor, in particular the angle of attack function of the Mach number.
- Correction of the impact pressure for the compressible flow in consideration of the angle of attack and angle of sideslip.
- Calculation of the specific time constant for the temperature sensor in consideration of the density and Mach number, subsequently filtering the actual temperature.
- Evaluation of a region in which the statistical faults are to be expected. Thus statistical errors can be generated and added to the output values if needed.
- Calculation of the transfer-function (output in voltage) of the transducers.

The resulting data is sent to another TRANSPUTER, which carries out a digital to analogue conversion. Thus the flow and temperature data can be transmitted to the FALKE Shuttle on-board computer via a simple plug. During the real flight this plug transmits the data generated by the Rosemount air data sensor and the temperature sensor. The processor which simulates the air

data sensor, sends the arriving variables of state to another TRANSPUTER. This one simulates the attitude and heading reference system LTR-81 from "Litef". This means:

- All arriving data is delayed for twenty milliseconds.
- The data, being valid for the centre of gravity, is converted into the position of the attitude and heading reference system.
- The data is expressed in terms of those units in which the LTR-81 sends to the ARINC-bus.
- Status indicating values describing the internal condition of the AHRS are generated.
- The data is converted into the ARINC 429 word format. Every four milliseconds eight new ARINC words are sent onto the bus, after evaluating out of a table which words have to be taken. This imitates the processor of the LTR-81 exactly.

The ARINC 429 words reach the FALKE Shuttle on-board computer via another plug. During the real flight this plug transmits the data generated by the attitude and heading reference system. This allows an easy change from the last hardware-in-the-loop simulation to the real flight conditions; just the three plugs with the interfaces to the real-time simulation have to be undone. Instead of the simulation computer, the Rosemount air data sensor and the attitude and heading reference system have to be connected to the flight control computer.

All other parts of Fig. 15 are used for the analysis of results, respectively for the control of the simulation. The eight-channel-printer is connected with the TRANSPUTER, simulating the attitude and heading reference system, via another TRANSPUTER capable of a digital to analogue conversion. This printer is useful for the presentation of the variables of state and other values describing the simulation. The development of a special simulation run can therefore be examined at once. Beneath the printer three further TRANSPUTERs can be seen, which are connected to a monitor. On this monitor a three dimensional view of the FALKE Shuttle is shown, describing its position during the test in real-time. This results in a very expressive presentation of the response of the flight vehivle to the maneuvers with respect to a variation of some parameters.

Finally in the bottom left part of Fig. 15 a personal computer (PC) with a built-in TRANSPUTER, the so-called host TRANSPUTER, completes the hardware. This processor is the interface between the PC/user and the TRANSPUTER-network needed for the simulation. A change of parameters , as for example the initial conditions, is transmitted to the real-time simulation in this way. In the other direction, output status values from the simulation reach the PC. This is necessary, for example, when the angle of attack leaves the range of the tables. Additionally, if needed, all variables of state can be transmitted to the PC during a simulation run. Afterwards they may be analysed off-line.

7. 3d real-time display

We would like to view the results of a simulation as quickly as possible. In this way we could detect errors or the influence of parameters much more easily. Since the simulation runs on a network of TRANSPUTERs, only one processor has a communication channel to the screen of the host. This communication is too clumsy and too slow to be used as an online display. On the other hand, it would require too much memory to store all important values at every time step.

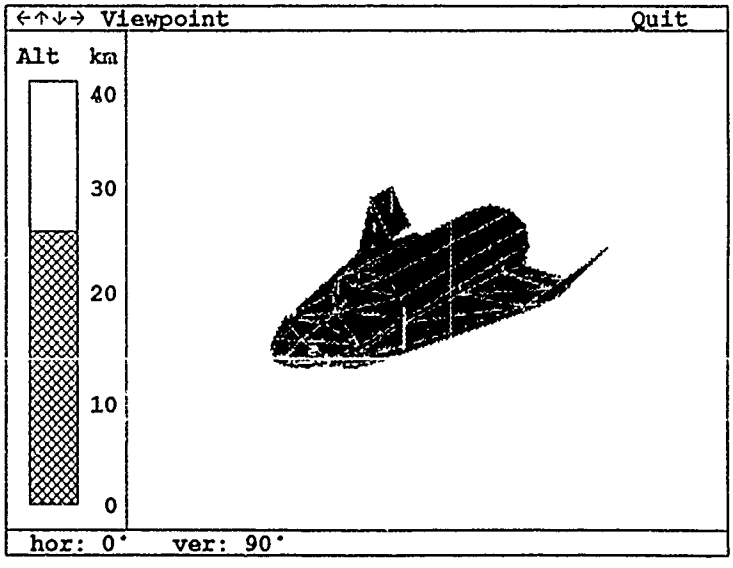


Fig. 16 A sample display (PC-version)

For this reasons we have developed a realtime display. This program also runs on a network of TRANSPUTERs that is connected to the simulation network. The simulation only sends the data to the display network and is not slowed down by storing or handling information.

The realtime display shows the flying object (in this case the FALKE Shuttle) as it moves according to the flight parameters the simulation calculates. This display allows an online valuation of the simulation, since responses to maneuvers can be seen very well. A sample screen is shown in Fig. 16.

A similar display is used during the real flight of the FALKE Shuttle. Since the FALKE Shuttle is sending down the height, the yaw, pitch and roll angle, we can again display the FALKE Shuttle as it is flying. This display is implemented on a Personal Computer. In this way it can be used on most computers with several graphic cards.

Both systems have helped much in the testing and realization of the hardware-in-the-loop simulation.

7.1 Transformation pipeline

All transformations and coordinate systems are written in homogeneous coordinates. Thus, all transformations are represented by 4 by 4 matrices. The object (in this case the FALKE Shuttle) is represented by a set of plane convex polygons. The polygons are transformed using a pipeline of coordinate systems. The object itself is described in the Model System. The World System represents the environment, i.e. the height or movement over ground, perhaps a reference grid or other objects. The View Reference System is chosen such that the viewpoint is situated on the z-axis and the viewplane is the xy-plane. Using projection, a part of the viewpyramid is transformed into a standard cube, described in the Normal Reference System. From here, transformations to various devices may take place. Here, the xy-plane is mapped onto a screen window. The coordinate systems are shown in Fig. 17.

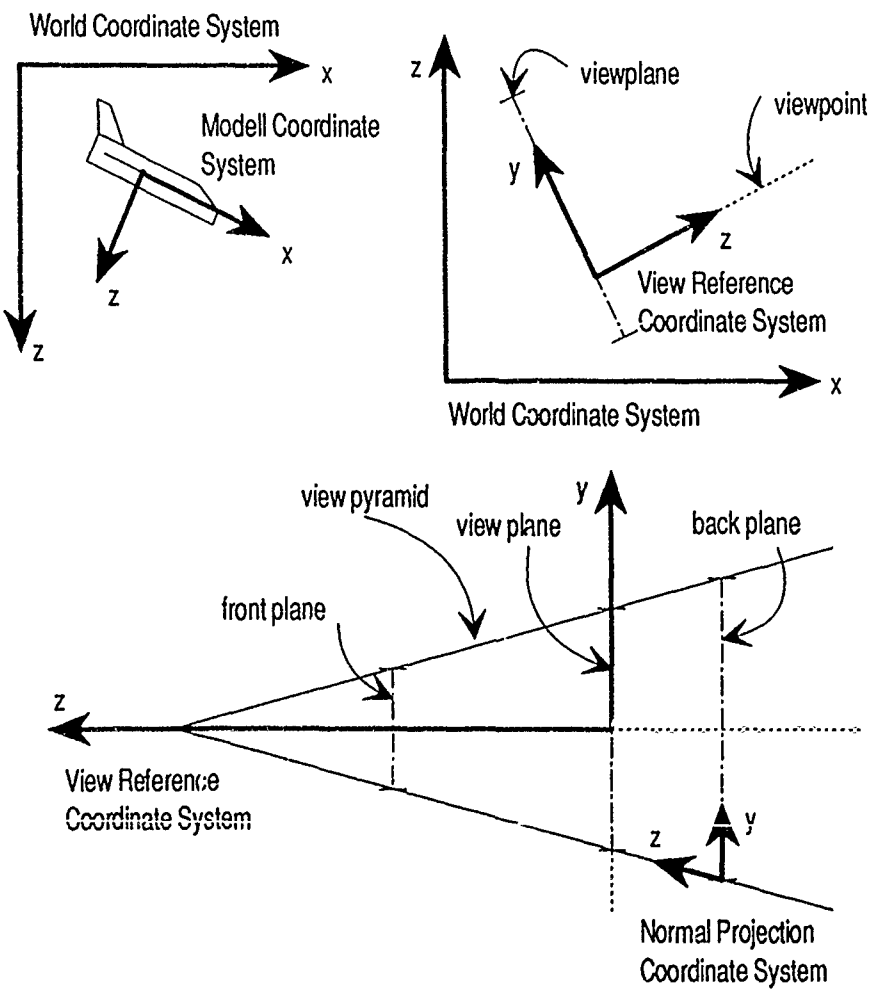


Fig. 17 The display transformation coordinate systems

There are several display options. The object may be shown as a wire model. This is used often for the PC-based system because it is the quickest method. For this, the object coordinates only have

to be multiplied by the final transformation matrix resulting from the above transformation pipeline. The next stage is a volume model. For this, a Hidden Surface Algorithm has to be used such that only visible polygons are shown. From several possible algorithms we have chosen the BSP (binary space partition) tree method which suits parallel processing. The polygons are preordered in a tree and are painted from back to front on the screen. In this volume model, the polygons may have different colors. They are shaded according to the angle between a light source and the plane of the polygon. This is called "hard" shading, since there is a color jump at the edge between two polygons. If the color is interpolated between the edges of the polygons, we get a "soft" shading model, where there is a continuous color distribution. This soft shading is a very time intensive algorithm, since every pixel has to be computed and displayed alone.

7.2 Implementation on a TRANSPUTER network

The algorithm is implemented on a TRANSPUTER network in the OCCAM language using several processes. They are connected to the "outer world" with two communication channels. The first channel should be connected to the host. From here the object data is sent to the network at the start of the program. The second channel is connected to the other TRANSPUTER network that calculates the flight data. Using the first channel again, the viewpoint, the size of the object and several other parameters may be changed during the display process. In this implementation, always the full volume model, shaded hard or soft, is shown. The process network is shown in Fig. 18.

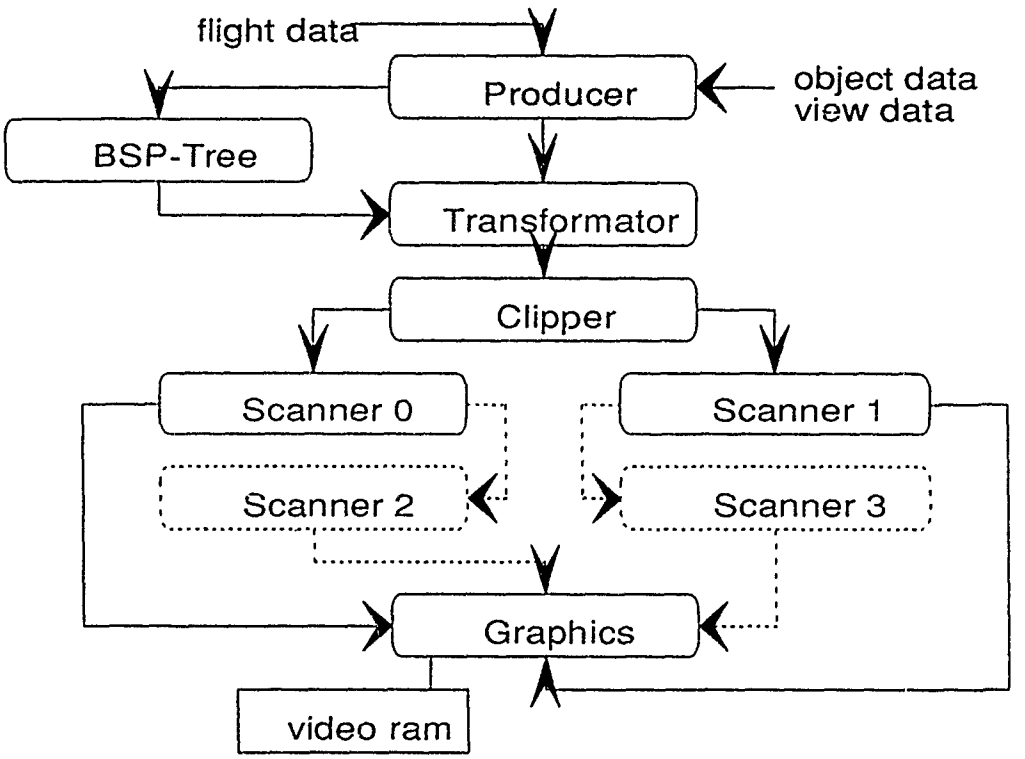


Fig. 18 The realtime display process network

The Producer process handles and distributes the object data, calculates the transformation matrices and handles the incoming flight data. The simulation network calculating the data and the display network may work completely asynchronous. The BSP-Tree process orders the polygons according to the viewpoint and sends them in the proper order to the Transformation process. Here the polygon coordinates are transformed into the Screen Coordinate System.The Clipper process clips the incoming polygon data at the screen borders, such that only the visible parts are shown. It distributes the data to the two Scanner processes. These processes decompose the incoming polygons into horizontal lines. If the object is soft shaded, there is an interpolation of the colors at the edges of the polygon. These horizontal lines and their color are sent to the Graphics process. The first Scanner process handles the even numbered lines, the second the odd lines of the polygon. In this way, the time intensive part of the algorithm is divided in two. This may be expanded to s processes, such that Scanner process number n handles all line numbers $k = n \pmod s$.

The Graphics process is executed on the only TRANSPUTER that has access to the video memory. It receives (in parallel) horizontal line data from the Scanner processes and maps them into the video memory. The disadvantage, that only one TRANSPUTER has access to the video memory, leads to a bottleneck. If hard shading is used, the Graphics process is by far the slowest. If soft shading is used, the Scanner processes are almost as slow. All other processes are faster by a

factor of 10. The disadvantage may be overcome by an update of the graphics hardware. The time usage of the processes are given in Table 1.

	hard	soft		hard	soft
Producer	2.1	2.1	Clipper	1.1	1.1
BSP-Tree	1.2	1.2	Scanner	14.7	71.4
Transformator	2.3	2.6	Graphics	25.0	27.8
	[msec/image]			[msec/image]	

The overall performance of the network: using hard shading (where technical objects like the FALKE Shuttle are displayed better) we can show up to 28 images per second on a 6 TRANSPUTER network. The time delay between receiving the flight data and showing the image is about 46 milliseconds. Using soft shading we produce up to 10 images per second with a time delay of 106 milliseconds. The image rate may be almost doubled by installing four instead of two Scanner processes.

7.3 Implementation on a Personal Computer

The algorithm is implemented on an IBM Personal Computer using TURBO PASCAL. The program has to order the polygons (if a volume model is required), calculate the transformation, transform the object data and display them on the screen. The program may display all models, from the wire model to the soft shaded volume model. It runs on every IBM compatible system with a monochrome graphics card or a high resolution color card. It may be connected to the flight data delivering system (i.e. the telemetry) through the serial port of the computer. Depending on the computer, the graphics card and the model displayed, we can show from 1 to 25 images per second. Some typical values are shown in Table 2.

PC	MHz	Copr	Card	wire	vol
386	33	yes	VGA	26.9	6.7
386	25	no	VGA	8.4	3.1
AT	16	no	Herc	3.5	1.3
AT	12	yes	EGA	7.3	1.6
AT	10	no	VGA	2.9	1.0
AT	10	no	Herc	2.3	0.9
AT	8	no	VGA	2.4	0.9
				[images/sec]	

The system can also be used to view flight data stored in a file that was produced by a former simulation on a VAX machine or IBM mainframe computer.

8. Results

A flight of the FALKE Shuttle has not been carried out until now, the same applies to a hardware-in-the-loop test of the complete system (as of: February 1990). But the on-board flight control computer has been tested at the DLR Institute for Flight Mechanics in Braunschweig. The main purpose was to find out whether the controller works stable and efficient. The block diagram of this test can be seen in Fig. 19. The electrohydraulics gets its set-point encoded as a current, but the simulation expects a voltage. Thus a current-to-voltage converter is needed inbetween. Further electromagnetic interferences on all analogue signals can be expected. Therefore a lowpass filter is needed between the simulation and on-board computer. Between this flight control computer and TRANSPUTER network of the real-time simulation three interfaces are placed:

- Commanded control surface positions are transfered to the simulation (via a current-to-voltage converter).
- ARINC words generated by the simulation system are transfered to the on-board computer (via an electronic box).
- Analogue Rosemount air data signals and the temperature value generated by the simulation system are transfered to the flight control computer (via lowpass filters), as well as the effective control surface positions.

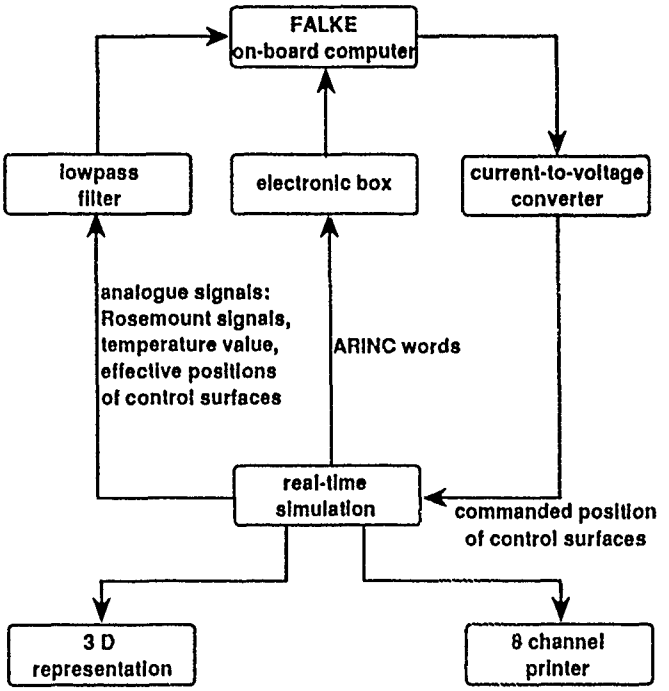


Fig. 19 Block diagram for the on-board computer test

The actual test program, including the variation of several parameters, could be accomplished within one week. The real-time simulation on the TRANSPUTER network meets all the needed requirements discussed in the second chapter. In particular it turned out to be very advantageous, that the interfaces between FALKE Shuttle and simulation could be tested separately without great effort before the actual test program was started. Concerning the on-board computer, the following statements were made [9]:

- The sequencer works properly, i.e. controller, maneuvers and recovery system are switched on and off correctly.
- The controller works sturdily in most cases.
- A rate of turn during the release of FALKE Shuttle from the balloon is compensated quickly after the controller is switched on for the first time.
- An additional stimulation of the flight dynamics is regulated quickly.
- A raising of the delay time of the sensory units (AHRS) - e.g. forty milliseconds instead of twenty milliseconds - is non-critical.
- A reduction in the control surface effectiveness is tolerable in a considerable scope; the same applies to the variation in the moments of inertia and the mass.
- A displacement of the centre of gravity is non-critical within few centimetres.
- The wind tunnel measurements in the DNW (German Dutch wind tunnel) indicate that the pitching moment coefficient is obviously higher than the value described in the Aerodynamic Design Data Book [3]. In the test a higher pitching moment leads to the situation that the prescribed five kilometre flight test region of FALKE Shuttle cannot be maintained during the mission.
- The controller is switched on for the first time when the Mach number reaches Mach = 0.25. This value is computed from the Rosemount voltages, being very small at high altitudes. Thus an error of one LSB (least significant bit) in the analogue to digital conversion leads to large deviations. It can be seen that the controller is activated very late. When the controller is activated after a maneuver has been carried out, the safety load factor can be exceeded and thus the recovery system is deployed.
- The set-point for the pitch angle is Θ = -85 degrees. When FALKE Shuttle reaches a pitch angle of Θ = -90 degrees, the system becomes unstable.
- The telemetry could be accelerated during the tests by neglecting some of the less important channels.

On the basis of this experience, some important recommandations before the actual flight test could be given [9]. The simulation results will contribute to a lower risk and more efficient FALKE flight test programme.

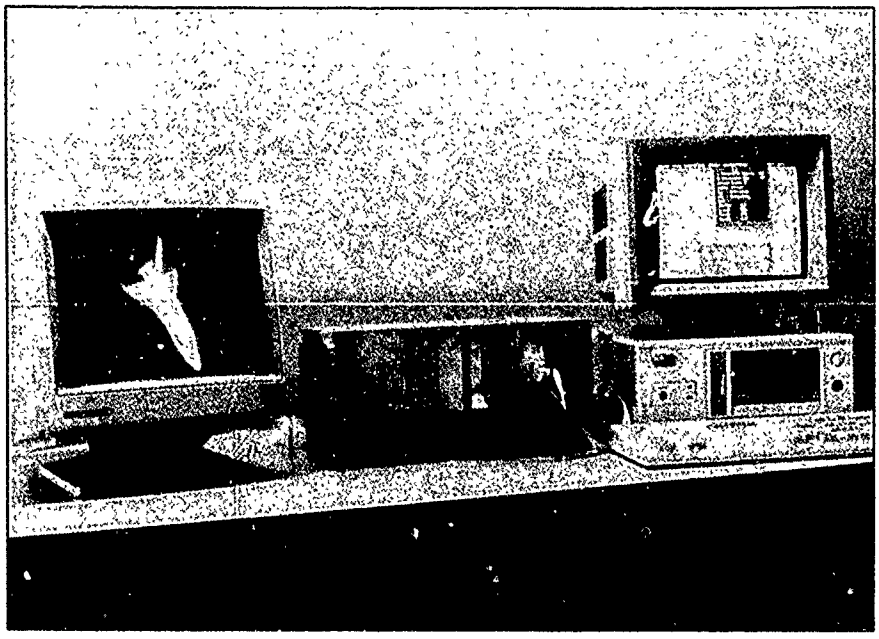


Fig. 20 The complete simulation sytem

9. Conclusions

A new technology was demonstrated for the design of real-time simulation systems. The consistent mapping of real world processes into a corresponding hardware-software system based on TRANSPUTERs and OCCAM provides a clear and more flexible data processing structure with improved real-time capabilities. Programming of the processes is achieved in a high level language (like Pascal or C) which incorporates the methods for parallel programming. Software development is independent of the amount of processor modules to be integrated into the target system. Only logics and physics of the real world processes must be met by the software.

The discussed real-time simulation system clearly demonstrates the advantages of OCCAM-software and TRANSPUTER-hardware as a suitable combination for real-time simulations. Such systems have modular structures, which are capable of getting extended, and clearly defined interfaces connect parts of the software. Such a TRANSPUTER network can be directly linked to test objects via suitable interfaces. Hence the complete system is mobile and in-the-field-tests become possible. There is no difference in software development for non-real-time and real-time applications. Finally, on-line graphic data analysis is easily implemented using the same technology.

10. References

1. Parameter Identification. AGARD Lecture Series No. 104, 1979.
2. N.N.: FALKE Informationsbroschüre der OHB System GmbH.
3. N.N.: Aerodynamic Design Data Book. Vol. 1, Orbiter Vehicle STS 1 Rockwell International, Contract No. NAS 9-14000 (1980).
4. D. Fischenberg: Auswertung von Windkanalmessungen an der FALKE-Space-Shuttle-Konfiguration. DLR-IB 111-89/32, Braunschweig 1989.
5. C. A. R. Hoare: Communicating Sequential Processes. Comm.ACM 21 (8), 666-677 (1978).
6. N.N.: IMS T800 TRANSPUTER. INMOS 72 TRN 117 02, April 1987.
7. G. Lehmann, C.-H. Oertel, B. Gelhaar: A new approach in helicopter real-time simulation. Fifteenth European Rotorcraft Forum, Paper No. 62, Amsterdam 1989.
8. D. Fischenberg: Projekt FALKE Flugbahnsimulation und Identifizierung. DLR-IB 111-89/01, Braunschweig 1989.
9. C.-H. Oertel: Test des FALKE-Bordrechners (20.11.89 - 7.12.89). DLR-IB 111-90/03, Braunschweig 1990.

COMPUTER GRAPHICS IN HARDWARE-IN-THE-LOOP MISSILE SIMULATION

by

Dr. B.J.Holden
Missile Simulation Branch (Code 3914)
Naval Weapons Center
China Lake, California 93555
United States

Summary

Performance of computer graphics workstations has increased dramatically in recent years. These machines are currently being applied to a wide variety of scientific and engineering problems, and can be used very effectively in hardware-in-the-loop (HWIL) simulation.. This paper briefly explains the architecture of these machines and summarizes their current capabilities. It then discusses the application of graphics workstations to simulation visualization and to the very difficult problem of Computer Generated Imagery for HWIL simulation of imaging missile systems.

Introduction

Rapid advances in integrated circuit technology in recent years have brought astonishing improvements in all forms of digital computation, but nowhere is this more evident than in the area of computer graphics workstations. The insatiable thirst for graphics compute power, brought on by the realization of just how valuable an analytical tool visualization is, has caused a rapidly expanding and extremely competitive market, in which the performance of graphics workstations in the $100,000 range has increased by a factor of 100 in the last five years. These powerful yet relatively economical machines have proven extremely beneficial to a wide variety of users. They are finding applications in molecular biology, computational fluid dynamics, atmospheric research, commercial animation, medical imaging, geophysics, image processing, aircraft simulators and a host of other fields. They are also bringing valuable new tools to the Hardware-in-the-Loop simulation community. As the focus of our efforts in missile seeker development shifts to imaging systems, these tools will become essential for image processing, image analysis, visualization, and generation of realistic target and background scenes for laboratory testing of imaging seeker hardware.

In this paper I will discuss three aspects of computer graphics -

What computer graphics workstations can do;
The use of computer graphics in HWIL simulation visualization;
The use of computer graphics in HWIL target generation.

What Computer Graphics Workstations Can Do

Most of the new graphics workstations have many features in common. They utilize very fast state of the art chip sets including floating point units. In many cases these are RISC processors. RISC (for Reduced Instruction Set Computer) processors have become very powerful and very popular in recent years. They were designed on the premise that out of a computer's total instruction set most application programs use only a small subset of simple instructions most of the time. A RISC designed processor is optimized to run these simple instructions very fast, with the goal of having many of them execute in a single clock cycle. With clocks running at 20 to 30 Mhz, this has resulted in processors running at 15-25 MIPs (millions of instructions per second). This design philosophy has been very successful and many of the fastest workstations on the market today are now RISC design. A comparison of relative speeds on a particular application benchmark from the Naval Weapons Center (NAVWPNCEN) for various common computers is shown in Fig. 1. These figures should be not be interpreted as being representative of the overall performance of these machines on a large range of application programs. However they do

indicate that the RISC machines are very fast, and on particular applications can compare very favorably with large computers. The Silicon Graphics, which uses the R3000 RISC chip set from MIPS Corporation does very well, considering its price.

ANN Benchmark Program
Relative Speed Measurements
Forward Propagation Rate (cycles/sec)

Platform	FORTRAN	C	Price
Cray XMP (1-Processor)	4,671 (Vectorized)	305 (Non-vectorized)	$5 M
SGI 4D/240GTX (4-Processor)	2353 (w/ 4 Processors)	227 (w/ 1 Processor)	$160K
SGI 4D/70GT	512	110	$100K
VAX 8530	174	44	$200K
Macintosh II	10	7.3	$5K

Fig. 1 Relative Processing Speeds

Many of these workstations are now being offered as multiprocessor systems. This gives the user the option of building a more powerful system by plugging in additional CPU boards. Some offer optimizing compilers which automatically split an application program to run on multiple CPU's.

Most of these systems have a Unix based operating system, with symmetric multiprocessing for the multiprocessor versions, and are built around a VME I/O bus. Designing systems for this open architecture has been the key to the incredible pace of performance escalation in the workstation market. Since the VME bus provides the backbone of the system, new processor and graphics boards can be plugged in as they become available. This allows extremely rapid utilization of new technology without having to design an entire new system.

Some graphics workstations use traditional CPU's in a fast parallel architecture to do the graphics computations, while others do the graphics processing on specialized hardware. A block diagram of a system that uses special graphics hardware, a Silicon Graphics 4D/70GT, is shown in Fig. 2. This is one of the graphics computers that is in use in the Simlab and will be used as an example in this paper.

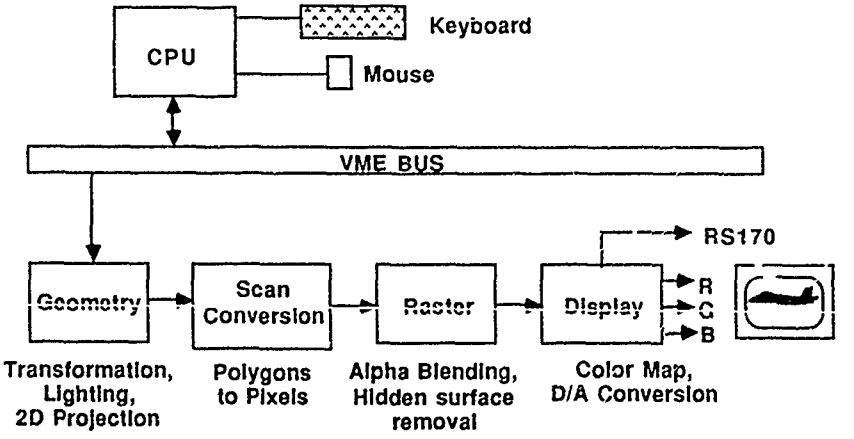


Fig. 2 Silicon Graphics 4D/70GT Block Diagram

With this architecture, the graphics program runs in the CPU, which then feeds object descriptions to the graphics hardware for rendering. Rendering can be defined as the operation of transforming a three

dimensional scene, with mathematically described objects and light sources, into a 2-D projection on a display as it would be seen by an observer from a defined perspective. A machine such as the Silicon Graphics is designed to do this operation very fast. The hardware performs coordinate transformations and graphics operations to calculate specific color values for each of 1.3 million pixels (picture elements) on a 1280 by 1024 pixel display.

The software representation in the graphics computer takes the form of graphics library commands (Silicon Graphics supplied software) which operate on geo-metrically defined objects. An object is constructed of a set of facets, or polygons. Each polygon carries information such as the 3-D world coordinates of its vertices, a surface normal vector for lighting calculations, color, and surface properties.

The geometry subsystem rotates, translates, and scales each of the 3-D vertex coordinates and transforms the surface normals. It also does lighting calculations with multiple light sources, based on the surface properties and on the relative positions of light source, polygon surface normal, and viewpoint. Finally, it projects the lighted polygons onto the 2-D x-y display screen space, while retaining a z, or depth value, for use in hidden surface removal. To support these compute intensive operations in real time it contains 5 parallel floating point processors, each running at 20-30 MFLOPs (Millions of Floating Point Operations per second).

The scan conversion subsystem reduces the projected vertex data to pixels. It computes an RGB (Red-Green-Blue) color value for each pixel in the display by interpolating the color values at the vertices along each raster scan line. The raster subsystem does hidden surface removal, and also blending and texturing functions on a pixel by pixel basis, as each pixel is written to the frame buffer. The display system reads the pixels from the frame buffer in a horizontal scan format and sends each of the three color components through a Digital to Analog converter and then to the display monitor.

This dedicated hardware can produce extremely fast graphics. The new multiprocessor versions of the workstation can dedicate one CPU just to feeding the graphics pipeline. The time required to render a scene depends on the total number of polygons it contains, and the rate at which a machine can render transformed, lighted, and smooth shaded polygons with hidden surface removal is considered a general figure of merit for graphics systems. These performance figures have been improving rapidly. Fig. 3 compares the performance figures for the last three generations of Silicon Graphics machines.

	1987 4D/70GT	1988 4D/240GTX	1990 4D/240VGX
Compute Performance			
CPU Clock	12.5 MHz	4x25 MHz	4x33 MHz
Integer Performance	10 MIPs	80 MIPs	100 MIPs
Floating Point	1.1 MFLOPs	16 MFLOPs	20 MFLOPs
Graphics Performance			
Polygons (Z-buffered, lighted, shaded, perspective)	120,000	135,000	1,000,000
Frame Buffer Pixel Access	1 million/sec	8 million/sec	18 million/sec

Fig. 3

Although improvements are being made in the software tools to utilize this powerful hardware, constructing the graphics models is still a very difficult and time consuming procedure. For HWIL simulation, detailed models of appropriate targets such as airplanes and ships must be built. As discussed earlier, this is commonly done by breaking the model down into many small polygons, each of which is described by the three dimensional coordinates of its vertices and by its surface properties, such as color and texture. The more detail that is required in the model, the smaller and more numerous the polygons must be. For a very detailed model the polygon list can become very large, and there is no flexibility to change the detail of the model as a function of its apparent range (size on the screen). This becomes a significant disadvantage at runtime when the graphics engine must process every polygon in the list, even if it becomes vanishingly small in the displayed image.

Another method is to define the model in terms of parametric surfaces. There are various techniques to do this, such as with Bezier or B-spline surfaces, but the general idea is that the surface of

the model is approximated by a set of smooth, curving, surface sections, or patches, each of wnich is defined by a set of polynomials. Ultimately, these mathematically defined polynomial surfaces must be broken down into small, flat polygons to be rendered by the graphics engine, but there are many advantages to this representation as compared to the polygon method. A curved surface on the target can be more accurately modeled, and the amount of data needed to represent the model in the form of polynomial control points is much less than for polygon vertices. However the major advantage comes at runtime because the number of polygons that each patch is subdivided into can be varied as a function of the size and orientation of the patch as displayed on the screen. This means that the number of polygons, or detail, in the model can be automatically varied with its apparent range, and the graphics engine does not waste time computing things that won't be seen.

Software tools to help the user build target models using parametric surfaces are being developed, but the task is still a very difficult one, particularly for complicated targets such as large ships. Another aspect of the problem is to define appropriate surface properties for each polygon or patch. For operation in the infrared portion of the spectrum the computer models must be derived from fundamental physics that includes both the reflective properties of the surface and its thermal emissivity properties, and validated against calibrated imagery from imaging IR sensors.

One of the most challenging aspects of computer image generation is the modelling of backgrounds. Sky, earth, and sea backgrounds each presents its own set of problems. Sky backgrounds must be able to present clouds. Earth backgrounds must reasonably model trees, mountains, and ground clutter. Sea backgrounds need wave action, whitecaps, and reflections. In general, natural objects do not lend themselves well to polygonal representation. Many ingenious techniques such as fractals and texture mapping have been devised to deal with this problem. Fractal geometry is a mathematical technique for generating models of irregular or fragmented objects by doing recursive transformations on a curve or surface, each of which increases the amount of detailed features on the surface. With this technique the amount of detail in a natural object can be varied as a function of apparent range. Texture mapping effectively paints a predetermined 2D image over the surface of 3D polygons. This can lend dramatic realism with a minimum of computational effort, such as painting an orange peel texture onto a smooth sphere.

The computer graphics techniques such as those mentioned above have long been used to render realistic single images, but in order to be of use for CGI (Computer Generated Imagery) in closed loop HWIL simulation the computations must run fast enough to execute in real time, i.e at the frame rate of the system under test. Each new generation of graphics engine is able to implement more of these techniques in hardware, thus increasing our ability to generate realistic images. Current machines are capable of doing transparency, texture mapping, anti-aliased edges and simple atmospheric effects in real time. The demand for photorealism in computer graphics is being driven by requirements from many applications, most of which are in the commercial sector. Because of this demand we can expect the hardware and software tools for generating photorealistic realtime graphics to continue to improve at a substantial rate

HWIL Simulation Visualization

Scientific visualization has become a very common phrase in recent years. Originally it was associated primarily with supercomputers since they were the only processors fast enough to produce real time graphics. As the power of lower priced computing platforms increased, the use of graphics to assist scientists and engineers has proliferated and spread into every aspect of science and technology.

The key to the effectiveness of visualization as a means of analyzing complex systems is that it can focus the vast power of the human visual system onto the problem. There is no computer that can approach this capability. The human visual system has the ability to assimilate huge masses of data and instantly distill it down to a few significant elements; to absorb the collective effect of an entire scene, or to discard most of the information and focus on a single item of interest; to immediately recognize a spatial or temporal pattern or to detect a small variation from the expected behavior.

Our objective in introducing visualization to the NAVWPNCEN Simlab was to bring this tool into the HWIL simulation world. In the Simlab there is a continuing effort to promote not just a HWIL facility, but to create an "Engineer in the Loop" environment, which provides the engineer/analyst with the best possible tools to simulate, evaluate, and interact with the system under development in real time. This affords the engineer an opportunity to gain insight and experiment with new ideas, rather than just gathering reams of plots and numbers and carrying them away.

A typical Simlab HWIL workstation is shown in Fig. 4. The Applied Dynamics AD100 simulation computer gives us the speed to run the simulations and software support to interactively change simulation parameters at will. The addition of the visualization completes the picture by greatly improving the analyst's ability to observe and evaluate the system behavior.

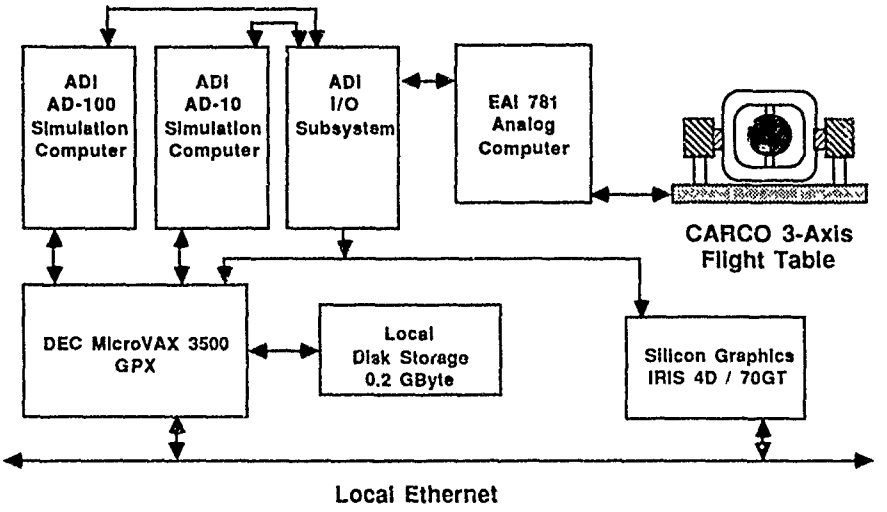


Fig. 4 SIMLAB Workstation

In the NAVWPNCEN Simlab the realtime visualization was accomplished by building a data link between the AD100 and a Silicon Graphics 4D/70GT, as shown in Fig. 5. The data link was designed to use the Sense and Control lines from the AD100 I/O system, a DR11W emulator on the Silicon Graphics, and a FIFO (First In First Out) buffer that was designed and built in the Simlab.

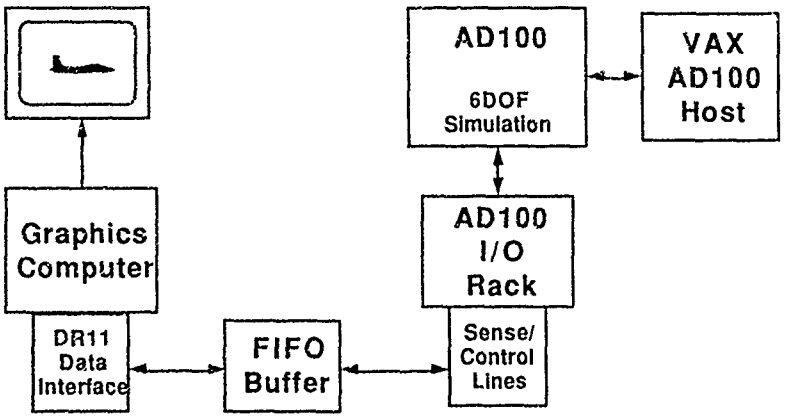


Fig. 5 HWIL Visualization with Realtime Data Link

The Sense and Control lines are a standard option for the AD100 I/O system. They are digital input and output channels which allow 16 bit parallel digital words to be passed from the AD100 to the external world at the rate of 10 MHz. The DR11W, which plugs into the Silicon Graphics VME bus, is a controller that can pass data between the external world and VME bus in either single word or block mode transfers of 16 bit digital words.

The FIFO buffer between the AD100 and DR11W is not a standard random access memory, but instead works like a huge shift register with data going in one end and coming out the other. It does the required signal level translation between the two machines, as well as buffering the data to give some "elasticity" to the data link. The FIFO buffer is 16K words deep and the AD100 can be writing into the top of the buffer while the DR11W is reading from the bottom. This buffering is an advantage when trying to synchronize data transfers between two machines with different data rates and also simplifies the communications protocol.

3-Dimensional models of all objects in the scene, including missile, target, countermeasures and background, are initially constructed as described in the previous section. Graphics library routines are used to program the relative motion of these objects as a function of position values computed by the AD100. During operation the AD100 sends the required state information via the data link and the Silicon Graphics renders and displays the correct 3-D scene in real time. The link is fast enough to send information every frame, which may be desired if the link is also being used for data logging. However it is usually not necessary to update the display at the frame rate of the simulation. A smooth motion display can be achieved with update rates of about 15 Hz.

The graphics display can be enhanced with as much realism and detail as the graphics programmer is willing to provide. The encounter can easily be rendered from any perspective, such as from the missile, from the target, or from any fixed or moving position in space. A particularly effective display results from a perspective just behind and above the missile. The ability to "watch" the missile fly in this manner gives the analyst vastly improved insight into stability and control problems.

The data used to drive the graphics can be stored and used later to replay the simulation run under different conditions, such as slower or faster speed, or from a different perspective. The graphics output can be converted to NTSC or PAL video format for documentation or presentation purposes.

The same technique can be used to visualize other simulation data. The display can be customized to include additional graphics windows which can simultaneously display trajectory information, flight data, or missile specific parameters. Simulation data output in the form of plots, strip charts, and data tables are still essential for detailed analysis but are tedious, time consuming, and require substantial knowledge and experience to interpret. On the other hand data output in the form of realtime graphical depictions of flight instruments, head-up-displays, or tracking gates can be rapidly interpreted and immediately correlated with the graphically observed system behavior. This capability, when combined with the AD100 interactive control of simulation parameters, makes an extremely powerful developmental tool.

HWIL Target Generation for Imaging Systems

The rapidly increasing capabilities of these graphics workstations present the possibility for solving an increasingly thorny problem. It is apparent that the next generation of missiles is shifting to one form or another of imaging seeker, such as staring focal plane arrays or image scanning systems. Many of these seekers will have detectors in more than one spectral band. In order to support the development and testing of these systems in the HWIL laboratory we must develop the hardware and software tools to provide a realistic target environment and the means to analyze both the fidelity of the target model and the performance of the missile system in that environment. There are many issues involved in solving this problem, and it is apparent that computer graphics must play a significant role in any solution.

As an imaging missile system progresses through its design cycle from initial concept to final test and evaluation we are constantly faced with the requirement of presenting its seeker with some kind of visual scene Initially this scene may be composed of simple geometric patterns for algorithm research. In final HWIL testing a detailed scene with realistic target and background is required. At each step of the cycle it is very important that we be able to precisely control and reliably reproduce the image that is presented to the seeker elements. This is essential to proper interpretation of the test results. If you don't know really know what went in, you don't really know what the the system was tracking on. Further, for closed loop testing it is necessary that the presented image change each tracker frame to properly represent what the seeker would see as the missile flies down its trajectory and reacts to guidance commands

These requirements demand speed, reproducibility and flexibility that can only be met by using computer generated imagery. The remainder of this section will be devoted to discussing the general sequence of steps involved in the design of an imaging missile system and the how computer graphics tools will be used in each of those steps. It is not intended to be an exhaustive list, but rather an overview of the primary issues to illustrate the central role that computer graphics will play in the design process.

The first step in the design cycle of an imaging missile system should be to build a capability for developing tracking algorithms. This implies building a strong knowledge base in image processing and doing a rigorous analysis of the feature space of the types of images that the missile will be designed to operate against. Real world digitized video frames of appropriate targets and backgrounds should be analyzed to determine what features such as intensity distributions, spatial frequencies, edge information, etc. are available that let us discriminate target from background. Based on this analysis a judgment can be make as to what combination of a) types of features present in the target/background image, b) image

processing operations, and c) available signal processing technology will yield the most effective, yet physically realizable acquisition and tracking algorithms for that missile's operating requirements.

A fast computer graphics workstation with attached special image processing hardware is the best platform for this type of work. The image processing can be done in software, but is much faster with dedicated hardware boards including digitizer, frame store, pipeline pixel processor, and real-time convolver. These boards can be plugged directly into the VME bus of the graphics computer and come with user software routines.

This graphics workstation makes an ideal tool for interactive engineering development. After an all-digital simulation of the tracker is developed it can be tested against various combinations of real world video and computer generated patterns to evaluate the robustness of the algorithms. The ability of the engineer to graphically observe the tracker performance on a frame by frame basis, make changes and immediately see the visual results of those changes results in a very efficient development process.

The next step should be to create realistic computer generated target and background images. Ultimately the effectiveness of the CGI approach hinges on our ability to do this well. This is an area that will require a large investment of time and effort. There are many well known techniques for building computer models, as mentioned in the first section of this paper and these need to be applied to construction of aircraft and ship models, and appropriate earth, sky and sea background. Edges, intensity distributions and lighting conditions need to be carefully controlled to emulate the real world. Special effects such as smoke, fog, haze, shadows, transparency and countermeasures will also be needed. Each of these elements must be correctly modeled in each spectral region of interest, such as visible, and near and far infrared.

When developing CGI it is especially important to understand what the seeker and tracker will see, as opposed to what the human eye will see. Human vision is highly processed both spatially and temporally in the eye before being sent to the brain. Processing in the retina does intensity averaging, automatic gain control, edge enhancement and motion detection on the visual image. In other words, an image or video sequence that looks realistic to the eye may look quite different to the tracker. A good example of this is image dither. This is a technique used on some graphics computers whereby the image is deliberately jittered at a frequency greater than the eye can follow. This smooths the edges of a displayed object as it appears to the human observer but to the tracker the edges appear to be fluctuating wildly.

Validation of the computer models will be a difficult and continuing process. This process will be aided considerably if real world imagery of the desired targets are available. Digital frames of the computed model can be fed through the image processing operations chosen for the tracker and the resulting extracted features compared with a real world video scene of the same target at the same aspect A similar comparison can then be made by feeding the real and computed images into the all digital tracker simulation and evaluating the tracker performance in each case. After hardware is available, tracker performance on captive carry flights can be compared with performance in the laboratory against the CGI. The insight gained from this analysis can be used to refine the CGI models.

Once this done, the way is open for intensive laboratory testing of the tracker simulation and hardware. Any combination of CGI target and background can be used, including a computer generated target merged with a real world background. The simulation scenarios can be varied at will. A key element of the analysis is the ability to control every pixel in the image and to absolutely reproduce an image sequence, including edges, intensities, and lighting conditions, down to the last detail. In this aspect CGI is more effective than flight testing, since everytning that the tracker sees can be strictly controlled and the features it extracts and processes can be identified.

The next step is to integrate the computer generated imagery into the HWIL simulation using the actual missile hardware. By far the best way to do this is to present a target image of proper spectral content to the seeker. This requires both generating the image and projecting it to the seeker optics so that the image appears to be at infinity and has sufficient resolution to appear continuous. This problem is a very challenging one, particularly for a multi-spectral seeker. Much effort is now being focused on developing dynamic visible and infrared image projectors and the associated projection optics. These projectors will be driven by a fast graphics computer which construct a realistic signature in the proper spectrum to present to the seeker. This configuration is shown in Fig 6. While there are many promising approaches in development, present solutions tend to be enormously complex and expensive, particularly for the IR, and not very effective.

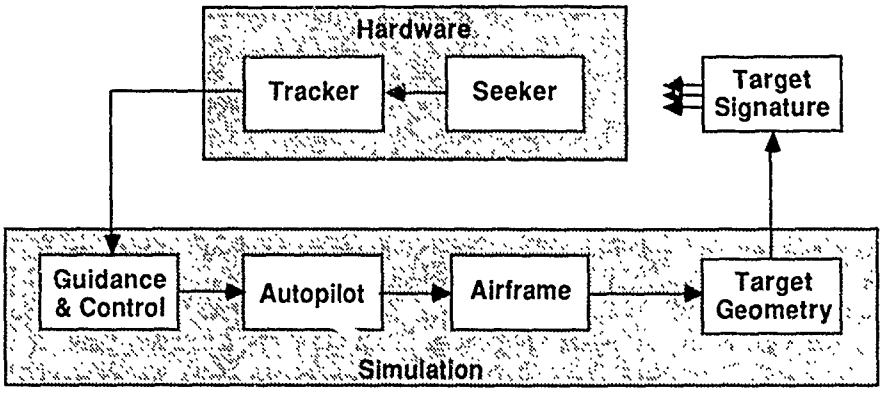


Fig. 6 Seeker Hardware in the Loop

Until this technology is much further improved, a better solution may be to treat the seeker and tracker requirements separately. Design problems associated with seekers tend to be such things as detector performance, optics, and mechanics, much of which can be analyzed in either a simple closed loop test or an open-loop environment using images that do not require specific target and background content. On the other hand, most of the design complexity of the imaging missile system is in the tracker, with its associated signal processing, embedded software, image processing algorithms and control dynamics. Thus, during the phase of development when tracker performance is the critical issue, we can avoid many of these very difficult image presentation and projection problems by injecting a simulated image directly into the tracker.

With this "tracker-in-the-loop" approach the seeker hardware is not used. The visual scene is simulated as a synthetic video image, exactly as it would appear if it was coming from the seeker, and injected directly into the tracker hardware, as shown in Fig. 7. Since the seeker is not present , a seeker model must be developed to simulate the effects of the seeker on the incoming video. Effects such as image jitter, detector noise, intensity variations due to finite detector size, and net MTF (Modulation Transfer Function) of the optics must be understood and correctly modeled.

This configuration allows testing of everything past the seeker, including tracking, guidance, and control algorithms in a closed-loop or open-loop mode. There is still a need to test the performance of the seeker-tracker combination on a flight table where seeker gyros can be exercised against a moving target and seeker-body coupling can be analyzed, but with this approach most of the development of the tracker could be done at individual tracker-in-the-loop workstations. This is especially important if a laboratory needs to provide simulation facilities for multiple missile systems.

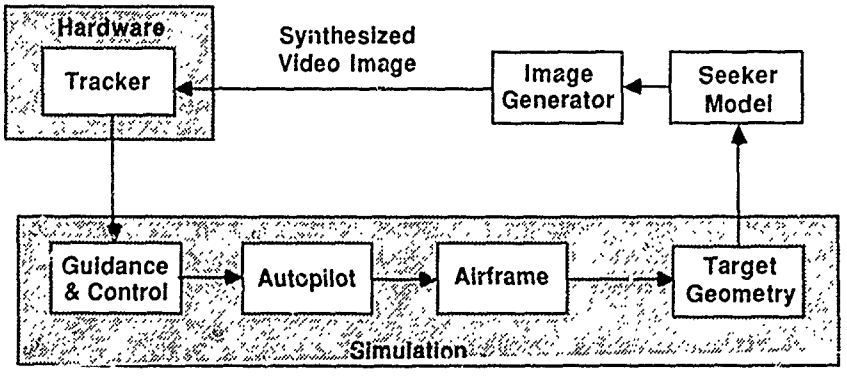


Fig. 7 Tracker Hardware in the Loop

Generation of the synthetic video image is not a simple task. It is very important to generate the video signals to look exactly the same as the video that the seeker would produce. In most cases the seeker video will be in digital format, going into the tracker in parallel digital signal lines. This means that the graphics computer used to generate the synthetic video image must have an external port to output a computed image in digital form, before it is converted to analog video to send to the monitor. In addition, the video going into the tracker must be synchronized to the missile's master digital clock. In order to accomplish this, the graphics computer must have a programmable video controller, so that the digital video can be read out of image memory in synchronism with the tracker pixel clock, and with the right number of horizontal and vertical pixels to properly simulate the actual seeker field of view. A block

diagram of the digital video port requirements is shown in Fig. 8. This capability is not available on many graphics computers which typically output only analog video in high resolution (19 inch graphics monitor), NTSC, or PAL video formats.

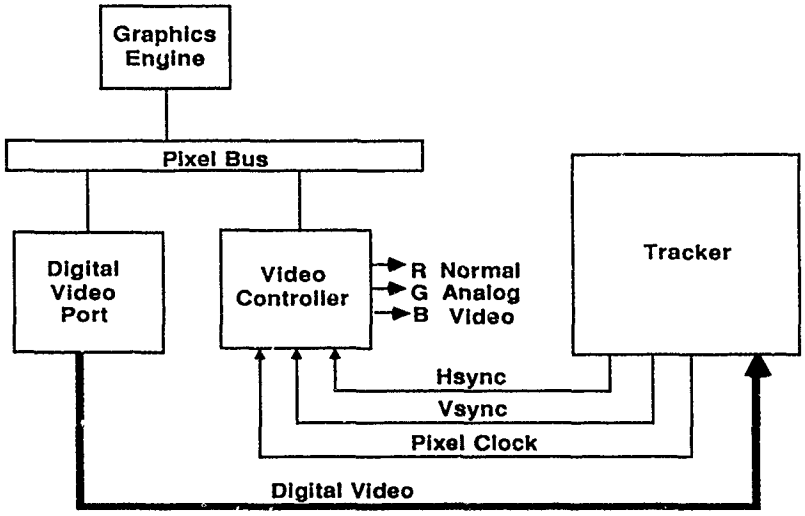


Fig. 8 Digital Video Port

Multiple digital video outputs for simulating multispectral sensors can also be generated. Since graphics computers are designed for computing Red-Green-Blue color, there are three color channels available which can be used to compute intensities in three different spectral bands simultaneously. These channels can be output separately to simulate outputs from multiple seekers in different bands.

Once the digital video connection and synchronization is accomplished the tracker can be tested in an open-loop mode against still frames, or animation loops consisting of a series of precomputed images read out of image memory. At this point the performance of the tracker hardware can be verified against the digital simulation. This is an important validation step. Since the embedded software is not usually emulated in the digital simulation, it verifies that the software is implementing the correct tracking algorithms.

Closing the loop in the HWIL simulation puts severe additional demands on the image generation system. Fig. 9 shows a block diagram of a closed loop imaging tracker simulation. An Applied Dynamics AD100 simulation computer is used to do the kinematics and aerodynamics. In addition, a real time special video processor is needed for the following reasons.

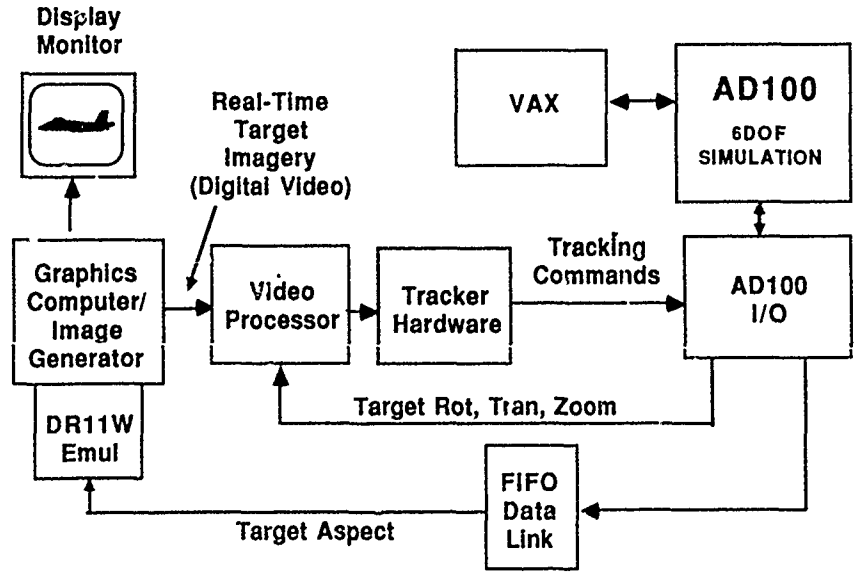


Fig. 9 HWIL Imaging Tracker Simulation

For a truly closed loop simulation the image generated in each video frame must accurately model the seeker's field of view. This means that the target position in each video frame must reflect the change in the seeker's viewpoint due to guidance commands given during the previous frame. Fig 10 shows these timing requirements. Any additional delay may introduce instability in the loop. Since it is desirable to send the video processor the latest state information available just before the start of the next tracker frame there is very little time for the graphics computer to generate a new scene.

Because of this, a special high speed video processor is required to do image translation, rotation, and zooming on the generated image. These operations, which are the extremely time critical elements of the relative target-missile motion, can be done by dedicated hardware during the blank time between images. A high speed data path from the AD100 to the video processor carries the required state information each frame.

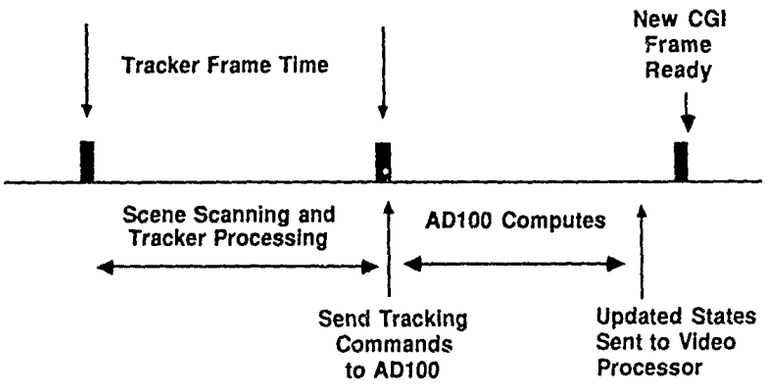


Fig. 10 Image Generation Timing Requirements

If the missile trajectory maintains a fixed aspect to the target, a single image of the target, translated, rotated and zoomed in real time by the video processor, can be used for the entire missile flight. This allows use of a single precomputed realistic target image. This approach has worked well at NAVWPNCEN.

When it becomes necessary to have a varying aspect angle to the target, as in detailed simulation of the terminal phase, then another approach is necessary. For relatively simple target models a computer such as a Silicon Graphics can render the entire scene within a few tracker frame times. Changes in aspect angle of the target are not as time critical as the other elements mentioned above, and a lag of several frames in the update rate is acceptable. A lower speed data path from the AD100 to the graphics computer can carry the aspect information. The CGI must still run through the video processor. This approach allows total closed loop operation without restrictions on missile trajectory and is currently under development at NAVWPNCEN. As graphics computer rendering speed increases, the complexity of the target images that can be generated will continue to increase. Fig. 11 shows the capabilities of currently available machines.

Measured Performance of Silicon Graphics for X29 Model

Machine	Number Polygons	Rendering Time	Achieved Frame Rate
4D/70GT	573	25 ms	30 Hz
4D/240GTX	573	10 ms	60 Hz

Fig. 11

If highly realistic imagery and varying target aspect are required, then the only solution is to bring more processing power to bear on the problem. A supercomputer such as a Cray in combination with a graphics computer could render an entire detailed target and background scene within the required frame time. This approach is also under development at NAVWPNCEN.

A third technique for doing closed loop simulation within these stringent timing requirements is to slow the tracker clock. If it can be slowed down enough to give the graphics computer time to render an entire new scene, the high speed video processor is no longer required. This is a powerful technique if the tracker can be designed to operate this way, although it does not test the tracker signal processing at its actual operating frequency. It will work with all-digital hardware, but presents problems if the digital tracker must interface with analog control loops.

Once the entire HWIL simulation is running, the closed loop tracker performance can be evaluated, the all-digital simulation can be validated, and the CGI HWIL simulation can be validated against flight tests. All of these steps in the design cycle, from basic image processing background work to final flight testing, will be very dependent on fast computer graphics workstations. They are going to be as essential to the HWIL simulation laboratory as a flight table and simulation computer. With their use in image generation, simulation visualization, and as fast compute engines, they are becoming an integral part of HWIL simulation.

MODELLING OF LAND BASED AIR DEFENCE SYSTEMS IN RESEARCH AND PROCUREMENT SUPPORT

by

K A Hurst and S Flynn
Ministry of Defence (Procurement Executive)
Royal Aerospace Establishment
Farnborough
Hampshire, GU14 6TD
U.K.

SUMMARY

This paper describes the ADVOCATE (Air Defence Verification of Options by Computer Analysis of Target Engagements) land based air defence simulation program. The key features of the simulation program are given together with a description of how weapon systems and threats have been modelled. The way in which ADVOCATE has been used is discussed and several weaknesses and strengths are identified as well as key program enhancements presently under way. The role of ADVOCATE as part of a wider weapon assessment process is also given some consideration throughout the paper.

1 THE NEED FOR FEW ON FEW SIMULATION

Major weapon procurement decisions, which invariably result in considerable expenditure, must be based on a thorough understanding and endorsement of the operational requirement, and a critical evaluation of any technical solutions which might meet this requirement. These solutions will inevitably originate from competing industrial concerns and may be variations on existing systems or completely new concepts. In all cases an assessment of how effectively a system might meet both military effectiveness and cost constraints needs to be made and it is during this process that a variety of modelling tools are required. Simulation is, except on rare occasions, the only means available for comparing candidate systems - simply because of the high cost of trials. Even if weapon trials are possible they cannot, under normal circumstances, evaluate how a system performs in a realistic military environment.

During system development, considerable use is made of simulation programs and design tools which assess the performance of individual weapon sub-systems, such as surveillance and tracking radars, missile propulsion, missile seeker and so on. This could be considered the "engineering level" of simulations where the emphasis tends to be on predicting the performance of a single weapon against a single threat. At the other extreme to this is the campaign or wargaming level of modelling, where wider military issues such as the relative benefits or balance between the various weapon systems available in a given theatre of operations can be assessed on a "many-on-many" basis. Such campaign models typically include many elements of the land and air battle, such as tanks, aircraft, helicopters and air defence assets, and reflect the interaction between them but with only a limited amount of detail in the modelling of individual weapon systems.

A third level of modelling that falls between the extremely detailed performance simulations and the wargame/campaign models may be identified, where the level of sub-system modelling is sufficient for the results to be sensitive to changes in their assumed characteristics. This "few-on-few" type of model, of which the ADVOCATE suite is an example, can be used not only to assess the relative merits of sub-system options and evaluate new system concepts but can also provide results for use by the larger scale, campaign, models as well as assist in making procurement decisions between competing systems for an existing requirement. The few-on-few simulation forms a valuable bridge between engineering and campaign modelling because of the lower levels of effort required to perform a study. This increases the utility of a program such as ADVOCATE since it can be used to provide feedback on whether particular lines of research may prove militarily useful and hence influence the direction of future work.

2 THE ADVOCATE AIR DEFENCE SIMULATION PROGRAM

2.1 Overview of ADVOCATE

ADVOCATE stands for Air Defence Verification of Options by Computer Analysis of Target Engagements and represents a collection of associated data pre-processing, simulation models and output data analysis tools, the majority of which are written in PASCAL. The main program is a critical event, Monte-Carlo simulation which permits the user to represent a wide variety of land based air defence system concepts. The program is restricted in its present form to examining the performance of ground based air defence units being used to defend point targets or an area against attacks by airborne threats, which can include fixed wing aircraft, helicopters and tactical weapons. A typical scenario might have six Air Defence Units (ADU) in fixed locations

Copyright (C) Controller, HMSO London 1990

defending a point target against 20 airborne threats with the simulation representing an elapsed real time of several minutes. The ADUs are typically deployed within a gaming area of approximately 25 x 25 km. A run would typically comprise of about 26 replications to obtain statistical results. The main features of ADVOCATE are summarised in Figure 1.

The inclusion of realistic terrain is essential when modelling short range surface to air systems because screening of a target by local terrain features and local obstacles, such as large buildings or woods, can be a major limiting factor to system performance. The terrain is derived from a digital terrain database and there is the potential for ADVOCATE to model any geographical area in the world for which mapping data exists. As well as the terrain data, which consists of a series of spot heights and an indication of the gross culture (woodland, built-up area etc) on a regular grid, there is also a local obstacle database which is superimposed on the basic terrain information. This local obstacle data is specific to a particular ADU position and contains information on the heights and positions of buildings, pylons, trees and so on.

Airborne threats are defined by type, such as fixed wing aircraft or helicopter, and by the tracks along which they fly. Tracks are specified by a series of waypoints and nominal heights above the ground. The model flies the target between waypoints using a series of straight lines and arcs of circles in plan using a realistic terrain following algorithm and taking into account the manoeuvre limits of a particular class of air vehicle. An attack phase can be defined where the target performs a "pop-up" manoeuvre prior to weapon release at a pre-defined point. Multiple threats of the same type can follow a lead target along the same track with a time delay or they can use a parallel track offset by a defined distance from the original. It is possible for threats to launch missiles which can then become separate threats in their own right. Modelling of electronic countermeasures (ECM) is at present limited although provision has been made for more detailed simulation of this important aspect. Threat behaviour does not vary from one replication to the next, although targets can be killed by missiles fired from ADUs and hence be deleted from the scenario for the duration of a single replication. ADUs themselves can be subject to attack and can be killed.

Multiple ADU deployments need to be modelled so that effects such as overkill can be accounted for, where several ADUs may engage a single target which has already been damaged and hence waste ammunition. In addition, provision has been made for ADVOCATE to model groups of ADUs which have been netted (capable of exchanging information) although this would currently require additional software to describe the information being passed. This information might include track data, IFF declarations and possibly whether a target has been previously engaged.

In any scenario the air defence consists of ADUs in fixed, pre-determined locations which cannot be varied either during or between replications. The ADUs may be of the same or mixed types so, for example, ADVOCATE can cope with scenarios where a shoulder launched system is used for point defence while a more capable, longer range, system is used to provide general area coverage. In many scenarios there are more potential ADU sites available than units to fill them and in these circumstances locations are chosen to give good all-round coverage of the defended area. The selection of sites is an important factor in modelling and is one area in which ADVOCATE is to be improved.

2.2 Structure of ADVOCATE

Figure 2 depicts the overall structure of the ADVOCATE suite of programs and gives an indication of the data required during a study and some of the output which is available. The main program, to which the acronym ADVOCATE strictly applies, is normally run many times to obtain statistical information on the performance of the ADUs. Since the behaviour of the threat does not vary between replications several pre-processors are used to calculate deterministic threat data at the start of a batch of runs, thus appreciably reducing running times. The main simulation uses this pre-processed data, along with additional data in the form of look-up tables, to simulate the air defence system under examination. The output from the simulation is in the form of a single data file which details all events which occurred during runs. Clearly for a large number of replications of a complex scenario this could involve a considerable amount of data. Post-processor analysis programs are used to extract information which provide measures of system performance relevant to the assessment criteria adopted for the study.

It is essential that ADVOCATE can accommodate a wide variety of possible sy concepts. To this end it was designed as a framework into which models of an ADU could be inserted, with well defined interfaces between elements of the simulation and a data structure to handle the exchange of data across these interfaces. This framework allows the user to add new modules to reflect new system concepts or to include greater detail in the modelling of specific sub-system operations and to build up a collection of application specific modules which can be reused. Figure 3 shows how the main program is structured in three tiers, the highest level performing the overall control of the simulation and the passing of data between modules, a second system level which defines the sequence of events which the ADU performs during an engagement and finally a sub-system level which details specific sub-systems which are contained within the ADU.

The top level of the program performs functions which are independent of the system and scenario under study and includes reading input data, module initialisation, control of simulation time, maintenance of the event list by which the simulation is progressed, sending results to the output file, control of replications and so on.

The second, system, level contains a description of all the events which occur during an engagement of an air target by the ADU together with the decisions which are taken throughout the sequence. The inclusion of logical decision processes means an ADU may perform a different sequence of events during the engagement of a target depending on the circumstances. Figure 4 is an extract from a typical event diagram for an ADU, a coded version of which forms part of an ADVOCATE system model. Some of the system models currently available simulate the following:

- command to line of sight missile systems
- active radar homing missile systems
- shoulder-launched systems in line of sight and noming variants.

There is also a system model of the threat and terrain which controls the sequencing of events in reaction to occurrences such as a target changing its status. If the target were killed for example no further events related to it would occur during that particular replication. Another example is when a weapon released by a threat impacts a user defined location, which would normally be the location being defended by the ADUs.

The third level of the main program is the sub-system level, the modules of which are called by the system models from the level above and which represent the activities which occur between events such as those shown in Figure 4. Examples of the many sub-system models which are available are:

- Surveillance
- Track formation
- Target selection
- Missile launch
- Terminal guidance
- Lethality.

The main function of these sub-system models is to calculate the outcome and duration of an activity, the results of which are passed back to the system model level and used to update the status of the ADU or threat, schedule further events and possibly cancel future events. Detailed or complex calculations are also performed at the sub-system level where possible. As indicated in Figure 3, each sub-system model can be called by more than one system model.

To minimize run times, many of the sub-system models make use of look-up tables which contain data generated either from existing data in other forms or by a variety of supporting programs, however the option is available for the sub-system to perform the detailed calculations itself. The following describes several of the existing sub-system models and how they represent some of the functions within an ADU.

Target detection begins with the unmasking of a target from behind the local terrain and the establishment of a line of sight to the ADU. The process is performed using a look-up table of single scan detection probabilities as a function of range. This is flexible enough to represent radar, electro-optical and visual detection, although it is dependent on the availability of suitable data. Track formation is normally based on the criteria of "N detections out of M scans". The term "track" in this context relates to the sensor and not to the pre-defined flightpath of the threat.

Threat evaluation and weapon assignment allocates priorities to all threats for which tracks have been formed and determines an optimum missile firing time for each. This decision takes into account the coverage diagram of the weapon system, available rate of fire and so on. This process must be regularly repeated to take account of changes in threat priorities due to targets being screened by terrain, targets being killed and the formation of new tracks. For a shoulder-launched system these rules tend to be much simpler.

Weapon flight is normally modelled by a look-up table. This look-up table contains a series of intercept times and positions which would occur if the target were to be engaged at different points along its track. Interpolation is used when the actual position of the target along its track does not correspond with a data point in the look-up table. This approach is adopted for each target track within the scenario. Target manoeuvre is taken into account automatically since the target tracks from the scenario are used to generate the look-up tables. The fact that the target tracks are pre-defined for a particular scenario means that targets cannot respond in an interactive manner to being engaged and this is one of the potential disadvantages of the present version of ADVOCATE. However it is felt that this limitation does not significantly affect the overall results obtained. The in-flight reliability of an individual missile is modelled by a single probability value. Missiles can be wasted due to not being fired after an irreversible start-up procedure, of missile gyros, thermal batteries and so on, has been initiated.

Missile lethality depending on the availability of data, is modelled as a map of kill probability over the weapon coverage diagram and hence takes into account, in a coarse way, of the relative geometry of the engagement. Whenever possible different categories of kill ought to be modelled so that the effect of only damaging a target, rather than immediately causing a visible catastrophic kill, can be assessed. This can be important because a target might be too damaged to complete its mission but still capable of controlled flight in which case it may be engaged by other ADUs further along its flightpath.

2.3 Input Pre-processors

The input pre-processors consist of a mix of ADVOCATE specific programs for generating the deterministic target data and a number of general purpose routines which have been modified to provide output which is compatible with ADVOCATE data input requirements. The relationship of the pre-processors to the rest of the ADVOCATE suite is shown in Figure 1. Pre-processors specific to ADVOCATE include:

- Target flightpath computation which subdivides the flightpath (as originally defined by the waypoints input by the user) and assigns information such as height, velocity and acceleration to each of these subdivisions. These subdivisions are also allocated the time at which they occur, measured from the start of the simulation. A terrain avoidance algorithm is used if requested by the user, provided the threat is not in its target acquisition or weapon delivery mode.

- Computation of crest lines in terrain as seen from each ADU site and modification of crest lines by local obstacles. This is used to determine when a line-of-sight exists between the target and ADU.

- Computation of line-of-sight opening and closing for each target/ADU pair. Using the output, from the previous program, a look-up table is generated for all tracks which defines when the target can be seen.

- Suppression of line-of-sight events beyond the maximum detection range of the ADU sensors.

- Blindzone pre-processor which generates events associated with entry to and exit from blind zones for each ADU/target pair. These zones are not related to terrain screening but represent limitations such as surveillance sensor fields of view, or sectors where engagements have been deliberately inhibited by the user.

The output from the pre-processors described above are a series of data records which allows the main program to determine the state of the target whenever an event occurs which is associated with the target.

- Missile flight pre-processor which constructs a look-up table which, by interpolation, can give intercept time and position, together with an indication of engagement success, for weapon launches at all points on a target track.

A number of the look-up tables used in the main simulation are generated by general purpose support programs which include the following:

Radar performance program which calculates single scan detection probabilities based on standard descriptions of the target, ECM conditions and conventional radar performance parameters.

Missile coverage program to calculate the weapon system coverage which is subsequently used in threat evaluation and weapon assignment calculations and in the missile flight pre-processor previously mentioned. Different versions of this program are available for line-of-sight and homing systems. Additional data for these programs may be required from aerodynamic prediction codes to give estimates of, amongst other parameters, missile lift and drag performance.

Lethality model which can range from simple manual calculation methods through to highly complex computer programs which model target vulnerability and warhead performance in considerable detail. However the more complex methods are normally too specialised and time-consuming to run to be appropriate to small scale studies. It is interesting to note that miss distance is not explicitly used in ADVOCATE, its value is effectively incorporated in the lethality data which is assumed for different parts of the weapon coverage diagram.

2.4 Output Post-processors

The output from ADVOCATE consists of a large file containing a log of all the events which have occurred during a run. This file is analysed afterwards to extract the required information. The relationship of the output post-processors to the ADVOCATE suite is shown in Figure 2. Post-processing can be performed using the following aids:

- A standard analysis program giving information which experience shows is useful, eg number of targets killed before and after weapon release, amount of ammunition used, overkills, interception range statistics etc.

- General purpose programs to produce graphs, histograms, and statistics of any variable in the output file; event summary listings of key events in single replications; charts summarising the outcome of all target/ADU interactions in all replications; range/time plots showing the main events of one replication pictorially.

- Specially written analysis programs. All output data is written to file in a well-defined format which is part of a series of data management programs known as the Modelling Support Environment (MSE). This data structure can easily be adopted for use by custom analysis programs to extract any data of interest to the user.

2.5 Modelling Support Environment (MSE)

A program such as ADVOCATE requires a large amount of input data and normally produces even greater amounts of output information. In modelling activities of this sort data is normally written to files and manipulated in a somewhat arbitrary way which is unique to the project in question. The Modelling Support Environment (MSE) was developed to provide a well-defined, extensible data handling and manipulation framework which could be used by any software with large input/output data requirements. The intention is to avoid the need to continually rewrite input/output routines to do essentially the same job time after time for different software projects.

The result is a suite of programs which provide a common user and programming interface for generating files which contain a self-contained description of the format and identity of the data they contain and which is stored in an efficient manner which is transparent to the user. A series of complementary data manipulation routines are available for interrogating MSE files and performing a range of functions such as writing data to alternative files in a user defined manner or plotting of data. Programmers can use MSE files via an input/output package, available in PASCAL, and can also interrogate files or perform data analysis using an interactive program.

An MSE data file consists of four sections, the first of which is a description on the length of all four sections together with a user supplied comment on the file's contents. The second section is a file history detailing when the file was created and by what programs as well as a record of any subsequent modifications made to the file. The third section provides information on the formats used to store the data which is held in the fourth and final section of the file. The third section can also contain information on the units (m/s, kg etc) of each data item. The fact that the file contains format information means that the user has a great deal of flexibility in defining the data structure to be used.

At the interactive level, routines are available for alteration of individual data items or file header and comment information. Complete files can be merged or truncated as well as deleted or duplicated. The user can selectively list data items and redirect them to new files. For data analysis there is a routine which can provide statistical information such as mean values, minima, maxima, standard deviation and variance on selected data items. Selected data items can be plotted as graphs or a histogram of frequency or probability can be obtained.

3 USE OF ADVOCATE

3.1 Research Applications

Although ADVOCATE models most of an air defence unit's sub-systems in a relatively simple way, the results obtained are nevertheless sensitive to changes in the performance characteristics of those sub-systems. This means ADVOCATE can be used to examine the relative benefits of changes to a particular sub-system or to make a choice between several ways of implementing the same function. The availability of realistic terrain is also important because many short range systems can potentially be limited by screening effects. The flexibility available within the program makes ADVOCATE suitable for research work and has been used frequently to perform a wide range of trade-off studies.

An example of a research activity using ADVOCATE was a series of studies which examined the trade-offs between weapon system range, missile speed and cost. The range of a system is typically related to the threat to be countered and the terrain in which the system will operate. The terrain can influence system performance by, for example, screening a low flying target until it is only a short distance from an ADU. In addition, once a line-of-sight is established to a target it may be interrupted intermittently, as the target flies behind terrain features. In such circumstances, where lines-of-sight may only briefly exist before a target is out of range, then missile speed becomes important in ensuring a target can be engaged while still visible. Clearly if the missile is command guided then the existence of a line-of-sight for the duration of an engagement will be critical to achieving adequate performance.

The approach adopted was quite straightforward and consisted of deriving a number of notional system concepts, the only differences between them being the surveillance sensor maximum range and the missile range and speed. These systems were fed into ADVOCATE and a systematic parametric study performed. One important outcome of this study was confirmation of the influence that terrain has on system performance.

An extension of the range/speed study attempted to evaluate how many ADUs of a particular type are required to afford a specified level of protection to a single target or area. This is a complex situation where one needs to balance system range against, for example, the constraints imposed by the local terrain and cost. Cost of the system will be important because there is the option of either using many short range systems or a few, more capable but more costly, longer range ones. When additional factors such as system mobility around the battlefield, susceptibility to countermeasures and the need to maintain adequate air defence even after sustaining losses to your own ADUs it is clear that the problem is not easily solved and requires the application of a variety of assessment tools and methodologies. Other research studies, which used ADVOCATE, have examined such issues as guidance methods as well as novel system architectures.

3.2 Procurement Support

Once beyond the initial research stages and into development a system must still be assessed against a military requirement and, in particular, during competitive procurement activities there will be a need to independently evaluate the likely performance of systems being proposed by industry. This is also an area where programs such as ADVOCATE have a role to play although the situation is complicated by the fact that at some stage the procurement of new equipment will need to be justified in terms of the relative benefit of its purchase as opposed to some other piece of military hardware. Demonstration of the military worth of new equipment will often involve modelling at the campaign, many-on-many, level for which ADVOCATE is not suited since it only models ground based air defences against airborne threats and does not give a direct measure of how the employment of a particular air defence system might influence the overall outcome of a land battle. However, results from ADVOCATE, which relate to the aggregate performance of a group of ADUs in a particular scenario, can be used as input to the higher level models. Considerable thought has gone into designing a study methodology which could be used to obtain realistic cost effectiveness measures for a new ADU at the full campaign level and this work is continuing.

ADVOCATE has already been used, together with a wide variety of other performance assessment programs, during the evaluation of competing systems for one particular requirement. During this work another air defence model, as well as ADVOCATE, was used and it was encouraging to note that the two gave similar results. Such "calibration" of one model against another is problematical and not often performed.

4 EXPERIENCE OF ADVOCATE IN USE

4.1 Strengths

During the initial stages of a project or research programme the detailed specification of a weapon system is impossible, although there is normally a requirement to measure system effectiveness on a continuous basis as designs evolve. ADVOCATE has proven flexible enough in the way it models a system to be of use even when a concept has not been fully defined. Since the input data framework was designed to cope with a wide variety of system concepts, it is possible to write a new system model and incorporate it into the existing ADVOCATE structure with minimal changes to the central programs. Flexibility also makes it easy to add more detail to sub-system models if necessary and to increase the amount of output detail available from them.

The program is also well suited to large parametric studies to examine the relative importance and sensitivities of overall performance to a wide range of sub-system characteristics, such as missile range or speed, sensor ranges, guidance methods and so on. The influence of terrain can be quantified and the performance of a system against a wide variety of threats is easily performed. The lack of extremely detailed models of system components is a positive benefit when such parametric studies are required and this also assists when evaluating new system architectures which employ radically new technologies.

The use of the MSE format for all data used in ADVOCATE has proved useful because of the self-documenting nature of MSE files.

4.2 Weaknesses

One of the major potential weaknesses of ADVOCATE is the fact that the behaviour of a threat does not vary or react in any way to the ADUs actions, except when they are removed after being killed. In reality one would expect countermeasures to be employed in an intelligent, reactive manner and for a threat to alter its trajectory to avoid an ADU once detected or to try to evade an attack. However, to date, this particular limitation has not restricted the use of the program and it is not believed to have influenced the validity of results obtained.

From the users viewpoint one weakness of the ADVOCATE suite is the relatively simple user interface, which means that at present an in-depth knowledge of the program is required to use it correctly. The generation of scenarios, which includes the siting of ADUs and the definition of target tracks, must at present be done by hand. When output is produced it is in the form of a list of events which occurred during the run and it is very difficult to interpret these results to gain a picture of how the targets and ADUs interacted.

5 FUTURE DEVELOPMENTS

5.1 User Interface

To overcome the problems associated with input and output mentioned above a new, graphics based, input/output interface is presently under development, the basis of which will be a digital map and 3-D terrain visualisation system. During scenario development the system can be used to evaluate candidate ADU sites in a more realistic manner and to obtain a good idea of the weapon coverage obtained by superposition of screening information on a map and by 3-D perspective views from the site in question. Target track definition will also be greatly simplified by using a digital map to automatically input and modify waypoints.

During output analysis both the map and 3-D views will be used to provide an animated replay of the results so that the interaction of targets and ADUs can be more clearly seen. The aim is for a non-expert ADVOCATE user to be able to see the ADU perform all the stages of an engagement from detection to missile strike and to visualise how complex multiple ADU/multiple target scenarios progress.

5.2 Expansion of Supporting Programs

As already indicated, a program such as ADVOCATE cannot, by itself, provide a full evaluation of the cost effectiveness of a particular system because of the need to measure the impact of air defence on other elements of the land battle. Speculative future developments of ADVOCATE may attempt to incorporate a variety of other, presently stand-alone, models into a systematic study methodology to improve both the realism of the simulation and the validity of military effectiveness indicators.

Possible models which might be incorporated into ADVOCATE studies in the future include procedural airspace control and command and control systems which in a modern, complex airspace environment may significantly affect overall ADU effectiveness. It may also be possible to include a program to evaluate the effect of enemy raids on military targets and try to incorporate the effects of friendly air defences as calculated by ADVOCATE. This may go some way to demonstrating the value of a particular system in militarily meaningful terms.

6 CONCLUSIONS

This paper has described the ADVOCATE air defence simulation model, its structure, principal features and how it can be used. Although there are a number of limitations in how it models some aspects, especially the threat, they have not been found in practice to severely limit the program's use.

Programs such as ADVOCATE have a clear role to play in both research and in equipment procurement although they can only be considered as a part of a much larger assessment process. ADVOCATE's flexibility and lack of unnecessary detail makes it particularly suitable for trade-off and sensitivity studies.

For the future ADVOCATE will be enhanced by the provision of a graphical interface which will simplify and make more realistic the input of critical data and also allow the user to visualise and interpret, in a qualitative sense, the large amounts of complex output generated.

PROGRAM - Monte Carlo simulation

Critical Event

Few-on-few

Written in PASCAL

Runs on IBM 4361 or a VAX.

ADU - can be netted

sub systems can include:

Surveillance

Tracker

Launcher

Missile

THREAT - can fly in formation

types include:

Fixed wing

Helicopter

Missile.

ENVIRONMENT - realistic terrain

meteorological conditions

ECM

FIG 1. MAIN FEATURES OF ADVOCATE.

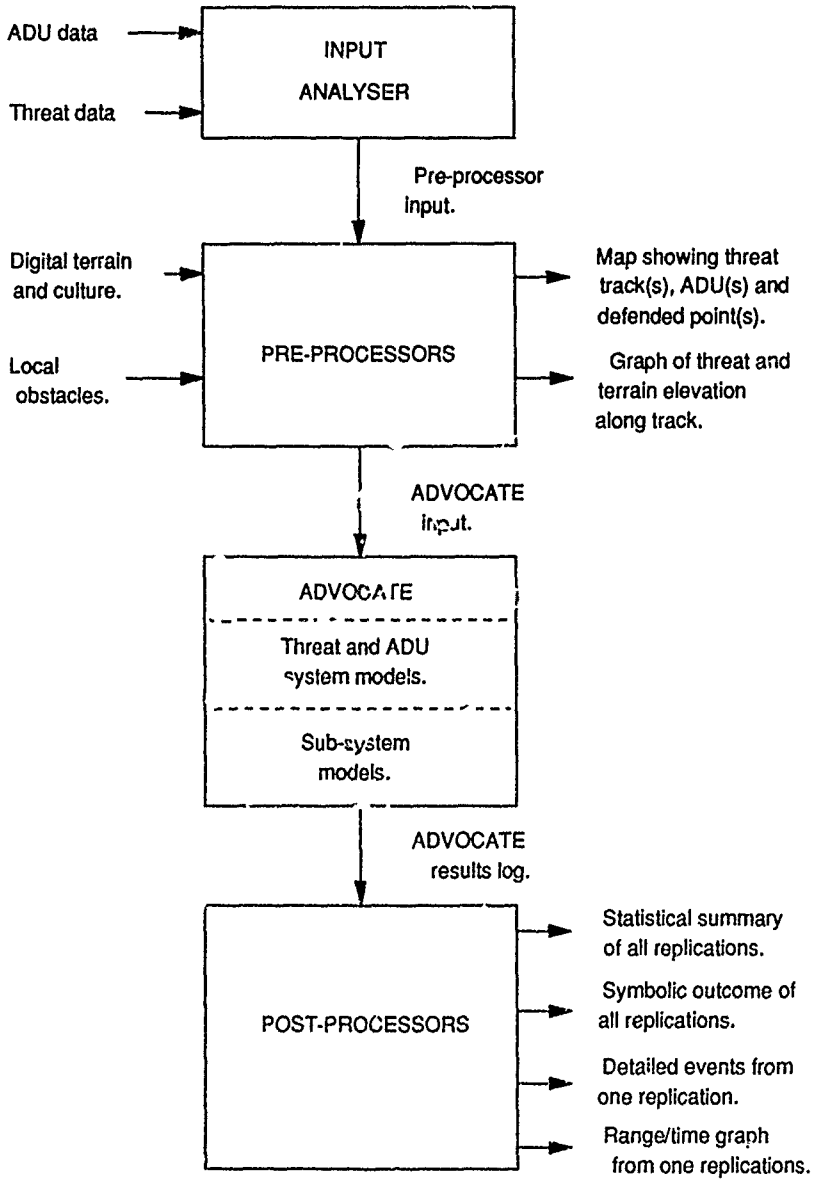


FIG 2. BLOCK DIAGRAM OF ADVOCATE SUITE.

COCKPIT MOCK UP CMU
A DESIGN AND DEVELOPMENT TOOL

by

Dipl.-Math. Christoph Weber
Elektronik-System-GmbH
Vogelweideplatz 9
8000 München 80
Germany

Summary

Designing a modern helicopter cockpit, ergonomics, operational and technical aspects have to be considered. For ensuring a low cost development schedule the CMU is a flexible, inexpensive design and development tool for optimization of the Man Machine Interface (MMI). The ESG CMU, realized in close cooperation with the user, is a full-size model cockpit of future helicopters such as NH 90 and PAH-2. The future user is integrated in the experimental closed-loop simulation with our CMU.

Problem situation

The design and development of new helicopters and thus of new and modern cockpits always requires the integration of latest technologies and of the human being in the latter. In the past the primary emphasis in this connection was placed on the technology in general, although, however, the crew is directly involved in the operation, i.e. by the tasks of helicopter command and control and mission performance. A further aspect is that both the physical characteristics, including the mind and the human intellect have hardly changed in contrary to the technology, so that it appears that man becomes the weakest link in the man-machine system. The primary objective today must be to relieve the man in the cockpit. Thus the crew-system interface is an essential key for not exceeding the limits of psychical and physical crew stress also under extreme conditions, such as

- Lowest flight
- Night missions
- Operations under extremely bad visibility conditions.

The problem becomes even more obvious by the fact that an enormous increase in data volumes as the result of new, additional sensors, and inclusion in command and control systems leads to a steady data quantity and workload increase, thus exceeding the limits of errorfree data processing by the crew.

The design and layout of an advanced cockpit, e.g. for the NH90 and PAH-2 is influenced by ergonomic, operational and technical aspects, whereby to optimum layout of the MMI's plays an important part.

In order to consider the latter, i.e.

- Optimization of the MMI -

in connection with the future NH90 and PAH-2 helicopters, ESG performs the following experimental programmes in parallel with the development efforts:

CMU (S)	Cockpit Mock Up (Side by Side)
CMU (T)	Cockpit Mock Up (Tandem)
AVT	Equipment test facility

Both the CMU (S) and CMU (T) are a flexible, costfavourable cockpit design tool for a side-by-side cockpit (NH90) and a tandem cockpit (PAH-2).

The AVT is an equipment test facility enabling the analysis of advanced equipment components in addition to the MMI aspects of a modern cockpit in the course of flight trials. The AVT, however, is not the subject of this paper.

Tasks and Objectives

As the task and objective in connection with the CMU (S) and CMU (T) are identical, the only difference being the version of the respective cockpit design tools due to the different functional requirements of the NH90 and PAH-2, only the CMU (T), briefly addressed as CMU is described in the following.

The essential task of the CMU design and development tool is the conversion of theoretical conceptual designs to experimental hardware for analysing the MMI for the future PAH-2 under quasi-real operational conditions.

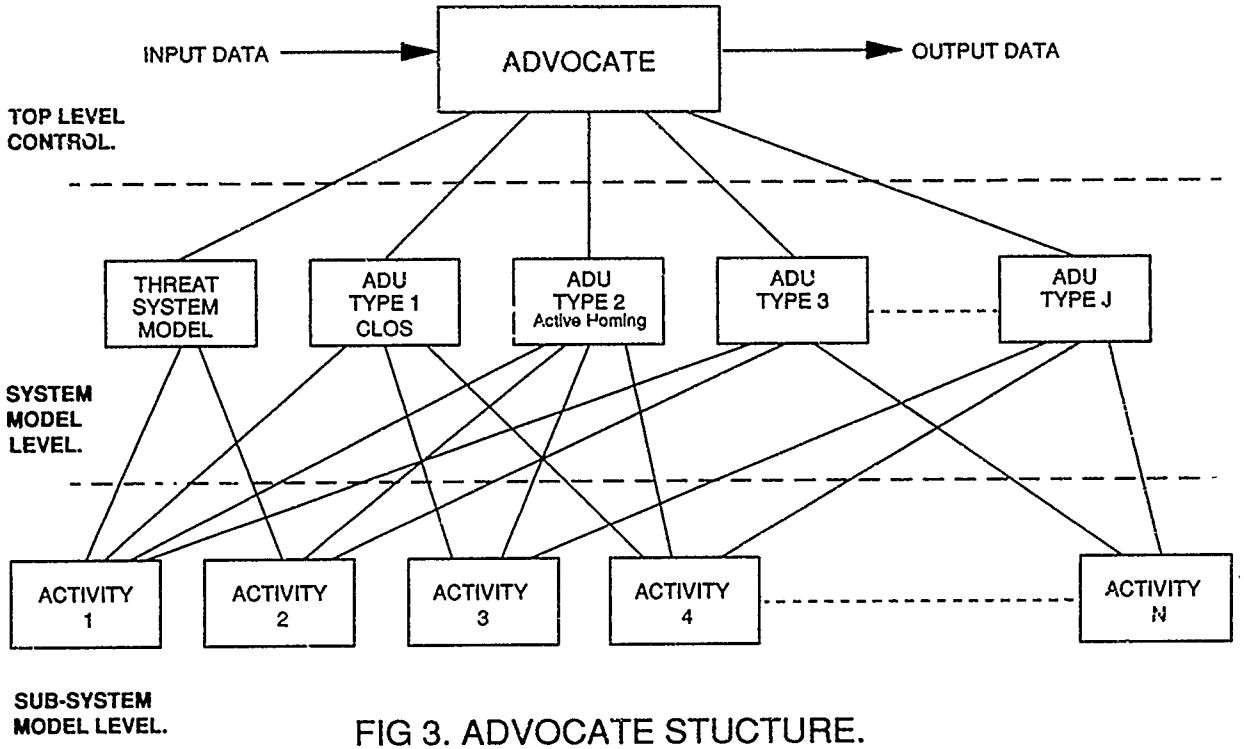


FIG 3. ADVOCATE STUCTURE.

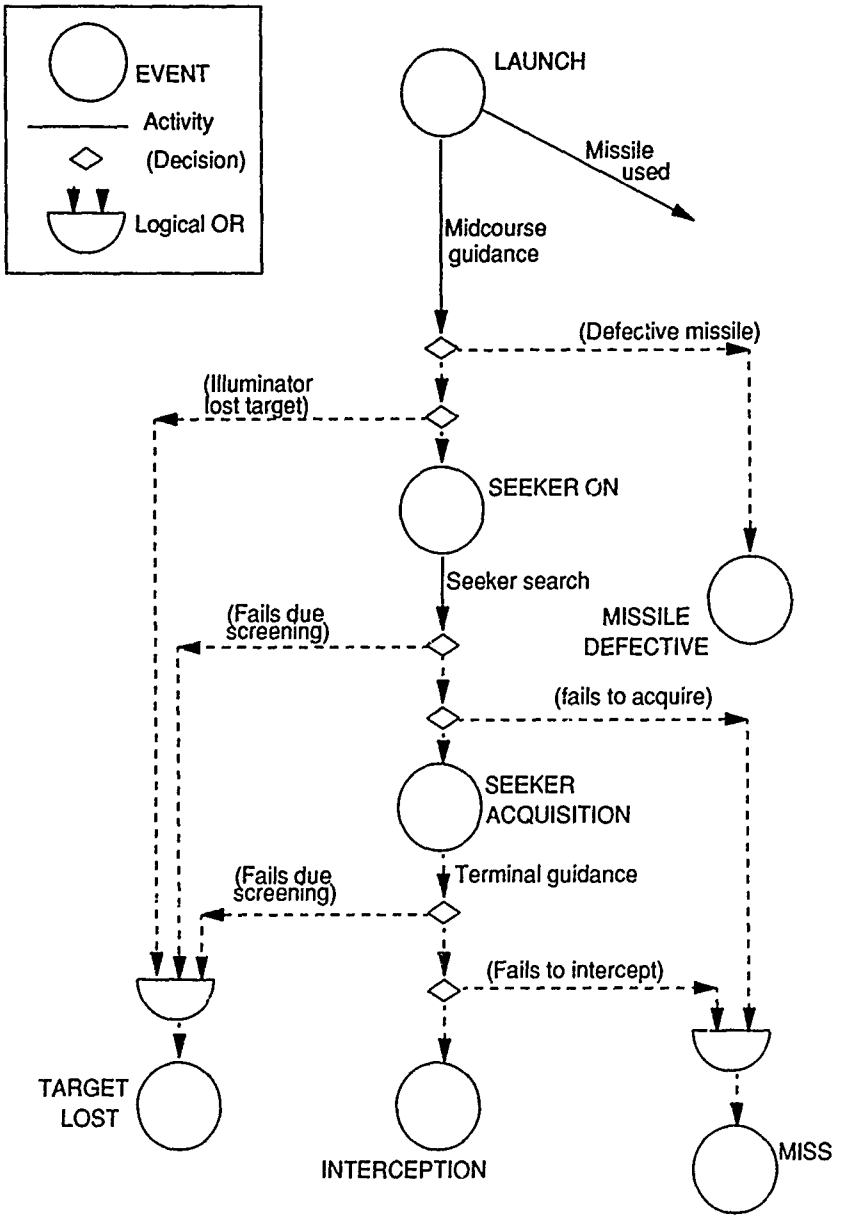


FIG 4. EXAMPLE OF PART OF A SYSTEM LOGIC DIAGRAM.

INTEGRATION OF A REALISTIC AIRLINE/AIRCREW/AIRCRAFT COMPONENT IN ATC SIMULATIONS

by

André Benoît and Sip Swierstra

EUROCONTROL
European Organisation for the Safety of Air Navigation
Engineering Directorate
72, rue de la Loi, B - 1040 Brussels (Belgium)

ABSTRACT

Present trends indicate that air traffic density will double over the next 10-15 years. Will air traffic services permit this? The level of automation achieved in the aircraft itself allows a flight to be programmed and then conducted with little or no subsequent human intervention. In contrast, at the executive level, the air traffic authorities handle each flight as a succession of individual short segments and are not in a position to take much account of aircraft capabilities.

This paper will describe the work carried out by the Engineering Directorate of the EUROCONTROL Agency with a view to integrating airline requirements, crew reactions and aircraft capabilities in simulations aimed at assessing future air traffic handling procedures. Such procedures involve the 4-D guidance of aircraft which may possess the entire range of 2-D, 3-D and 4-D navigation capabilities.

Emphasis is placed on specific aspects covering two essentially different areas :

(a) assessment of future 4-D ground/air guidance procedures under realistic conditions, implying the use of full-scale flight simulators operated by airline crews ;

(b) assessment of the overall air traffic control loop, including controller/pilot/aircraft interfaces, in a realistic manner.

In the first case, use is made of full-scale flight simulators operated by airline crews, duly integrated into an overall facility including a ground-based control station manned by professional air traffic controllers.

In the second case, since some hundred aircraft may be involved simultaneously, full-scale flight simulators and pilots have had to be replaced by suitable models and pilot substitutes or "pseudo-pilots".

The solutions proposed in the two areas have been tested and presented successfully to controllers, pilots and pseudo-pilots.

The cooperation of Aéroformation (Toulouse, France), Belgian World Airlines, SABENA (Brussels, Belgium), British Airways (London, UK), City Hopper, Nationaal Luchtvaart Maatschappij (Amsterdam, Netherlands), Deutsche Lufthansa (Frankfurt, Germany), National Aerospace Laboratory, NLR (Amsterdam, Netherlands) and the Belgian Régie des Voies Aériennes/Regie der Luchtwegen (Brussels, Belgium) is gratefully acknowledged.

INTEGRATION DU VECTEUR COMPAGNIE AERIENNE/EQUIPAGE/AERONEF

DANS LES SIMULATIONS ATC

par

André Benoît et Sip Swierstra.

EUROCONTROL
Organisation européenne pour la sécurité de la navigation aérienne
Direction technique
72, rue de la Loi, B - 1040 Bruxelles (Belgique)

SOMMAIRE

D'après les tendances actuelles la densité du trafic aérien devrait doubler au cours des 10 à 15 prochaines années. Les services de la circulation aérienne seront-ils capables d'y faire face? Le niveau d'automatisation des aéronefs permet de programmer un vol puis de l'exécuter sans intervention humaine ou avec une intervention minimale. En revanche, au niveau opérationnel, les services de la circulation aérienne traitent chaque vol comme une succession de courts segments, et ne sont pas en mesure d'exploiter au mieux les capacités des aéronefs.

Le présent article décrit les travaux menés par la Direction technique de l'Agence EUROCONTROL en vue d'inclure les besoins des compagnies aériennes, les réactions des équipages et les capacités des aéronefs dans des simulations visant à évaluer les procédures futures de prise en charge du trafic. Ces procédures incluent le guidage quadridimensionnel d'aéronefs pouvant présenter la gamme complète des possibilités de navigation (bi-, tri- et quadridimensionnelle).

L'accent est mis sur des questions spécifiques relevant de deux domaines fondamentalement différents :

a) l'évaluation des procédures futures de guidage air/sol quadridimensionnel en conditions réelles, ce qui implique le recours à des simulateurs de vol grandeur nature pilotés par des équipages de compagnies aériennes ;

b) l'évaluation, de manière réaliste, de toute la chaîne de contrôle de la circulation aérienne, y compris les interfaces contrôleur/pilote/aéronef.

Dans le premier cas, on fait appel à des simulateurs de vol grandeur nature, pilotés par des équipages et dûment intégrés dans une installation globale comprenant une station de contrôle au sol gérée par des contrôleurs professionnels.

Dans le second cas, comme l'évaluation peut impliquer jusqu'à cent aéronefs simultanément, les simulateurs de vol grandeur nature et les pilotes doivent être remplacés par des modèles ad hoc et des "pseudo-pilotes".

Les solutions proposées dans ces deux domaines ont été expérimentées et présentées avec succès aux contrôleurs, pilotes et pseudo-pilotes.

Nous remercions de leur collaboration Aéroformation (Toulouse, France), Belgian World Airlines SABENA (Bruxelles Belgique), British Airways (Londres, Royaume-Uni), City Hopper, Nationaal Luchtvaart Maatschappij (Amsterdam, Pays-Bas), Deutsche Lufthansa (Francfort, République fédérale d'Allemagne), National Aerospace Laboratory, NLR (Amsterdam, Pays-Bas) et la Régie belge des voies aériennes/Regie der Luchtwegen (Bruxelles, Belgique).

1. FRAMEWORK

As an introduction to our paper presented at this Symposium devoted to simulation, we shall indicate briefly the field, context and range of application considered and show the originality and scope of our contribution to the future role of simulation in air traffic control.

The application considered covers on-line air traffic handling, a field historically known as air traffic control.

Air traffic control is a large scale system in which the human being plays the essential role. On-line, all decisions are taken by the controller on the basis of his knowledge of the current local traffic situation. In spite of all efforts undertaken to date, this will remain the case still for an appreciable period of time.

Accordingly, any simulation of an advanced control procedure or newly proposed sub- or complete system will have to **incorporate the controller/system interface** or reflect it in a highly realistic manner.

In the field of applied research, aids to on-line regulation of traffic and automated assistance for the guidance of all aircraft concerned are being developed. In particular, techniques are now proposed for controlling time-of-arrival constrained trajectories with an accuracy better than 10 seconds - in spite of all the perturbations effecting the conduct of a flight - that is to say, consistent with airborne 4-D navigation. This, clearly, implies an additional simulation requirement, namely the **introduction of the air component aircraft/avionics/pilot, response characteristics included, in an accurate and reliable manner.**

2. GUIDANCE OF AIRCRAFT v. ON-LINE TRAFFIC MANAGEMENT

The discussion which follows is general ; it applies to the assessment of any ATC procedure or sub-system proposed for the improvement of the currently observed difficulties (saturation, congestion, delays, costs). Nevertheless, for the sake of convenience, we shall on occasion refer to the system developed at the Agency, initially to control the flow of traffic entering an extended area "centered" around a main airport and extending up to 300 nm around it - depending on jurisdiction - as including and surrounding a main terminal and possibly a series of secondary airports, an area referred to as a Zone of Convergence, ZOC (Refs. 1 and 2).

Such a sub-system has been developed with implementation constraints in mind. It is now possible to introduce it in a full-scale traditional ATC simulation facility or at an existing operational centre as an additional set of essential functions as suggested in the diagram in Figure 1, which reflects the Belgian airspace organisation (approach, en-route and upper airspace centres). Obviously, the box "ZOC ATM module", includes both traffic management and flight guidance functions.

Clearly, the two functions, namely the **on-line management of traffic**, which is undertaken at least when a new aircraft enters the area concerned, and the **guidance of each individual flight**, are closely coupled. Nevertheless, in terms of development, validation and assessment, two essential separate steps were envisaged :

- simulation of guidance techniques, airline crews and air traffic controllers in the loop;
- simulation of traffic management strategies, with pilot and controller actions duly represented;

before the conduct of full-scale ATC simulations at various levels of sophistication in the representation of the aircraft/avionics/pilot component.

3. GUIDANCE OF AIRCRAFT : USE OF FULL-SCALE FLIGHT SIMULATORS

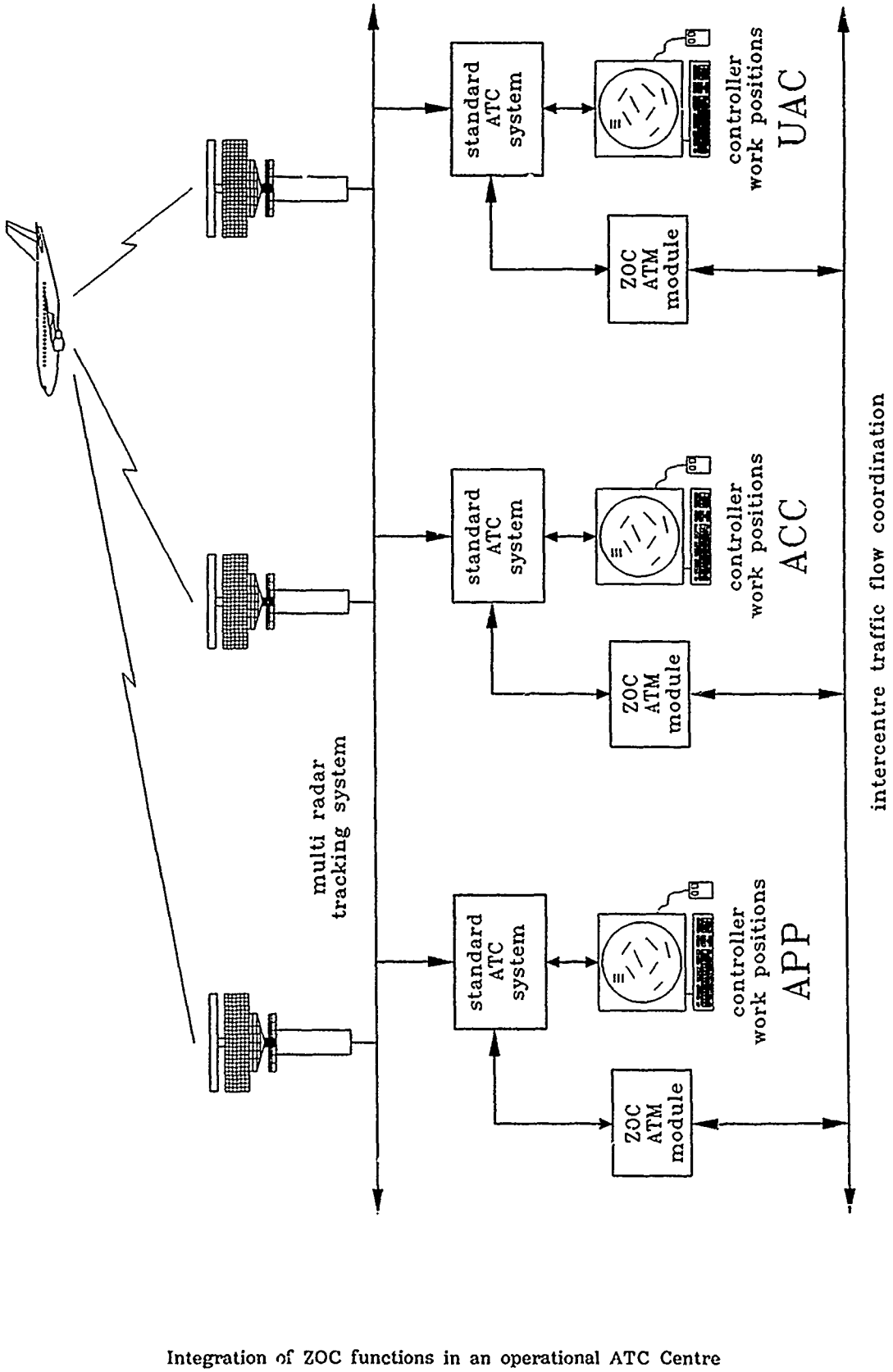
To assess the guidance of aircraft sub-system, use has been made of full-scale flight simulators operated by airline pilots. The simulation environment is schematically shown in Figure 2. Full-scale flight simulators are used simultaneously (B-737 and DC-10 at SABENA, Brussels, Belgium ; B-737 and B-757 at British Airways, London, UK ; Airbus A-310 at Aéroformation, Toulouse, France) and additional traffic is generated and operated either by pilots or pseudo-pilots or both using the ACCESS flight simulator in manual mode, semi-automatic mode or both, respectively (Ref. 3). In these validation exercises, **the control of the traffic** is performed by professional air traffic controllers.

In such simulations, the emphasis has been placed on the **human role** and consequently on the **definition of the related interfaces** involving either the controller or the pilot or both.

As a result, it has been possible to refine the guidance advisories to meet both the system objectives and the pilots' and controllers' working requirements. Further, a wide range of perturbations could be covered and the stability of the guidance of flights sub-system duly assessed.

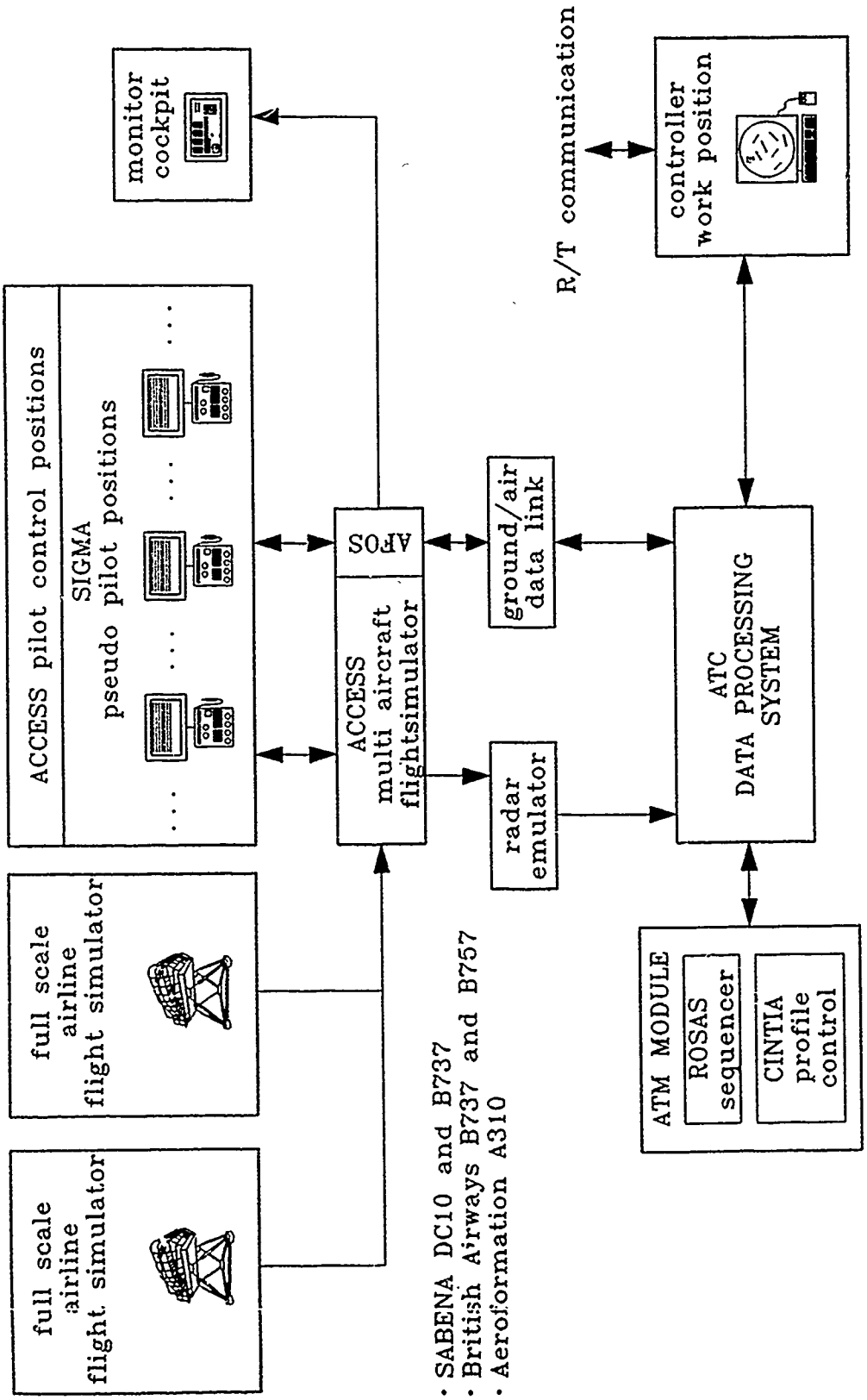
4. MANAGEMENT OF TRAFFIC : EXTENSIVE USE OF ACCESS FLIGHT SIMULATORS

In order to test various management strategies or to assess the impact of alterations in a basic reference strategy, or to illustrate possible advanced modes of operation in air traffic control, extensive use is made of the ACCESS flight simulator capabilities (Ref. 3).



Integration of ZOC functions in an operational ATC Centre

Figure 1



Assessment of ground-based guidance procedures
Use of full-scale flight simulators

Figure 2

All possible modes of operation of ACCESS can be used - separately or combined - to simulate situations ranging, for instance, from :

- the introduction of a management/guidance component into an existing ATC simulation facility or operational centre, to

- the illustration of a possible futuristic fully automatic ATC operating system.

The advantages of conducting such simulations with ACCESS are those inherent to ACCESS itself, namely economy - in equipment and manpower -, realism of the air component, ease of operation.

5. ASSESSMENT IN REAL LIFE : USE OF ACTUAL AIRCRAFT

At his stage of development - preparatory to the implementation of the ZOC function in operational life - it is planned to assess both management of traffic and guidance of flights functions, incorporating one real aircraft in the simulation. Two series of tests, appreciably different in nature, are envisaged to this end.

The first would include a twin-propeller aircraft with standard navigation capability, see Figure 3. The second would constitute a preliminary test of the use of the ground/air data link capability - as associated with Mode-S - for ATC purposes. A diagram of the organisation of this exercise is given in Figure 4.

In both cases, the desired traffic load would be generated using the ACCESS capability.

6. CONCLUSIONS

A simulation facility has been developed, based on the use of one or, if necessary, several work stations, see Figure 5. It was initially developed :

(a) to reduce the need for full-scale ATC facilities, and

(b) to allow for accurate representation of the air component.

It now constitutes a powerful tool enabling a compiete ATC system to be simulated, incorporating

- advanced on-line traffic management and stable and reliable guidance of flights functions;
- human role and related interfaces at several levels of automation;
- air component, response characteristics included.

Further, the same simulation can incorporate simultaneously various representations of the air component, in particular :

- real aircraft ;
- full-scale flight simulators ;
- ACCESS flight simulators in manual mode, semi-automatic mode, or both ;

which makes it possible to develop, test, validate and assess any particular component with the actual or desired traffic density.

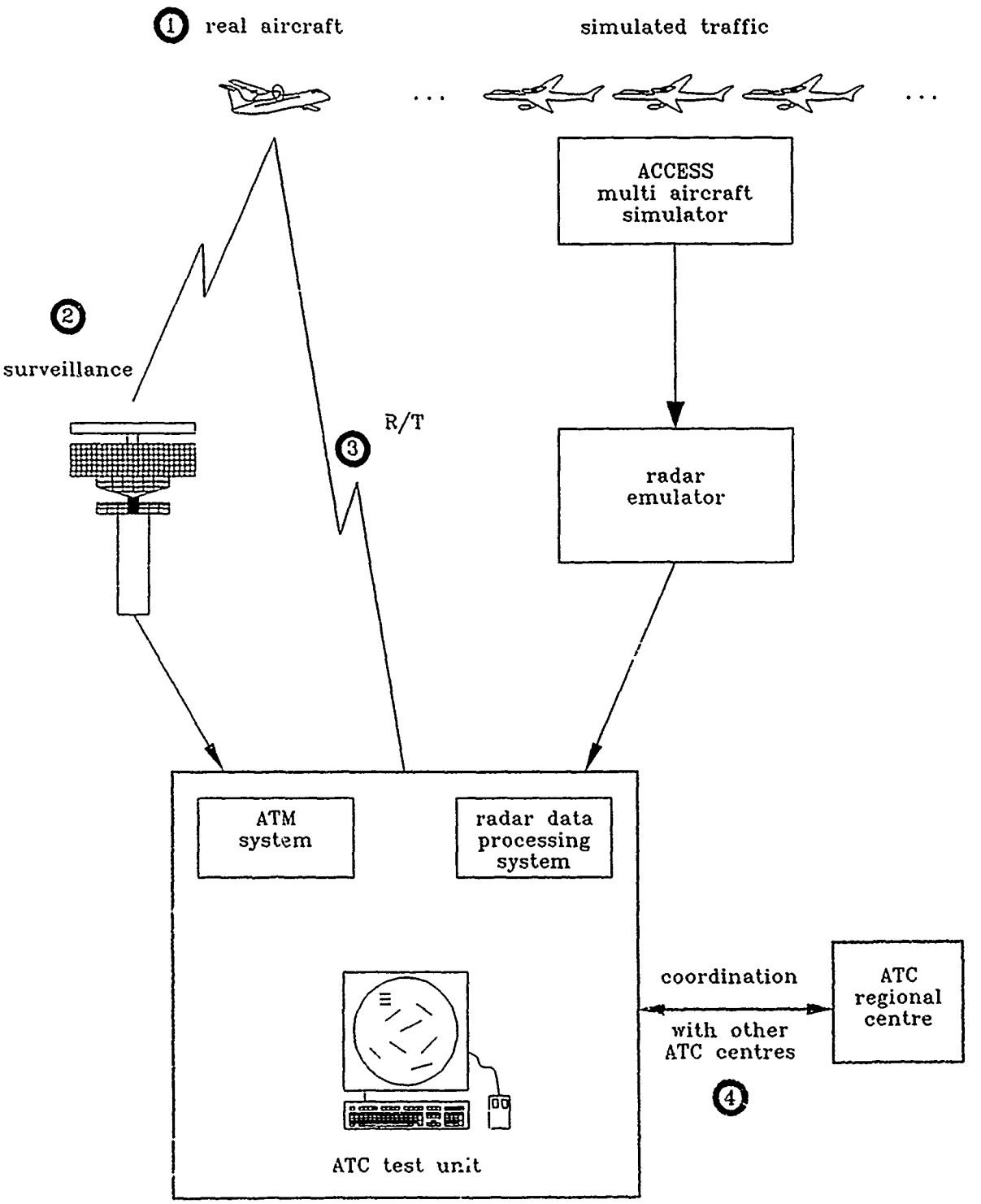
With the use of ACCESS in the automatic mode, the facility can be employed to illustrate several possible aspects of future levels of ATC automation. The facility is easily portable (It has been used in Brussels, London, Toulouse and at the EUROCONTROL Experimental Centre, Brétigny/Paris). Further, it is readily adaptable to local conditions (connections involving standard Ethernet and RS-232 interfaces). The potential of the facility will be further demonstrated at the Farnborough Air Show next September.

7. REFERENCES

1. "Air Traffic Management and Aircraft Guidance in and around a Main Terminal"
by A. Benoît and S. Swierstra,
EUROCONTROL Report 892009-E/A, December 1989.

2. "The Control of Inbound Flights : Basic Principles"
by A. Benoît and S. Swierstra,
EUROCONTROL Report 892011-E/892011-F, December 1989.

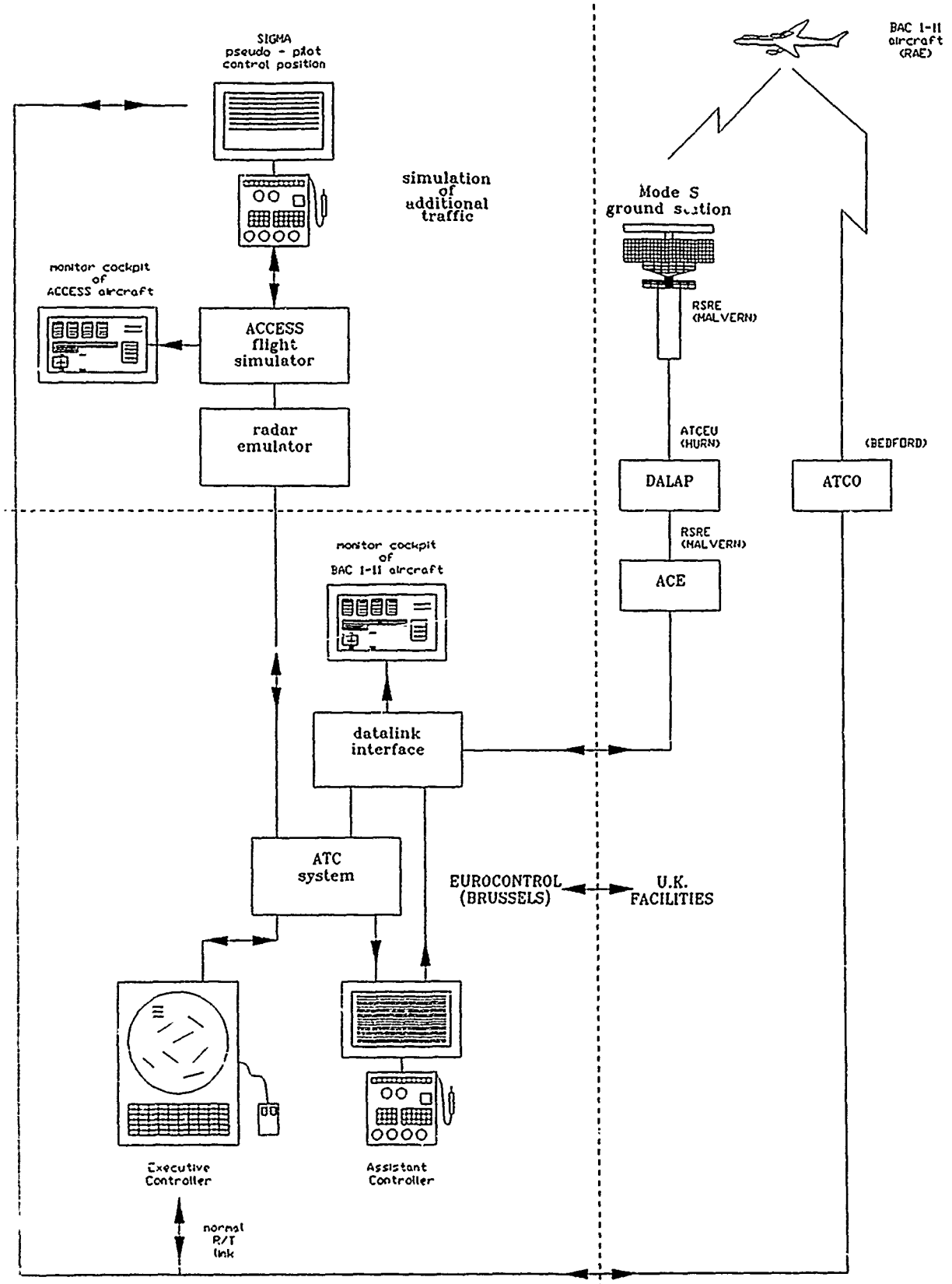
3. "Integration of aircraft capability into Air Traffic Handling Simulations"
by A. Benoît, Y. Delnatte and S. Swierstra,
EUROCONTROL Report 892010-E, December 1989.



ASSESSMENT OF GROUND-BASED 4-D GUIDANCE OF FLIGHTS

(Time-of-arrival constrained trajectories)
Tests conducted in present R/T communications environment

Figure 3



Integration of real aircraft in simulations
ZOC control in an A/G datalink environment

Figure 4

8. BIBLIOGRAPHY

"AIRCRAFT TRAJECTORIES : Computation - Prediction - Control"
A. Benoît, Editor
AGARD AG-301, May 1990

9. GLOSSARY

ACC	:	Area Control Center
ACCESS	:	Aircraft Control Console for Experiments and Simulations Systems.
ACE	:	ATC communications environment.
AFOS	:	Automatic Flight Operating System
APP	:	Approach control center
ATC	:	Air Traffic Control
ATCO	:	Air Traffic Control officer
ATH	:	Air Traffic Handling
ATM	:	Air Traffic Management
CINTIA	:	Control of INbound Trajectories for Individual Aircraft
DALAP	:	DAta Link Application Processor
ROSAS	:	Regional Organisation of the Sequencing and Scheduling of Aircraft
SIGMA	:	System for Input and Generation of Messages for ACCESS
ZOC	:	Zone Of Convergence

10. KEY-WORDS

Air Traffic Control - Air Traffic Management - Air Traffic Handling
Simulation - Flight Simulators - Automation of ATC/ATM
ACCESS - ATC - ATM - CINTIA - ROSAS - ZOC
Guidance of flights - Time-of-arrival constrained trajectories

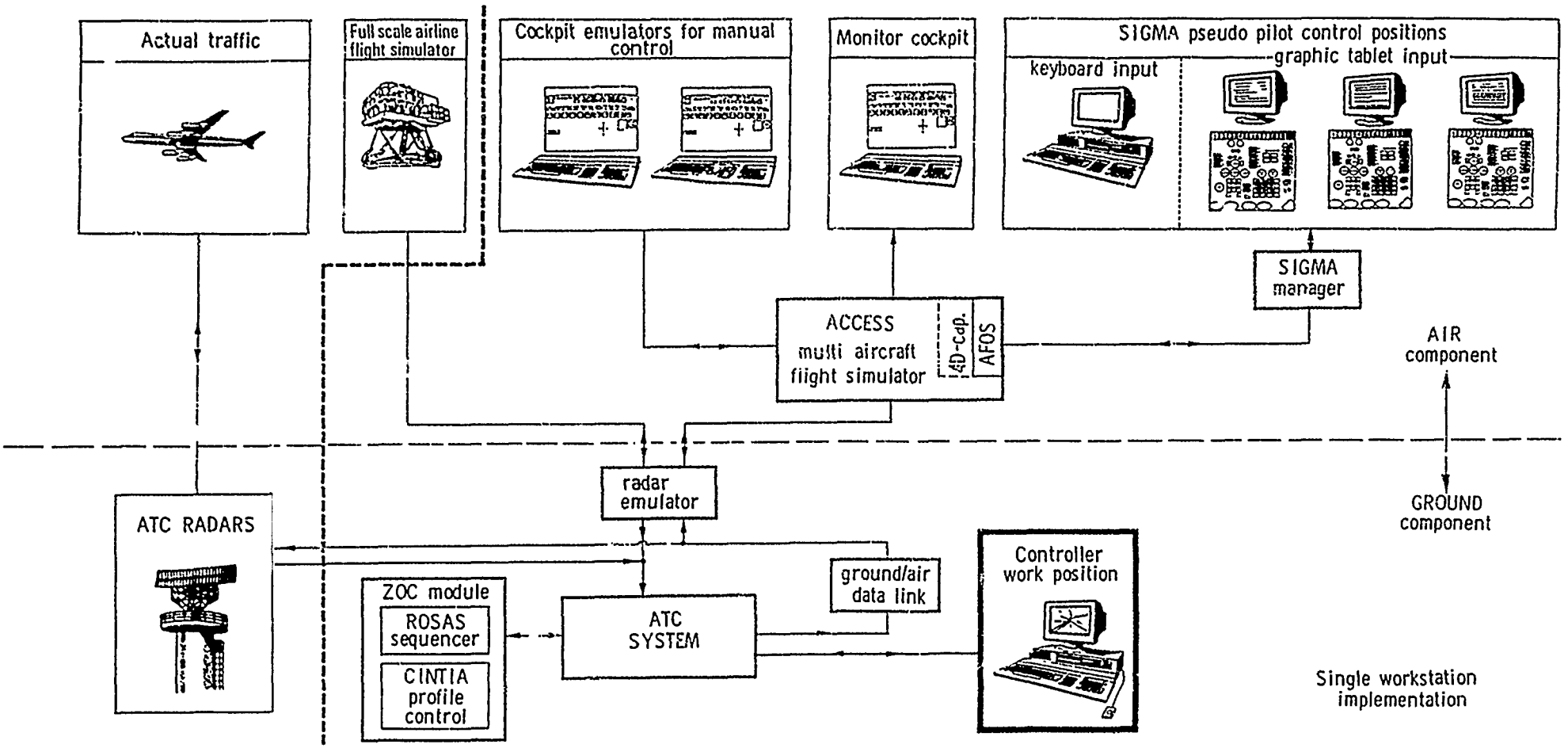


Illustration of the adaptability of the simulation facility
Single workstation implementation

Figure 5

NAVPACK - Simulation Tools for Design of High Performance Integrated Navigation Systems

Jan Z. Zywiel, John S.A. Hepburn, Bruno M. Scherzinger

Advanced Technology Centre, Honeywell Limited,
1345 Denison Street, Markham, Ontario, Canada, L3R 5V2.
Tel. (416) 479-5100

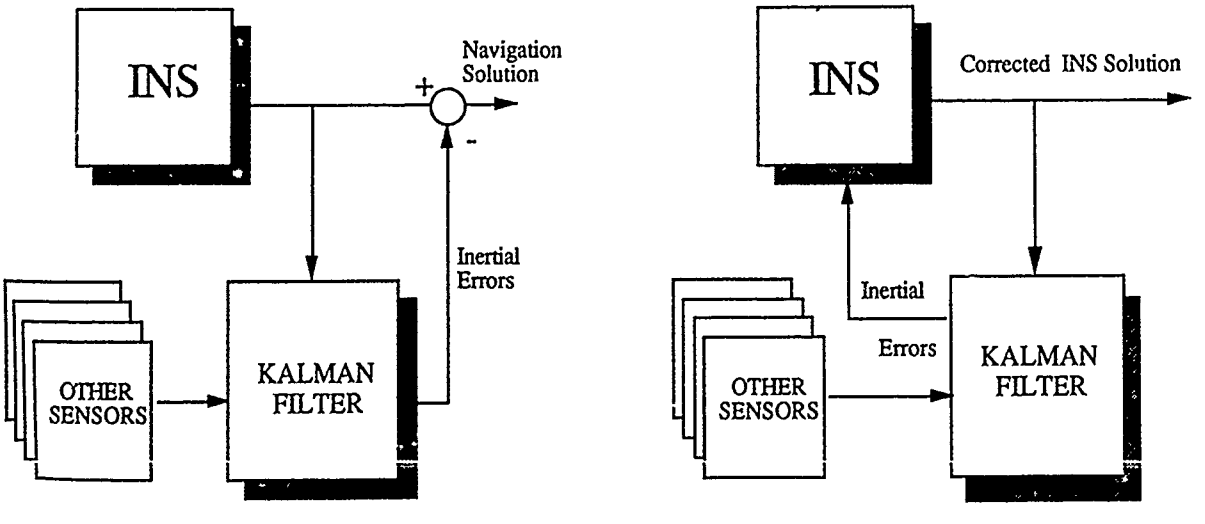
ABSTRACT

This paper describes the NAVPACK Software Package, developed by Honeywell's Advanced Technology Centre, for navigation systems simulation and analysis. The fundamental concept of NAVPACK is to create as modular a structure for the software as possible, with standard interfaces between separate programs and within individual programs. Therefore NAVPACK consists of distinct computer programs that perform individual simulation tasks. These programs are combined as needed at the operating system level to perform the required processing.

The NAVPACK software has been successfully used for supporting a number of programs undertaken by the Honeywell's Advanced Technology Centre. It was used in the development of the Helicopter Integrated Navigation System (HINS) for the Canadian Department of National Defence [1],[2]. HINS requirements called for a high performance, robust, and fault tolerant integrated navigation system. Elements of NAVPACK were employed for the development of a very high precision motion compensation system for high resolution, long range synthetic aperture radar (SARMC) for the United Kingdom Ministry of Defence [3]. The package was also used in Honeywell's work on a recently completed Marine Attitude Reference System (MARS) for the Canadian Department of National Defence, comprising an Inertial Navigation System (INS) capable of in motion alignment without aiding sensors.

INTRODUCTION

A high performance integrated inertial navigation system (such as the Honeywell HINS [1]) includes an INS and other aiding navigation sensors, such as a global positioning system (GPS) receiver for providing geographic position and velocity, a Doppler sensor for aircraft body velocity and a baro or radar altimeter, together with a Kalman filter. The Kalman filter uses the redundancy in the data from the navigation sensors as well as accurate dynamic, geometric and statistical models of the errors associated with the data in order to compute optimal estimates of the error in the INS output. Then the optimal integrated navigation solution is obtained by compensating the INS position, velocity and attitude outputs with the error estimate determined by the blending Kalman filter. This compensation may be implemented in either of the two modes. feedforward (open loop) or feedback (closed loop). In the former, the INS generates its solution without aiding (except perhaps by altitude) and the error compensation is applied externally to the INS outputs. In the latter case, the estimated error computed by the Kalman filter is used to compute resets which are applied to the INS -- a procedure known as error control -- so that its output is as close to the optimal blended solution as possible. Figure 1 illustrates the integrated navigation concept for both the feedforward and feedback modes.



(a) feedforward mode

(b) feedback mode

Figure 1
Integrated Navigation Concept

NAVPACK is a collection of computer programs that aid the designer of an integrated navigation system in all the stages of the system development, from the initial design and simulation to performance evaluation of a development model. NAVPACK consists of distinct computer programs that perform individual tasks, which may be classified into four categories. trajectory generation, data synthesis,

processing, and evaluation. These programs are combined as needed at the computer's operating system level to perform the required data processing. Standard interfaces between separate programs and within individual programs facilitate the use of the package and make NAVPACK a very flexible software tool. A big advantage of the NAVPACK structure is that the synthetic data, typically used in the earlier, simulation stages of navigation system development, may be later replaced by real sensor data without modifying the processing program. Figure 2 illustrates interactions between the major functional blocks within the NAVPACK structure.

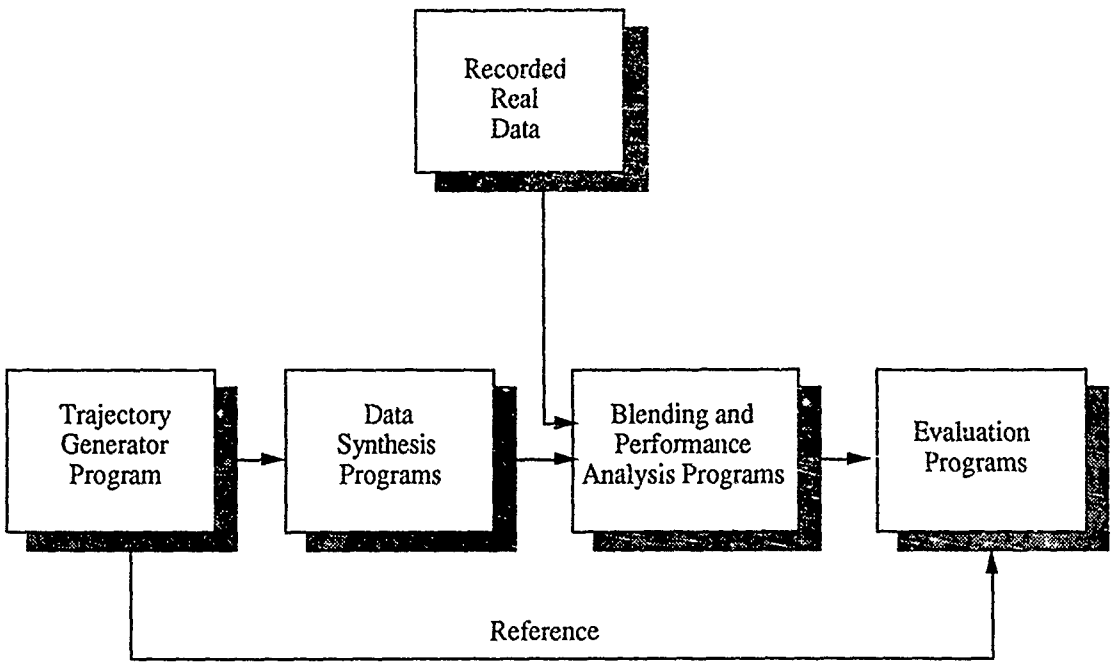


Figure 2: NAVPACK Architecture

FLIGHT PROFILE AND SYNTHETIC SENSOR DATA GENERATION

To simulate an integrated navigation system realistic synthetic sensor data must be generated for all the sensors or subsystems such as INS, GPS, etc. which comprise the system. The data has to exhibit typical error characteristics of the real navigation sensors. The sensor data synthesis is a two step process. First a trajectory is constructed using the trajectory generation program and then the sensor data is obtained for the individual sensors.

The trajectory generation program produces reference (i.e. error free) trajectory data, derived from a smooth nominal profile generated by cubic splines together with disturbances such as atmospheric turbulence produced by application-specific Gauss-Markov models. The user defines a trajectory by providing English text instructions specifying initial conditions and maneouvres. The stand alone data synthesis programs apply deterministic and statistical errors to the error free trajectory, resulting in realistic synthetic sensor data for simulated navigation units. These errors are controlled by the user through error parameters. Certain errors may be deselected by simply specifying appropriate parameters. For example, in the synthesis of a strapdown Inertial Measurement Unit (IMU) with conventional mechanical gyroscopes the user would set the dither amplitudes to zero as they are applicable only to Ring Laser Gyroscopes (RLG).

Some of the synthesis programs can synthesize sensor failures, for simulation of navigation system with failure detection, isolation and reconfiguration capabilities. Failure synthesis is accomplished by an alteration of sensor error parameters at the time of a simulated failure in such a way that the characteristics of the synthetic output correspond to a failure condition. These conditions are specified by failure parameters defining types of failures, times of their occurrences and the magnitudes.

The library of synthetic sensor data generation programs presently includes 11 most typical syntheses of: air data system, baro altimeter, distance measurement equipment (DME), Doppler radar, gimbaled INS, GPS, strapdown IMU, strapdown INS, strapdown magnetometer, and vertical gyro. It is expected that this library will grow to include other synthesis programs. The trajectory generation and synthesis programs are described in greater detail in [4]

BLENDING AND PERFORMANCE ANALYSIS PROGRAMS

This collection consists of computer programs that read sensor data either synthetic or recorded real - and perform sensor blending to produce the optimal navigation solution or a statistical measure of the accuracy of this solution. These are the covariance analysis and the full scale simulation/processing programs. In addition this group includes programs which further postprocess the blended navigation solution to aid in the performance analysis effort. These are the resets removal and the optimal smoother programs and they are typically used in postprocessing of recorded real sensor data.

Covariance Analysis

It is not always possible to implement a truly optimal integrated navigation processor in a real time system. Computer memory and throughput limitations may compromise the design of such a system and a suboptimal implementation may become a necessity. If this

is indeed the case, the designer has to determine the sensitivity of the system to any mismodeling between the optimal and the suboptimal solutions. This can be addressed with covariance analysis.

In the so called *Optimal Error Covariance Propagation* mode the program reads trajectory data and uses system dynamics, statistics and measurement models to propagate the error covariance. The U-D factorized Kalman filter [5] computes the optimal gains [6] which are used in covariance propagation and can be recorded to a file. The covariance propagation processing may include prefiltering of measurements if required. The output covariance provides a good indication of performance statistics of the integrated navigation system. It is also possible to run the covariance analysis in the *Suboptimal Error Covariance Propagation* mode. Here the program propagates the error covariance using a recorded gain history and the suboptimal covariance update is implemented [6]. Measurements may also be prefiltered.

The covariance analysis program is often used to conduct the *Sensitivity Analysis* to evaluate the sensitivity of the performance of the navigation system to specific error sources. This is performed in three steps. First the optimal covariance propagation is performed using the actual filter model of the investigated navigation system, and the optimal gain history is recorded. Then in the second step the suboptimal covariance propagation of the truth model, i.e. of the model believed to be the most complete and accurate, is carried out using the previously recorded gains. Finally the sensitivity evaluation is performed by comparison of the covariance from the filter model (optimal) with that of the truth model (suboptimal). The concept of the sensitivity analysis is illustrated in Figure 3.

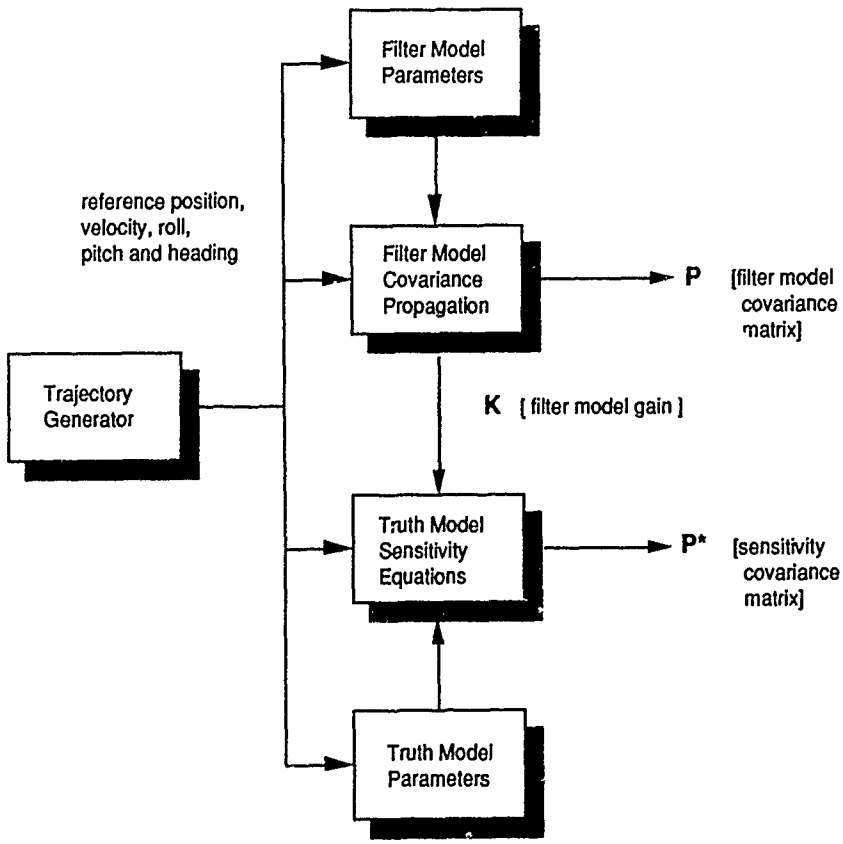


Figure 3: Sensitivity Analysis Concept

Figure 4 shows a sample Covariance Analysis plot. The broken line indicates the RMS error corresponding to the truth model covariance, while the solid line refers to the filter model. It can be seen here that the filter model is unduly optimistic in its error covariance estimate and that error sources which contribute to the errors in the truth model are not accounted for in the filter model.

It is often desirable to determine the contributions of individual error sources to the overall estimation error in an integrated navigation system - the so called *error budget*. An error budget is carried out using the covariance analysis program. First the Kalman gain history of the optimal covariance propagation run is calculated and stored. This run is used as a reference for the error budget. Next the covariance propagation of the truth model is run several times in the suboptimal mode to generate a set of actual RMS performance histories from the stored filter gain. In each run all error sources and initial conditions are set to zero except the one being evaluated. Obtained performance figures are then compared with the reference figures to give error contributions.

Full Scale Simulation/Processing

The covariance analysis program propagates the error estimate covariance of the investigated navigation system, thus providing a good measure of the expected system error statistics. It does not estimate the navigation errors themselves and therefore does not produce the actual error-corrected navigation output such as position, velocity, and attitude. This however is done by the Full Scale Simulation/Processing program. This program integrates data from individual navigation subsystems, taking advantage of their complementary strengths so as to provide a navigation solution which is more accurate and reliable than that of the individual subsystems. On input, the program accepts sensor or navigation subsystem data generated by the Synthesis Package, or recorded real data. It computes blended navigation position, velocity and attitude. In addition, other parameters required for performance evaluation are output, such as Kalman filter error state covariance, measurement residuals, and innovation covariance. Figure 5 illustrates the functionality of the Full Scale Simulation Program.

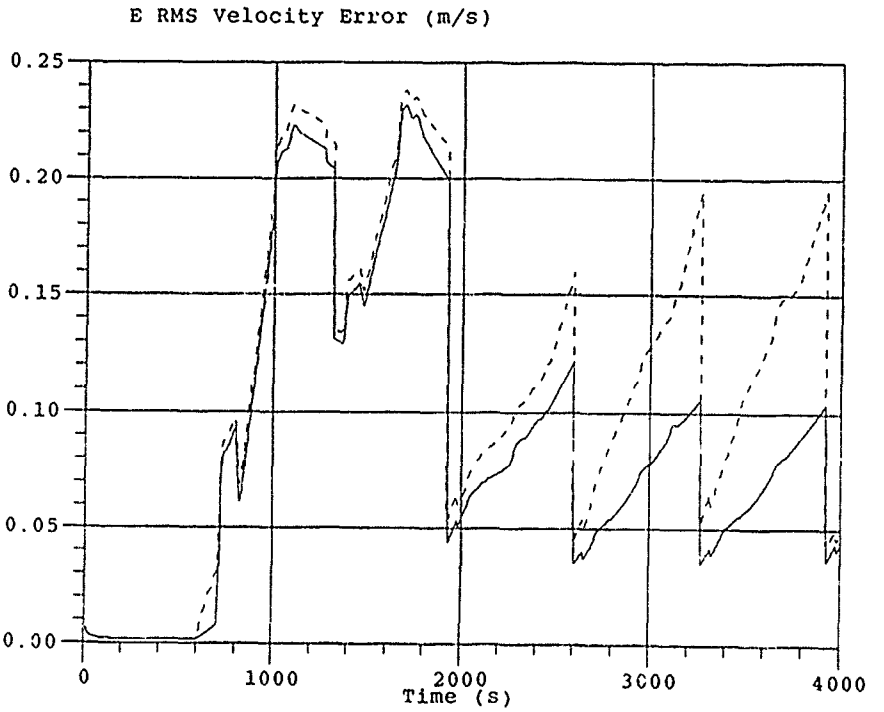


Figure 4: RMS Error from the Truth and Filter Models

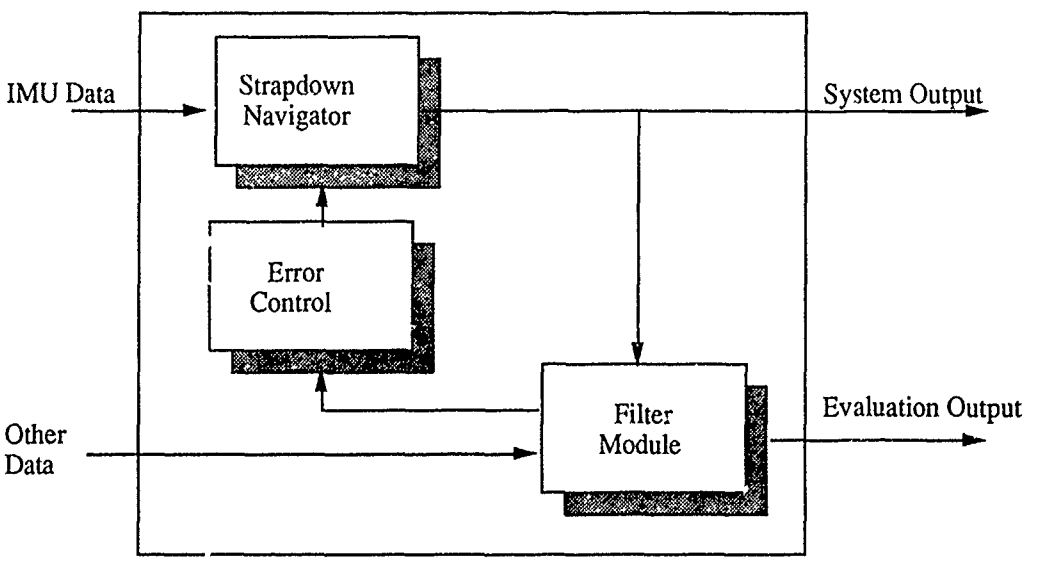


Figure 5: Functional Diagram of the Simulation/Processing. Closed Loop Operation

Here the Navigator Module represents a strap-down navigator which integrates incremental angles and velocities from the IMU and outputs position, velocity and attitude. The Filter Module incorporates a generic Kalman filter and user specific procedures which perform initialization, measurement construction, and definitions of system dynamics, statistics, and measurement models. The Filter Module has a built in mechanism for selection of an arbitrary subset of system states and measurements which provides the user with flexibility needed in Kalman filter design and the overall system performance analysis.

The Kalman filter provides optimal estimates of the inertial errors (error state vector) and associated error covariances. The implementation of the Kalman filter incorporates Prefilter, Update, and Extrapolation functions. The Prefilter computes a weighted

average of the error measurements (observations) computed during each Kalman filter update interval and outputs a single averaged measurement for processing by the filter at the end of each update interval. This allows the Kalman filter to be iterated at a low rate while incorporating higher-rate measurement data into the filtering process.

The update algorithm processes prefiltered measurements to obtain error state and covariance estimates, while the extrapolation propagates the error state and covariance between the updates. They are implemented using Bierman's U-D factorized method [5] where the estimation error covariance is expressed in terms of its upper triangular (U) and diagonal (D) factors, viz.:

$$P = UDU^T.$$

Both the update and extrapolation are done entirely in terms of the U-D factors. The extrapolation of the covariance involves the use of a Modified Weighted Gram-Schmidt Orthogonalization algorithm [5]. Sequential scalar measurement processing is used in the update algorithm, resulting in increased algorithm efficiency and flexibility, and allowing for simple processing of irregular measurements or easy rejections of erroneous measurements. This approach also ensures numerical stability.

The Error Control Module resets the Navigator's integrators using error estimates from the filter, thus closing the Navigator-Filter loop. The error control function can be deselected in which case the error estimates from the filter would be applied externally to the Navigator's outputs (see Figure 1 (a)).

Figure 6 illustrates heading output from postprocessing of recorded helicopter data using the Full Scale Simulation program. The sensor data were recorded during development flight trials for the HINS system [1],[2] currently under development for the Canadian Department of National Defence. The solid line indicates the error constructed by differencing the HINS heading output with the reference. The broken lines form the RMS error envelope using RMS error estimates from the Kalman filter.

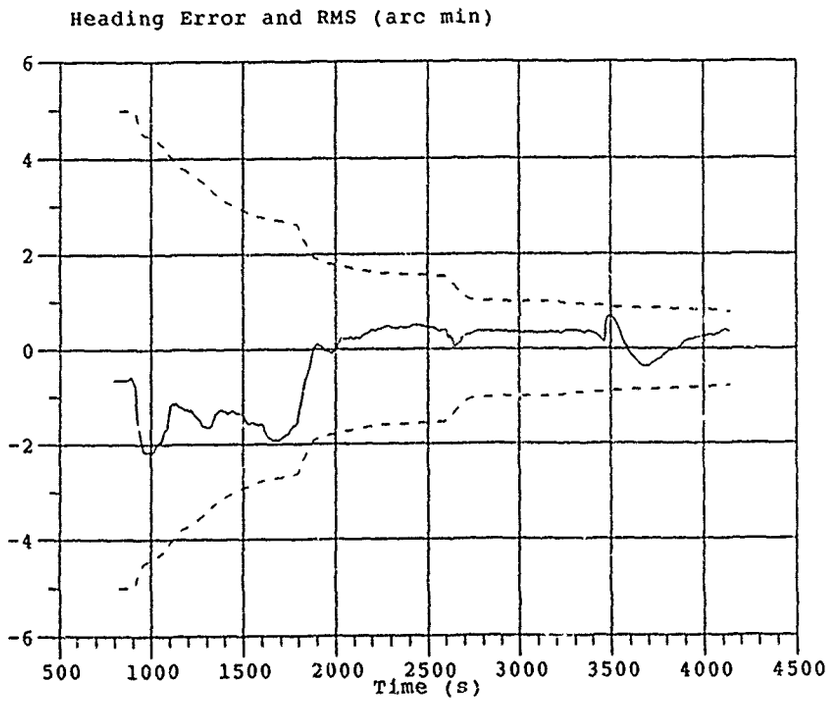


Figure 6: Processing Real Data Using Simulation Platform
Heading Error and Filter RMS

Resets Processing

High performance integrated inertial navigation systems use a Kalman filter to compute the optimal navigation solution from the data provided by an inertial navigation system (INS) and various navigation aids such as GPS and Doppler. The error estimates of the Kalman filter are used to reset the INS when the navigation system operates in feedback mode. A Reset Removal Procedure (RRP) has been developed by Honeywell's Advanced Technology Centre for recovering, during post flight processing, a high fidelity approximation to the unaided "pure inertial" navigation solution that would have been produced by the INS had it not been operated in feedback mode. The procedure is of great value in carrying out a performance evaluation of an integrated inertial navigation system, by effectively broadening the scope and extent of actual physical flight trials. The RRP is described and discussed in detail in [7].

On any particular flight, both feedforward and feedback modes may be used. For example, the navigation system may operate in feedback mode until it is adequately aligned, after which it may run in feedforward mode. Switching to feedforward mode may be desirable to attain higher system immunity to sensor failures - if the INS were operating in closed loop mode and the performance of an aiding sensor degraded, the INS solution might be corrupted before the failure had been detected.

Robustness of the navigation system in the presence of navigation sensor failures is extremely important for the demanding operational roles that systems such as HINS are required to play. HINS has a failure detection, isolation and reconfiguration (FDIR) capability which allows it to continue to obtain the optimal navigation solution for the available healthy sensors after one or more sensor failures.

In order to evaluate thoroughly the performance of such an integrated system, it is thus necessary to investigate the performance obtained for various subsets of the navigation sensors. Since flight trials are expensive, it is desirable to obtain as much information from each flight as possible. Accordingly, during the flight, the data output by the various navigation sensors are recorded along with Kalman filter data and any INS reset vectors for post flight analysis. If the INS completes an accurate ground alignment prior to the flight, the whole mission may be carried out in feedforward mode, so that the INS is unaided (apart from stabilizing the vertical channel). Then in post flight processing of the recorded data, the performance of the navigation system for various subsets of the navigation sensors may be accurately evaluated. However, if the INS is not well aligned before the flight, operation in feedback mode is essential, at least until the INS is adequately aligned. In this case, the resulting recorded INS solution is not "pure inertial", so it cannot be used directly for evaluating the performance of the navigation system with different configurations from that which was actually flown.

The Reset Removal Procedure (RRP) allows a high fidelity approximation of the "pure inertial" INS solution to be recovered, during post flight processing, from that generated during operation in feedback mode using the reset vectors and Kalman filter data. In other words, the RRP computes an approximation of the INS solution that would have been generated had the INS not been operated in feedback mode. This "open loop" INS solution may then be integrated with any desired combinations of aiding navigation data enabling the evaluation of the performance of the navigation system under a variety of scenarios.

Optimal Smoother

The Optimal Smoother is an implementation of a modified Bryson-Frazier smoother [6]. It is fully compatible with the NAVPACK Covariance Analysis and Full Scale Simulation/Processing programs. The Smoother operates on outputs from these programs to generate improved (smoothed) error estimate and/or estimation error covariance time histories. In applications in integrated navigation, the Smoother is used to post-process navigation error estimates and thereby obtain a more accurate estimate of navigation errors. These are then typically used to perform post-mission corrections on the recorded navigation data. The corrected position and velocity may become a reference against which the uncorrected navigation data is evaluated.

EVALUATION PROGRAMS AND THE BEST ESTIMATE OF TRAJECTORY

The evaluation programs compare the data being evaluated with corresponding reference data and plot the results. Navigation errors can be plotted together with Kalman filter (or smoother) RMS error estimates as in Figure 6.

In the simulation environment this evaluation is rather straightforward since the reference navigation data is naturally given by the Trajectory Generator's output. This is not the case however when recorded data from real sensors from flight tests is used. During these tests the ideal situation would be to have an independent system such as CIRIS at the Central Inertial Guidance Test Facility (CIGTF) at the Holloman Air force Base, New Mexico, to provide the reference data. This approach is in most cases prohibitively expensive.

An alternate method of providing the reference is to construct the Best Estimate of Trajectory (BET) by blending available sensor data such as GPS and INS with additional data such as position and zero velocity measurements at surveyed reference points. If the INS data was recorded in the feedback mode, the unaided INS data must be recovered using the Resets Removal Program, before this blending function can be performed. Then the accuracy of the blended reference trajectory can be further improved by back-smoothing using an optimal smoother.

To support this method, data from all of the sensors has to be recorded independently of its use by the system in generating the integrated navigation solution. This allows sensors to be configured out of the system's navigation solution during actual tests, yet still be recorded so they could be used to generate the reference data.

SUMMARY

The NAVPACK Software Package has been developed by Honeywell's Advanced Technology Centre, for integrated navigation systems simulation and analysis. The package consists of trajectory generator, navigation sensor synthesis programs to simulate sensor performance, full scale simulator/processor, covariance analysis, optimal smoother and INS reset removal programs. The interfaces between the programs were designed to enable the user to combine these programs at the operating system level to perform simulation, post-processing and performance evaluation analyses. The NAVPACK software has been successfully used for supporting a number of programs undertaken by the Honeywell's Advanced Technology Centre such as the Helicopter Integrated Navigation System (HINS).

REFERENCES

[1] Zywiel, J. C. Webb, H. Russell, Test Program for Honeywell/DND Helicopter Integrated Navigation System (HINS), IEEE PLANS'90 Conference, Las Vegas, Nevada, 1990

[2] West-Vukovich, G., J. Zywiel, B. Scherzinger, H. Russell, S. Burke, The Honeywell/DND Helicopter Integrated Navigation System (HINS), IEEE AES Magazine pgs. 18-28, March 1989.

[3] Hepburn, J.S.A. and C.P. Doyle, Motion Compensation for ASTOR Long Range SAR. IEEE PLANS'90 Conference, Las Vegas, Nevada, 1990

[4] Zywiel, J. Z., J S.A. Hepburn, C.P. Doyle, B.M. Scherzinger, NAVPACK - A Package of Simulation Tools for Integrated Navigation Applications, IEEE PLANS'88 Conference, Orlando, Florida, 1988.

[5] Bierman, G.J., Factorization Methods for Discrete Sequential Estimation, Academic Press, New York, 1977.

[6] Arthur Gelb (editor), Applied Optimal Estimation, The MIT Press, Cambridge, Mass., 1974.

[7] Hepburn, J.S.A., J.Z. Zywiel and D.B. Reid, INS Reset Removal for Enhanced Post Flight Performance Analysis, IEEE PLANS'90 Conference, Las Vegas, Nevada, 1990

Application of Modelling and Signal Processing in Air Defence

Dr.Ing N.Murty Vepa Dr.Ing Wolfgang Kreuzer
Co.traves GmbH
Winterspürer Straße 17-19
D-7768 STOCKACH
GERMANY

SUMMARY

Modelling plays an important role in improving the performance of air defence systems. Simple and effective models that match the real world are necessary, to process sensor signals and to estimate / predict flight path trajectories of air targets in real time, to the accuracy needed by the air defence fire control systems. The scope in air defence consists of modelling of sensors and effectors, noise and errors of sensors, flight path profiles of targets, target noise (e.g. glint) and ballistics. Models and actual measurements form the basis for fire control algorithm design. Based on the experience gained in the development of a number of high performance air defence systems, this paper describes the techniques of modelling and sensor signal processing in real time. Critical problems that will be encountered in land based, land vehicle mounted and ship based air defence systems are pointed out. Modelling of target types and target maneuvers is discussed. Methods of employing modelling and stochastic optimal estimation techniques to estimate and predict target flight trajectory in real time are explained. Both active and passive sensors are dealt with and factors to be considered in multi sensor suite selection and integration are pointed out, and a novel concept for an integrated sensor (WHISS) is briefly described. Use of multi sensor data fusion and advantages gained as a result of multi sensor synergism are presented.

State of the art stochastic optimal estimation techniques enabled to develop a target type / maneuver adaptive filter bank, that estimates the target flight trajectory accurately in real time, and predicts the flight path to the accuracies needed by the fire control system. Various target types and their flight path profiles and navigation laws are taken into consideration to tackle all types of threat situations. Compensation of hull motions / deformations, which are estimated from the measurements of the inertial sensors in real time, provides the capability to transfer target data from one module to another at a different location without losing the accuracy, which is very important for flexibility in the use of resources, and therefore contributes to enhance the potential applications of the system. Use of multi sensors and data fusion improves the coverage, eliminates multi path effects, and contributes to the reduction of reaction time and improvement of performance using redundancies. Real time algorithms and their realization, which involves real time parallel processing, is a challenging task. Expert Systems can be employed for interpretation of data, planning, coordination and engagement tasks. All these complex tasks have to be optimally combined and integrated to realize a high performance air defence system, for which total simulation of the system is one of the important means.

LIST OF SYMBOLS

$\underline{a}$	Acceleration	γ	Elevation Angle
$\underline{b}$	Acceleration (Flight Path Fixed)	γ_G	Elevation Angle (Gun)
$\underline{b}_{PN}$	Acceleration due to Proportional Navigation	γ_R	Elevation Angle Measurement (Radar)
$\underline{C}$	Measurement Matrix	$\delta\gamma_R$	Off Bore Sight Angle in Elevation (Radar)
C_{PN}	Navigation Constant	$\delta\sigma_R$	Off Bore Sight Angle in Azimuth (Radar)
d, $\underline{d}$	Distance	Δ_{rR}	Measurement Error of Range (Radar)
$\underline{D}$	Parallax Vector	Δv	Velocity Increments
$\underline{e}$	Unit Vector	$\Delta\Theta$	Angular Increments
$\underline{f}, \underline{g}(.)$	Vector Functions of (.)	$\Delta\gamma_R$	Measurement Error of Elevation (Radar)
$\underline{K}$	Kalman Gain	$\Delta\sigma_R$	Measurement Error of Azimuth (Radar)
N(.,.)	Normal Distribution (Mean, Variance)	Λ	Probability of Maneuver Change
$\underline{p}$	Position	ν	Measurement Noise
$\underline{P}$	Covariance Matrix of Estimation	Θ	Model Parameter
$\underline{P}'$	Covariance Matrix for one Step Prediction	ξ	Modelling Error

Pb(.)	Probability of (.)	ρ	Roll Angle
r	Range	σ	Azimuth Angle
$\underline{R}$	Covariance Matrix of Measurement Noise	σ_G	Azimuth Angle (Gun)
res	Residue	σ_{LOS}	Angle of Line of Sight
s	Laplace Operator	τ_m	Time Constant of Maneuver
$\underline{I}$	Transfer Matrix of one Step Time Delay	$\underline{\Phi}$	Transition Matrix
$\underline{I}_{ab}$	Transformation Matrix (b->a)	χ_M	Course Angle of Missile
v, $\underline{v}$	Velocity	ψ	Attitude Angle in Azimuth
$\underline{v}_M$	Velocity of Missile	ψ_R	Azimuth Angle Measurement (Radar)
W	Weighting Factor	τ_{tg}	Time to go
$\underline{x}$	State Vector	$\langle .,. \rangle$	Scalar Product
$\hat{\underline{x}}$	Estimated State	Matrix:	Thick, Capital Letter, Underlined
$\underline{x}'$	One Step Predicted State	Vector:	Thick, Small Letter, Underlined

1 INTRODUCTION

Fig. 1 illustrates signal processing applied in the fire control of an air defence system. Models (of targets, sensors and effectors) are widely employed in the Simulation, design, development and operation of air defence systems. Validated models are a treasure, which form the basis for sensor signal processing (e.g. for air space surveillance, target classification / identification, flight trajectory estimation and prediction, kill assessment etc.) and fire control. To meet the real time requirements, models have to be simple and effective in matching the real world. Targets with high maneuver capabilities and various flight path profiles, a multi target threat environment, all weather capabilty, and resistance to counter measures necessitate a multi sensor system consisting of both passive and active sensors. While missile based defence is effective; due to the short reaction times needed for last ditch defence against leakers, automatic rapid fire guns are employed together in a multi layer defence. Because of no (or limited in the case of smart ammunition) correction capability of ammunition after it is fired, accuracy requirements of gun based systems are very stringent, compared to the missile based air defence. Processing of a multi target threat scenario in an all weather environment requires a careful selection and integration of the multi sensor suite. To achieve the total synergy offered by a multi sensor system, optimal estimation and data fusion techniques based on stochastic theory, which can be realised in real time are needed. Dynamic characteristics of the platform (land based, terrestrial vehicle / ship mounted) as well as structural motions of hull degrade the system performance and hence have to be measured and compensated for, in real time. Modelling & simulation of the total system by integrating all the above complex tasks is an important means to optimize and realize a high performance air defence system.

Based on the development and manufacturing experience of SKYGUARD, GEPARD and SEAGUARD, this paper deals with modelling and multi sensor signal processing in air defence. Multi sensor suite selection and integration are discussed and the functional principle of an integrated novel multi sensor WHISS, which is under development is briefly described. Models for targets, sensors and effectors are dealt with and modern optimal estimation techniques for estimation and prediction of target flight trajectories are presented. Influence of hull motions and methods of compensation are explained. Advantages offered by multi sensor data fusion in air defence are pointed out. Real time processing implementation aspects, and possible use as well as state of the art of expert system technology are presented.

2 MULTI SENSOR INTEGRATION

The deficiencies of active (Radar / Laser) and passive (IR / acoustic / optical) sensors for both surveillance and tracking of air targets are well known, and it is a common knowledge that a solitary sensor can not perform all the required functions, day and night under all weather conditions for all threat scenaria. As such, there is a need to integrate multiple sensors of different type, which increases resistance to counter measures and provides a graceful performance degradation capability in case of failures. To realize multi sensor integration and achieve the total benefits of synergy (with the motto "The whole is greater than the sum of the parts"), proper care has to be taken in the selection and physical integration of sensors, as well as in sensor data fusion. Functional and operational requirements of the mission, constraints on the reaction

times, envelopes of coverage, threat characteristics and performance deficiencies of each individual sensor are some of the factors that guide the selection process of the multi sensor suite. Multi sensor integration requirements, sensor survivability, reconfigurability and mission adaptability, as well as functional redundancy and backup capability are key factors that effect the choice of sensor location.

While integration of different types of individual sensors is still the common practice, there are recent attempts to develop mixed sensors, which combine both active and passive sensors. WHISS (WHIspering Search System) which combines both Infrared Search System (IRSS) and Search Radar (Ref. 1), is one of the typical examples of a mixed sensor system in which both active and passive sensors are integrated in an optimal way. Fig. 2 illustrates the main functions of WHISS (viz. 1. Passive search, 2. Radar activation for a short interval, and 3. Correlation of IR and Radar target data), which are briefly explained as follows.

1. Passive search: Thermal radiation received by the IR search system is processed in the pre screener for possible targets, which are analysed in detail in the target extractor by comparing the target information with the clutter map. Possible targets are reported to the Radar.

2. Radar activation for a short interval: Search Radar with its rotating antenna transmits a beam for a very short interval in the direction given by the IRSS, and delivers the received echo signal on the Radar / IRSS target correlator.

3.Correlation of target information: If Radar confirms the target detected by IRSS, target information is reported to the tracking system. Otherwise target coordinates are registered in the clutter map, which helps to control Radar emission in that direction.

3 MODELLING

3.1 General Considerations

Mathematical models are a compact way to summarize our knowledge about a process, and are fundamental to science and engineering. They are derived from basic laws of physics and experimental data, and are validated for their adequacy of application. It is important to avoid both over simplification and over complication. Models are the heart of good system engineering, and choosing / deriving right models that match with the real world is crucial to success. As mathematical models, generally transfer functions, difference / differential equations, state space representation, Max. / Min. values and boundary curves, statistical techniques and tabular representations are employed. Models are employed in design, simulation and operation of an air defence system for air space surveillance, threat analysis, weapon planning, allocation, scheduling and engagement as well as for fire control purposes. An air defence system consists of a sensor suite, command and control (C^2) system and effectors (gun / missile based) supported by electronic support / counter measures. As flight trajectory of the ammunition round can not be corrected, gun based systems have comparatively stringent requirements on the accuracy of modelling and signal processing. In this paper, gun based systems are taken into consideration, the requirements of which in general cover missile based systems as well.

High accuracy target trajectory prediction requirements of a gun based air defence system demand modelling (Fig. 3) of targets, sensors, processor (C^2) system and effectors in such a way that system dynamic characteristics and accuracies, as well as reaction times are truely represented. Models are extensively employed during concept, design and development phases for the analysis of reaction times, errors and stability, as well as budgeting of errors and reaction times. Validated models are further employed in the operational software for tracking of targets, estimation and prediction of flight trajectories, and weapon engagement. Pre established models, as well as adaptive models (which adapt online to the particular operational conditions) are employed.

3.2 Modelling of Fire Control Sub Systems

Pre established models can be employed to the elements of the fire control process, the characteristics of which are reproducible. One time accurate modelling is sufficient enough to determine the mathematical models for this class. Models which are matched to changing conditions of operation, based on direct measurements also belong to this class of models, in which pre determined sensitivity functions and coefficients are employed. A typical example for this is the ballistic model. Mathematical models for sensors, effectors and processor (C^2) system belong to this class.

A sensor system usually consists of the sub systems i) Inertial attitude reference unit ii) Servo system iii) Synchros / Coders iv) Tachos / Rate gyros v) Radar / Electro optical tracker and vi) Laser range measuring unit. In addition to i) to iv), an effector system includes gun and ammunition. All sub system characteristics that influence the performance have to be taken into consideration in the modelling. For the fire control process of gun based air defence, various sub systems can be modelled as illustrated in Table 1. Ballistics of chosen ammunition is one of the important models

in the fire control computation. Generally 3 DOF (Degree Of Freedom) models instead of 6 DOF are used for the trajectory of the projectile to reduce online computational load. Special models which are based on validated firing tables of the given ammunition are employed for the computation of the fire control. The values listed in the table are approximated by analytical functions with the aim of minimizing the number of parameters and computation effort. Meteorological data (temperature, pressure and wind) and muzzle velocity of the ammunition round effect the ballistic computations significantly. As such their influence is taken into consideration in the mathematical models.

3.3 Target Modelling

Fig. 4 gives an overview of the target models. General target models for all possible classes are implemented and the choice of a particular model is made depending on the actual mission. As a special case, target model that describes the trajectory most precisely can be selected online automatically or by an operator. Due to the adaptation needed, this type of models are called adaptive models. Referring to Fig. 4, depending on the mission it may be necessary to differentiate and choose between general models (which are often employed) and target specific models, which permit a very accurate characterization of trajectory. Depending on the accuracy of modelling target kinematics, noise in the state estimation can be suppressed, and extrapolation error for the time of flight of the projectile can be reduced. As such, target modelling deserves special attention. Target kinematics (Eq. 1) can be described by the dynamics and the measurements.

$$\begin{aligned} d/dt(\underline{x}) &= \underline{f}(\underline{x},\underline{\theta},t) + \underline{\xi} \\ \underline{y} &= \underline{g}(\underline{x},\underline{\theta},t) + \underline{v} \end{aligned} \qquad (1)$$

A general target model usually employed is from "Singer" (Ref. 2), which is modelled in the stationary cartesian coordinates. In this model (Fig. 5) acceleration components along the coordinate axes are considered to be independent of each other and can be represented by a first order Markov process as in Eq. 2. With a proper selection of the parameters, it is possible to represent various classes of targets corresponding to their maneuverability. Target velocity and position are determined by integration.

$$\begin{aligned} d/dt(\underline{a}) &= -(1/\tau_m)\cdot\underline{a} + \underline{\xi} \\ d/dt(\underline{v}) &= \underline{a} \\ d/dt(\underline{p}) &= \underline{v} \end{aligned} \qquad (2)$$

One of the major drawbacks of general models is that target classes are characterized not by the deterministic part of the model, but by modelling errors. Modelling errors can be reduced by employing target specific models (air targets in Fig. 4), in which apriori knowledge can be taken into consideration. As target model becomes more precise, noise suppression and trajectory prediction will be improved. However, as target kinematics is described more and more precisely, it is necessary to be accurate in assigning the target to the model in the online process. A mistakenly wrong assignment can amount to a significantly large modelling error, even though it is set to a minimum in the model.

By taking special kinematic characteristics into consideration, modelling the flight paths of aircrafts and drones can be significantly improved. Their maneuvers are mainly effected by transversal aerodynamic forces. So, the direction of the resulting acceleration is normal to the flight trajectory. By modelling the target accelerations in the flight path fixed coordinate system (Fig. 6), (with different maneuverabilities along the length axis in the flight direction and transversal to it), this effect can be taken into consideration. As integration for determining the velocity and position is carried out in the stationary reference system of the fire control process, transformation from the flight path fixed system is needed, which depends on the target kinematic state, resulting in a nonlinear model as follows.

$$\begin{aligned} d/dt(\underline{p}) &= \underline{v} \\ d/dt(\underline{v}) &= \underline{T}_{ab}\cdot\underline{b} \\ d/dt(\underline{b}) &= -(1/\tau_m)\cdot\underline{b} + \underline{\xi} \end{aligned} \qquad (3)$$

Transversal aerodynamic force required for a maneuver is generated through the lifting force of the wings. The direction of possible flight acceleration is indicated (Fig. 7) by the roll angle, which can be taken into consideration in the target model. If this roll angle is indicated by the tracking sensor, reaction time needed to know a maneuver is significantly reduced.

Modelling of target kinematics is possible in the case of missiles as well, especially when position of the object under attack is known (Ref. 3), e.g. in the case of ships with gun based last ditch defence. For missiles, the angle of LOS (line of

sight) is measured and the course is corrected (e.g. by proportional navigation (Fig. 8) where change of course angle is proportional to the angular rate of LOS).

$$d/dt(X_M) = C_{PN} \cdot d/dt(\sigma_{LOS}) \qquad (4)$$

From relative kinematics of the target missile and the object under attack, most of the target acceleration can be determined in the flight path fixed coordinated system, and can be taken into consideration in the deterministic part of the target model.

$$\underline{b}_{PN} = C_{PN} \cdot (T_{tg}\underline{v}_M + \underline{d}/\langle \underline{e}_d, \underline{e}_v \rangle^2)/T_{tg}^2$$
$$T_{tg} = d/v_M \qquad (5)$$

Accelerations, not considered above, can be modelled and included in the target model as a 1'st order Markov process. Velocity and position can be obtained by integrating in the stationary coordinate system of the fire control process. Similar to aircrafts, the resulting target specific model of the missile is also nonlinear.

$$d/dt(\underline{p}) = \underline{v}$$
$$d/dt(\underline{v}) = \underline{T}_{sb} \cdot \underline{b}_{PN} + \underline{a} \qquad (6)$$
$$d/dt(\underline{a}) = -(1/T_m) \cdot \underline{a} + \underline{\xi}$$

Helicopters can be modelled with sufficient accuracy using the general target model.

3.4 Model Adaptation

Mathematical models are chosen for a typical target class and are adapted online to the individual characteristics of the target to be engaged. One of the methods of model adaptation is to extend the target states by the model parameters to be identified, which does not permit the use of simple filters due to the resulting nonlinearity. Further, computation requirements needed for model extention are overproportional, and time needed for an accurate identification of model parameters is too large. As such this method is applied rarely.

Adaptation can also be implemented using model balancing. Models can be balanced online by employing either (i) model balancing using weighted superimposition (Fig. 9) or (ii) selecting from target specific models (Fig. 10). Both methods have access to the given set of target models, and determine the adaptive model from them. They are characterised by a reasonably good speed of adaptation. In model balancing by weighted superimposition, the extent of modelling errors for each model is determined, based on which weighting factors for model superimposition are determined in such a way that a good consistency between the process and the adapted model is obtained. This method is applied for the case of general target models, where state estimation and model balancing are done by means of a general purpose filter bank. Model balancing by model selection is mainly applied for the case of target specific models. Model selection is done either manually by an operator, or automatically by means of online data processing. The criteria for model selection are the special kinematic features of a particular target class and target state.

4 FIRE CONTROL PROCESSING

Fire control processing, the main task of which is to compute weapon pointing parameters from target measurements for maximum probability of hit, is the core of operational data processing of a modern air defence system. In addition, it computes signals necessary for sensor follow up and for supporting the fire control doctrine. Fig. 11 illustrates main data flow and influence of models in fire control data processing, which is implemented in three main stages viz. 1) Pre filter 2) Tracking filter and 3) Prediction and ballistic computation.

4.1 Pre Filter

From the available measurements in the various sensor reference frames, pre filter generates the target measurements in the reference coordinate system of the fire control process. Fig. 12 illustrates the combination of signals in pre filter, which consist of 1) Bore sight error of the tracking sensor 2) Sensor follow up angle and 3) Attitude reference in the case of a single tracking sensor. Corresponding signals have to be taken into consideration in the case of multiple tracking sensors. In the design, dynamic transfer function and in the implementation, position and attitude information of the sensors are employed. As such, pre filter structure and design are strongly influenced by the dynamic characeristics and location of sensors. Further, plausibilty tests for sensor measurements are carried out based on predicted values. In case of redundant measurements with multiple tracking sensors, the measurements are verified by comparing with each other, taking specific sensor characteristics into consideration (e.g. moving away of the range gate in the case of Radar jamming).

4.2 Tracking Filter

In the tracking filter, target state (position, velocity and acceleration) is estimated based on stochastic optimal estimation techniques. As position and velocity can be determined by integration of acceleration in a stationary reference frame, computations are done in a quasi inertial reference coordinate system. While estimation results are on one hand dependent on the quality of measurement (noise), on the other hand the employed target model has a significant influence. Application of good models with very small modelling errors suppresses noise and is the basis for trajectory prediction. Taking computation requirements into consideration, linear models which are uncoupled in all three axes are preferably employed. As target model, Singer model can be employed. Original measurements, which are uncoupled in polar coordinates, are as well assumed to be uncoupled in the cartesian system. Various target maneuvers are taken into consideration by choosing a number of simple target models with different parameter sets. By means of adaptive superimposition, the target model with best fit is determined, which leads to a general purpose filter bank explained as follows. The states of each of the n chosen target models are estimated by means of the corresponding elementary Kalman filters, the structure of which is shown in Fig. 13, together with the resulting filter bank. The filter equations can be written as follows:

$$
\begin{aligned}
\underline{K}_k &= \underline{P'}_k \cdot \underline{C}^T{}_k \cdot (\underline{C}_k \cdot \underline{P'}_k \cdot \underline{C}^T{}_k + \underline{R}_k)^{-1} \\
\hat{\underline{X}}_k &= \underline{X'}_k + \underline{K}_k \cdot (\underline{Y}_k - \underline{C}_k \cdot \underline{X'}_k) \\
\underline{P}_k &= \underline{P'}_k - \underline{K}_k \cdot \underline{C}_k \cdot \underline{P'}_k \qquad (7) \\
\underline{X'}_{k+1} &= \underline{\Phi}_k \cdot \hat{\underline{X}}_k \\
\underline{P'}_{k+1} &= \underline{\Phi}_k \cdot \underline{P}_k \cdot \underline{\Phi}^T{}_k + \underline{Q}_k
\end{aligned}
$$

By evaluation of the residues of each elementary filter, the weighting factors of each of the models / elementary filters can be determined. Assuming quasi stationary target maneuvers, the weighting factors correspond to the probability of the particular target model. The weighted sum of each estimate is the result of the adaptive estimate. The corresponding error covariance can be computed accordingly as given below.

$$
\begin{aligned}
W_{i,k} &= Pb(res_{i,k}) \cdot \underline{\Lambda} \cdot W_{i,k-1} / \sum_j Pb(res_{j,k}) \cdot \underline{\Lambda} \cdot W_{i,k-1} \\
\hat{\underline{X}}_k &= \sum_j W_{j,k} \cdot \hat{\underline{X}}_{j,k} \qquad (8) \\
\underline{P}_k &= \sum_j W_{j,k} \cdot (\underline{P}_{j,k} + [\hat{\underline{X}}_{j,k} - \hat{\underline{X}}_k] \cdot [\hat{\underline{X}}_{j,k} - \hat{\underline{X}}_k]^T)
\end{aligned}
$$

This filter bank can be employed for a wide spectrum of targets and as such is named as "General Purpose Filter Bank" (GPFB). A drawback of the GPFB is that the maneuver characteristics of the targets are described only through modelling errors: i.e. GPFB has a narrow band width for low maneuver targets and a very good noise suppression. However, it has a relatively large band width and poor noise suppression for high maneuvering targets. Unfortunately, a simple and accurate mathematical model which can be applied in general for all possible target trajectories does not exist. Under special conditions, matched target specific models can be employed, resulting in good noise suppression for high maneuvering targets. As already mentioned in the previous section, target specific models are nonlinear and make extended Kalman filters necessary. The filter bank can be built corresponding to the tree structure of the target models (Fig. 4). Adaptation to the conditions valid at a given time is done by selecting the special filters (of target specific models), on the basis of the estimation results of GPFB. A selection logic examines whether assumptions for the validity of a special filter are fulfilled or not. If they are, the special filter takes over the state estimation process, and the GPFB processes further in the background. When the assumptions for the validity of the special filter are no more satisfied, the state estimation task is handed back to the GPFB.

4.3 Prediction and Ballistics

In gun based air defence, pointing angle and angular rate of the gun have to be computed (Fig. 14), for which crossing point of trajectories of both target and projectile are determined iteratively. From the estimated state (obtained from the tracking filter) and the mathematical model, target trajectory is predicted, where as flight path of the projectile is generated from the ballistic model. For precise hit point prediction, an usable target model and accurate modelling of the ballistics of the projectile are necessary. Ballistics are dependent on the meteorological data and muzzle velocity of the ammunition. The ballistic model makes it possible to correct the gun sight in elevation, and takes into consideration angular momentum of the projectile in azimuth. Wind is an additional source of error and is taken into consideration for computing gun pointing parameters.

5 SIMULATION

Mathematical simulation is an important means for testing and traceability (demonstration) of performance of air defence systems. Employing simulation, functional & operational performance can be proposed, corrections and changes can be initiated before realizing in hard ware. Statements regarding tactical mission planning, and system specifications resulting from it, can be made employing Monte Carlo simulation. Following (Fig. 15) is an example for functional and operational simulation that determines the performance of the fire control algorithms of a gun based anti aircraft tank (AAT).

Trajectories of various targets are measured, registered and corrected for errors by smoothing, and are arranged as per target class, maneuver characteristics etc. and are recorded on a data medium. Simulation program, recorded target trajectories and measurement noise (modelled based on sensor characteristics and environment) are employed in the simulation. Simulation program consists of fire control algorithms to be evaluated, and mathematical models for the sensor / effector system of the AAT. On one hand, the model has to compute the measurements of the AAT from the given target trajectory, which are taken as input to the fire control algorithms. On the other hand, pointing values of the gun have to follow the set values computed from the fire control algorithms. Thus, dynamical interaction between the equipment and the fire control computations can be taken into consideration, and incorrect over estimation of performance of the algorithms that may result due to idealised assumptions can be avoided. Set values for weapon pointing are registered and are evaluated by comparing with the ideal reference values, which are determined from the exact target trajectories. Using vulnerability models (in the most simplified version: effective target area), hit and kill probabilities can be estimated. Model for sensor / effector system of the AAT, illustrated in Fig. 16, describes the dynamics together with the servo system. Nonlinearities (e.g. in the tracking Radar), which influence the overall system performance can be included further.

6 MULTI SENSOR DATA FUSION

Synergism that results through the optimal use of both active and passive sensors improves (i) overall system's detection range, track accuracy, and integrated target identification, Kill assessment and ECCM / EMCON capabilities (ii) track initiation range and performance estimates of individual sensors. Further it contributes to (1) a reduction of false alarm rates and system false tracks, (2) an improvement of track maintenance for difficult targets, (3) a more graceful system degradation, and (4) elimination of multi path effects.

Assuming that detections of each sensor are independent events, overall detection probability of the total system with n sensors, at any range and for a given environment can be written as

$$Pd = 1-(1-Pd1)(1-Pd2)..(1-Pdn) \qquad (9)$$

By means of cueing one sensor with any other to search a specific area, detection range of the individual sensor can be increased further, e.g.: If cues (range & angle) are provided to the IR system, it can slow down the scan rate or lower its detection threshold at the appropriate location to improve the sensitivity.

Features of the operational environment can be estimated by using different sensors, e.g.: MFR (Multi Functional Radar) can help determine the extent of ducting phenomena and rain environment, regions of rain can be determined from VSR (Volume Search Radar) signals, and ESM can help locate and determine the exact nature of the ECM (Electronic Counter Measures). Logical combination of these reports can be used to get an useful environmental scenario (ducting, weather, jamming etc.), which allows to optimize the performance of each sensor. By means of cueing one sensor to another, the second sensor can allocate resources and initiate track immediately, which reduces the time for track initiation and thus increases the track initiation range. Different type of sensors can contribute to the fused system track accuracy by providing accurate measurements in various dimensions and using different regions of the electromagnetic spectrum.

Multi sensor synergism helps to track individual / group of targets, which can be difficult for individual sensors. This includes crossing, merging, separating, low probability of detection, high speed, near range, high acceleration or low elevation targets with multi path propagation. If two targets cross, all the accumulated knowledge of the two targets might be swapped (due to confusion) with an individual sensor. However, multi sensor data fusion uses all the available measured parameters to maintain tracks, and avoids a track swap. Fig. 17 shows the view of crossing targets as seen from an IRSTS (Infra Red Search and Track System) and Radar. The IRSTS could mistake the crossing targets for maneuvering targets and swap the track data. However Radar's range data helps to resolve the confusion. Targets with low probability of detection can be detected and tracked using multi sensor synergism. E.g. for the case of poor Radar detection of a low RCS target, system fused track file obtained through other sensors can provide enough information to periodically cue the Radar or allow the Radar to maintain the undetectable / poorly detectable target's track. In a similar way Radar

range data or the ESM's ability to descriminate particular targets could help the IRSTS maintain track of nearby or high dynamic targets.

The low elevation target with multi path propagation has plagued Radars and the so called mirror effect (Fig. 18) is detrimental to the performance of an air defence system. IRSTS can help resolve this problem by its relative indifference to multipath propagation. Multi sensor data can be used to expand the target signature which results in an improved target classification / identification capability. Multiple targets that are closely spaced are difficult to be resolved by a single sensor, which can be better solved using multi sensor data. E.g. Radar can help resolve two targets closely spaced in angle and being tracked as a single target by the IRSTS. Multi sensors can contribute to an improved kill assessment capability. E.g. Radar provides target SNR, doppler changes, cessation of down link if it exists, as well as multiple returns (from fragments). IR sensor provides angle position changes, changes in target signature as well as war head detonation detection. ESM indicates cessation of target emission. All the above mentioned advantages due to multi sensor synergism can be realised by controlling and combining multi sensor data. Ref. 4 illustrates different methods of data representation and association, including Dempster-Shafer method of evidential reasoning, which helps to utilize fully each sensor's information regardless of its form.

7 INFLUENCE OF HULL MOTIONS

Motions of hull structures of sensor / effector platforms may introduce significant errors which in turn degrade the performance of air defence (especially gun based) systems. Fig. 19 illustrates canting of the hull of an AAT, resulting due to the reaction forces of a fire burst. If this is not compensated by proper means, no distinction can be made between hull motion and real target motion. Hull motion may be interpreted as target maneuver, causing significant errors in the prediction of target motion, which degrades the performance of the fire control system. Real time compensation of hull motions, using gyro measurements is a means of overcoming this problem. The problem of hull motions plays a much more significant role in naval vessels due to bending and flexing of hull structures. The benefits offered by multi sensor integration and data fusion can be incorporated and utilised fully, only when needed alignment accuracies between various sensor and effector modules can be guaranteed under all environmental and operating conditions. Usually, each high performance sensor is inertially stabilized on naval vessels using a local inertial reference unit. From gyro and accelerometer measurements of the corresponding inertial reference units, it is possible to estimate in real time dynamic misalignments between any two modules resulting due to hull motions. Ref. 5 investigates a high accuracy dynamic alignment system, which estimates the misalignment angles between any two modules based on angular / velocity incremental measurements provided by the strapdown inertial reference units locally incorporated. Fig. 20 illustrates a dynamic alignment system for a modular system (with n modules). It allows unrestricted transfer of target data from any sensor to any other without loss of accuracy, improves the performance and makes it possible to exploit all the potential applications.

8 REAL TIME ASPECTS

An air defence system requires considerable amount of embedded computing resources. The embedded system must interact with multiple external devices and respond in "real time". Since external actions are asynchronous and potentially simultaneous, a concurrent structure is evident. A concurrent program specifies two or more sequential programs, that may be executed at the same time. This concurrency may be actual, having two or more processors, or apparent with interleaved execution on a single processor. The traditional approach to control concurrent processes has been through the use of a real time executive, based on the model of many tasks executing on a single processor. Executive is the program, that looks at the current state of the system and decides which task shall be executed next.The standard implementation language for embedded systems is Ada. Model of concurrency in Ada is different from conventional approaches. Its view of multi tasking is that tasks are concurrent asynchronous processes, potentially each with its own processor. The intrinsic feature for task communication and synchronisation is the rendezvous. Ada has no need for a real time executive. Instead there is a hidden run time system that underlies the Ada tasking model, but is not visible to the programmer.

If the required resources exceed a single processor, a multi processor system is needed. For the realisation of a multi processor system two questions are of global evidence: 1. Which architecture is to be used? 2. How are the processes distributed among the processors? Possible hardware architectures are: i. Multi Processor System (MP): Base of the architecture is a memory, which is accessible from all processors. Data is common on the memory. ii. Multiple Processors System (MPS): This architecture consists of pairs of processors and local memories, which are linked by a network. There is no common memory. Data exchange is made by means of the network. Possible principles for the distribution of the application are: 1. Dedicated Function Allocation: Functions are allocated to the processors. 2. Traffic Sharing: Independent data streams are exchanged between the processors. Distribution of the functions can be performed in a static or dynamic manner. 3. Dynamic Load Balancing: Processes are distributed according

to the current load of the processors. Architecture and distribution principles are chosen depending on the application. For larger systems, an MPS architecture and dynamic load balancing are prefered.

9 APPLICATION OF EXPERT SYSTEMS

Expert systems can support the following functions in air defence (Ref. 6). 1) Interpretation of data from various sensors for target identification and classification including IFF 2) Threat evaluation, Planning / Coordination / Engagement of sensors and effectors. Most of these tasks are time critical and have to be performed in real time, for which expert system technology is not yet mature. Features that effect the real time behaviour (Ref. 7) are used for comparison and evaluation of the expert system shells available on the market, and are summarized briefly as follows.

Features that effect real time behaviour:

1.Knowledge base: The size of it (which is mainly a function of number of rules and facts) determines the needed memory size. As the inference machine verifies the total rule base for each evaluation cycle, it has a tremendous effect on the run time characteristic of the expert system, most of the computer time being required for pattern matching.

2.Computing time: As path to a solution is determined dynamically by interpretation of the knowledge base, time needed for computation is normally larger compared to sequential / hard wired programs. Methods of implementation also effect computing time considerably (e.g. Garbage collection in Lisp).

3.Synchronization with external events: Real time expert system should have a process interface, which allows to read sensor data in real time. The expert system / implementation language should have the possibility to transfer sensor data directly into the knowledge base, time efficiently.

4.Controlled erasing of facts: As the time validity of sensor data, and the facts derived from, change during system operation, techniques must be implemented to guarantee the consistency of the knowledge base, and past knowledge elements which are no more valid should be removed.

5.Guaranteed response time: All results that can be used should be delivered within a defined time interval. As expert systems, in general employ non deterministic search strategies, it is not possible to guarantee the response times by dimensioning hard ware.

6.Continuous operation: Existing expert systems end their evaluation cycle if there are no more rules to be processed. Expert systems capable of real time processing should hand over to a waiting status which is revoked as new input data is received. Further, it is necessary to be able to interrupt the processing of knowledge by external events.

7.Integration of subprograms written in conventional languages: The expert system must have the possibility of integrating subprograms written in conventional languages e.g procedures for signal processing, data reduction, feature extraction and I/O processing.

Table 2 shows a comparision of the shells available in the market for the above real time features. Investigations have shown that none of them are suitable for the development of real time expert systems for application in air defence. The main draw backs are 1) no process interface 2) no possibility of permanent operation because of garbage collection 3) too much overhead as far as memory and computation time are concerned, which is due to the wide application spectrum of expert system shells and 4) no possibility of extention for including new system functions.

While small or medium sized projects can be realized by simple rule based expert system shells which are capable to generate hand wired Ada programs, one needs in general a comfortable and high performance shell such as G2. In order to integrate the expert system in the operating system, possiblities to convert expert systems to Ada are needed, for which no development tools are offered today in the market. Moreover performance capabilities of today's hardware are not adequate to fulfill the stringent real time requirements. Today no commercial development system exists which guarantees a successful application of this technology for realising real time expert systems. Innovative developments (e.g. parallel computer), however, may provide successful solutions in the future.

10 CONCLUSIONS

Modelling plays an important role in sensor signal processing, target flight trajectory estimation and prediction, as well as in the computation of the ballistics of the projectile, the real time implementation of which is essential to the succesful performance of an air defence system. Application of modern state of the art optimal stochastic estimation techniques, in combination with realistic target modelling, result in GPFB (General Purpose Filter Bank) or extended Kalman Filter, which can be adapted to

every mission situation to estimate and predict the target trajectory to the needed accuracy. Sensor data alignment (which consists of time validity, parallax correction as well as alignment and transformation of coordinates) in real time is crucial to implement multi sensor data fusion and realise the benefits of synergy resulting from it. Compensation of hull motions / deformations in real time enables to align data and provides the capability to transfer target data from any sensor to any other sensor / effector at a different location. While expert systems promise a number of applications which can further enhance the performance of an air defence system, current state of the art of expert system technology is not suited for real time applications.

11 REFERENCES

(1) B. Erzinger, "WHISS - Das flÜsternde Suchsystem", Hausmitteilung der Contraves AG, ZÜrich, CONTACT, No. 2, 1987.

(2) R. A. Singer, "Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets", IEEE Trans. Aerospace and Electronic Systems. Vol. AES-6. No.4, July 1970, pp 473 - 483.

(3) W. Kreuzer, "Kalmanfiltertechnik in der Feuerleitung von SEAGUARD und GEPARD", CCG-Lehrgang A1.04: Zielidentifizierung, Zielverfolgung mit Kalmanfilter, Sept. 1988.

(4) S. S. Blackman, "Multiple Target Tracking with Radar Application". 1986, Artech House Inc.

(5) N. M. Vepa, "A Dynamic Alignment System for Applications on Flexible Platforms such as Ships", Symposium Gyro Technology, Sept. 19/20, 1989, Stuttgart, Germany, pp 16.0 - 16.13.

(6) J. E. Franklin et al., "Expert System Technology for the Military Selected Samples", Proceedings of the IEEE, Vol. 76, No. 10, October 1988 pp 1327 - 1366.

(7) T. KrÜger "Real Time Expert Systems for Air Scenario Evaluation", Contraves internal Report, 1989.

Sub System	Model		
	Dynamics	Boundaries	Errors
Initial Attitude Reference Unit	1	Measurement Range Environment (Shock, Vibration)	Drift
Synchro / Coder	1	Measurement Range	Offset
Tacho	1	Measurement Range	Offset Scaling Error
Servo System	$\frac{1}{(T_1+1)(T_2+1)}$	Angular Velocity / Acceleration limits	Offset
Tracking Radar	e^{-Ts}	Range Antenna Beamwidth	Measurement Noise; Target Noise
Tracker	e^{-Ts}	Resolution	Measurement Noise
Laser Range Measurement Unit	e^{-Ts}	Range	Measurement Noise
Gun	-	Rate of Firing	Weapon Dispersion
Ammunition	Firing Table 3/6 DOF	Range	Ballistic Dispersion
Processor Unit	e^{-Ts}	Operations/sec.	Rounding Errors

Table 1: Modelling of Sub Systems

	Lisp Based Shells							Workstation Shells		
	Real Time									
Features	Picon	G2	Babylon	Knowledge Craft	Joshua	ART	KEE	OPS5	S1	Nexpert
Processing Speed Rules/sec.	?	?	?	?	200	?	?	?	?	?
Possibility of Tuning	o	o	o	o	X	o	o	o	o	o
Guaranted Response Time	(X) Garb.	(X)	o	o	o	o	o	o	o	o
Controlled Removal of out dated Facts	o	X	o	o	o	X	X	o	o	X
Synchronisation with external Events	X	X	o	o	o	o	o	o	o	o
Continuous Operation	(X) Garb.	X	o	o	o	o	o	o	o	o
Integration of Conventional Modules	Lisp	o	Lisp	Lisp	Lisp, Ada	Lisp	Lisp	Pasc. Ada, Fort.	C od. Lis	Pascal, Ada, Fortran

x: fulfilled o: not fulfilled ?: no data available

Table 2: Comparison of Expert System Shells

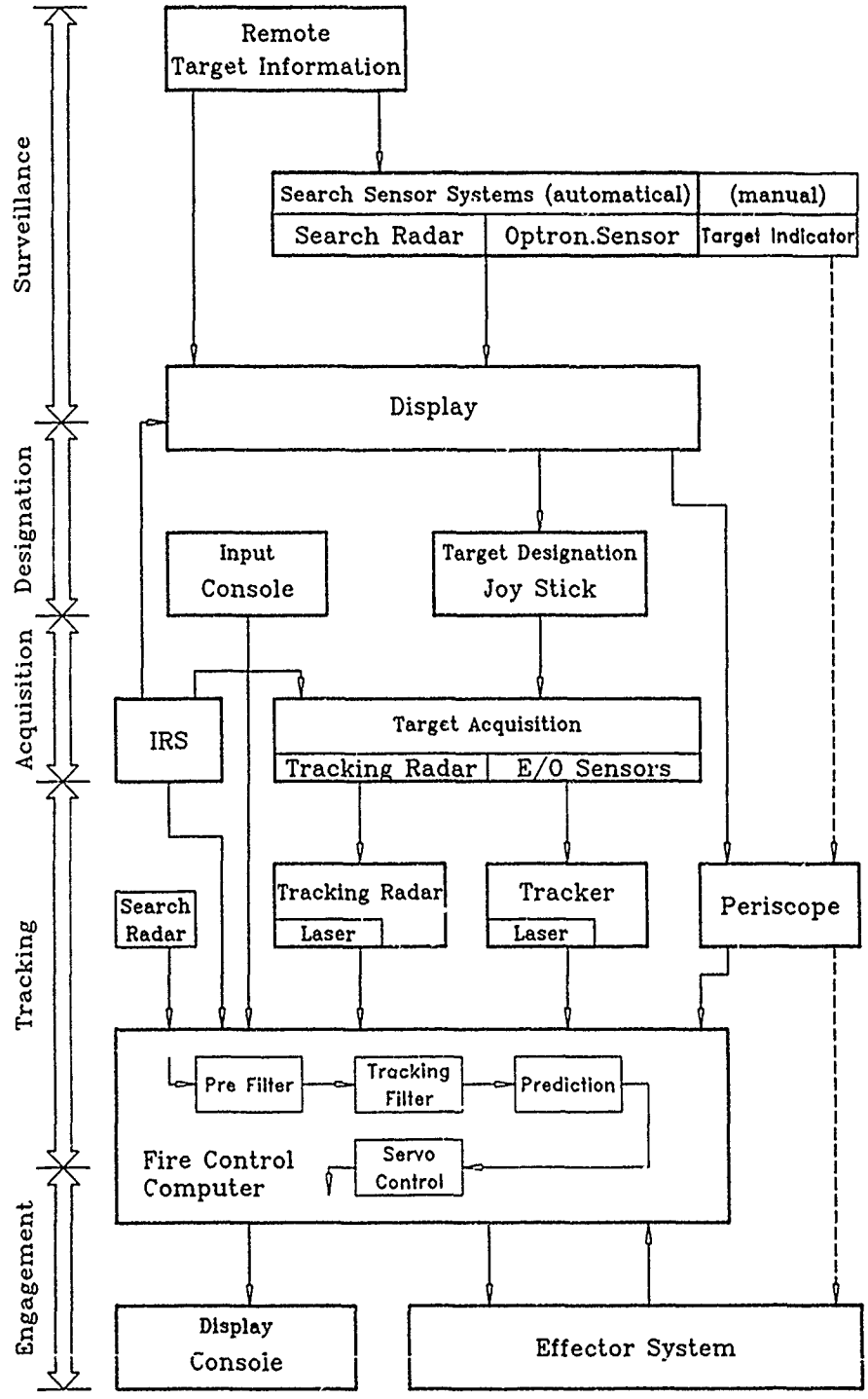


Fig. 1: Signal Processing in Air Defence

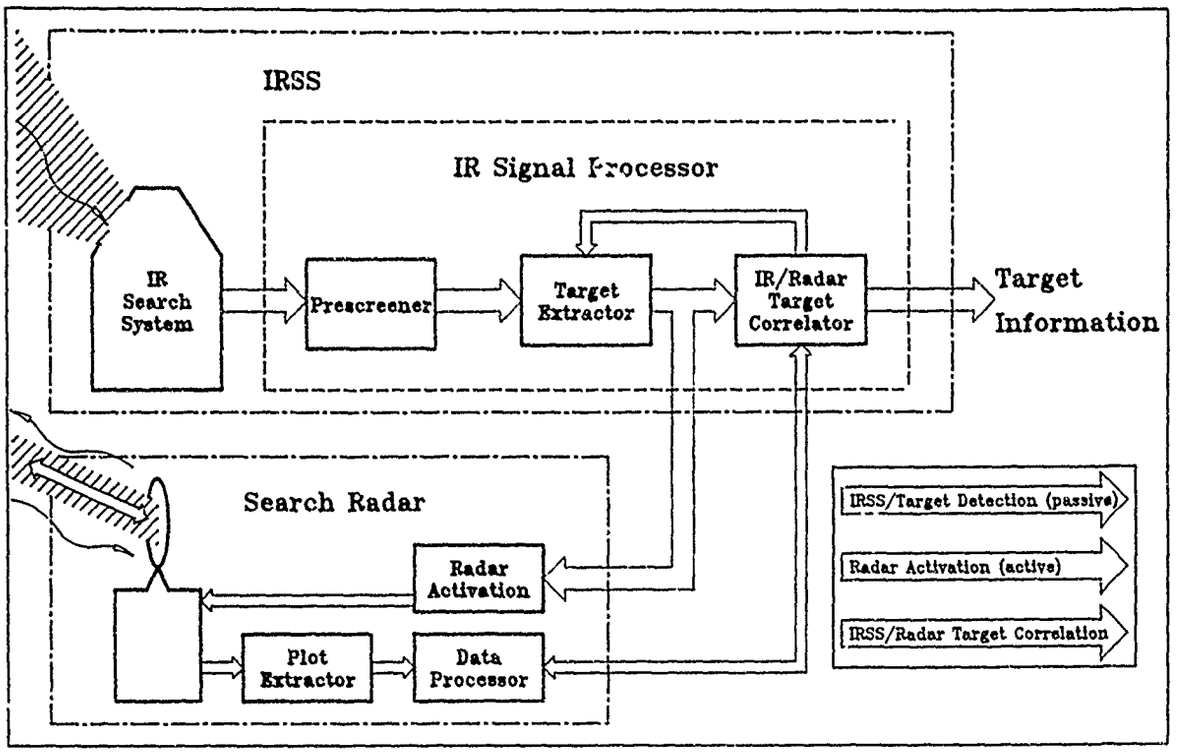


Fig. 2: WHIspering Search System (WHISS)

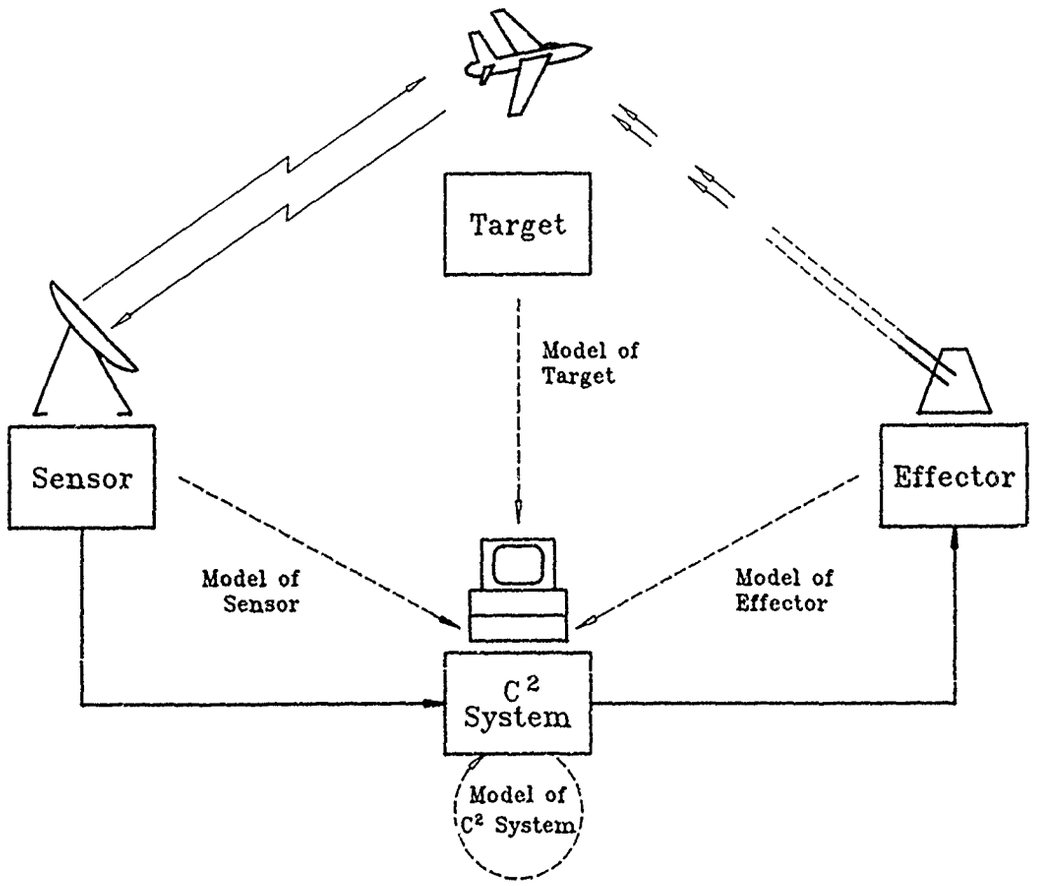


Fig. 3: Models in Air Defence System

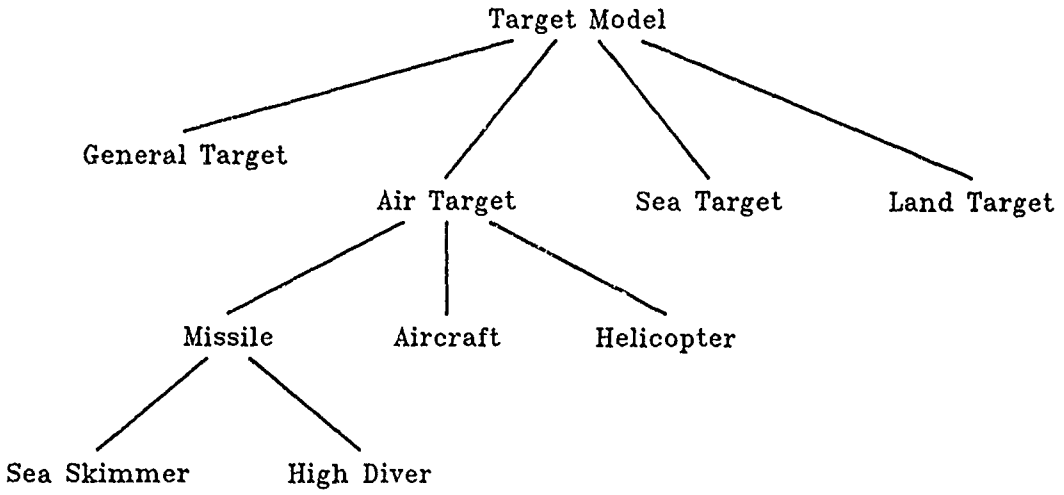


Fig. 4: Tree Structure of Target Models

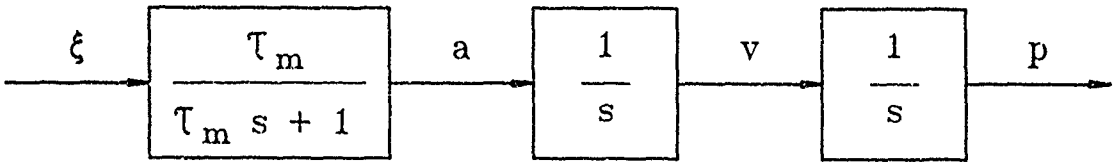


Fig. 5: "Singer" Model

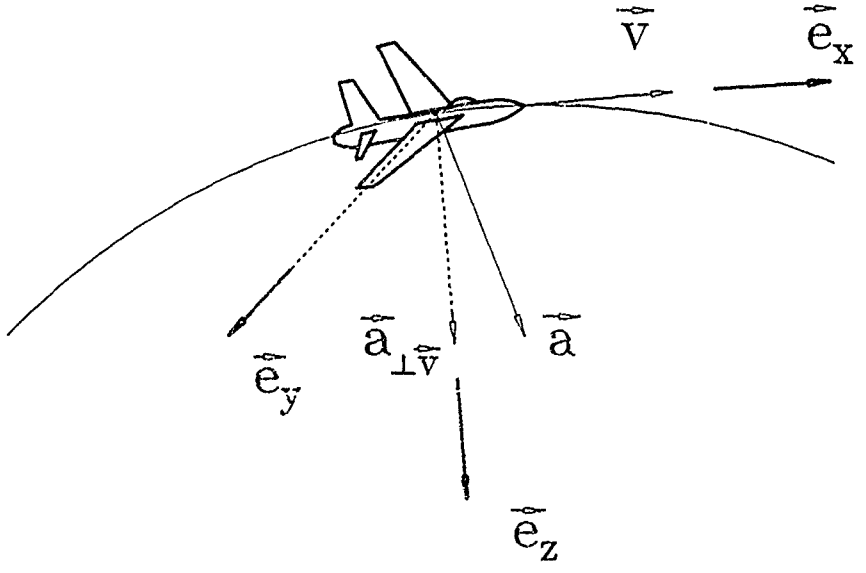


Fig. 6: Modelling in Flight Path Fixed Coordinates

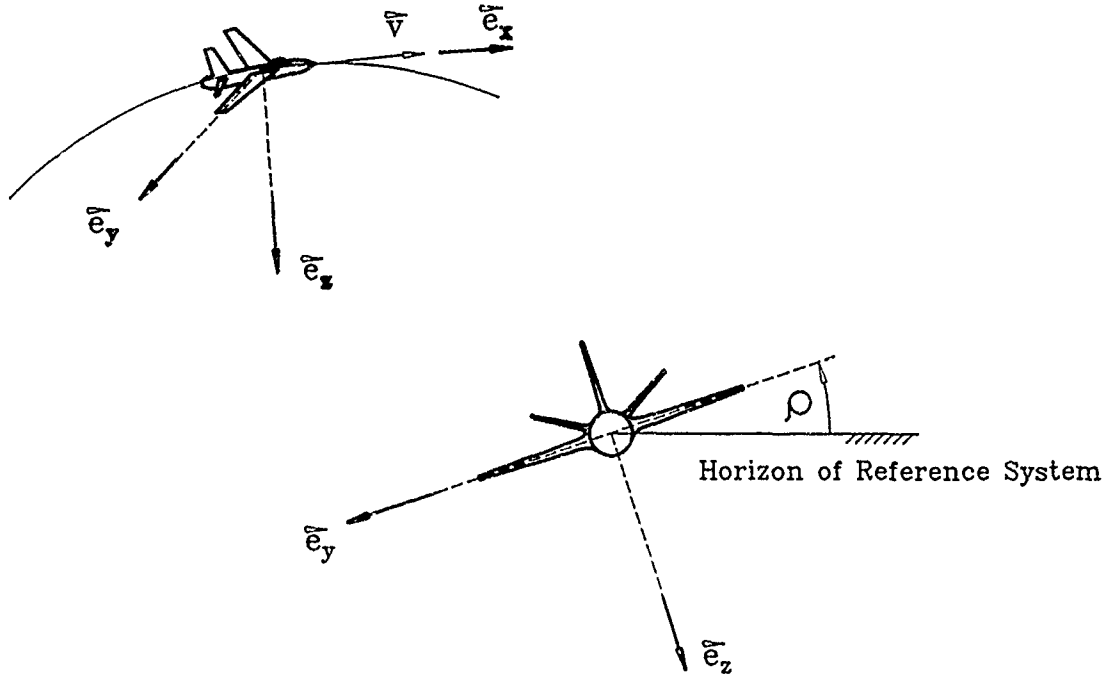


Fig. 7: Roll Angle as an Indicator of Maneuver

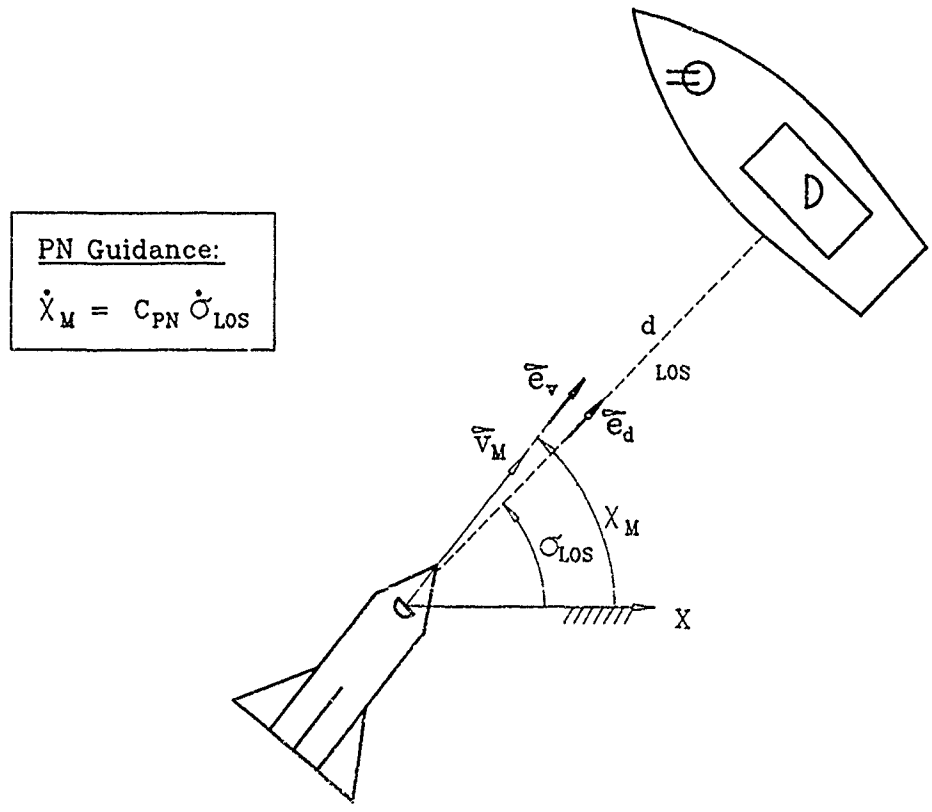


Fig. 8: Principle of Proportional Navigation

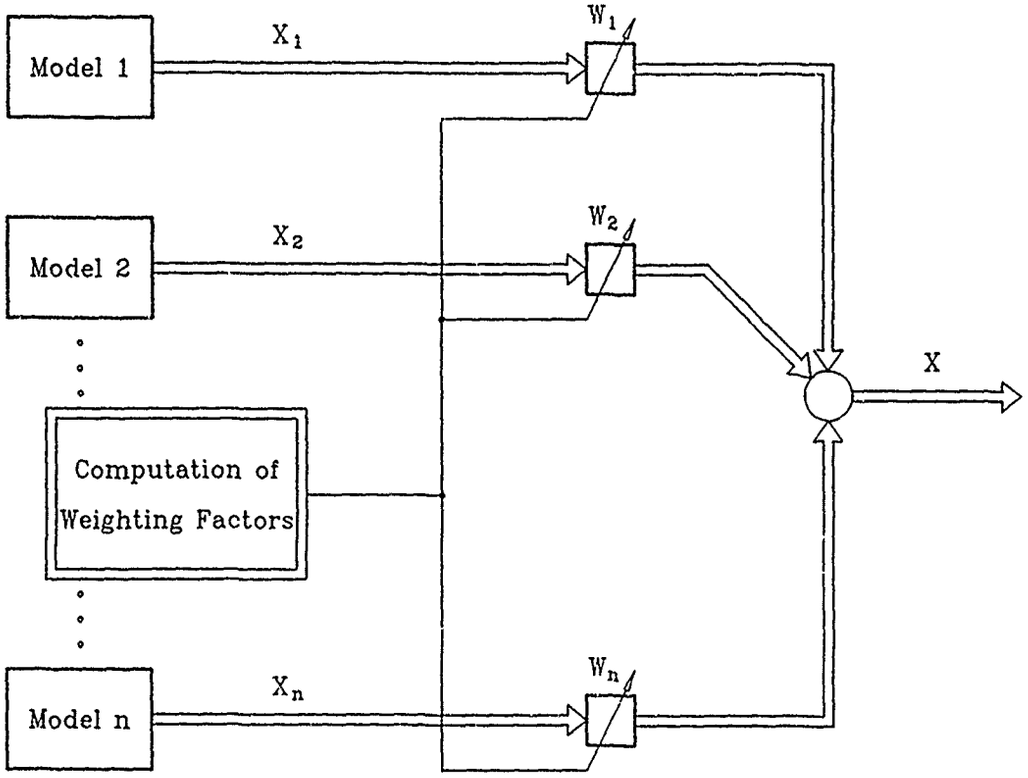


Fig. 9: Adaption by Weighted Superimposition

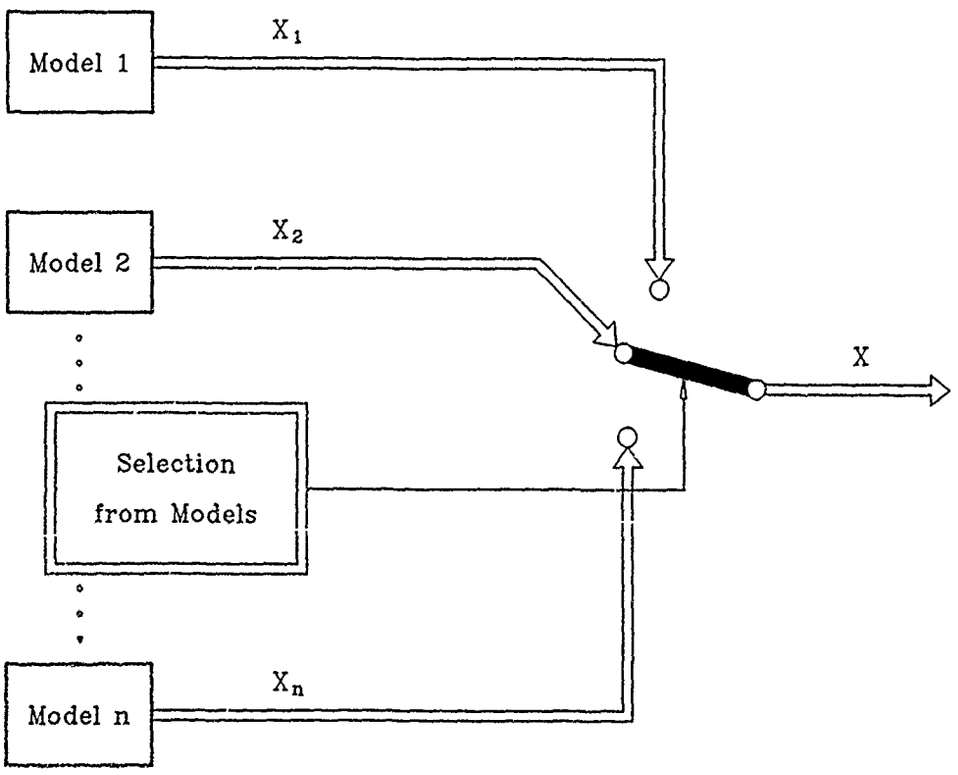


Fig. 10: Adaption by Model Selection

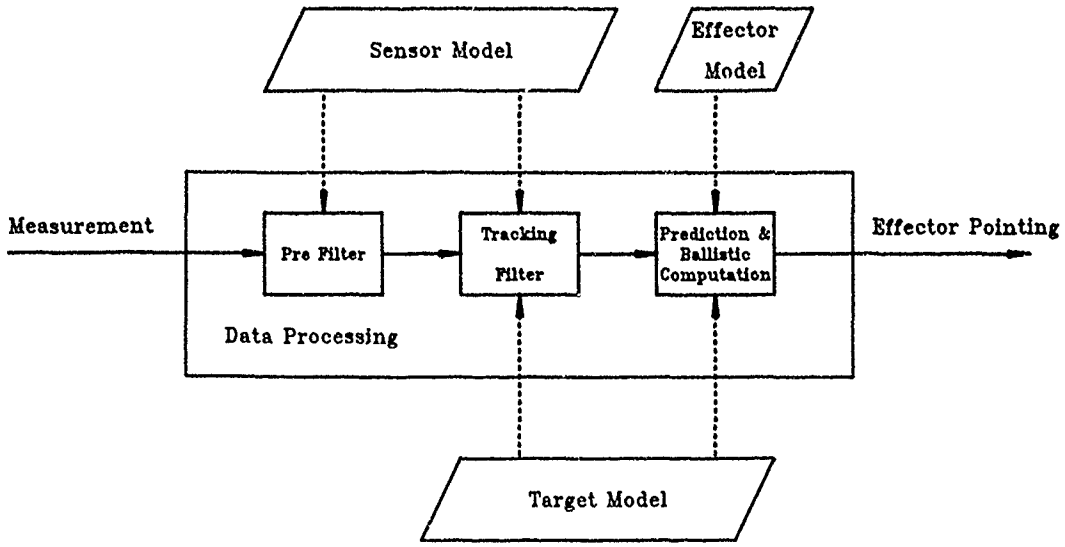


Fig. 11: Influence of Models in Fire Control Process

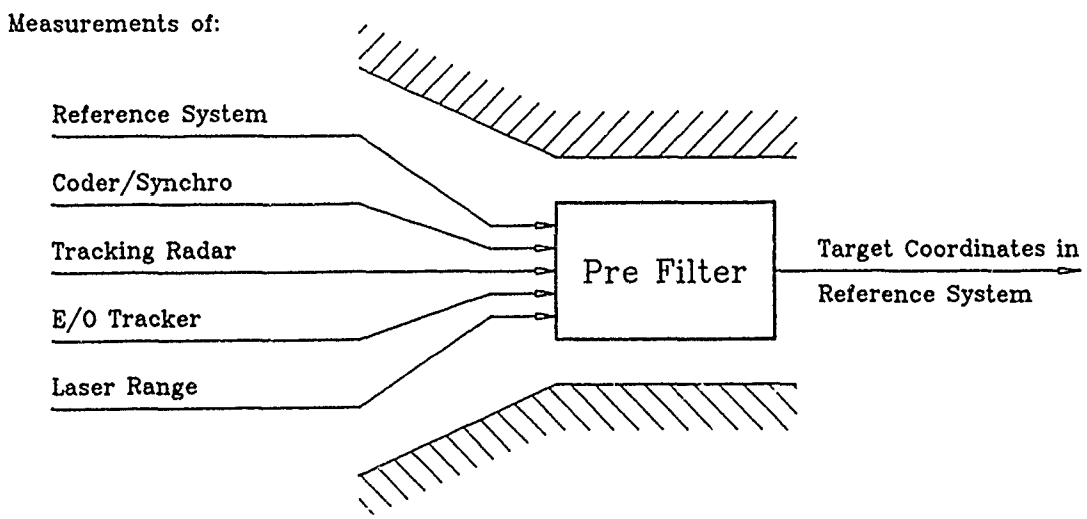


Fig. 12: Pre Filter

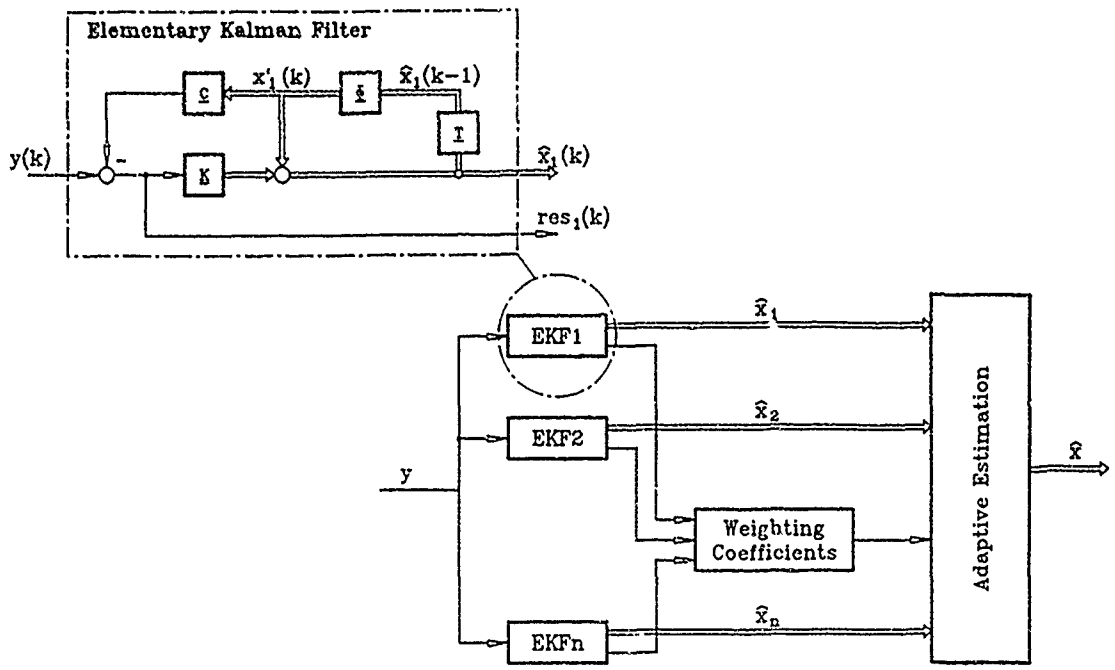


Fig. 13: General Purpose Filter Bank

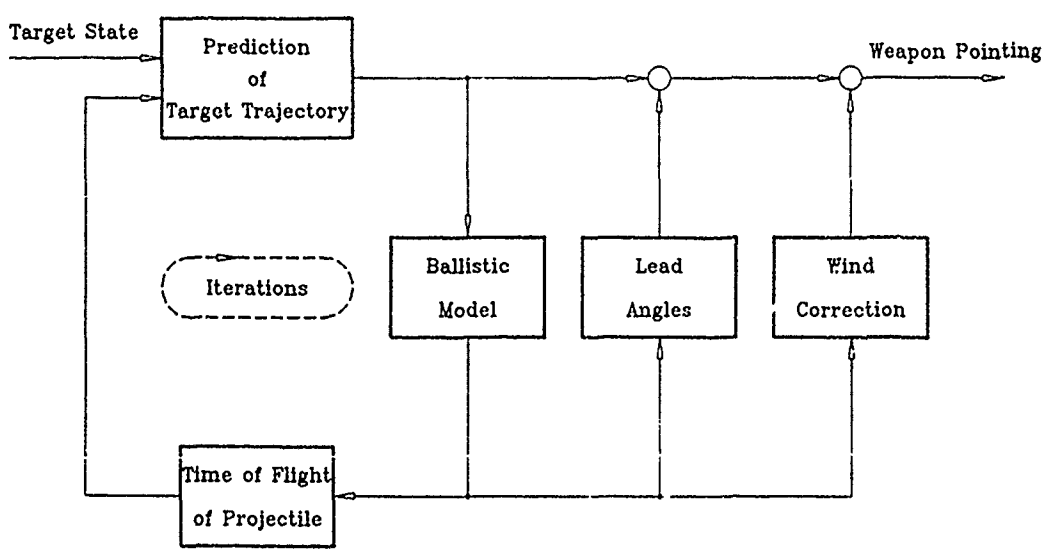


Fig. 14: Computation of Weapon Pointing

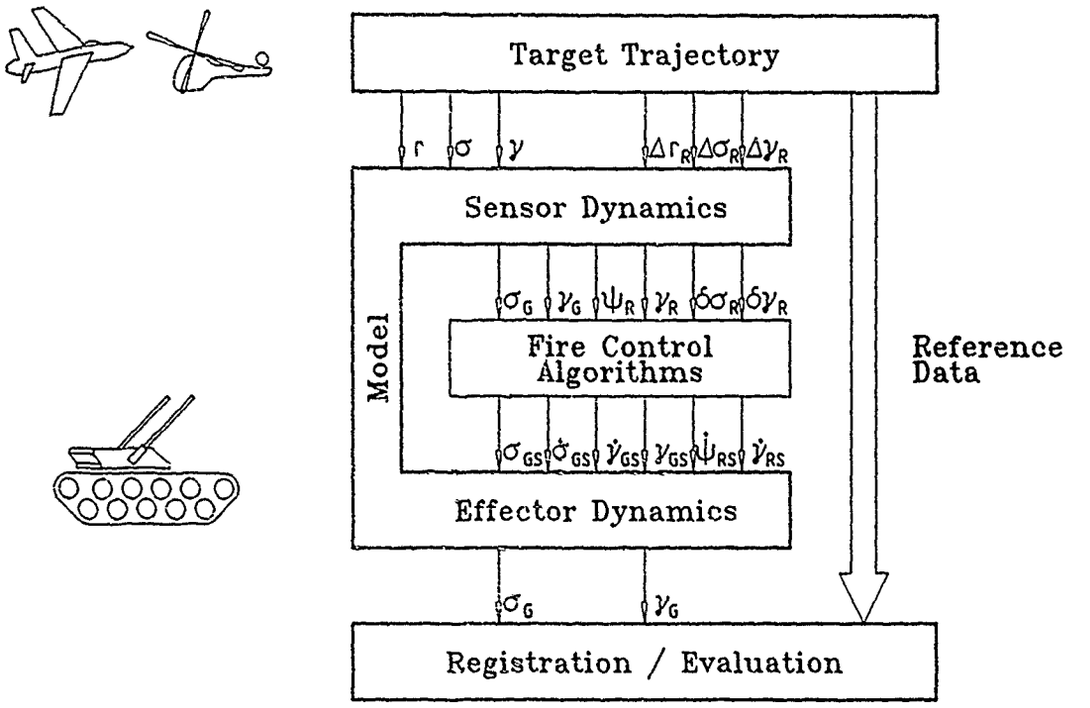


Fig. 15: Simulation of Anti Aircraft Tank

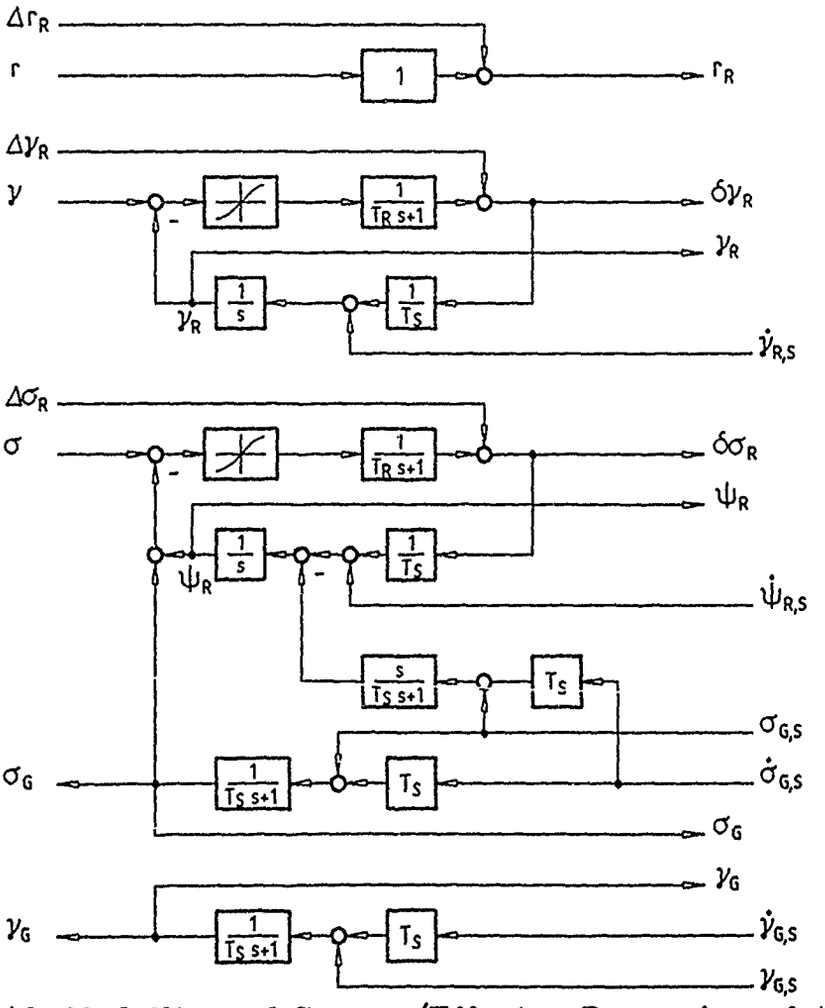


Fig. 16: Modelling of Sensor/Effector Dynamics of AAT

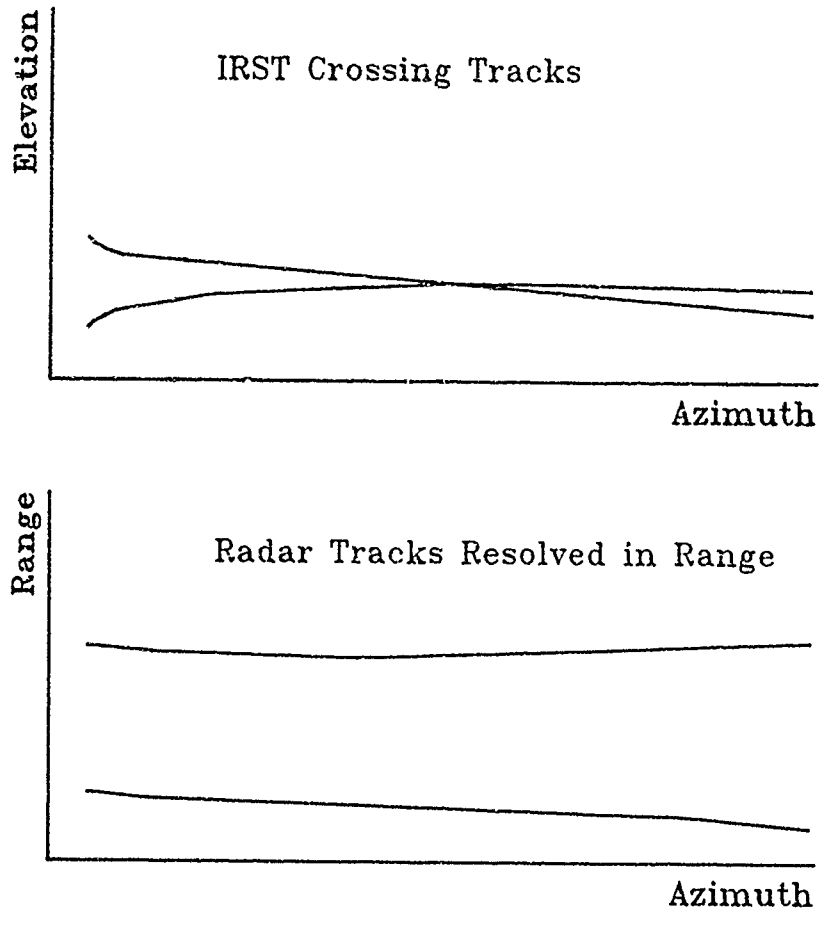


Fig. 17: Crossing Targets

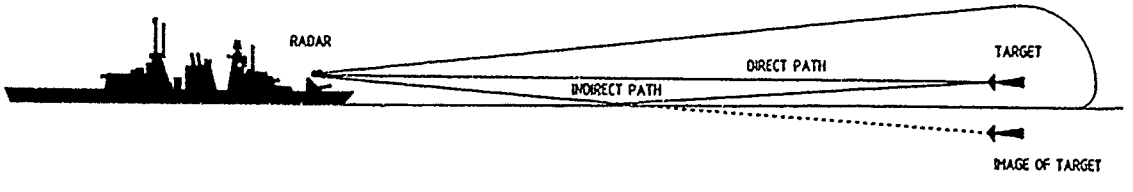


Fig. 18: Mirror Effect

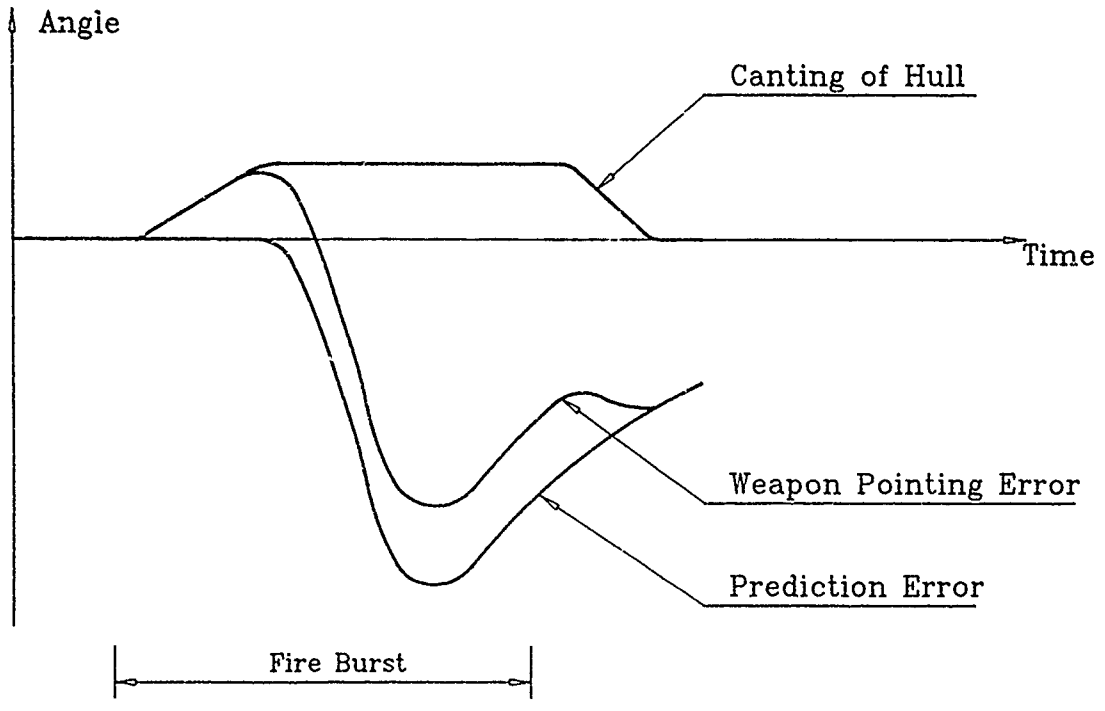


Fig. 19: Influence of Fire Burst on an AAT

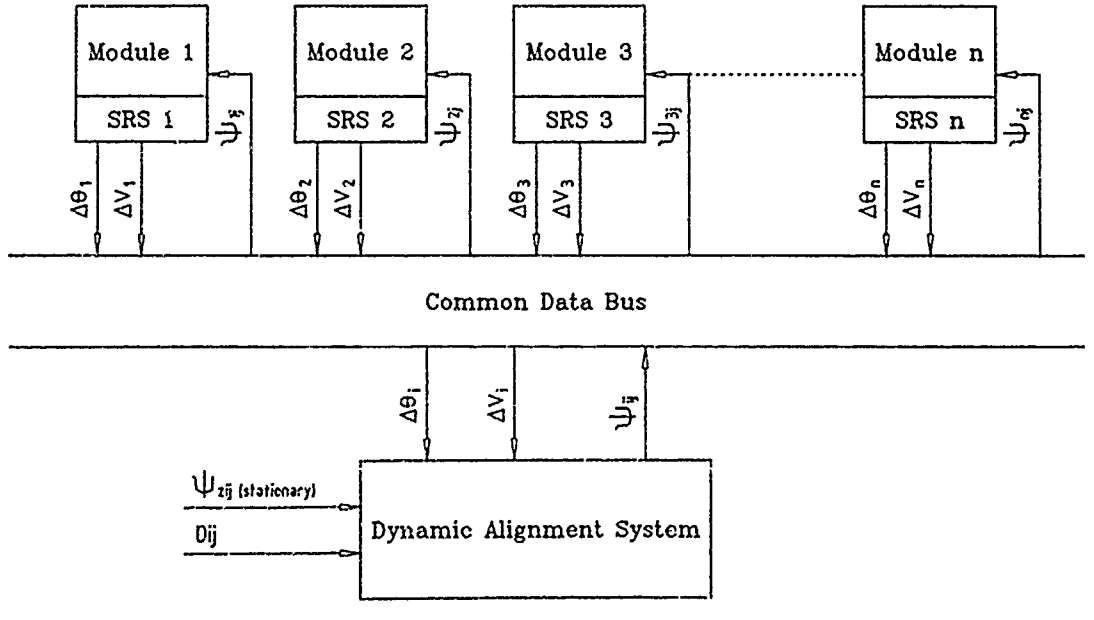


Fig. 20: Dynamic Alignment of System with N Modules

THE USE OF SYSTEM SIMULATION DURING THE DEFINITION PHASE OF THE PASSENGER TRANSPORT AIRCRAFT MPC75

by
Dieter Dey and August Kröger.
Deutsche Airbus GmbH
PO.Box 950109
D-2103 Hamburg 95
West-Germany

Summary

The paper starts with some general remarks concerning the tasks to be performed during the definition phase of a civil passenger aircraft and states the importance of the use of simulation as a design tool.

A more detailed differentiation of the terms "systems" and "simulation" is given with the emphasis on realtime simulation.

The present use of simulation in four areas is described: for systems engineering and know-how accumulation; for aircraft systems automation, monitoring and handling in failure cases; for tests of programmed avionic boxes, specially the fly-by-wire system and for flight simulation with and without pilot in the loop.

1. Tasks of the Definition Phase

It is known that about 85% of the life-cycle-cost of a main system like an aircraft will be determined during the definition phase.

We know that the development cost of a new aircraft will really burden the financial capacity of a company, therefore, the effort and the time for the development will be restricted.

The task is to define an aircraft together with its systems having a performance competitive on the market in 1995. That means to generate qualified design decisions, start with system integration engineering and find out development risks to be worked at.

Looking at the technological situation we see:

- a dramatical increase of processor speed and memory size with constant or decreasing hardware-cost
- more pretentious requirements for new systems with an increased use of processors also in primarily mechanical systems
- an increased coupling of all aircraft systems, specially for fly-by-wire and autoflight, caution and warning as well as centralized maintenance
- an increase of software-development cost and the growth of necessary investments for new technology (processors, operating systems and related tools).

So it is a big challenge for an a/c manufacturer to build up during the definition phase the tools for the definition and integration of digital systems, especially because information processing is not his traditional domain and he may be dependent on suppliers.

In the present situation the increased use of simulation facilities is absolutely necessary because

- simulation projects will integrate the knowledge of different specialists and therefore support system-oriented thinking,
- the quality of simulation results will indicate the status of the aircraft and systems definition,
- system simulation will be used by the engineers as a "sparring partner" for requirements-, design- and performance-analysis improving so the quality of their work,
- simulation models should be structured in accordance to the real system, so they can be used later on during the development and certification phase as test tools for black boxes,
- simulations should be used as a translation tool for the intentions of engineers and the experience of the pilots as users.

Ten years ago it still had to be argumented for the cost effective use of simulation and simulators for the development of a civil aircraft [1] . Today this approach seems to be well accepted.

Boeing reported [2], that they have increased their simulator use from 3600 h in 1976 to 41000 h in 1982, when the 757 and the 767 were developed using about 75% of the capacity of the simulation center. The most intensive users have been the stability and control group and the flight deck integration and avionics group.

2. Defining the Terms "System" and "Simulation"

It was very interesting for us to find already within the AGARD Conference Proceedings No. 79 on simulation from 1970 already the attempt of Mr. Brüning to define those both expressions. He at that time looked into seven dictionaries, found different explanations and at the end asked whether the definition of 1964, he liked best, is still valid.

As we know, a system is defined as a group of things working together to perform a defineable task. Which things belong to a system depend on the borderline drawn in accordance to the problem to be studied.

For the description of the aircraft we use a hierarchy of systems. We call the aircraft the main system. In relation to that, systems are the engines, the APU and the fuel environmental control, electrical and hydraulic system.

If we want to address our attention to parts of those systems, we call them subsystems. The smallest parts considered are the components.

Simulation is defined by the imitation of a system by a model and the analysis of the behaviour of that system model. For our paper we only want to look at computer-based simulations.

We normally look on the behaviour of a system as a function of time. When we are to calculate the reaction of a system-model to an input due to the normally actual elapsed time, we call that real time simulation.

To get a system-model three steps are necessary (see fig.1).

First, the system to be simulated has to be analysed and a model has to be defined which is adequate for the questions the simulation should answer. Therefore, different models of a system will be used in accordance to the aspects to be looked at, for example

- the influence of different component parameters on the system performance,
- the determination of the behaviour at minimum or maximum inputs,
- the reaction on disturbances,
- the effects of failures,
- the handling procedures,
- the performance of the system with the pilot in the loop.

The intention is to design the simplest allowable model.

After the definition of the model, in a second step, it has to be implemented as accurate as possible in the computer. The computer model has to be verified against the system description to show the proper implementation together with the limits of accordance.

The third step is the proof of the validity of the computer based model compared with the actual system. So for the same inputs simulated and real outputs will be compared and often the system model will be updated.

3. Simulations in the Definition Phase

Four types of simulations are already or will be built up. First the simulation of single systems for the purpose of system engineering, second a model for system monitoring and failure propagation, third a test bench for stimulating, measure and analysing processor-based black boxes and fourth an experimental flight simulator will be described in the following.

3.1 Simulation of Systems

The basic mechanical systems of the aircraft are:

Powerplant (engine 1, 2), auxiliary power unit, fuel system, air system (including environmental control and pneumatic), electrical system, hydraulic system, landing gear and flight control. For each system a simulation model will be programmed to represent the structure of the system and the performance of its components so that the total system performance can be analysed. The model should run in real time, but this is not absolutely necessary and depends on the modelled aspects. The system model will accumulate knowledge of different specialists, it will help to find problems early in the design process, it should be able to answer engineering questions, being a "sparring partner" for the specialist. With the simulation it should be possible to avoid some hardware tests. During the test and integration period for the system, its simulation should be used to help in solving unforeseen problems.

The model should be improved during the development process, when new information is available. At the end it will represent together with a system and model description, the total knowledge about the system and its capabilities. Such a model can be used to preserve know-how after the development phase and to transport know-how. It therefore can be used for solving maintenance problems and for the development of modifications. It also can be used for training purposes.

As an example the hydraulic system at the Airbus 320 is shown in figure 2. The principle layout of each of the three systems is shown in figure 3.

Hydraulic fluid is located in a reservoir pressurized by bleed air. A pump feeds the hydraulic liquid into an accumulator and through a pressure controller into the supply tubes. Fast actuators for ailerons or spoilers may have their individual accumulators. Some of the consumers like landing gear, slats and flaps are seperated by a priority valve. The coupling of the green and yellow system is performed by a power transfer unit which can be looked at in a single system as a pump or a consumer, controlled by the system pressure.

For the simulation of the hydraulic system the different components like pumps, accumulators, valves and consumers are programmed as modules and linked together. The performance (the adequacy) of the component models has to be chosen accordance to the questions to be answered.

In addition to the model a control program and some support programs are necessary which allow

- the setting of values at the beginning,
- the possibility of steering certain values in accordance to a definable time function,
- the introduction of errors by intentional setting of dynamical values,
- the storage of every value of interest as a function of time and in addition,
- some plotting and analysis programs.

3.2 Centralized System Monitoring

The optimal display of information for the pilot about the conditions of his basic systems and the display of information together with the handling procedures in the case of malfunctions have to be evaluated. Therefore, a graphical workstation based panel and display simulator was built [3]. The physical arrangement of the monitors to simulate the overhead panels and the engine and warning display together with the system display and the push button control is shown in figure 4.

Figure 5 and 6 give examples for the display pages and the overhead panels. The graphics are based on the graphic tool "Data Views" (DV), which consists of an editor to draw the images and a set of library routines to implement the generated images in a program and to control the color and position of symbols and characters. All monitors have touch-sensitive screens, so that all push buttons and control knobs can be activated.

The simulated displays and controls together with models of the basic aircraft systems are running on two SUN 312/60 workstations with graphic processors. The two computers and a third one for SW-development and data analysis are coupled by Ethernet.

Figure 7 shows the coupling of the simulated models so that a propagation of failures will occur and can be studied. A control program with the capabilities mentioned in section 3.1 gives the opportunity to initialize failures and store every value of interest for later analysis. With this tool the monitoring algorithms for the systems can be tested. The structure of the whole system is shown in figure 8.

Each model is built up with the structure of the original system. So in the program submodels are defined with the goal to have less interfacing values. Most of the values have a physical meaning like in the real system. Many submodels are equipment models like valves, motors and pumps. Each model should simulate with allowable simplification the real system. They can be controlled in real time, so with this approach the actual functionality is available including failure propagation.

3.3 Flight Control Test Station

3.3.1 MPC 75 Flight Control / Autoflight System Concept

For MPC 75 a 'fly by computer' control system is in the definition phase. The core of MPC 75 integrated primary and secondary flight control/ autoflight system are 2 groups of dissimilar computers containing transputer and ASIC (Application Specific IC) hardware.

As an example pitch control is described in more detail in figure 9. The concept for pitch control laws under normal operation foresees load factor demand with load factor limitation and pitch rate damping leading to

- neutral static stability by permanent autotrim
- turn compensation up to 35 deg bank steady turn
- handling qualities not affected by speed, weight and C.G. variation
- gust, thrust and configuration changes, airbraking etc. with minimum disturbance on short term flight path and attitude.

The ground control law with direct and maximum elevator deflection available is phased in at touch down and phased out after lift off. Below the transition altitude flare law is activated where autotrim is cancelled and conventional stick force stability is reestablished.

There is a flight envelope protection with high angle of attack limitation, high speed limitation and alpha floor control law. High angle of attack law is activated when the AOA is above a limit leading to positive static stability and stall protection with safe operation at maximum lift. High speed law introduces positive static stability above maximum speed (V_{MO}/M_{MO}) and reduces speed/Mach excursions to a minimum without load factor exceeding limitation. If filtered AOA exceeds the alpha floor limitation e.g. in excessive wind shear conditions Take off/Go Around engine thrust is activated disregarding actual thrust lever position. To proof this concept a predeveloped system will be available this year for intensive examination on our flight control teststation.

3.3.2 Test Station Description

For flight control/autoflight system feasibility studies up to prototype validation a test station is available including simulation of aerodynamic loads on control surfaces by computer controlled hydraulic actuators.

The most powerful part of the test station is the avionc signal conversion and test computer with its test control computer and test registration and analysis computer, shown in figure 10.

This system is used as an open loop test bench with predefined computer generated test signal inputs as well as in a closed loop mode with the complete six degrees of freedom flight simulation. This test environment has been established from 1983 to 1988 in parallel with a fly-by-wire study. The program has been sponsored by the West German Ministry of Technology and Research. So the system will be available for all German aircraft and system manufacturer for future projets including flight control and autoflight system development.

3.3.3 Conducted Projects

One of the main technological new features of the Airbus A310-300 and A300-600 is the C.G. control system (CGCS). Fuel is pumped between the main tank system within the wing and the additional tank in the horizontal stabilizer. If necessary 1 or 2 additional center tanks can be installed at the front end of the rear cargo bay to replace cargo payload by additional range. For long range max fuel is increased from appr. 45 tons to 62 tons. A C.G. control computer monitors the fuel flow between additonal center tanks and main tanks and controls the flow to and from the tail tank to hold C.G. within the limits for drag reduction. During development of this system the complete fuel system has been modelled for simulation, the control laws have been developed and validated and the prototypes have been tested for flight test release. Within an automatic 70 hour test program all functions of the CGCC have been activated and 250 parameter have been recorded for analysis.

During certification of civil transport aircraft it has to be demonstrated that flutter speed is greater than 1.2 of the design diving speed. For Airbus A310-300 an experimental flutter margin augmentation system (FMAS) has been developed, which has been used in flight test for demonstration of damping characteristics as well as a means for excitation of flutter modes. Within the simulation system 5 oscilatory modes have been simulated with 1000 Hz sample rate. The FMAS computer working with a sample rate of 250 Hz has been connected via the analogue interface. It has been demonstrated that the control laws damp the flutter critical oscillation and do not influence the other modes in a negative way.

Within the life extension program for the Transall C 160 military transport aircraft the complete avionic systems will be renewed. For this task artificial feel units and control column including autopilot servormotors have been installed into the simulator and a/c simulation software has been made available. For preparation of flight tests planed for 1991 autopilot control laws will be validated.

In parallel with the test system development a fly-by-wire feasibility study has been made within a joint project by three companies :

- Deutsche Airbus GmbH for test system flight control system concept
- BGT (Bodenseewerk Gerätetechnik) for the fly by wire computer system
- LAT (Liebherr Aerotechnik) for parallel active electro-hydraulic servo actuators.

The system concept is based on the requirements for certification of a safety critical function, that means failure probability 10^{-9} per flight hour and the capacity of a computer system to implement active control function as gust load alleviation and variable camber.

The experimental system consists of a quad-redundant, fault tolerant multicomputer system and a quadruplex actuator electronic unit providing redundancy management and force synchronization of parallel active electro-hydraulic servo actuators.

This experimental system has been tested in the new rig up to aircraft simulation in the loop. The control laws of the fly-by-wire system as well as the control and redundancy management concept for the system including parallel redundant actuators have been verified by tests.

3.4 Flight Simulation

3.4.1 Task of the Simulator

At the moment a first version for MPC 75 flight dynamics is available and first simulated manually controlled flights have been done. Within the definition phase of MPC 75 the main task on this tool is the development and adaption of simulator software and hardware to the specific features and data base of this aircraft. In this phase it is most important to response fast to aircraft and system modifications to assist decisions when alternative solutions have to be evaluated. So we are preparing and working on

- validation of mathematical models for aircraft characteristics as aerodynamics, mass distribution, engine dynamics and landing gear dynamics
- handling quality investigations for highly augmented flight control functions including alternate control laws down to direct laws depending on sensor system failure status
- autoflight control law validation within the nonlinear closed loop
- pilot evaluation of criticality of system failures such as loss of individual control surfaces
- pilot evaluation of modified or higher integrated cockpit displays and controls
- flight control and autoflight hardware in the loop prototype testing for system integrity, performance, failure effects and pilot workload
- assistance for certification of computerized flight control system, autoflight system, flight management system including autoland, category 2 and category 3 landing capability.

3.4.2 System Overview

The main components of the develooment simulator system are shown in figure 10 :

- 3 MODCOMP 32/87 realtime simulation computers with 2 megabyte shared memory and 8 megabyte memory each
- components for system software development as CPU D, terminals, printers, laser plotter, tape units and 1 gigabyte disk memory
- control and display station to operate real time simulation processes in a simplified way without cockpit
- cockpit with display and control devices, visual system and aural system
- simulation test data registration and display computer

Depending on the necessary computer performance and I/O requirements for individual task oriented simulation versions (total aircraft or individual system) all components or only individual components are active for the test. So different investigations on different parts of the system can be done in parallel. For example it is possible to run an Airbus a/c simulation with autoflight on CPU A, CPU B and the cockpit and to do long term automatic testing for a/c system prototype on the CPU C with the avionics interface.

3.4.3 Simulation Computers and Software

The core of the simulation system are three 32 bit real time computers MODCOMP 32/87 exchanging data via a common shared memory and transmitting data via 16 bit DMA channels. The simulation tasks are split into these 3 units. A typical distribution of programs is

- CPU A: autoflight, flight controls
- CPU B: 6 DOF flight simulation
- CPU C: additional aircraft systems

Typical cycle times for CPU A and B are 20 to 30 milliseconds and for CPU C down to 1 to 5 milliseconds.

The programs have to represent the mathematical models for

- rigid body motion for 6 degrees of freedom with no limitation concerning attitude due to quaternion model implementation
- aerodynamic forces and moments taking into account aircraft flexibility due to load factor
- variable mass distribution and moments of inertia due to fuel and payload
- landing gear dynamics including brakes and variable runway friction
- flight control and autoflight system including sensor simulation and failure models
- cockpit displays and indications
- fuel management, hydraulic and electric systems
- ground surface, navigation facilities e.g. ILS transmitter modelling including signal noise model
- atmospheric conditions (wind, gust, windshear)

3.4.4 Simplified Simulation Control

For simplified realtime operation of flight simulation the control and display station is used instead of the complex cockpit systems. The main features of this station are

- data conversion capability between a/c simulation computers and control & display station
- indication of all relevant parameters on LEDs, alphanumeric displays and analogue indicators
- inputs such as switches, pushbuttons, simplified throttle levers, primary and secondary flight controls for simulation control

For individual simulation tasks the definitions for indications and controls is done by user supplied software within the simulation computers. So it is a very flexible tool for system tests.

3.4.5 Cockpit Systems for Simulation

The fixed base simulator and its cockpit are shown in figures 11 and 12. For establishing simple data exchange between the simulation computer system and the cockpit the individual subsystems are connected by different lanes.

A/C position and attitude data are transmitted to the visual system (Singer Link Miles, Image 2 T). This is a dusk and night system with texture. The system meets FAA phase II requirements and could be upgraded to phase III requirements of FAR121 appendix H and advisory circular 120-40. The system produces computer generated images from a vision database for 3 channels on 4 CRTs. Airport area databases with 40 nautical miles diameter are available for 14 different locations. Modifications of these databases or creation of own additonal databases can be done easily by an interactive modelling facility processing map and photographic information. Weather simulation is provided on the system for clouds, visibility, fog, lighting, wet runways or snow covered airfields. These features are used to demonstrate compliance with all weather operation requirements. The application of texture pattern provides the essential height and speed cues when close to the ground, e.g. for the evaluation of flare control laws.

The aural system generates sound on 4 channels for 6 speakers within the cockpit. The generator is able to reproduce original aural cues, by sound sampling methods or to synthesize sounds of different types. The system is used for the generation of

- aerodynamic noise influenced by speed, flap setting and airbrakes,
- engines
- landing gear movement and ground roll noise
- rain and hail
- flap/slat operation
- a lot of system chimes and artificial voice messages

The aural system is located within a separate microcomputer system. During simulation the sound program is controlled by data from the simulation computer via RS 232 serial interface.

Flight and system status information to be displayed in the cockpit are transmitted via DMA transfer to 4 workstation computers SUN4/280 generating graphic display information. This part of the simulator is a powerful flexible tool for investigation of alternative display data presentation, resulting in validated specifications of the operational systems. So it is possible to generate a completely new display page within a few days and the system allows to switch between alternative solutions within a simulated flight.

The cockpit interface transmits signals between the simulation computers and the cockpit hardware such as

- control column or side stick positions
- rudder position
- artificial feel system control
- trim wheels and switches
- flap/slat and airbrake levers
- landing gear and brake control
- throttle control including reverse thrust
- flight control unit for autoflight functions

The hardware base for this interface is a VME-bus computer system. The interface to the simulation computer is a 16 bit DMA lane. For the cockpit systems all types of signals such as ARINC 429, analogue or discretes are a vailable.

3.4.6 Data Acquisition

A defined set of data within the shared memory is available at the quicklook display workstation SUN 3/260. The data aquisition program allows the selection of parameters, its scaling and time scale for representation as well as the definition of data to be stored on disk during real time simulation. Alternatively to time history display an animation program is available to show for example elastic deformation of the wing in real time.

4. Prospective View

For the definition phase of the MPC 75 the general approches for the aircraft systems, the available tools and their flexibility has been reported. It is planed to reach a development go ahead for the aircraft in 1991. Till than with these tools we will get good system specifications and the knowledge basis to control that program.

Literature

[1] "Einsatz und Erfahrungen der Simulation bei der Entwicklung der VFW 614 und der VAK 191", D. Dey and K.-H. Unterreimer, VFW-Fokker GmbH, DGLR-Vortrag-Nr. 77-083, Symposium Entwicklungssimulation, 1977.

[2] "Boeing Gains Real-Time Flight Data", Benjamin M. Elson, Aviation Week and Space Technology, 17.01.83.

[3] "Cockpit- und Flugsteuerungsprüfstand", J.H. Renken u.a., Schlußbericht, ZKP-Vorhaben LVP 8660 I/3, 1989.

[4] "Untersuchungen zur rechnerunterstützte Fehlerdiagnose im Cockpit", J. Lauber, Diplomarbeit RWTH-Aachen, 1989.

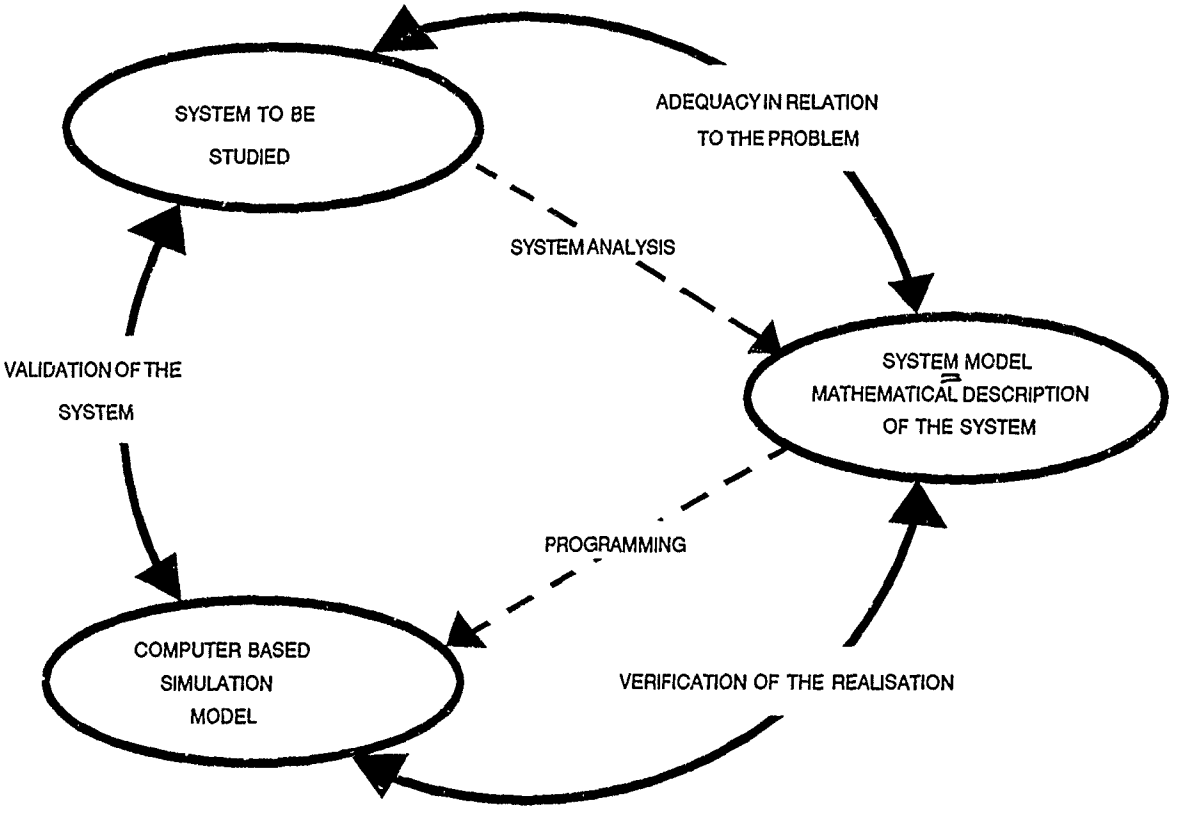


Figure 1 Generating a System Model for Simulation

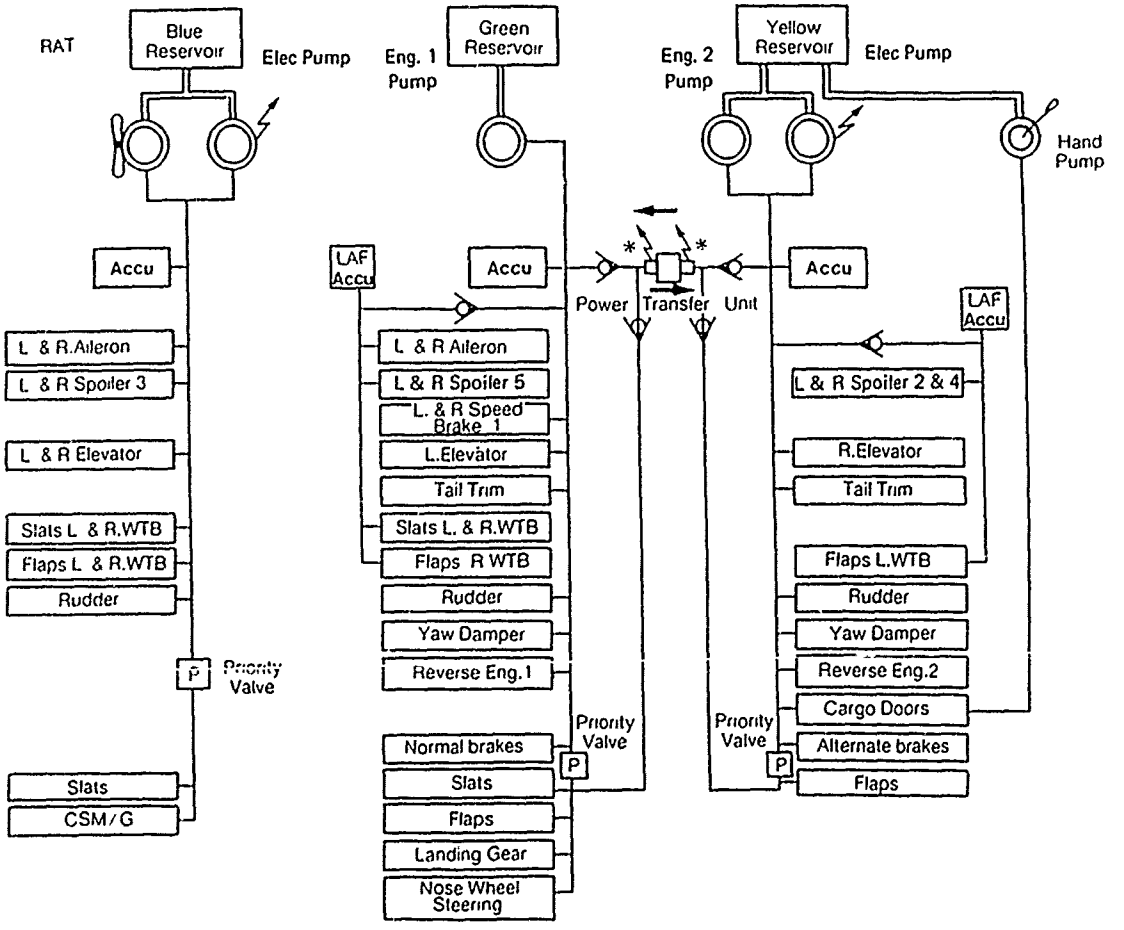


Figure 2 Example of a Hydraulic System

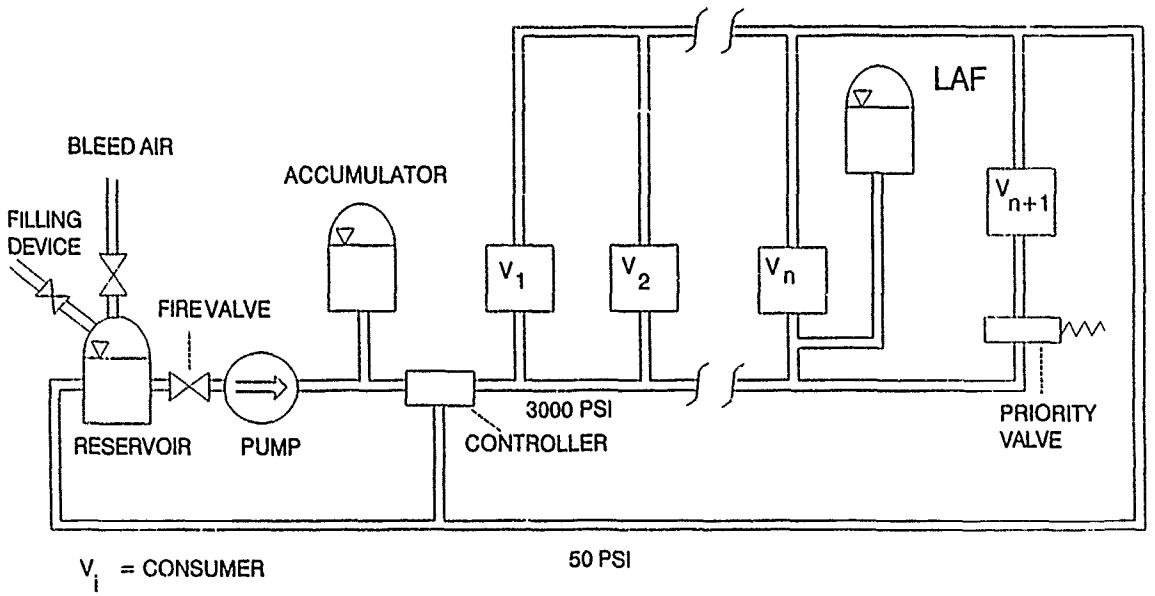


Figure 3 Principle layout of a Hydraulic System

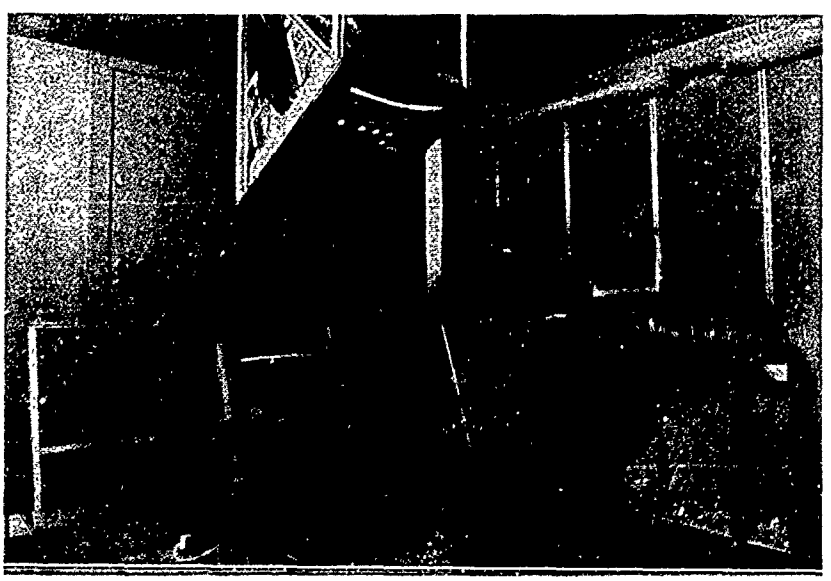


Figure 4 Systems Monitoring Test Facility

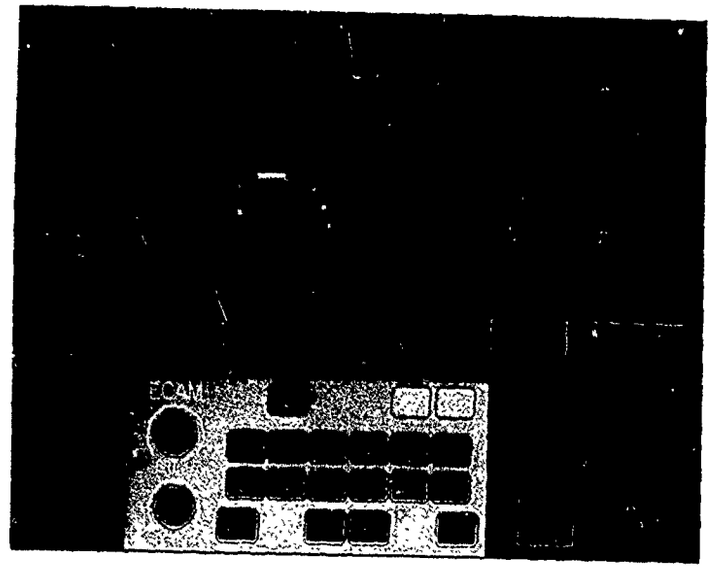


Figure 5 Example for the Engine and Systems Display

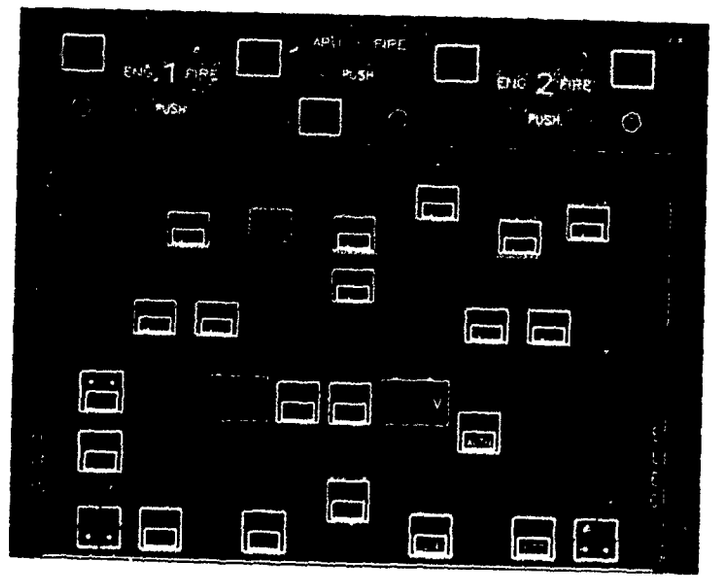


Figure 6 Example for the Overhead Panel Simulation

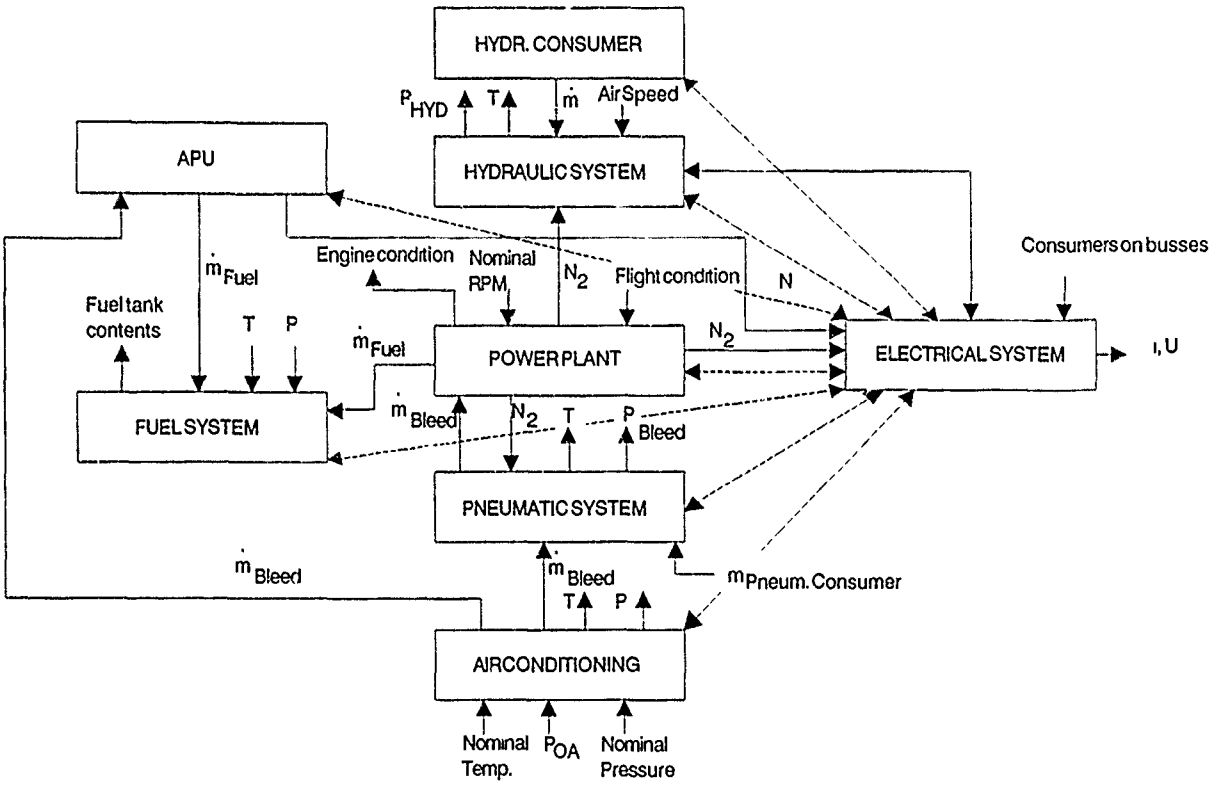


Figure 7 Coupling of the simulated Systems

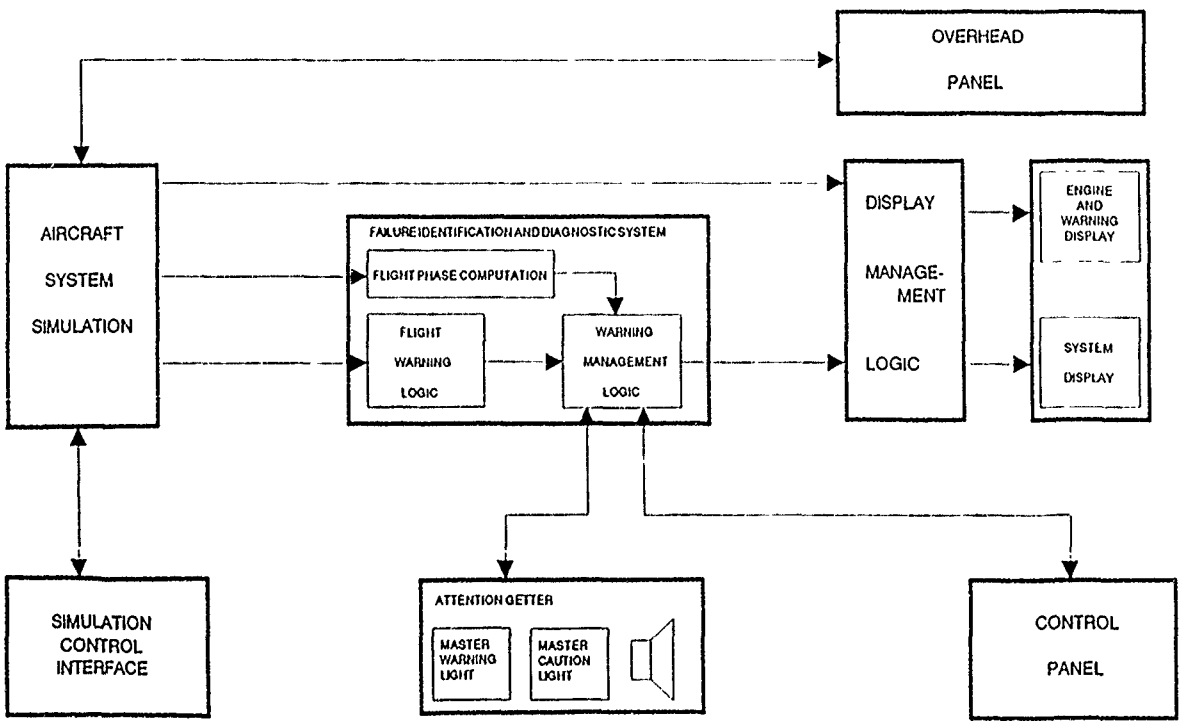


Figure 8 Structure of the Systems Monitoring Test Facility

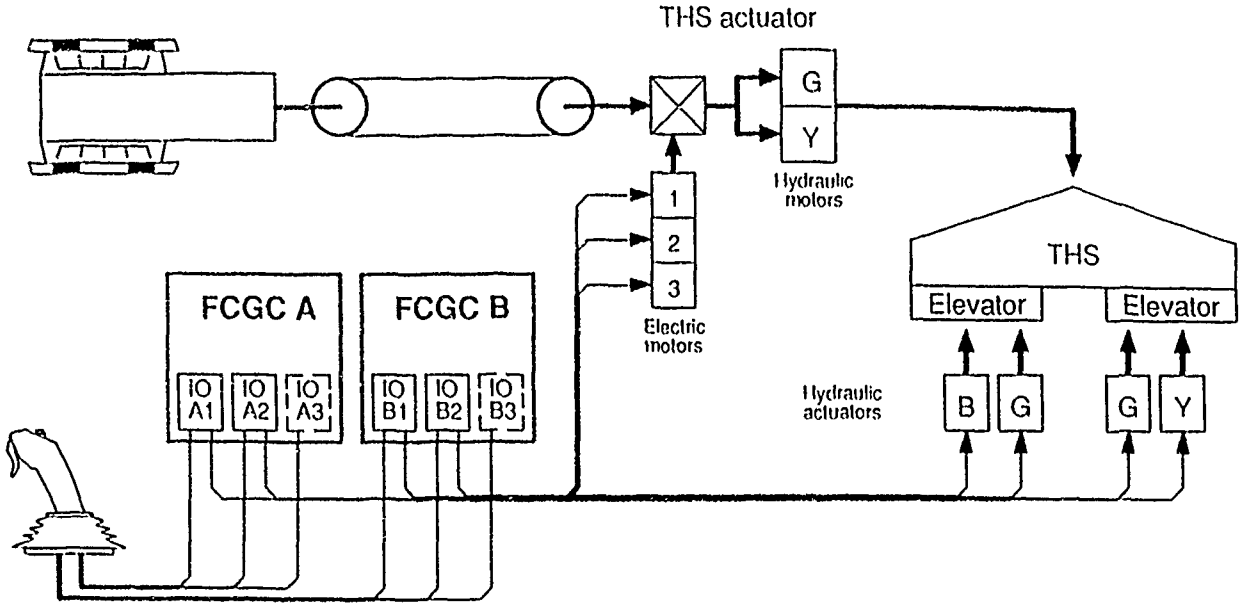


Figure 9 Pitch Control of MPC75

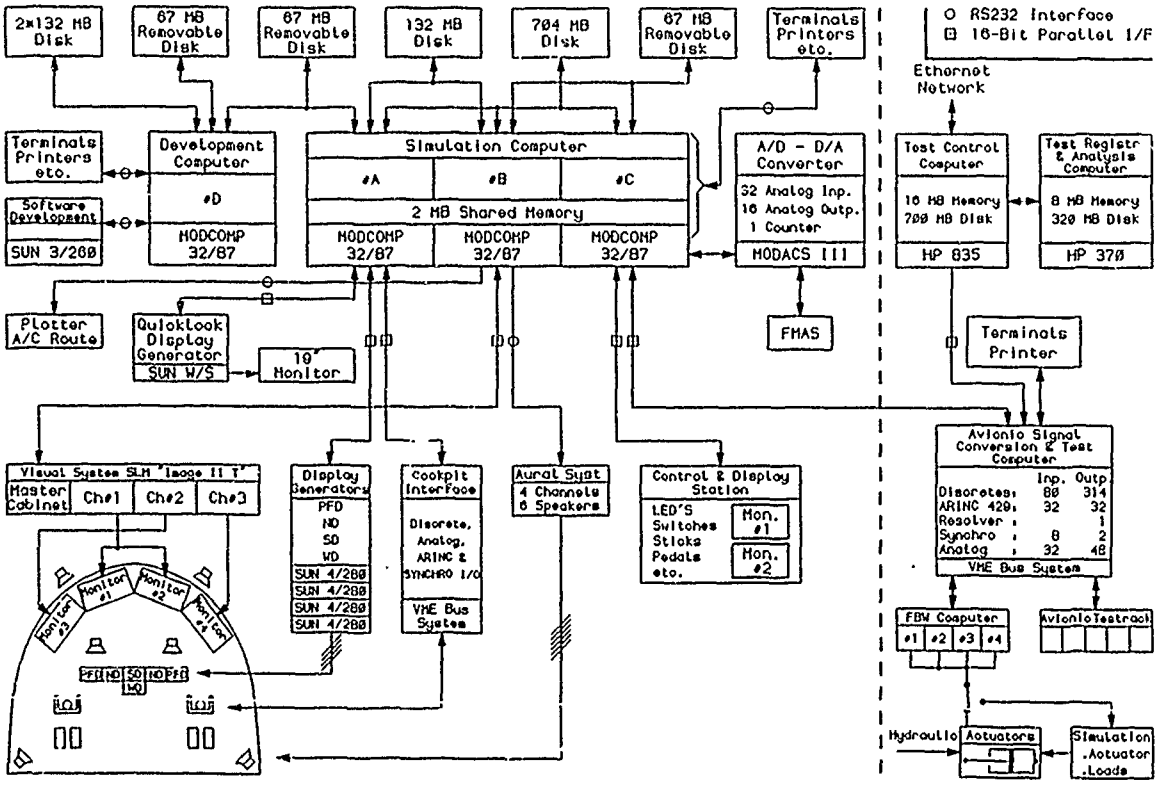


Figure 10 Structure of the Flight Simulator and Flight Control Test Station

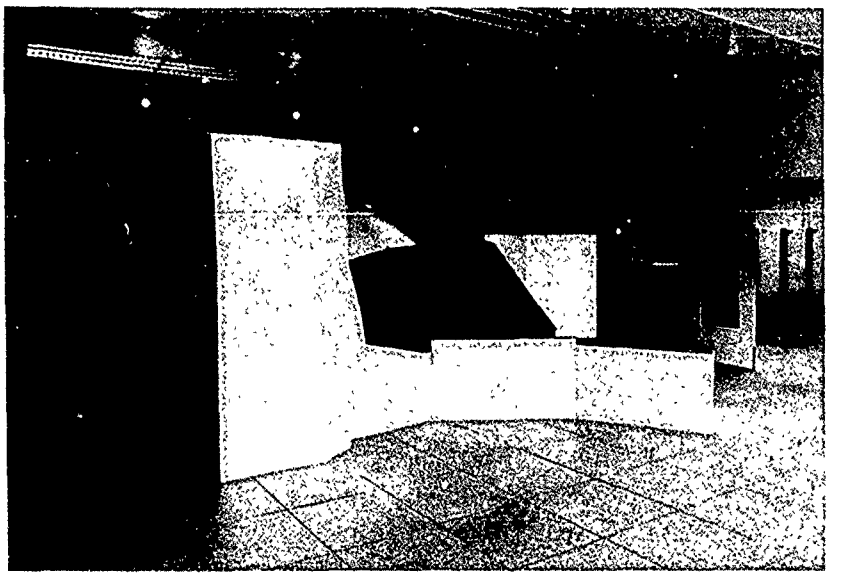


Figure 11 Fixed Based Simulator

Figure 12 Cockpit of the Simulator

COMTESS

Combat Mission Training Evaluation and Simulation System

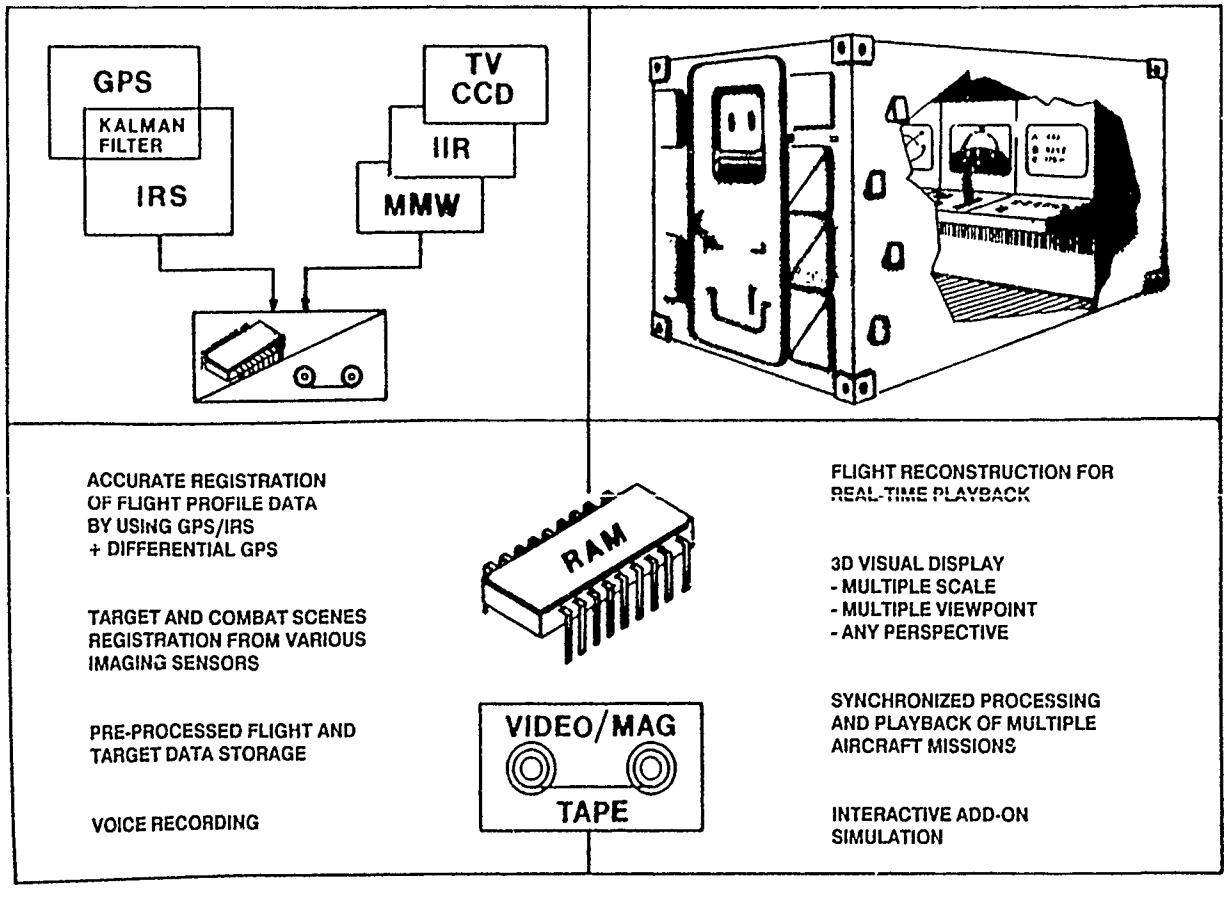
by W. Kraft, U. Krogmann, H.P. Müller, E. Platt
Bodenseewerk Geraetetechnik GmbH - BGT
P.O. Box 10 11 55
D-7770 Ueberlingen
West Germany

INTRODUCTION AND SUMMARY

NATO air forces need to achieve and maintain combat readiness of their air crew against increasing challenge and complexity of expected combat scenarios and, at the same time, decreasing availability of training sorties due to rising costs and weakening public acceptance. To counter the problem, training mission effectiveness needs to be enhanced and optimized.

ACMI (Air Combat Maneuvering Installation) is of great training quality benefit for westerly air forces, particularly for air-to-air tactics training. However, it is complex, expensive and stationary, as it needs a dedicated ground environment. Modern life flying for combat training requires tools for flight registration and reconstruction in any airspace and over any distance at affordable costs, for both air-to-air and air-to-ground missions.

BGT's Combat Mission Training Evaluation and Simulation System - COMTESS has been designed against these requirements and uses GPS (Global Positioning System) for accurate and continuous position determination, leading to registration of trajectories and maneuvers. Besides the dependence on GPS and on power supply at the aircraft interface, COMTESS is virtually autonomous. All airborne components are combined within a standard missile airframe (pod), e.g. Sidewinder, easily loaded on any Sidewinder-capable aircraft with no impact on maneuverability and performance. The pod-inherent modular sensor system is essentially based on GPS, using differential GPS for accuracy enhancement.



In addition to aircraft position and velocity vector, reference data such as attitude, angular rates and linear acceleration need to be determined and registrated for comprehensive mission reconstruction. Therefore, an inertial reference system (IRS) is used to interface with GPS via dedicated Kalman filter algorithms. It also serves to compensate for GPS antenna masking which will intermittently occur during highly dynamic maneuvers. The sensor package is complemented by imaging sensors, working in the visible, infrared and/or millimeter wavebands, for provision of target imagery and combat scenes generation as an optional part of the debriefing display, regardless of the meteorological conditions during flight.

Mission reconstruction and playback (debriefing) take place in the ground station. Any part of the mission can be looked at from any perspective, including variable scale, cockpit view, time lapse, slow motion, freeze, zoom-in and -out. Data of all mission-involved aircraft are processed to allow a synchronized playback of the whole mission with simultaneous real-time numerical and graphical display as well as imaging sensors display. A man-in-the-loop mode of operation allows add-on simulation of alternate or corrective maneuvers for correlative tactics evaluation. Simulated scenarios, e.g bad weather, CM/CCM, enemy air defense etc. can be combined with live mission playback for training envelope extension and, vice versa, COMTESS-recorded missions, or parts of it, can be integrated into full mission simulator flying.

The ground station can be installed in a standard shelter, e.g. ISO container, i.e. it is mobile and air-liftable. It can be used and operated by air crew without any need of particular expertise training or assistance. Monitoring and interaction by special staff, e.g. weapon instructor pilots is optional depending on mission complexity and objectives.

COMTESS provides accurate visual mission reconstruction and playback for detailled evaluation and debriefing of both air-to-air and ground attack missions, range independent and daily available at squadron level worldwide. The system allows for various combination of live flown missions with complex simulator flying. COMTESS is low cost, easy handling, and fits any Sidewinder compatible and/or MIL STD 1760 interface aircraft without any need of modification.

GPS APPLICATION FOR FLIGHT PROFILE DETERMINATION

1. General Remarks

The global positioning system (NAVSTAR GPS) is a powerful means to determine user position, velocity and the clock error by measuring pseudo-range and pseudo-range rate to four satellites. Due to the inherently high accuracies, in particular in position determination, GPS offers new attractive ways for on-line and real-time flight profile determination and recording needed within the scope of advanced pilot training systems. Simply adding GPS to the existing training facilities, however, does not yield the potential benefits offered by GPS. New concepts must be developed and implemented which account for

- new flight combat tactis
- advanced weapon systems (air-to-air/air-to-ground)
- limited availability of actual flight hours
- requirement for training at sqadron level.

2. Special Conditions for Flight Profile Determination and Recording

For the on-line determination of the flight profile and the motion state of high performance, high dynamic aircraft special provisions must be incorporated to cope with the following operational characteristics:

- the GPS receiver will be installed within a missile body which is mounted on a standard sidewinder launcher to the aircraft wing
- the GPS receiver is exposed to severe environmental conditions
- due to antenna masking during high manoeuvring phases the number of available satellites may be limited or it might be necessary to switch between different satellite configurations causing different satellite geometries
- the antenna is not mounted in the aircraft center line
- the flight trajectory data can be determined with reference to a ground based station thus allowing for the implementation of the Differential GPS mode of operation.

Since the motion state vector of the aircraft beside position and velocity comprises information about linear acceleration, angular rates as well as heading and attitude angle of the aircraft, an inertial reference system (IRS) is integrated with an appropriate GPS receiver offering the capability for a full motion state (i.e. profile) determination in high dynamic environment and with high precision.

The high dynamic capabilities of the carrying aircraft on the one hand and the possibility for a frequent antenna masking on the other requires a very tight integration of the GPS with the IRS. In contrast to the commonly applied way of integrating an autonomous GPS system with a likewise self contained IRS, in the approach as shown below in figure 2 the sensor-hardware elements of both subsystems are combined in an optimal way utilizing a common system software which contains a Kalman Filter and the appropriate algorithms. This yields a minimum hardware and software amount and offers high dynamic accuracy in view of the particular operational conditions as indicated before.

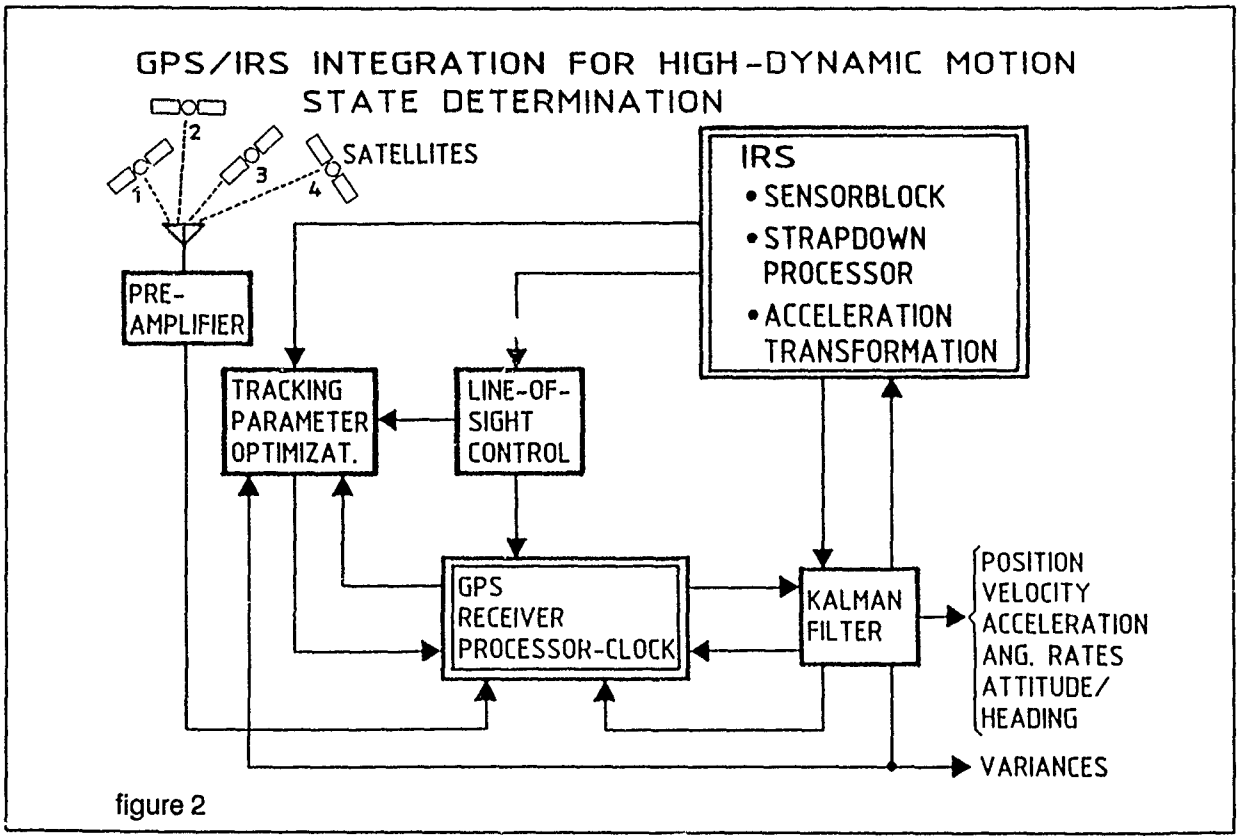


figure 2

As already mentioned all hardware modules needed for flight profile determination and recording must be implemented in a missile airframe partly leading to particular challenge as far as weight and volume of the equipment is concerned. Figure 3 delineates a realization concept for the combined GPS/IRS subssystem. It is based on recently developed inertial technology at BGT, e.g. as shown in figure 4 by way of a flight tested strapdown inertial midcourse guidance system for next generation air-to-air missiles, and available GPS receiver and processing technologies as well as hardware modules.

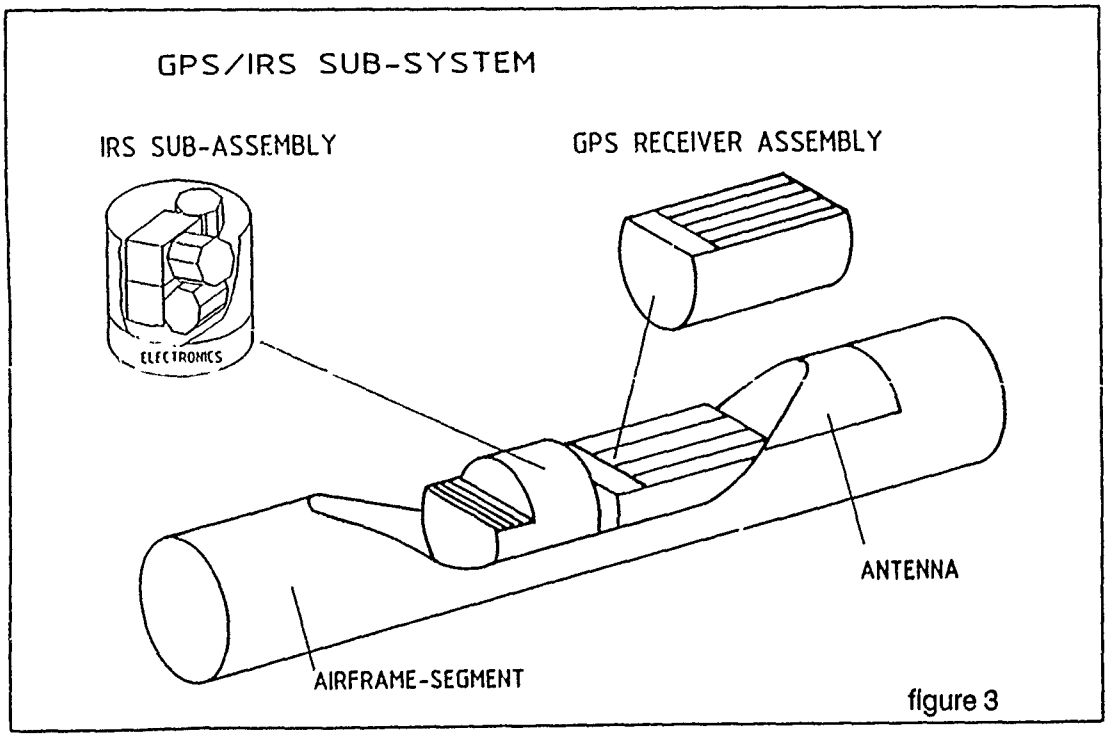


figure 3

STRAPDOWN
INERTIAL
REFERENCE
SYSTEM

FOR NEXT-
GENERATION
AIR-TO-AIR
MISSILES

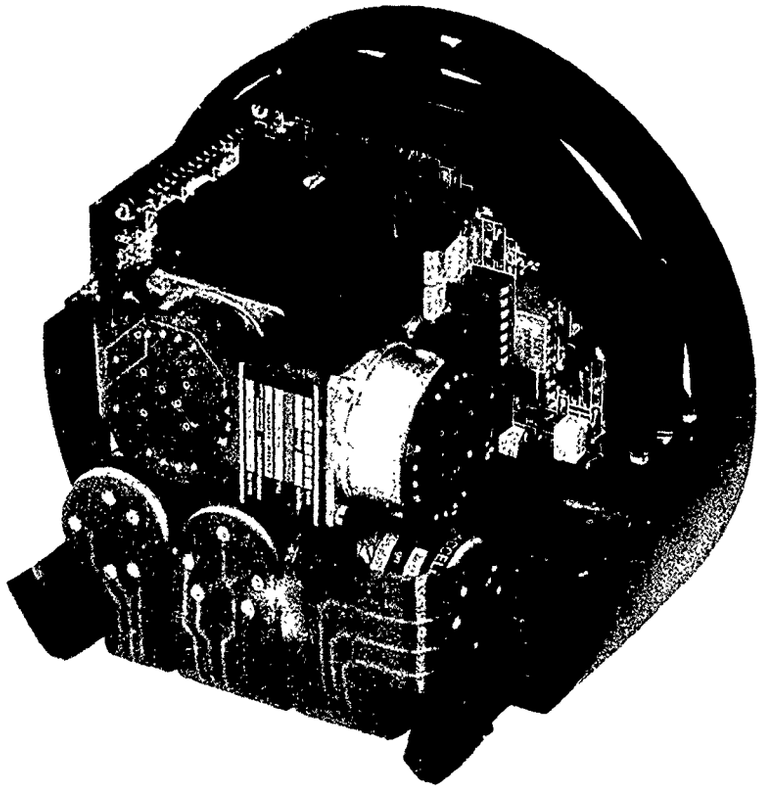


figure 4

3. Differential GPS Concept

Impressive absolute accuracies have been demonstrated for the NAVSTAR global positioning system during various tests and partly operational uses. However, particular applications like flight trajectory recording during training and exercise operations require even better accuracy than GPS is capable to deliver in the stand-alone mode.

The GPS-system includes a number of error sources which manifest themselves in a quasi-static offset-type error such that the measured GPS-position has an offset against the real position. During the transition from one satellite constellation to another, i.e. when the selected satellite constellation is changed due to the raising of a new satellite, step-function like position measurement errors of several meters amplitude have been observed. Furthermore, there is a noise content in each measurement which is receiver-dependent and oscillations are heterodyned. Altogether and globally, this leads to an offset type error in a stationary GPS application in the order of magnitude of 15 - 30 m.

Looking for GPS application in high-dynamic systems additional dynamic errors have to be taken into account which occur in mission phases replete with longitudinal, vertical and lateral acceleration as well as high rotary maneuvers. They are again receiver-dependent. In particular during low-level flying (A/G-missions) and within the scope of A/A-missions with frequent and large translatory and rotary maneuvers occasional antenna masking may occur, deteriorating the GPS performance if not considered appropriately. By the tight integration of the GPS with the IRS as shown in figure 2, dynamic errors as well as antenna masking effects are extensively reduced.

Differential GPS has emerged as a successful technique for achieving extremely precise navigation with GPS by reducing the static errors. It has been noted that the majority of error sources are related to the satellites and the propagation of their signals. The relative satellite geometry between two close terrestrial sites can be assumed to be very similar, because the satellites are operating at approximately 20000 km altitude. For this reason the range errors tend to be highly correlated in a local geographic area. Even considering that finalized data are not yet available, it can be stated that the local area appears to be as large as 300 - 500 km before significant spatial decorrelation of the range errors occurs. This area is sufficient when looking at the differential GPS application for flight profile determination and registration of fighter crew training flights within the normal radius of local training range and exercise area flying.

By the concept of differential GPS a number of common offset-type errors experienced by conventional GPS application are extensively reduced achieving few-meter or even better performance. To take advantage of the differential GPS concept a receiver reference station is located in the local area where greater accuracy is required. By employing this reference station as a second GPS receiver, and provided that its location is precisely known and thus can be compared to the position as derived from the satellite data, slowly varying, correlated errors can be isolated and eliminated. Based

on the assuption that correlated errors experienced via the receiver will be common to all users in a relatively close geographical area, by use of the reference station an appropriate estimate of its actual offset-type error can be obtained and transmitted to an airborne user where the latter may be able to compensate for the major portion of its own GPS system's errors.

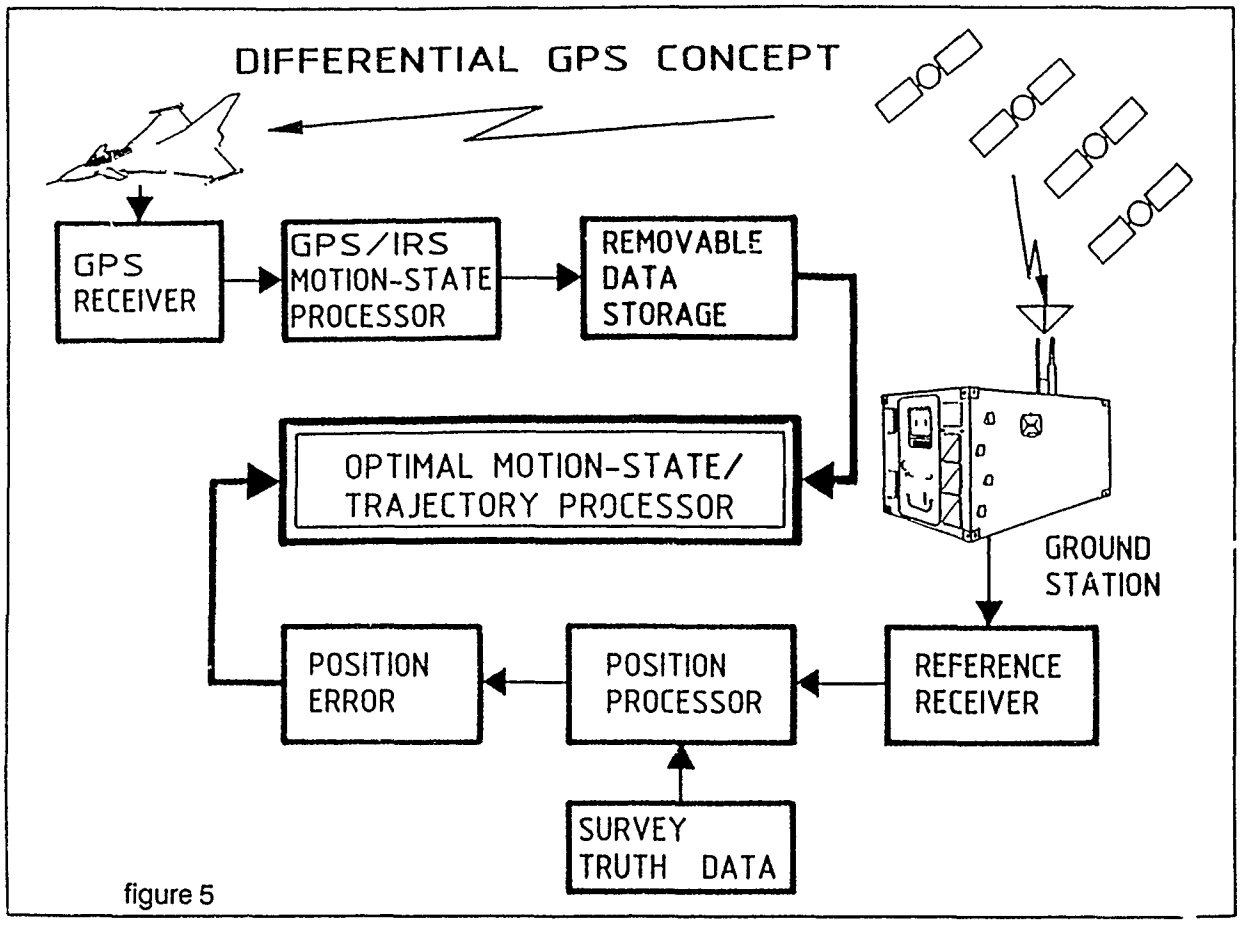


figure 5

There are various types of implementations of differential GPS discernible by the type of correction data and different data link options. If a real time differential GPS solution is required, up-linking of the differential corrections to the airborne user is mandatory. For the case under consideration here, the differential solution is only needed on the ground and thus can be calculated post-flight. Down-linking of the stand-alone solution in the mobile unit to the reference station on the ground by suitable removable storage is sufficient for this application where the role of differential GPS is for precise reconstruction of the aircraft's flight trajectory and motion status. The configuration is shown in figure 5.

Conclusively it shall be mentioned that a tight integration of GPS and IRS in combination with the application of a suitable differential GPS implementation, yields a viable and synergetic complementary solution which enables for the implementation of precise flight profile recording and reconstruction for high dynamic users operating in a somehow limited local area.

SYSTEM STRUCTURE

Data acquisition and recording systems covering the following aspects are required for the registration of flight profiles, aircraft data and target information during defensive and offensive missions:

- determination of flight profiles (position, speed) using GPS
- determination of other aircraft data (rates, heading, attitude) by means of an Inertial Reference System (IRS)
- image data of imaging sensors for the registration of target situations and the evaluation of flight profiles
- data recording equipment for storing the individual data streams.

Determination of these data in a field of operation with a minimum of restrictions and for the most various applications requires an overall structure which must take into account mainly the following aspects for in-flight data registration:

- use with different types of aircraft
- aircraft-independent measuring system
- use of standard interfaces/adaptors
- no restrictions of flight envelopes and aircraft maneuverability
- use of measuring system independently of training/exercise airspace location
- real-time recording of large quantities of data and high data rates
- easy handling of data carriers
- modular design favouring adaptation to specific mission requirements and optimization
- possibility fo upgrading with regard to changed requirements (growth potential).

In addition to these requirements the registered flight data have to be feeded into a ground station, prepared, compared with specified data and represented graphically for analysis by the staff involved upon conclusion of each training phase, i.e. for assessment, evaluation and debriefing.

In order to guarantee this performance the following requirements have been taken into account:

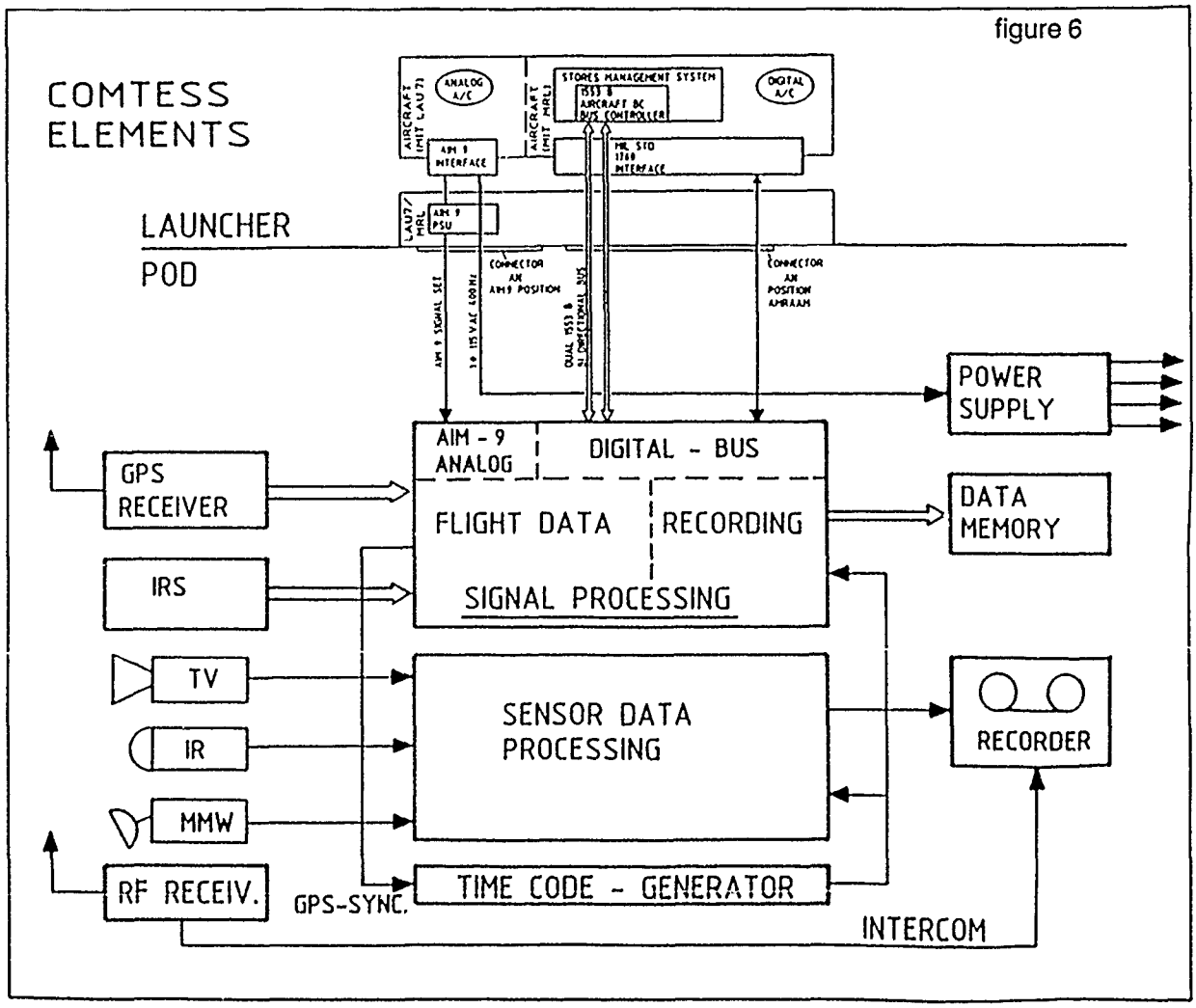
- simple transfer of flight date to the ground station
- design of basic function and operation of the ground station such that air crew can use ist
- option for functional extension for simulation of repeated flights with alternative mission scenarios, e.g. under the direction of a weapon instructor
- computer-aided data processing
- networked data sources.

These specified requirements have been the main design drivers during definition of the COMTESS solution which includes the following essential features/components:

1. Flight Data Acquisition System

- configuration of pod (mass, diameter, length, C/G) within the Sidewinder AIM-9 airframe dimensions
- flight data acquisition by means of Global Positioning System (GPS) and Inertial Reference System (IRS)
- additional information from imaging sensors (TV, IR, MMW)

figure 6



- signal processing for data preparation and reduction
- recording systems for storage of data, solid state for GPS/IRS and DAT for imaging sensors
- RF receiver for interphone and readio communication recording
- time code generator for synchronizing all data acquisition systems involved in the mission
- power supply for measuring equipment from airborne supply system (28 VDC, 115 VAC).

2. Ground Station

- data input station
- central processor
- flight trajectory processor
- scene generator
- image playback station
- hardcopy
- data link to external station

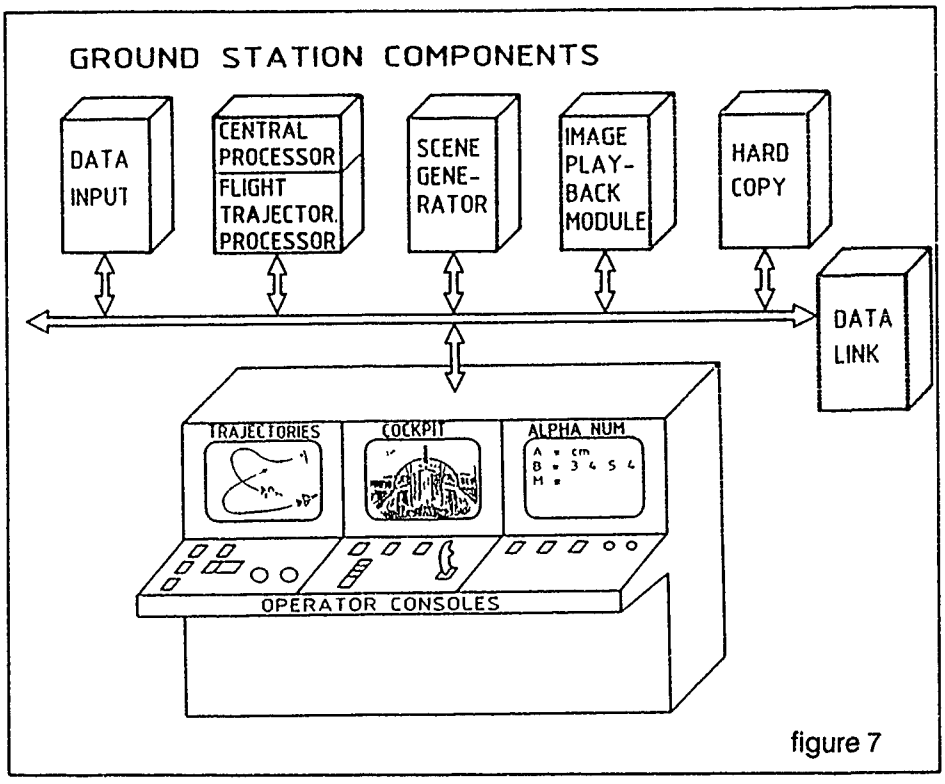


figure 7

SYSTEM MAIN COMPONENTS

1. Flight Data Acquisition and Registration

1.1 GPS Receiver

The application of a GPS receiver meeting the COMTESS requirements has other operational characteristics than usual applications in navigation systems. The receiver has tc be looked at taking into account particular criteria, such as

- fundamental receiver architecture
- type of signal processing
- internal flow of operations
- satellite selection procedure
- antenna configuration.

In addition, the GPS receiver assembly must fit into a structure corresponding to the Sidewinder missile body. The receiver configuration shown in figure 8 is particularly suitable for this application because its fully digital design and modular structure allows for miniaturization and adaptation to the specific applications.

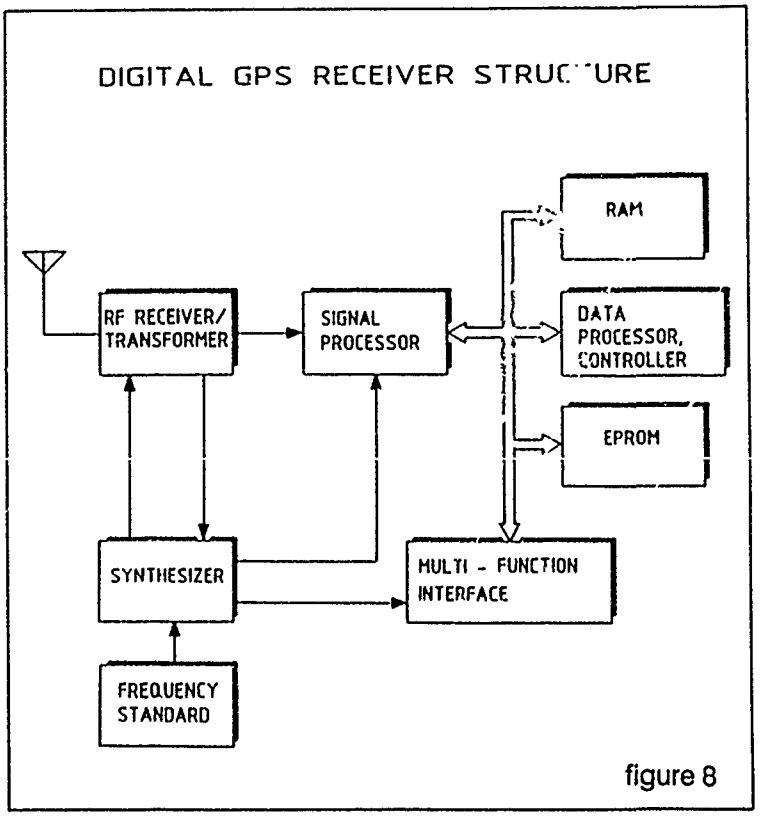


figure 8

1.2 Interial Reference System (IRS)

Besides the GPS data, additional important flight data such as rates and accelerations have to be registered for reconstructing the complete flight profile. This is accomplished by an Inertial Reference System (IRS). The main components of an IRS are the accelerometers and the rate sensors which are rigidly attached to a sensor block. This block is installed in the pod and is thus rigidly connected to the aircraft (strapdown system). As a result, the Inertial Reference System is exposed to the full aircraft dynamics and has to be designed to cope with the maximum maneuverability of the aircraft. The system used in COMTESS (figure 4 on page 4) is a BGT-developed strapdown system for next-generation missile midcourse guidance, incorporating dynamically tuned gyros (DTG) of a large bandwidth and represents a state-of-the-art system. It features compact design and wide measuring ranges for accelerations and angular rates.

1.3 Imaging Sensors

The representation and analysis of image data from electro-optical sensors allows the evaluation of target situations and the verification of flight profiles or parts of it. The use of different sensor systems enables for real image scenes generation in various wavelength ranges. These scenes extend the spectrum of applications of the overall system under the following aspects:

- optimization of the sensor system with regard to mission objectives
- generation of scenes corresponding to the weapon systems to be used, e.g. IR, Radar, TV, multimode
- use of the system independently of weather conditions.

On the basis of the recorded scenes and together with the flight profile and the aircraft data, image processing systems can generate comprehensive information packages and represent them graphically for post-flight assessment and evaluation. In connection with the use of different electro-optical systems, especially the following performance parameters play a decisive role:

- range in different weather conditions
- field of view
- resolution
- image frequency.

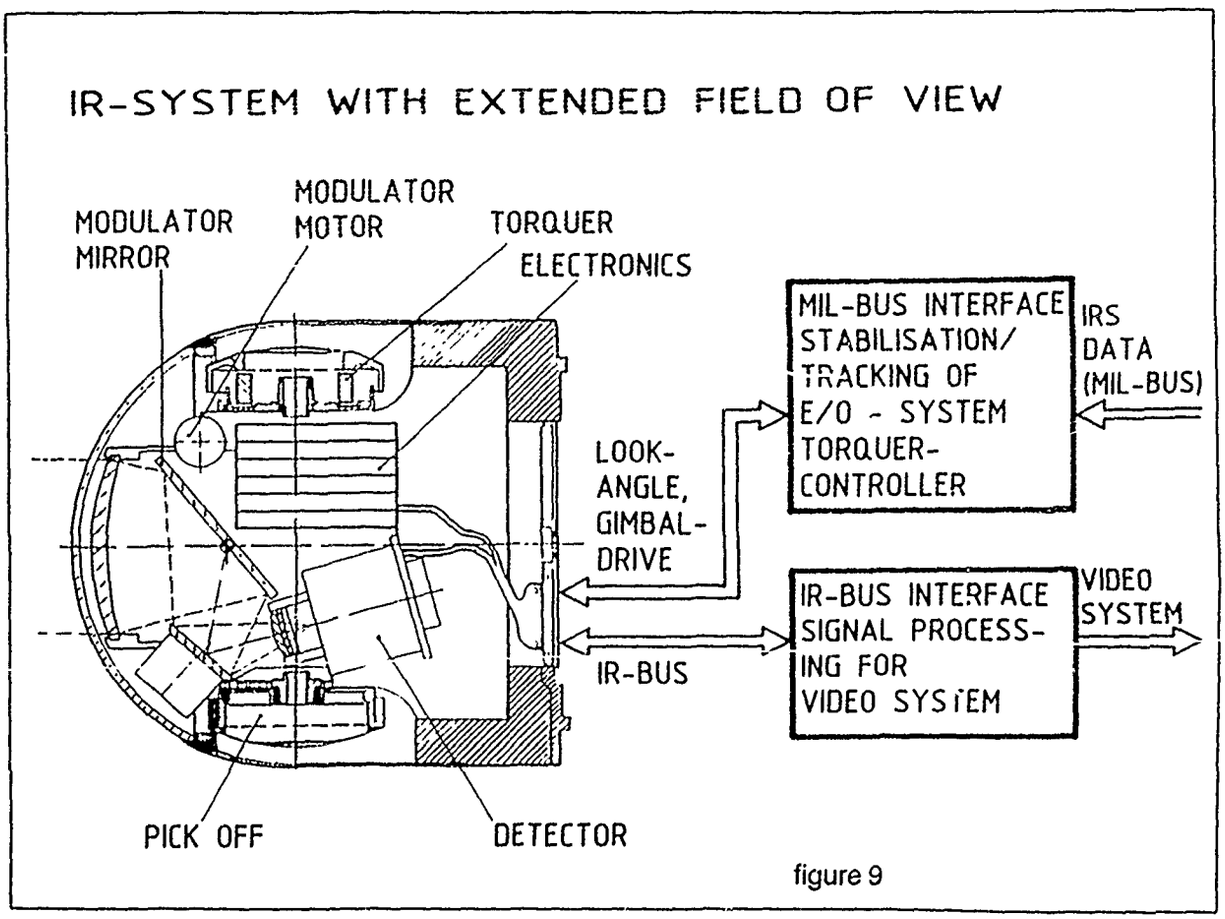


figure 9

BGT has developed and flight-tested sensors working in various modes, such as miniaturized CCD-TV, wide field of view IIR, IR/MMW and multi-spectral. These sensors are available for application to COMTESS.

Figure 9 schematically shows the example of an IR system which is capable of imaging large fields of view with a high image frequency and resolution. In this case, the infrared image is generated by means of a modulator mirror, i.e. the optical beam path passes an array of detector lines. For acquisition of larger fields of view the complete gimbal system is scanning. The IR signals generated in the detector are amplified, filtered and digitized in a signal electronics. Via a specific IR data bus they are then transmitted to a video electronics where they are conditioned for video recording.

1.4 Signal Processing System

The objective of the signal processing system is to collect and process the data from the GPS and IRS and to have the data stored in the removable recording unit. This sequence is provided by the multi-processor system software. Figure 10 shows the multi-processor system structure for the signal processing of IRS and GPS data, using digital interfaces. The internal processing speed is adapted to the data rate of GPS and IRS.

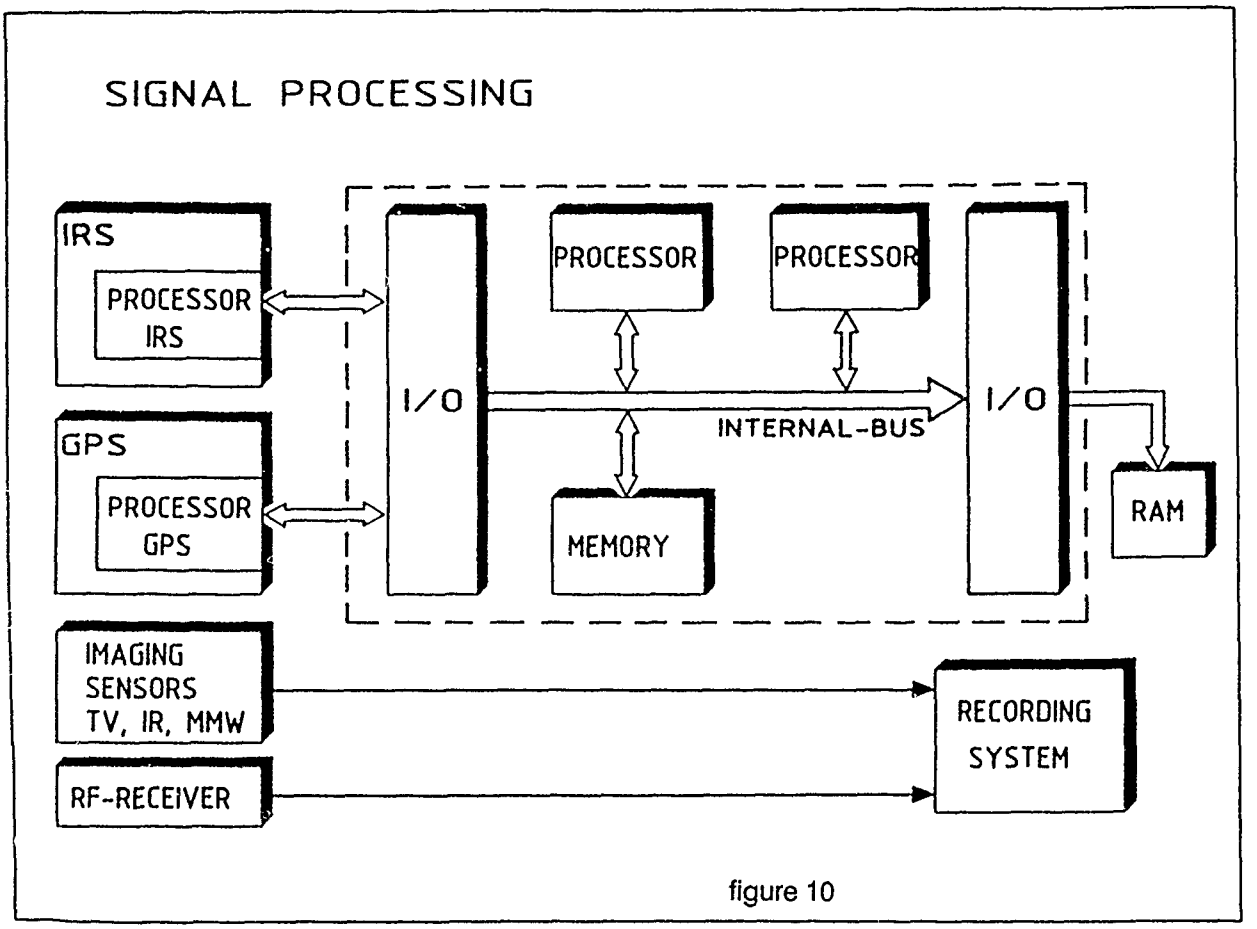


figure 10

1.5 Data Recording System

Depending on the signal types and formats, the central signal processing system prepares the sensor data so as to form two data streams. Because of the very high data rates of imaging sensors, storage is only possible on a magnetic tape, e.g. video recorder, digital audio tape (DAT), instrumentation recorder. The pre-processed data from the GPS receiver and the IRS are available in digital form which allows for storage in available solid state memories, e.g. RAM.

1.6 Pod Structure

The pod structure is an AIM-9 Sidewinder missile body and accommodates all system components needed for the acquisition and registration of flight profile and imaging sensors data. To support easy flight certification and to ensure full aircraft maneuverability, the Sidewinder characteristics in terms of mass, centre of gravity and electrical interface are retained.

Figure 11 shows the arrangement of equipment in the flight data registration pod (sensor pod).

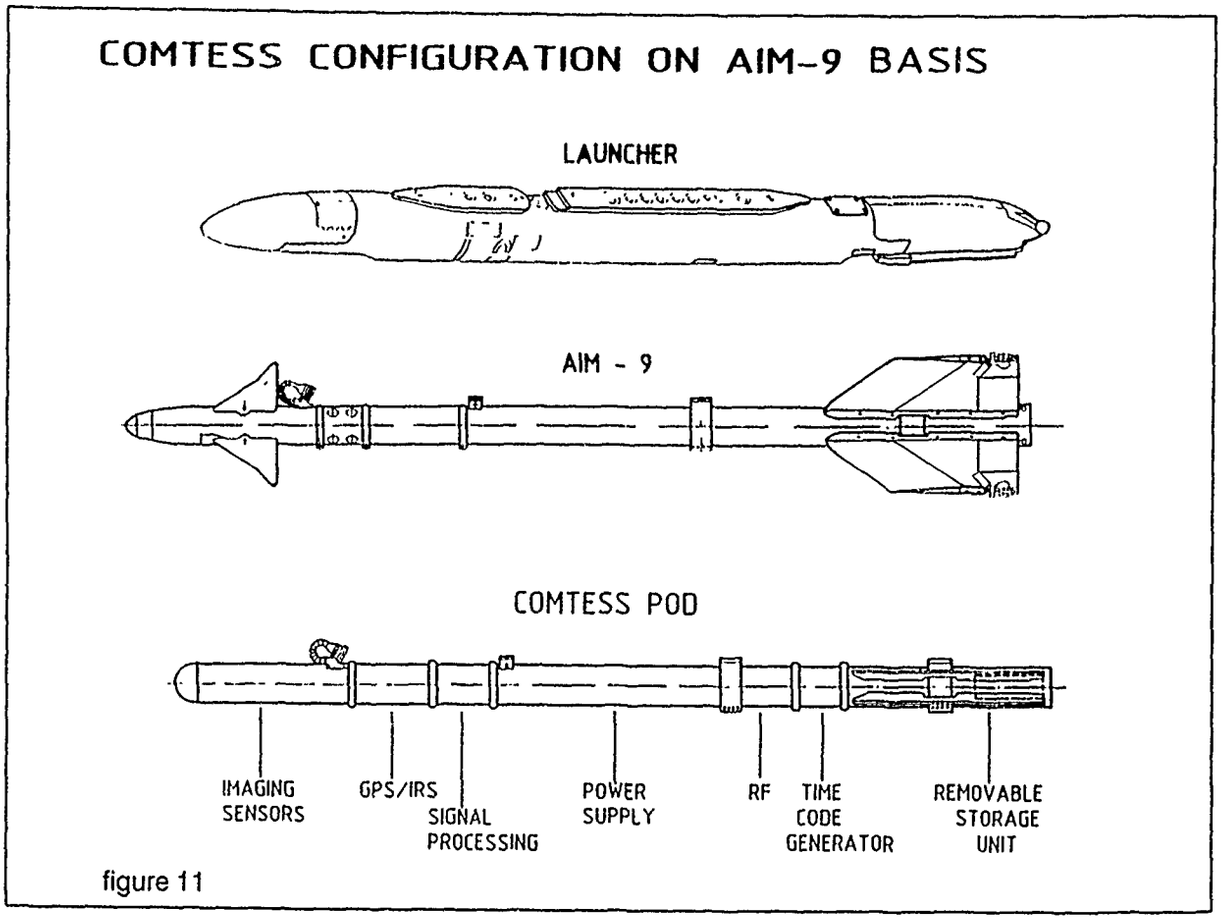


figure 11

2. The COMTESS Ground Station

2.1 Design Objectives

The COMTESS solution embraces data gathering, data recording and scene presentation, the first two items being part of the airborne COMTESS equipment whereas ground equipment is governed by data expansion, visualization, animation and presentation. Therefore, the COMTESS ground equipment relies strongly on skillness in modern computer architecture, hardware design and software techniques. In addition to purely replaying the actual flight data COMTESS allows for comparision between mission planning data and real flight data. As a third capability COMTESS can simulate alternate flight trajectories when used in its "man-in-the-loop" operation mode. Using a stick input the pilot can try out alternate maneuvres and evaluate their effectiveness. This mode, of course, should be solely regarded as a lower level aid to the pilot when discussing alternatives and is not intended to backup or to replace a training simulator at all. Fourth, COMTESS can optionally record and replay images of multi-spectral imaging sensors.

The COMTESS ground station highlights include:

- visual reconstruction of recorded flight- and trajectory data, i.e. complete mission profile
- comparision of planned mission and real mission data
- man-in-the-loop simulation for post-flight tactical discussions
- recording and replaying of the imaging sensors registrations
- playback of selected important cockpit displays (Radar, FLIR, TV etc.), depending on A/C interface capability.

2.2 Modular Design

The COMTESS modular design philosophy also applies to the ground station, resulting in realization of blocks for a range of stations with increasing capabilities.

The Central Processor (CP) Module is capable to support the basic functions for:

- reading the recorded trajectory data of a number of aircraft
- assembling the trajectory sets of multiple aircraft into a combined mission trajectory set
- trajectories playback as would be seen from a freely choosen viewpoint, using generic symbols for aircraft and trajectories and offering slow motion, zoom etc. whilst adding generic navigation and sight information
- continuous display of relevant flight data on the operator consoles.

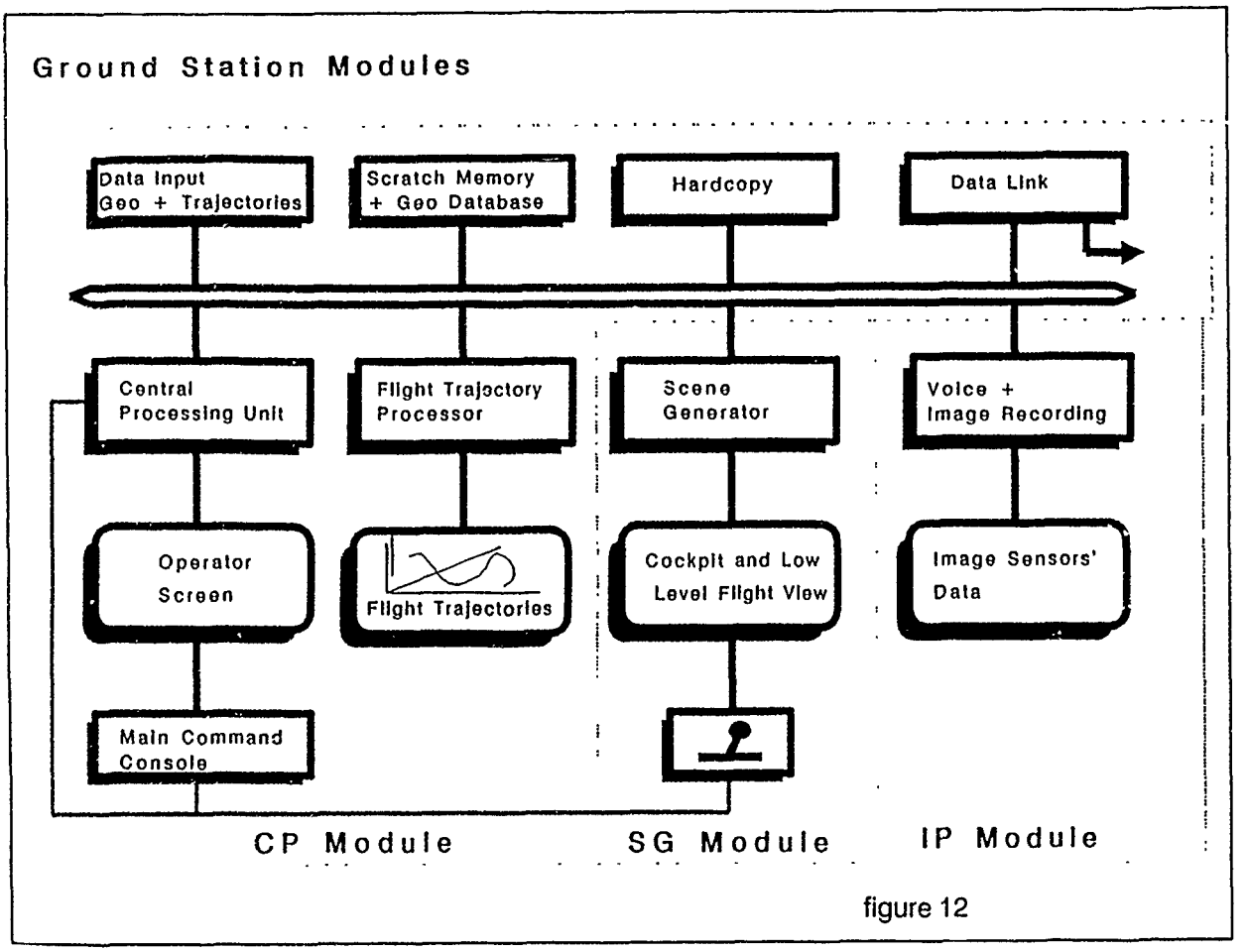


figure 12

Adding the Scene Generator (SG) Module to the basic CP Module results in extended capabilities:

- visualization of the scenes as seen with pilot's eyes, including realistic navigation fixes and surface representation, optionally based on digitized geographical data
- man-in-the-loop operation with the pilot generating mission variation data by manual stick control. These alternatively simulated data can be visually compared on the screen with recorded data for compliance. In addition, mission variations and alternate maneuvers can be generated as add-on simulation for comparison and tactics discussion/evaluation.

Adding a second SG Module results in a limited capability for dogfight engangement visualization with both pilots' cockpit views presented on two screens. Together with the generic 3-D-trajectory screen display this configuration offers an advanced environment for effective and useful dogfight debriefing. Because there are now two joy-sticks available it is also possible to "freeze" the replay of the trajectories at any point of time and to continue from there in a "dual flight simulator" configuration. This operation mode, of course, cannot replace a real dogfight simulator, but it offers a degree of try-and-error type of post-flight debriefing obtained never before in a low cost system.

The Image Playback (IP) Module is used for synchronous display of recorded imaging sensors' data, providing valuable augmentation of the debriefing information. Figure 12 shows the ground station modules.

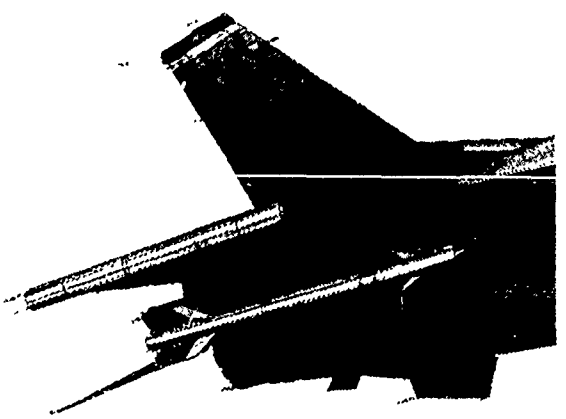
2.3 Ground Station Technology

All technology required for the previously presented components and capabilities is available off-the-shelf on chip basis or on basis of ASIC design technology. The Central Processor Module only requires standard chip types including signal processors, DRAMS etc. and is built around the industry standard VMEbus. The CP module has certainly been the most challenging component from the system designer's standpoint. Taking into account that there was not a training simulator targeted, a number of 3600 polygons (triangles) was found to be adequate for a semi-realistic surface representation. A screen resolution of 600 x 800 pixels at an information update rate of 25 pictures/sec. with a 75 Hz non-interlaced screen update rate guarantees for excellent brightness and geometrical resolution. These image generation characteristics result in approximately 7 megabytes/sec. of image data transfer load on the system bus, which is easily achieved by the VMEbus and which allows also for a dual image generator configuration.

As an effect of using ASIC's for basic graphic functions, such as z-buffer-technique, area overlay, area fill, look-up tables etc. the central processor's computational speed requirement for the SG module could be kept within the performance scope of modern 80 Mflops signal processors. BGT has extensive experience in the fields of computer architecture and ASIC design with the 17 Giga-ops BVR image parallel processor being the very best evidence for that claim.

Last but not least it is to mention that in addition to training quality enhancement, COMTESS provides an excellent and reliable tool for post-accident and post-crash investigation and analysis. The solid state removable memory unit which stores the GPS/IRS-determined flight profile data, is based on an available BGT-developed RAM Module, which survived almost 40.000 g in life firing out of a Leopard Tank Gun, giving evidence that all data needed for accurate reconstruction of critical flight portions would be available even in case of fatal aircraft destruction. This is a deliberate and valuable spin-off feature of COMTESS, to support a key objective of flight safety investigations and resulting conclusions - prevent flight accidents.

COMTESS is our answer to the increasing demand for better airborne training effectiveness, providing the tool for extensive post-flight evaluation and tactics optimization at costs far below conventional flight trainers.



INTEGRATED TECHNOLOGY DEVELOPMENT LABORATORIES

by

Donald E.Dewey
The Boeing Company
P.O. Box 3707
Seattle, Washington 98124-2207
United States

SUMMARY

New integrated avionics technologies are capable of providing the performance improvements needed for current military aircraft. However, integrated laboratory facilities are needed to fully realize the potential of these technologies. The Boeing Company has developed such a facility, a single laboratory capable of studying highly integrated avionics systems from research through full-scale development.

INTRODUCTION

As military aircraft have become more complex, engineers and scientists have begun to investigate new technologies and methods to increase aircraft system performance and mission effectiveness. In the early 1980s it became apparent that by (1) automating many of the functions normally performed by the pilot, (2) extracting and computing situation information from multiple sensors, and (3) simplifying the informational content of the pilot's displays and controls, significant improvements could be made in mission performance. Pilots would have greater awareness of their overall situation and environment, aircraft could be made to fly closer to the in-control limits, and aids could be provided to the pilots that would simplify and improve the battle and mission performance of the aircraft system.

Although it was clear that the new technologies being studied, referred to as integrated avionics, could provide performance improvements, traditional aircraft system laboratories were ill-equipped to support the research needed to fully realize the potential of these technologies. Laboratories were autonomous, standalone facilities and could not be tied together to integrate aircraft functions. High-speed networks were becoming available, but often the laboratories were widely separated geographically and available networks could not be used. Flight simulation systems, which included test cockpits and simulated external visual scenes, were difficult and costly to reprogram.

The Boeing Company studied the feasibility of constructing a single facility that would have all the needed standalone laboratories, yet enable these laboratories to be tied together to conduct simulations in an integrated avionics mode. The study showed that such a facility was feasible. With the availability of a new generation of high-resolution digital visual generators, high-speed host computers, and high-speed networks to support real-time simulation, it became evident that simulations could be conducted effectively in an integrated facility, and that it would be economically practical to construct such a facility.

INTEGRATED TECHNOLOGY DEVELOPMENT LABORATORIES

Boeing made a commitment to construct a single laboratory capable of studying highly integrated avionics systems, starting at the research phase, proceeding through demonstration and validation phases, and continuing finally to initial full-scale development phases. Construction began in 1984 on the 9,800-square-meter facility, which became fully operational in 1987. Located at the Boeing Developmental Center in Seattle, Washington, the Integrated Technology Development Laboratories (ITDL) (figure 1) composed the first totally integrated facility of this kind in the United States. In its early configuration, the ITDL contained separate flight control, digital avionics, and crewstation laboratories, plus three large domes for simulating external scenes, cockpit graphics, and sensors for flight simulation (figure 2). A tower for real-time sensor testing was added in 1988, as well as an M-versus-N multiple engagement simulation system. Construction will begin during 1990 on a 3,800-square-meter addition, which will enable more pre-full-scale development programs to share use of the facility.

The guidelines for the ITDL were established early and were adhered to throughout the design phase. They were as follows:

- Retain the autonomous functions of the technology laboratories (e.g., flight control) but provide for operation in the integrated mode.
- Provide the capability to run integrated classified and unclassified programs concurrently.
- Provide a resource base for simulation that could be shared by multiple programs in a classified environment.
- Design the laboratory communication networks so that each laboratory can support multiple users.
- Standardize computing resources as much as possible.
- Collocate simulator laboratory functions so that these laboratories can share common computing and other resources.

© The Boeing Company 1990

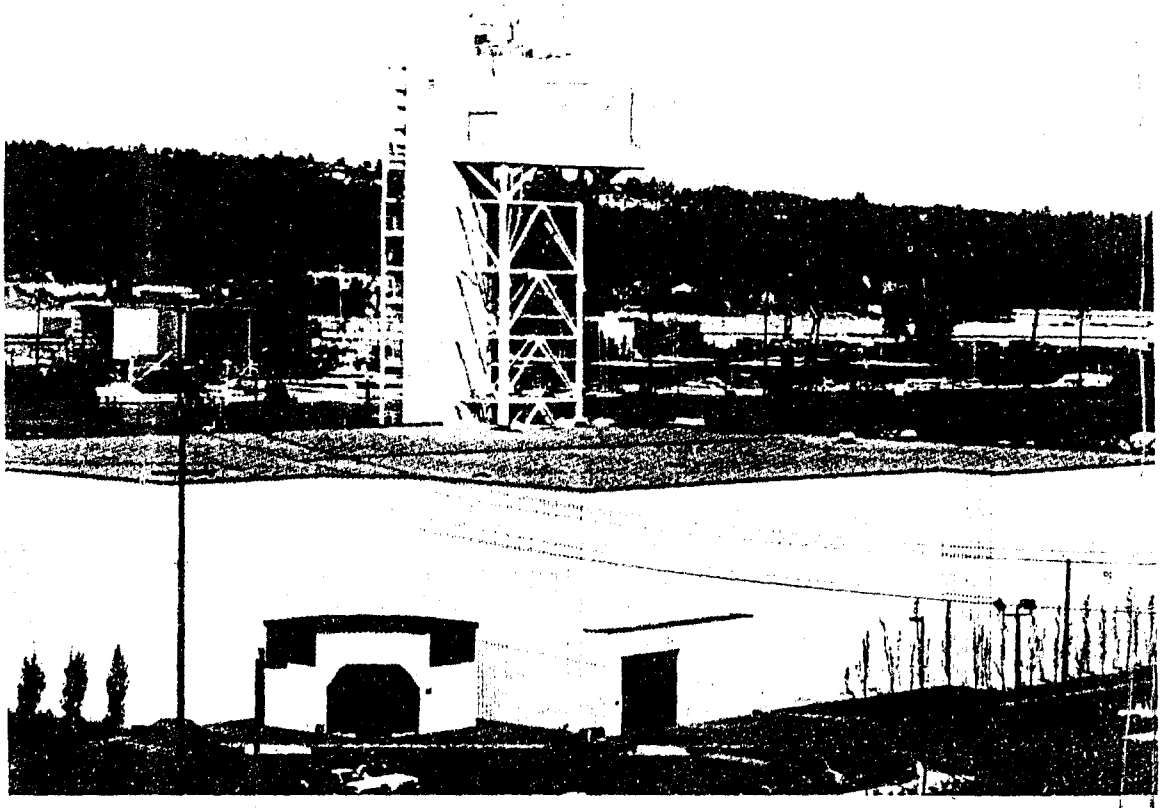


Figure 1. Integrated Technology Development Laboratories

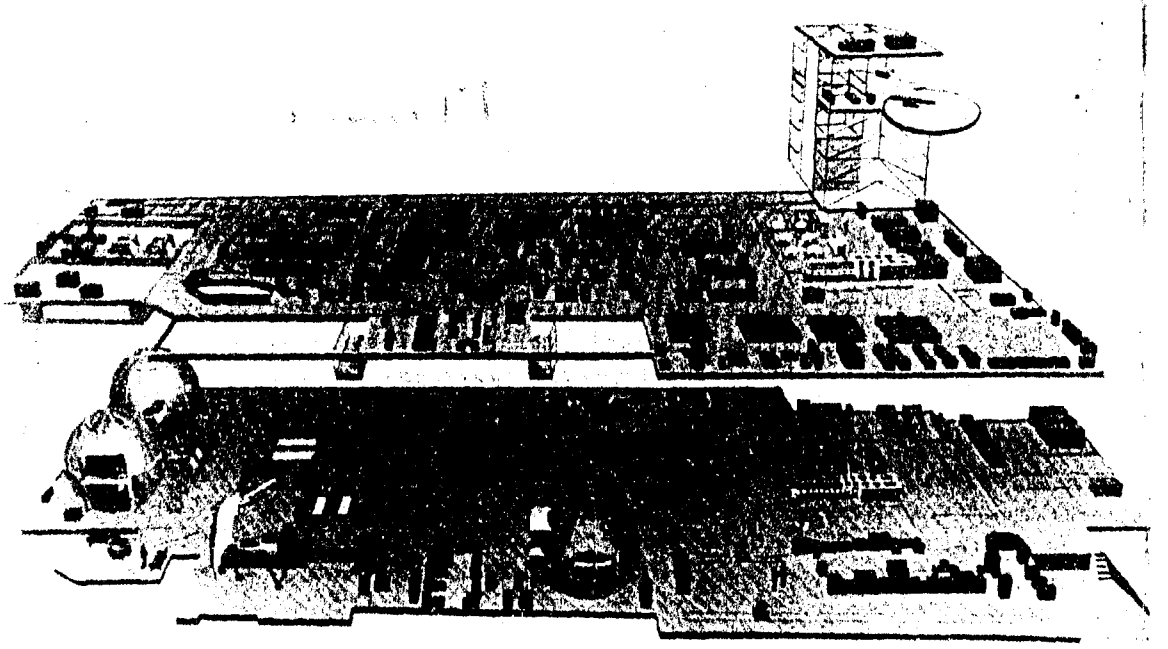


Figure 2. Integrated Technology Development Laboratories Configuration

DESIGN

The ITDL design evolved into a single facility containing multiple laboratories capable of independent operation or interconnected to time shared computer and flight simulation resources. This facility was placed in a total TEMPEST environment with physical security to allow multiple secure and nonsecure programs to coexist. This approach allows high-value assets such as flight simulation domes, motion bases, and computing resources to be reconfigured to support multiple

programs on a daily basis. Simultaneously, projects can use dedicated laboratory space for program development. For example, actual flight sensor hardware (including cooling and power) can be installed in the sensor tower to check real-time operation. Sensor output data can then be transmitted to the project laboratory to study sensor fusion algorithms with hardware in the loop.

A complete machine shop and electronic assembly area supports all ITDL projects. The 3,800-square-meter addition scheduled for completion in 1991 will provide additional laboratory, ITDL engineering staff, and cockpit development areas.

FLIGHT SIMULATION SYSTEM

At the heart of the ITDL is the man-in-the-loop flight simulation system (figure 3). The basic elements of the system include three 9-meter, 360-degree-field-of-view domes and two CompuScene IV computer image generation (CIG) systems that provide simulated high-threat, day/night, and all-weather environment scenes (figures 4 and 5). A database generation system develops the CIG backgrounds. A projection system projects images representing other aircraft and objects onto the dome surface, superimposed over a full-dome sky/horizon/ground background scene. These scenes are under real-time control of either a mathematical model or a pilot in another simulation cockpit. Silicon Graphics, Inc., computer systems generate graphics for all the displays in the interchangeable test cockpits. The crewstations are fully operational and instrumented fighter, bomber, and transport cockpits that can be reconfigured to meet specific requirements. Simulations are hosted on three Encore computer systems, which can be interconnected depending on simulation complexity, number of players, and so on.

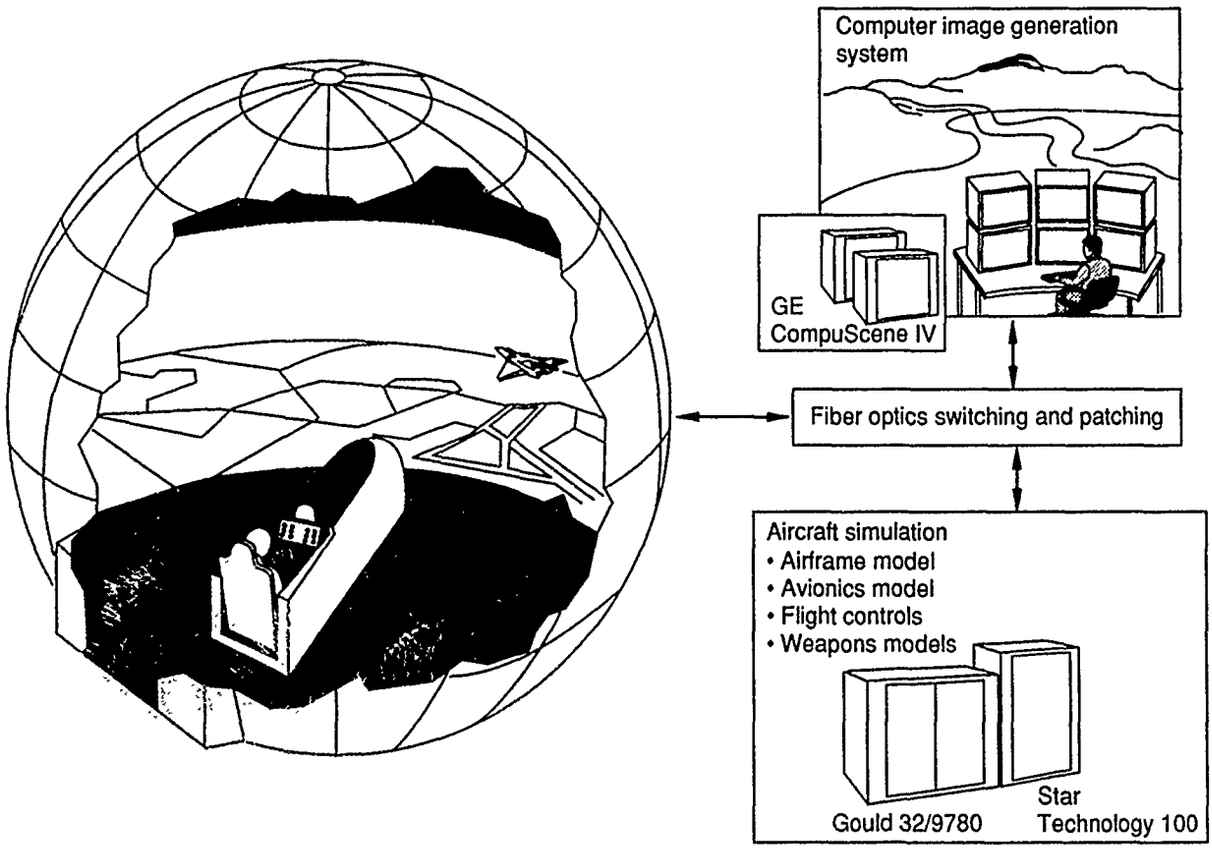


Figure 3. Man-in-the-Loop Flight Simulation System

In addition to the three domes and cockpits, there are 12 consoles (figure 6) that can be manned by pilots or operators to simulate and study multiple engagements in real time. Like the cockpits, the consoles can be reprogrammed to meet specific requirements. Both the cockpits and the consoles provide working platforms for real-time evaluation of control and display systems, including flight control system development and analysis, flight deck layout and development, crew and equipment interface studies, flight procedure training, and flying qualities evaluations. When used in the integrated mode (with the various standalone laboratories tied together), the ITDL can support research projects including sensor fusion mission effectiveness, target acquisition, and weapon delivery studies. The multiple engagement simulation system (figure 7), a software environment that controls the simulation in the integrated mode, drives and controls various aircraft, sensors, missiles, and ground threats that interact with each other in real time. The system provides a programmable testbed for a variety of projects that range in size from a single aircraft simulation to a large, complex M-versus-N air combat simulation. The system is designed to be able to be reprogrammed without massive and costly regeneration of software.

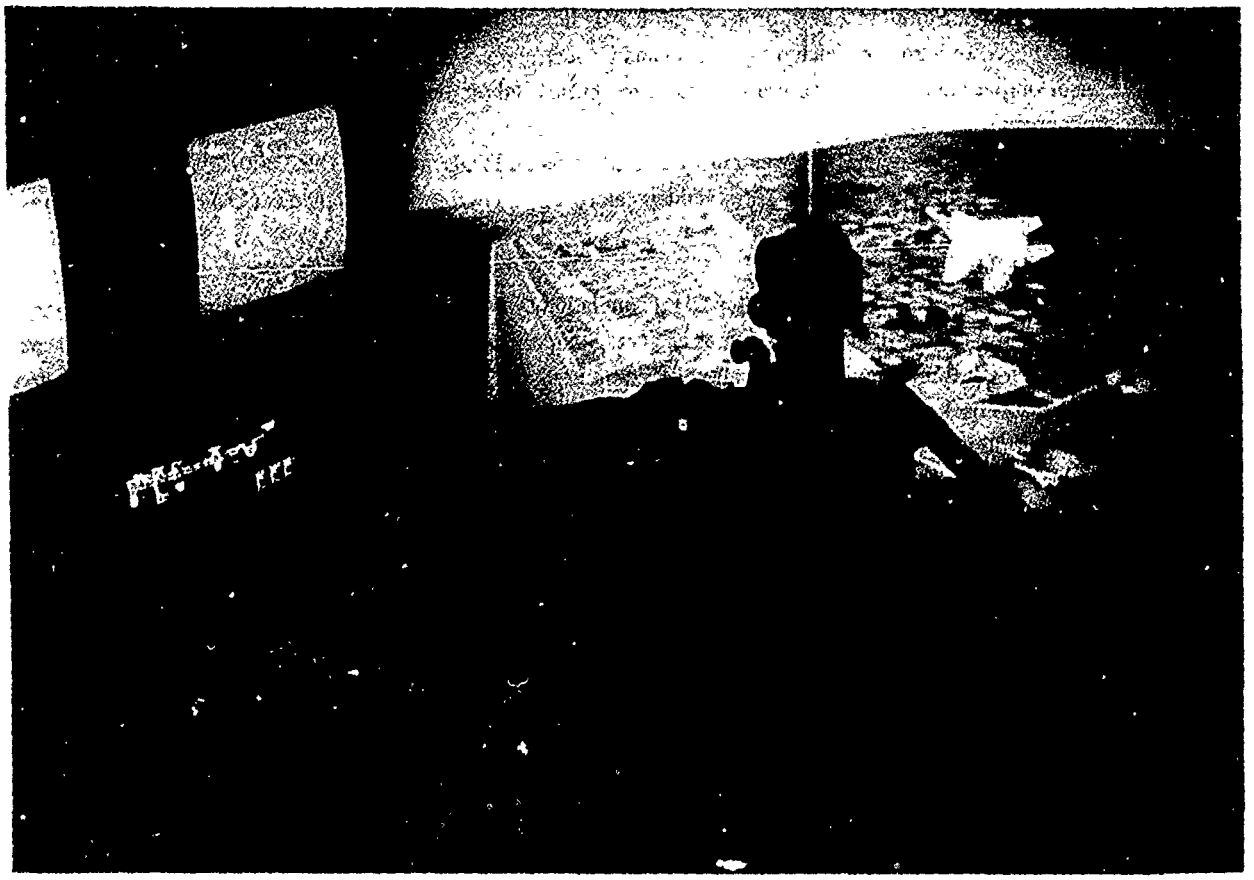


Figure 4. CompuScene IV CIG System

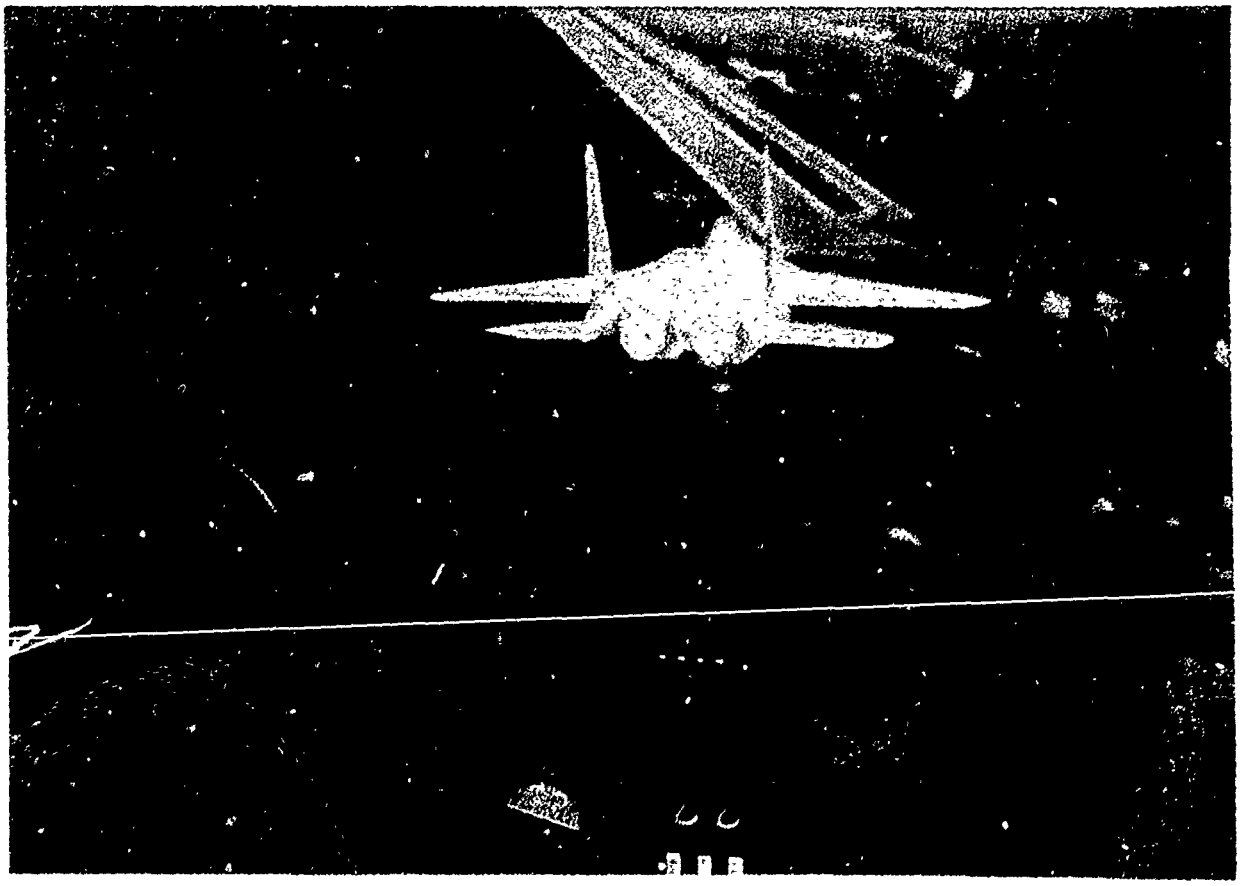


Figure 5. CompuScene IV CIG System

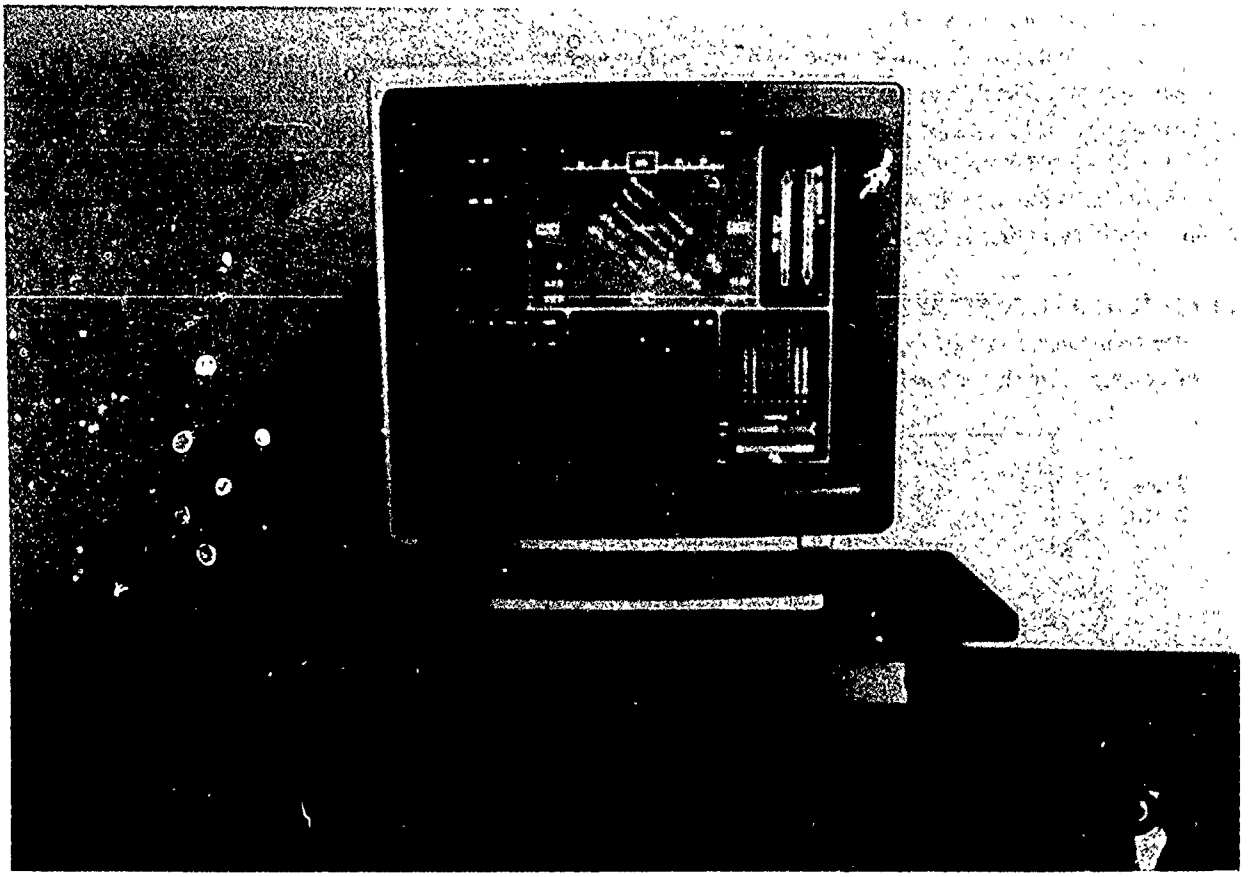


Figure 6. Integrated Technology Development Laboratories Console

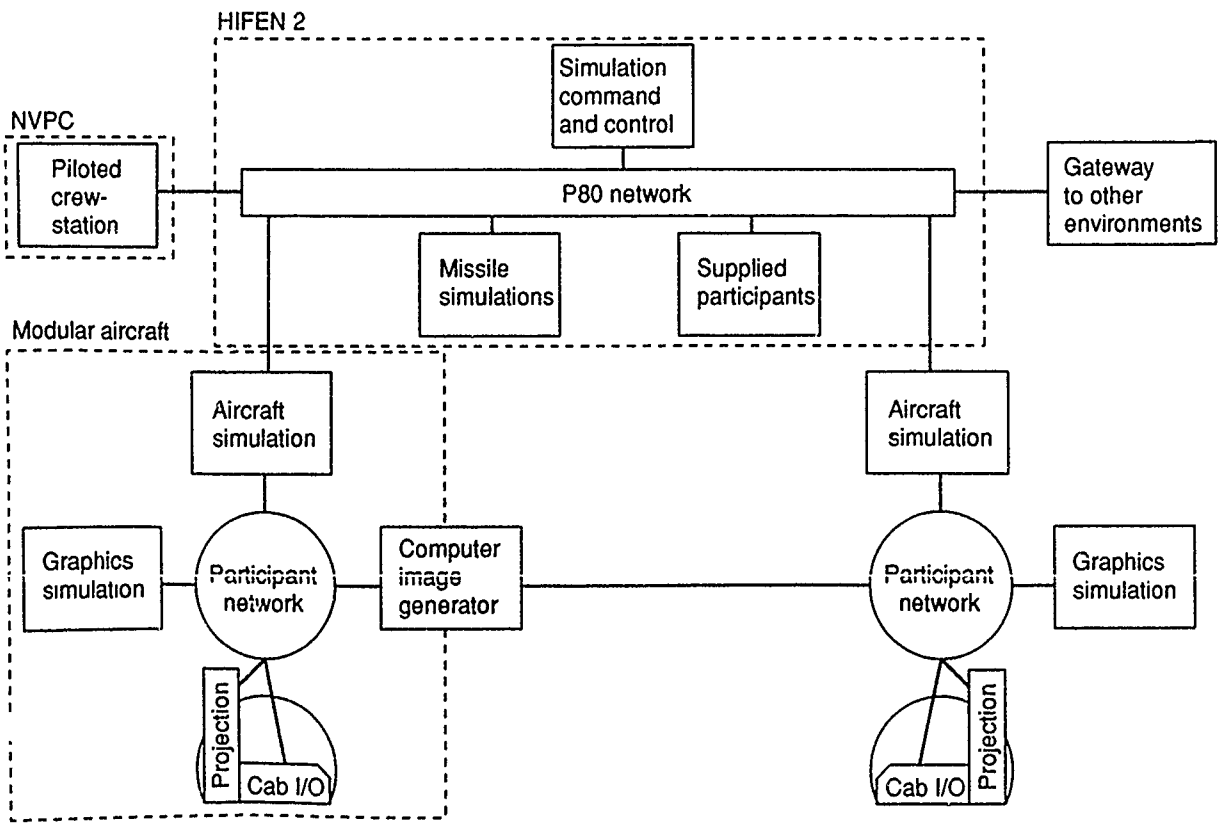


Figure 7. Multiple Engagement Simulation System

COMPUTATIONAL SYSTEM

Although dedicated program laboratories may contain their own computational systems, the ITDL has a number of computer systems available for scheduled development use. When not used for real-time simulation, any of the three Encore 9780 computers can be used for secure or unsecure development, along with a dedicated engineering development 9780. A 32-bit DEC VAX 8700 computer is also available for unclassified development. This system serves more than 200 terminals in the ITDL and is linked to other Boeing facilities for remote use. A Dasix computer-aided design system is used to develop complex, multilayer printed circuit boards used in the high-speed real-time simulation data networks.

COMMUNICATION NETWORK CONTROL SYSTEM

The communication network control (CNC) system provides the communication backbone for the ITDL (figures 8 and 9). Fiber optics were chosen for their bandwidth and reliability. Fiber-optic links run in solid tubes from each laboratory to

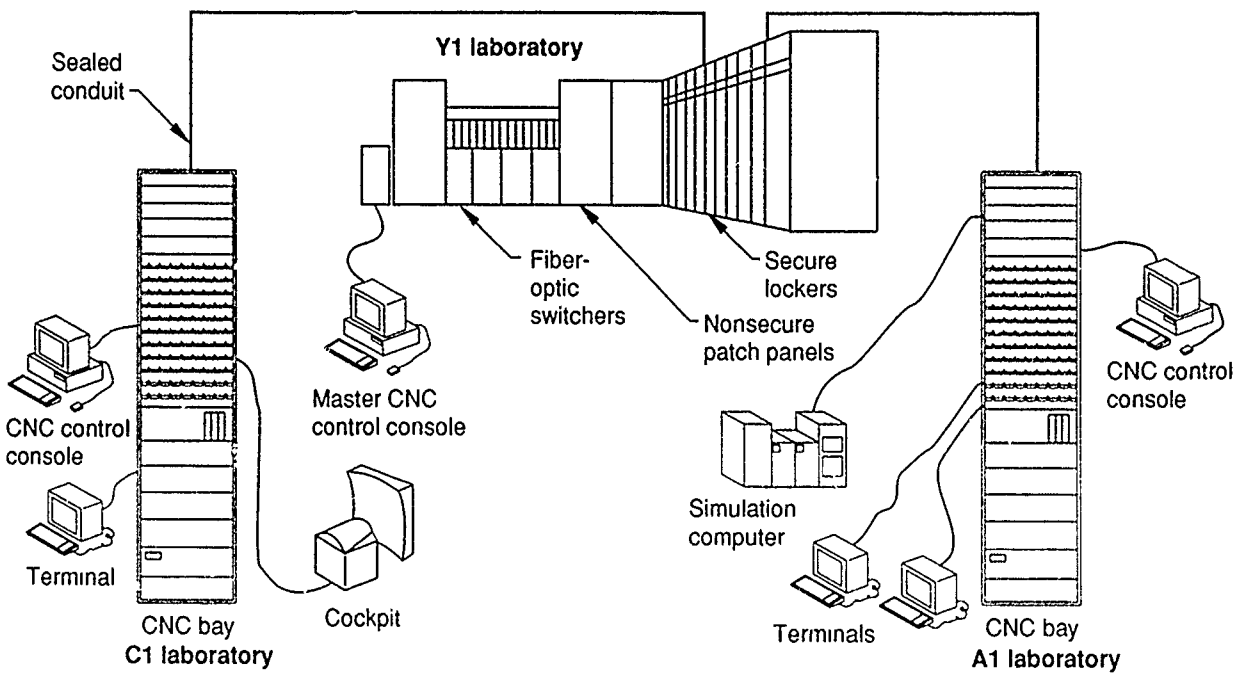


Figure 8. Integrated Technology Development Laboratories Network Configuration

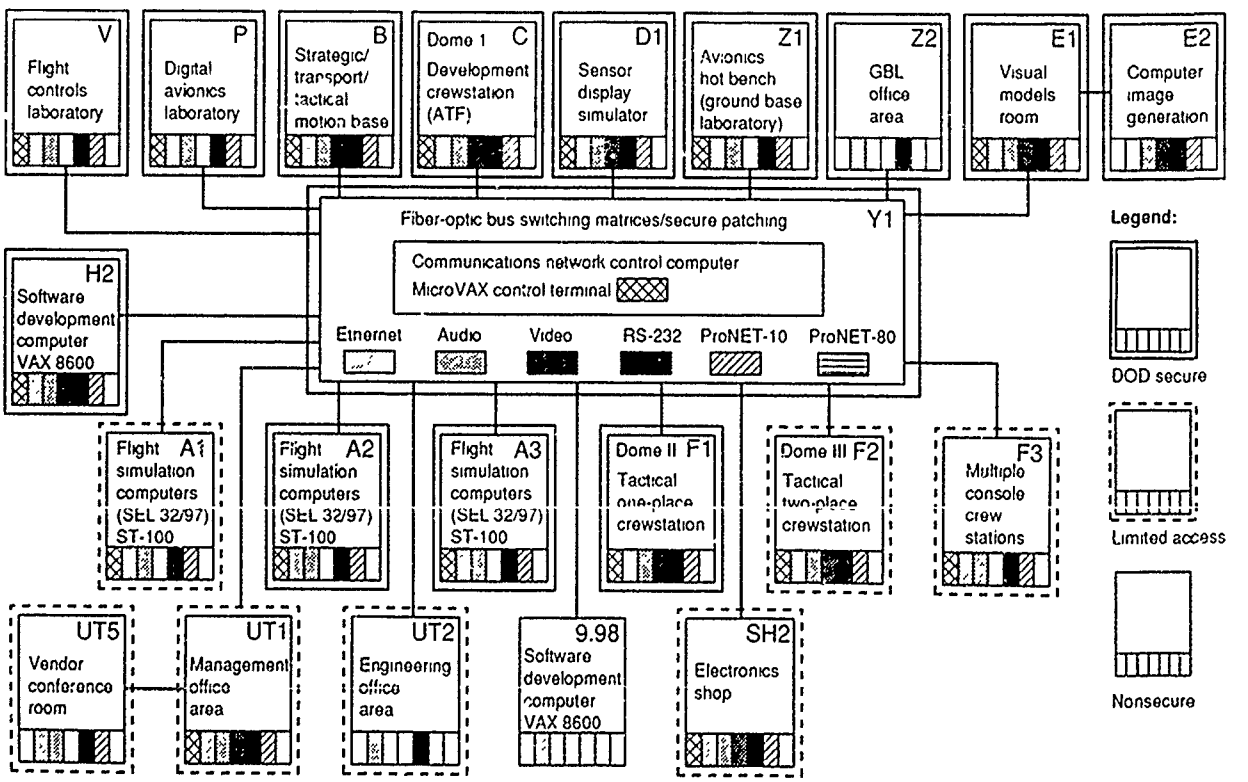


Figure 9. Integrated Technology Development Laboratories Network Configuration Diagram

a central secure switching room. Multiple protocols are supported, from high-speed datalinks for simulation to standard Ethernet and RS232 for communication. When unclassified, laboratory communication configurations are changed by a CNC-controlled fiber-optic switcher. Classified links are set up by hand through a set of securable lockers. These lockers can support multiple classified programs (not in the same laboratories) simultaneously. The total system is designed to permit the individual laboratories to operate separately or in integrated fashion and still meet U.S. Department of Defense 50-3 level security requirements.

ITDL USE

Since becoming operational in 1987, the ITDL has supported Boeing Military Airplanes division avionics demonstration and validation programs; a number of technology contracts, such as the Cockpit Automation Technology program; and independent research and development projects. Some of these research programs could not have been completed successfully without the use of the IDTL. The ITDL staff comprises more than 100 engineers and technicians on two shifts who design, maintain, and operate the systems and software in support of IDTL custom ers. Planned additions to the IDTL will further enhance its unique capabilities and enable the facility to provide support to a variety of new programs.

SIMULATION OF NAP-OF-EARTH FLIGHT IN HELICOPTERS

by
Gregory W. Condon
Chief, Flight Systems and Simulation Research Division
NASA Ames Research Center
Moffett Field, California 94035, U.S.A.

SUMMARY

NASA Ames Research Center, in conjunction with the co-located U.S. Army R&T Laboratory's Aeroflightdynamics Directorate, has conducted extensive simulation investigations of rotorcraft in the nap-of-the-Earth (NOE) environment and has developed facility capabilities specifically designed for this flight regime. This paper reports on the experience gained to date in applying these facilities to the NOE flight regime and on the results of specific experimental investigations conducted to understand the influence of both motion and visual scene on the fidelity of NOE simulation. Included are comparisons of results from concurrent piloted simulation and flight research investigations. The results of a recent simulation experiment to investigate simulator sickness in this flight regime is also discussed.

INTRODUCTION

Over the past decade great strides have been made in developing piloted simulation capability. The main enabling capability has been computer technology, which has permitted the calculation of sophisticated mathematical models of aircraft motions, the dynamic modeling of complex avionic systems, and, most visibly, the computer-generation of realistic images of the outside environment. These advances have resulted in major improvements in the "reality" of piloted flight simulators.

These advances in simulation capability have also substantially advanced the use of piloted simulation in the development, acquisition, and operation of all types of aircraft under the conditions of a broad spectrum of missions. The drivers for this use are cost and safety improvements. A current example is the use of ground-based simulation to certificate transport airline pilots without requiring flight time in the actual aircraft. Piloted simulation has also played an increasing role in the research, development, and acquisition of new aircraft by (1) discovering and remedying problems before flight-test articles are fabricated, (2) aiding in understanding anomalies during flight development, and (3) assisting procuring agencies (for military systems) in mission evaluation for scenarios that cannot be actually tested.

Obviously, the successful use of piloted simulation in these roles depends on the fidelity, both subjective and objective, of the simulator relative to the real airplane. In civil airline training, the assessment of fidelity is relatively straightforward, since the actual aircraft exists and almost all mission elements can be flown. However, in the case of the development of a new aircraft, the assessment of fidelity cannot be made directly, because the aircraft does not exist and, for military aircraft, certain mission segments cannot be flown. In this case, a priori confidence in simulator fidelity is crucial and much more difficult to accomplish.

Ames Research Center has been applying ground-based piloted simulation to the research and development of aircraft for over 20 years. Over the past 10 years there has been a major emphasis on the flight dynamics, guidance, and control of rotorcraft, usually in conjunction with the co-located Aeroflightdynamics Directorate of the U.S. Army R&T Laboratory. The mission scenarios of prime interest to this class of aircraft result in flight close to either the ground or obstacles at low airspeeds and at low g levels. Recent programs have included investigation of helicopter air combat in nap-of-the-Earth (NOE) flight; helicopter autorotative landings; tilt-rotor and helicopter scout/attack missions; helicopter accident investigation; and pilot night-vision systems in NOE. In particular, the driving element of rotorcraft simulation at Ames has been the mission requirement to fly in the near-Earth environment, down to and including NOE (Fig. 1).

Experience at Ames with rotorcraft piloted simulation for the near-Earth environment has shown that pilot acceptance is particularly sensitive to visual and motion cueing. For high-fidelity flight and mission evaluation in the near-Earth environment, the pilot requires precise information about the range to obstacles or terrain and about the rate of closure on those obstacles or terrain. The prime source of this information is the visual scene. The motion system provides feedback on the maneuvering characteristics of the usually unstable, nonlinear, highly coupled

dynamics of these vehicles. Accurate modeling of these basic dynamics is important but can, in general, be accomplished satisfactorily with state-of-the-art computers.

The specific issue of the fidelity of piloted simulation for rotorcraft in the near-Earth environment has been addressed over the last 10 years at Ames Research Center in conjunction with advanced R&D programs aimed at understanding and improving the flight dynamics, control, and guidance of rotorcraft operating in this environment. Assessments of helicopter simulation technology at Ames have been previously reported (Refs. 1,2). Those assessments addressed the key issues of visual and motion fidelity. Since the latest of these summary reports (Ref. 2) was published, the motion system of the Vertical Motion Simulator at Ames Research Center has been upgraded, and several research investigations specifically addressing simulator fidelity have been undertaken.

The objective of this paper is to provide an overview of the results of specific investigations into the factors influencing the validity of simulated NOE flight. Although a comprehensive program to quantify an understanding of simulator fidelity does not exist, the results of several individual studies conducted at Ames do provide some valuable insights. First, the Ames simulation capabilities are described, followed by a discussion of three investigations specifically concerned with simulation validity: (1) the influence of simulator motion and visual cue variations on helicopter autorotative landing; (2) a comparison of simulator and flight results for a UH-60 performing selected NOE maneuvers; and (3) the influence of simulator motion and maneuvering intensity in NOE flight on symptoms of simulator sickness. The paper concludes with a discussion of approaches to mitigate the effects of poor simulator fidelity on NOE simulation results.

AMES SIMULATION FACILITY

The primary simulation facility used for rotorcraft piloted simulation studies at Ames Research Center is the Vertical Motion Simulator (VMS) complex. The VMS, shown in Fig. 2 , is a six-degree-of-freedom, large-motion simulator with the motion capabilities listed in Table 1. As shown in Fig. 3, the cab is mounted on a gimbal system that provides independent pitch and roll rotation. This gimbal system is mounted on an independent cone that provides yaw rotation. The cone-with-gimbal-assembly then moves horizontally, perpendicular to the main beam, for translation in one axis; the complete carriage with cone and gimbal moves horizontally along the main beam for translation in the second axis. The entire beam/carriage/cone/gimbal moves vertically to generate the third degree of linear motion. The cab can be oriented with the x-axis either along the beam, for greater x than y motion, or transverse to the beam, for greater y than x motion. The gimbal, cone, and carriage assembly is a recent upgrade to the VMS, made in order to add the third linear degree of freedom and to substantially increase the rotational motion performance, particularly simultaneous rotational motion. A previous five-degree-of-freedom configuration (three rotational and two linear) is described in Ref. 1; the motion system performance is described in Table 1. The VMS, particularly with its recent upgrade, provides unparalleled six-degree-of-freedom motion capability.

The VMS system (Fig. 4) consists of the VMS motion simulator, two fixed-base simulator stations, and five interchangeable cabs. This system allows the development and checkout of cabs when not installed on the motion base, the changing of cabs quickly (in less than 1 day), and the conduct of concurrent fixed-base and motion-base simulations, either independently or linked together. Four of the cabs use various arrangements of collimated video monitors for presentation of simulated visual scenes generated either by Singer-Link DIG I or Evans and Sutherland CT-5A computer graphics systems. The insides of the four cabs are shown in Fig. 5, with example cockpit furnishings and Singer DIG simulated scenes. The fifth cab uses light valves with combining and projection optics to project an E&S CT-5A computer graphics scene on the inside of a 20-ft-diam dome, as shown in the artist's rendering in Fig. 6. This dome cab has been operated in fixed-base simulation and is currently undergoing rework operation in motion.

Each of the three investigations reported herein utilized one or more of the collimated-CRT cabs driven by the Singer DIG I computer-graphics system. It should be noted that the Singer DIG I system uses 10-year-old technology and is not representative of state-of-the-art capability. The specifications for the system are as follows: (1) full daylight scene capability; (2) four channels (windows); (3) 1024-line raster format; (4) 30-Hz update (non-interlaced); (5) 8,000 polygons; and (6) 256 edge crossings per scan line. The artificial, visual-enhancing scenes were so constructed as to provide the pilot with the needed range and range-rate motion cues in a quantifiable and controlled manner.

HELICOPTER AUTOROTATIVE LANDING

A joint NASA/FAA simulation program was conducted to provide background data to assist the FAA in developing certification criteria for helicopter training simulators. The program was specifically focused on pilot control

in the autorotative landing task (Refs. 3,4). The autorotative landing task imposes a great challenge on simulator fidelity, since landing requires that the pilot's attention be directed outside the cockpit at a time when he must rely almost solely on visual, motion, and sound cues. The results of a VMS experiment undertaken to enhance the understanding of landing performance and pilot control strategy under conditions of varying simulator motion-system performance and varying visual scene content and detail are reported in detail in Ref. 4. The general findings regarding motion and visual fidelity are synopsized below.

Visual Scene Content

The visual display was provided by a four-window computer-generated image. Three visual scenes were used during the autorotation landing task evaluations. Figures 7 through 9 depict each of these scenes just prior to touchdown.

A major objective of the experiment was to evaluate the influence of visual scene elements on landing performance and pilot workload. Individual scene elements were tailored to maximize the important cues of aircraft attitude and of range and range rate from the terrain. The evaluation was based mainly on pilot commentary.

Figure 7 shows an airfield scene with a black-and-white checkerboard landing zone. Pylons along the right side and pylons beyond the landing zone provided cues in addition to those contained in the basic airfield scene. The pylons along the right side provided height and velocity cues, whereas the tall pylons in the distance provided pitch-attitude cues during the landing flare. This scene provided adequate cues for most pilots.

Figure 8 illustrates further modifications to the airfield scene. Prominent are the shift to a gray-shaded checkerboard, smaller squares in the final portion of the checkerboard, and the addition of a person and vehicles surrounding the landing zone. This modified checkerboard landing scene provided a distinct improvement over the Fig. 7 scene. The lower-contrast gray shading of the cross-hatching appeared more natural and brighter to the pilots. The half-size squares of the final quarter of the landing zone provided a useful cue for judging the final touchdown rate. The smaller cross-hatched area of the final quarter of the landing zone, which was in full view during the final seconds before touchdown, provided the pilots a finer gradation for height control. The addition of the man and trucks around the landing zone provided easily recognized scene scaling. Compared with the rather abstract appearance of the original black and white checkerboard scene of Fig. 7, the human-scaled additions of the modified scene provided a much more usable scene. The lack of recognizable texture in the computer-generated image was compensated for by artificial scene elements such as the checkerboard, pylons, man, and truck, and provided the necessary cues.

The canyon scene of Fig. 9 provided a contrast to the abstraction of the checkerboard airfield scenes. Pilots commented favorably on the strong attitude cues provided by the trees, canyon wall, and floor junction. The double row of trees provided better velocity cues than those available on the airfield scenes. Pilots commented that no scene provided good height cues in the critical period just before touchdown. Height judgment just before touchdown contributed to the large dispersions in touchdown sink rate and rotor speed.

Motion System Performance

To identify the effects of motion-system performance on pilot task performance, four levels of motion cueing were investigated. These ranged from full VMS capability, values typical of a large-travel hexapod and a small motion "nudge" base, to fixed base. The measures used to evaluate the influences of motion-system performance on landing task performance were (1) changes in pilot control strategy, and (2) aircraft ground velocity at touchdown (within safe rotor rpm constraints).

The piloting technique that is taught for autorotative landing flare is a steady increase in collective stick to full throw just as the helicopter touches down. Figure 10 illustrates collective stick time-history traces for different motion levels by pilot B. In general, this pilot exercised the proper control technique with the full VMS motion. As the motion performance degraded, his use of collective control changed. Many landing flares with degraded motion performance showed signs of ballooning, stair-stepping, and overcontrol in the collective time-histories. Note that the maximum collective control was not used at touchdown when motion cues were not available. Pilots differed in their behavior, some showing poor control techniques even with full VMS motion. However, the trend shown did occur for several pilots.

The landing performance statistics for pilot B are plotted in Fig. 11 for an 8,000-lb baseline configuration. Although the touchdown sink rate degrades with reduced motion cueing, the fixed-base result is very similar to that

of the full motion. Note, however, that the fixed-base sink-rate result is obtained at the expense of a large variation in forward velocity. Pilot B reported that degraded motion cues could distract him more than fixed base. The low forward speed at touchdown for the nudge-base was achieved at the expense of low rotor speed. Both the hexapod and nudge-base motion levels tended to distract this pilot.

For pilot A, the touchdown sink rate improved with increased motion performance (Fig. 12). However, the touchdown forward velocity tended to be higher with greater deviation for increased motion cueing. Using the full VMS motion result as the standard, reduced motion cueing for pilot A resulted in a shift of landing strategy to deemphasize touchdown sink rate.

Higher vehicle gross weight had a dramatic effect on landing performance trends for pilot F (Fig. 13). In spite of the higher landing speed technique used by pilot F, the landing performance (particularly the touchdown sink rate and forward velocity) shows distinct degradation with degraded motion cues. Pilot F was less affected by motion-system variations at the lower gross weight. The higher gross-weight configuration forced a more critical flight task requiring use of all available simulator cues.

The landing performance results for variations in motion-cue levels for all pilots may be summarized as follows: (1) degraded motion cueing generally degraded landing performance, thus causing some shifts in landing strategy and control technique; (2) motion-level variations affected pilots who sought to obtain the best performance from the helicopter (lowest forward speed and low sink rate) more than it did those pilots who used a run-on landing technique; and (3) reducing the helicopter performance margin by increasing the aircraft gross weight created a more critical flight task, which caused some pilots to become more sensitive to motion-cue variations.

UH-60 SIMULATION VALIDATION

In the early 1980's, NASA and the U.S. Army conducted a systematic evaluation and validation of a U.S. Army UH-60 helicopter simulation on the Ames VMS for nap-of-the-Earth flight tasks. The results of the initial experiments in 1982 are reported in Refs. 5 and 6. Because of deficiencies discovered during these experiments and the keen interest of both agencies in continuing improvement of the fidelity of helicopter simulation for this flight regime (and for the UH-60 in particular), efforts have continued to address these shortcomings. The improvements that have been made will be briefly described and several overall findings of a recent simulation/flight evaluation of these improvements will be discussed. A detailed report is being prepared for publication elsewhere.

Early Experiments

The experiments in the early 1980s used the Ames VMS with a four-window CGI scene provided by the Singer DIG I image-generation system. The details of the experimental setup are described in Refs. 5 and 6.

Figure 14 shows a comparison of the mean and extreme Cooper-Harper handling-qualities ratings (HQR) (Ref. 7) between the VMS simulation and flight tests for three of the NOE maneuvers: bob-up (BU), sidestep (SS), and dash/quick-stop (D/QS). The mean ratings for all of the tasks in flight were Level 1, whereas in the simulator they were Level 2. In addition, there was no overlap in any of the ratings for any of the tasks. The pilot commentary identified the following deficiencies of the simulation: (1) inability to judge range and height as accurately as in flight; (2) larger thresholds of visual perception of motion; (3) insufficient damping in all axes; (4) vertical and roll pilot-induced-oscillations; (5) exaggerated control inputs; and (6) deceptive motion cues.

Not surprisingly, these problems were attributed to the following characteristics of the visual, motion, and modeling systems: (1) insufficient CGI scene field-of-view, content, and texture; (2) basic transport delay of 120 msec as a result of the inherent architecture of the CGI and host computers; and (3) phase distortion of the motion system.

Simulation Improvements

Over the past 5 years, extensive efforts have been undertaken to improve the fidelity of the VMS system and the validity of the UH-60 simulation. The improvements to the VMS included (1) doubling of the angular rate and acceleration performance, (2) addition of motion in the third translational axis, (3) incorporation of compensation for the CGI to reduce the overall transport delay to approximately 20 msec, and (4) incorporation of an Applied Dynamics Inc. AD100 host computer to reduce the model cycle time to 6.7 msec (with a 20-msec input/output cycle). The fields of view of the available cabs are unchanged (Fig. 15). Although the capabilities of the CGI system also remain

unchanged, the scenes for the specific NOE tasks have been tailored to increase content and detail, as will be discussed below. There has been a significant effort to improve the modeling of the UH-60 (Ref. 8).

Recent Experiments

The two recent UH-60 Black Hawk simulations were the first simulation validation experiments on the newly refurbished VMS. The first simulation was done concurrently with a flight test of the UH-60 aircraft at the NASA facility at Crows Landing Naval Auxiliary Landing Field (Calif.). This allowed a back-to-back comparison of flight and simulation, which is desirable when an actual vehicle is being simulated and fidelity assessments are being made.

The recent simulation experiments settled on three primary tasks to be used in assessing simulation fidelity: the bob-up (BU), the sidestep (SS), and the dash/quick-stop (D/QS). The experiments were set up so that the same task could be performed on the simulator as in flight. The selection of the bob-up/bob-down and the sidestep maneuvers allowed the use of a specially designed hover board for the flight tasks (Ref. 9). These boards were duplicated on the DIG-1 image generator used on the VMS simulation to get the one-to-one task performance desired. The boards were placed on a facsimile of the Crows Landing Airfield reproduced on the DIG-1. Figures 16 and 17 show the boards at Crows Landing and in the simulator, respectively. The dash/quick-stop maneuver was performed on the simulator in a setting representative of the task done at Crows Landing. Knowing the limitations of the simulator field of view for the dash/quick-stop, the task was modified to constrain pitch-attitude excursions within those limitations. The HQR results of the subjective evaluations given by the test pilots for the recent flight and two simulation experiments are shown in Fig. 18, along with the corresponding results from the earlier experiments (Fig. 14). The mean ratings for the recent experiments are denoted by the filled symbols and the extremes in ratings are denoted by the solid vertical bars. The earlier results are shown with open symbols and dashed vertical bars.

There has been a significant improvement in the validity of the simulation of the UH-60 for these NOE tasks. The ratings from the current simulation are only 0.5 to 1.5 ratings worse than the flight ratings, whereas in the previous experiments the spread was from 1.5 to 2.5 ratings worse. Pilot commentary provides initial insight into the characteristics of the simulation that still contribute to the differences.

Bob-up and sidestep tasks— Because of the restricted field of view of the CGI scene, the pilots were unable to see the stop point when they initiated the bob-up maneuver. Consequently, they could not lead the task as well as they could in flight, which resulted in higher workload required to mitigate overshoot and bobble when trying to arrest the vehicle at the stop point. In addition, some of the pilots commented that they perceived lighter heave damping in the simulator than in the aircraft. Although considerable progress has been made in improving the mathematical model (Ref. 8), the loader system dynamics, and the visual delay (Ref. 10), other residual visual scene problems may be still contributing to this lack of validity.

Although the hover boards did help in achieving closer agreement with flight by providing improved range and range rate cues, problems still exist with the image. The pilots commented that the reduced resolution and lack of depth perception (Fig. 19) in the simulator detracted from doing precision maneuvering in the simulator. Overall, the pilots said that they tended to perform the task with the same control strategy in the simulator as they did in flight.

Dash/quick-stop task— The mean HQR ratings for the dash/quick-stop task showed the best comparison between simulation and flight, but the pilots noted a difference in the piloting strategy used to accomplish this maneuver. In flight, they relied on the external scene to judge the vehicle's altitude and ground speed, with a cockpit instrument check to verify airspeed and height above the ground. In the simulation, they relied more on aircraft instruments to judge the vehicles attitude, height above the ground, and air speed with a check of the outside CGI scene to verify altitude. They gave two reasons for this change in strategy: first, they could not judge ground speed and altitude from the CGI scene owing to the lack of fine texture; and second, the restricted field of view limited altitude information during pitch changes.

SIMULATOR-INDUCED SICKNESS

An undesirable by-product of ground-based piloted simulation in which the realistic visual scenes available today are used is the phenomenon of simulator-induced sickness. This is a growing international problem (Ref. 11) with the incidence rates appearing to increase as more flight simulators and more complex visual systems are put into use (Ref. 12). In general, increased incidence of simulator sickness is associated with more intensive maneuvering, such as air-to-air combat and NOE flight. It has been hypothesized that simulator-induced sickness is a result of a conflict between the pilots' visual and vestibular systems, that is, actual or cognitively expected motion and visual

cues. The consequences of this conflict on the results of simulation, and on the well-being of subject pilots, are key issues that need to be addressed.

An initial joint NASA/Army simulation experiment has been conducted to investigate the causes, symptoms, and measures of simulator-induced sickness and to identify solutions to the problem (reported in detail in Ref. 13.) The large-motion capabilities of the Ames Vertical Motion Simulator provided a unique opportunity to study the effects of visual-motion dis-synchrony on simulator-induced sickness.

The objectives of the experiment were to (1) assess the incidence of simulator sickness under four simulator motion conditions, (2) validate physiological and behavioral measures of pilot performance and well-being, and (3) develop a quantitative measure of conflict between visual and inertial cues for motion sensing. Only the findings regarding the influence of variations in inertial motion cueing on the incidence of simulator sickness are discussed in this paper (refer to Ref. 13 for other details.)

Four simulator motion conditions were tested: (1) fixed-base, (2) VMS nominal, (3) increased lead, and (4) reduced motion bandwidth. The conditions were selected to represent the full range of motion-visual synchronization from the least, in the fixed-base condition, to the highest fidelity that VMS can provide. The specific characteristics of the visual and motion systems are defined in detail in Ref. 13. The intermediate conditions were selected to be representative of motion systems found in current and proposed military flight trainers. The increased-lead condition produced, relative to VMS nominal, exaggerated initial motion inputs in the rotational axes (roll, pitch, and yaw) followed by a more rapid motion washout. The reduced-motion-bandwidth condition was characterized by a decreased motion bandwidth which produced an increased temporal lag in the rotational axes.

Forty-eight Army helicopter pilots participated in the study, each randomly assigned to only one of the four simulator motion conditions. The flight task required each pilot to fly a simplified model of a single-seat UH-60 Blackhawk helicopter while pursuing a target aircraft at a specified interval. The motion of the target aircraft was recorded from prior flights in the VMS. Each pilot flew four 10-min segments distinguished by successively increasing demands on the amount of flight maneuvering required. The first segment involved very gentle maneuvering; the fourth segment was quite aggressive with bank angles frequently exceeding 90°. All four segments were flown at altitudes from 20 to 100 ft above the terrain. Pilots were provided with visual status information which informed them when they were either too close or too far from the target aircraft.

Every 5 min during the simulated flight, pilots were asked by the experimenter to provide a numerical rating, on a scale from 1 to 7, of their level of well being. A rating of 1 signified "I feel fine and symptom-free" and a rating of 7 signified "I am unable to continue and wish to terminate my flight." Pilots were encouraged to terminate their flight at any time if they began to feel uncomfortable or nauseated.

Pilot Discomfort Ratings

Figure 20 presents mean discomfort ratings for pilots grouped by motion condition. The data are presented for all four 10-min sessions, each of which required progressively more maneuvering by the pilot. Because of excessive discomfort, 23.0% of the pilots were unable to complete all four sessions in the increased-lead condition, 18.1% in the fixed-base condition, and 8.3% in both the VMS nominal and reduced-motion-bandwidth conditions.

As indicated in Fig. 20, pilots reported higher levels of discomfort in those conditions that required greater maneuvering. This is particularly evident in the increased-lead condition, in which pilot mean ratings of discomfort increased from 1.5 in low maneuvering to 3.4 in high maneuvering. A less rapid increase in reported discomfort was observed for the other three motion conditions.

The results also suggest an interaction between motion condition and maneuvering intensity. The reduced-motion-bandwidth condition produced greater mean discomfort in the two lowest maneuvering conditions, whereas the increased-lead condition produced more discomfort in the two highest maneuvering conditions. Overall, the VMS nominal condition appeared to be the most benign.

Simulator Side Effects

In general, the measures of simulator side effects corroborate the measures of pilot discomfort discussed above, in that large increases in reported symptoms were observed.

Immediate postflight data revealed increases of 20% or more (over preflight data) in reports of general discomfort, eye strain, salivation increase, sweating, nausea, difficulty concentrating, dizziness, and stomach awareness. One pilot who participated in the fixed-base condition vomited before exiting the simulator.

Data taken 30 min after flight revealed increases of 10% or more (over pretest reports) for general discomfort, eye strain, difficulty focusing, nausea, dizziness, and increased appetite. Across all motion conditions, there were substantial reports of symptoms up to 30 min after exiting the simulator. Prolonged symptoms of general discomfort and nausea appeared with greater frequency in the more attenuated motion conditions (fixed base, increased lead, and reduced motion bandwidth) than in the VMS nominal condition. This appears to follow the prediction of the sensory conflict theory, in that greater discrepancies between visual and inertial cues for motion exist in those three conditions. Long-term aftereffects, from 3 to 48 hr after completion of the simulation session, were found to be negligible.

The results presented above are from the first experiment on the VMS, which was undertaken to gain an understanding of the factors involved and their influence on simulator-induced sickness. The results indicate (1) that phase distortion of motion cues, particularly at high acceleration levels, leads to increased occurrence of symptoms of simulator-induced sickness, and (2) that long-term aftereffects were negligible. Nonetheless, further investigation is required to quantify the degree to which the differences are statistically meaningful and the measures are statistically valid.

CONCLUDING REMARKS

Based on these experiences at Ames in examining the influences of motion and visual fidelity of ground-based piloted simulation, and on the efforts undertaken to properly control these influences, the following concluding comments are offered:

1. It is crucial to satisfactory simulation validity that the piloting tasks be tailored to fit within the motion and visual scene capabilities of the simulator. Conversely, the simulation must designed to provide the cues necessary if the pilot is to perform the task as one would expect him to perform it in flight.

2. High-fidelity motion cues are required to improve task performance for near-Earth or nap-of-the-Earth flight tasks. As the difficulty of the task increases, the effects of motion cueing become more pronounced. Small motion cues, poorly tailored to the task, may degrade performance more than no motion cues (fixed-base).

3. When using a four-window CGI system, phase distortion in motion cueing, particularly at high acceleration levels, leads to increases in simulator-induced sickness. An increase in maneuver level leads to increases in simulation-induced sickness. Long-term aftereffects were found to be negligible.

4. Although the use of carefully controlled tasks can mitigate the effects of limited scene field-of-view, the current fields of view available on the Ames VMS need to be increased further to enable broader nap-of-the-Earth tasks to be flown with acceptable validity.

5. The addition of easily recognizable scaling objects contributes greatly to the pilot's ability to estimate range and range rate.

6. Accurate ground-speed and height sensing, which are crucial to nap-of-the-Earth flight, require fine scene texture.

7. The overall response lags in the simulator visual scene and motion system, and the poor synchrony between these lags, significantly affect pilot acceptability and performance, and the onset of symptoms of simulator sickness. The application of high-speed digital computers, CGI delay compensator techniques, and motion washout adjustment can mitigate these effects.

REFERENCES

1. Bray, Richard S.: Visual and Motion Cueing in Helicopter Simulation. AGARD Conference Proceedings No. 408, AGARD FMP Symposium on Flight Simulation, Cambridge, U.K., 1985.

2. Bray, Richard S.: Helicopter Simulation Technology: An Ames Research Center Perspective. NASA CP-2219, 1982, pp. 199-208.

3. Decker, W. A.; Adam, C. F.; and Gerdes, R. M.: Model development and Use of Simulators for Investigation of Autorotation. FAA Conference on Helicopter Simulation Proceedings, Atlanta, Ga., Apr. 1984.

4. Decker, W. A.; Adam, Charles F.; and Gerdes, Ronald M.: Pilot Use of Simulator Cues for Autorotation Landings. 42nd Annual National Forum and Technology Display of the American Helicopter Society, Washington, D.C., June 2-4, 1986.

5. Key, David L.; Hansen, Raymond S.; Cleveland, William B.; and Abbott, William Y.: Helicopter Simulation Validation Using Flight Data. AGARD Flight Mechanics Panel Symposium, Ground/Flight Test Techniques and Correlation, Cesme, Turkey, Oct. 1982; also in AGARD Confere ce Proceedings No. 339.

6. Clement, Warren F.; Cleveland, William B.; and Key, David L.: Assessment of Simulation Fidelity Using Measurements of Piloting Technique in Flight. AGARD Conference Proceedings No. 359, Monterey, Calif., 1984.

7. Cooper, George E.; and Robert P. Harper, Jr.: The Use of Pilot Rating in the Evaluation of Aircraft Handling Qualities. NASA TN D-5153, 1969.

8. Ballin, Mark G.; and Dalang-Secrétan, Marie-Alix: Validation of the Dynamic Response of a Blade-Element UH-60 Simulation Model in Hovering Flight. 46th Annual National Forum and Technology Display of the American Helicopter Society, Washington, D.C., May 21-23, 1990.

9. Schroeder, J. A.; Moralez, E.; and Merrick, V. K.: Simulation Evaluation of the Control System Command Monitoring Concept for the NASA V/S TOL Research Aircraft (VSRA). AIAA Paper 87-2255-CP, Monterey, Calif., 1987.

10. McFarland, Richard E.: Transport Delay Compensation for Computer-Generated Imagery Systems. Royal Aeronautical Society Conference, Flight Simulation: Recent Developments in Technology and Use, London, April 12 and 13, 1988.

11. Motion Cues in Flight Simulation and Simulator-Induced Sickness. AGARD Report No. CP-433, 1988.

12. McCauley, M. E.: Research Issues in Simulator Sickness. Proceedings of a Workshop. National Academy Press, Washington, D.C., 1984.

13. McCauley, Michael E.; Cook, Anthony M.; and Hettinger, Lawrence J.: Simulation Sickness. Proceedings of the NASA Steering Committee and a Summary of Recent Research, Royal Aeronautical Society Conference, Progress in Helicopter and V/STOL Aircraft Simulation, London, May 1 and 2, 1990.

ACKNOWLEDGEMENTS

This paper includes the results of the extensive investigative efforts of many people at the NASA Ames Research Center and the U.S. Army's Aeroflightdynamics Laboratory in trying to understand and improve the fidelity and validity of ground-based piloted simulation. The author particularly wants to thank Adolph Atencio of the U.S. Army's Aeroflightdynamics Laboratory for the information on the recent UH-60 investigations.

Table 1.– VMS Motion Specification

Axis	Nominal operational limits		
	Displacement	Velocity	Acceleration
	Performance after upgrade		
Vertical	17 m	5 m/sec	7 m/sec^2
Lateral	12 m	2.5 m/sec	4.5 m/sec^2
Longitudinal	2.4 m	1.2 m/sec	3 m/sec^2
Roll	18°	40°/sec	115°/sec^2
Pitch	18°	40°/sec	115°/sec^2
Yaw	24°	46°/sec	115°/sec^2
	Performance before upgrade		
Vertical	17 m	5 m/sec	7 m/sec^2
Lateral	12 m	2.5 m/sec	4.5 m/sec^2
Longitudinal	0	0	0
Roll	20°	20°/sec	60°/sec^2
Pitch	20°	20°/sec	60°/sec^2
Yaw	20°	20°/sec	60°/sec^2

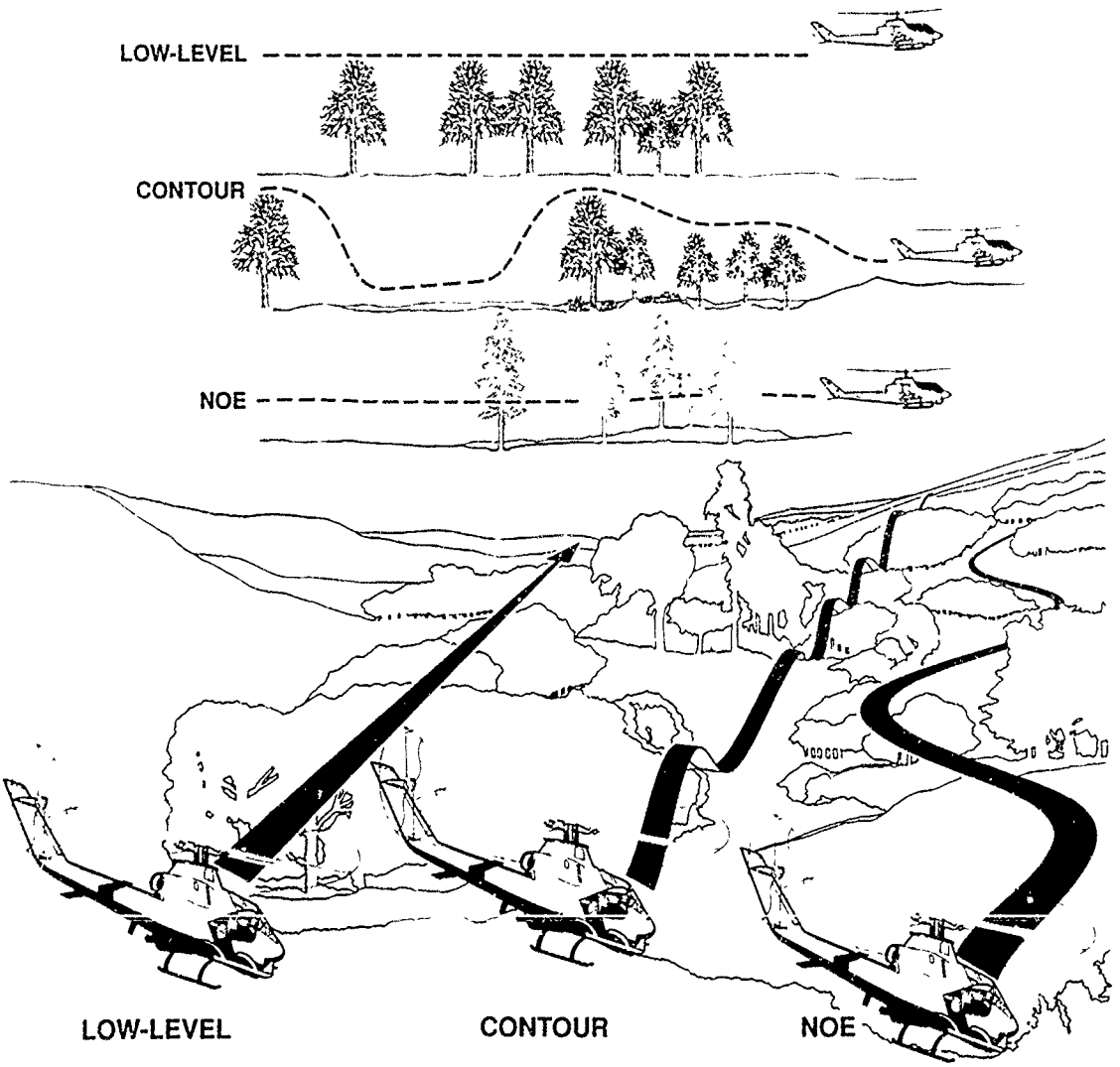


Figure 1.– Modes of helicopter flight near the ground.

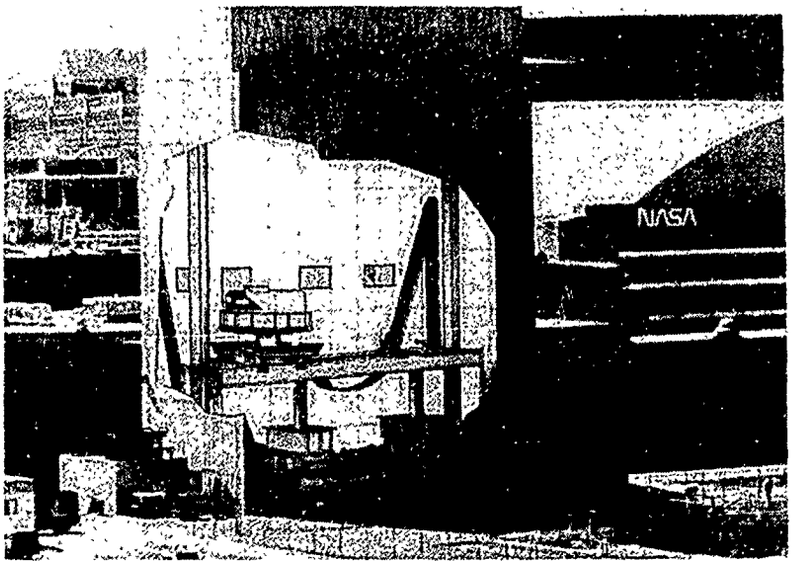


Figure 2.– Vertical Motion Simulator.

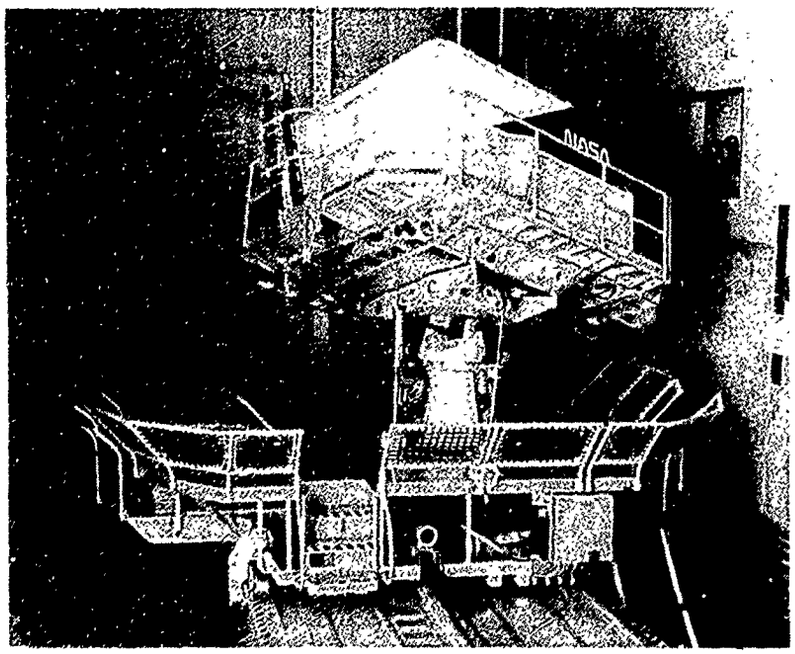


Figure 3.– Interchangeable cab mounted on VMS motion base.

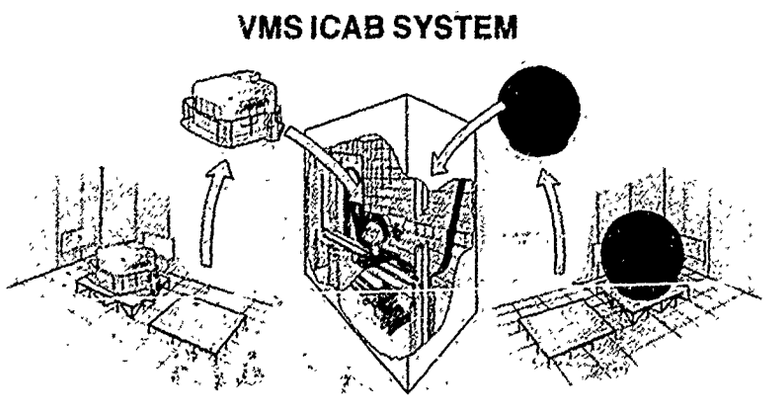
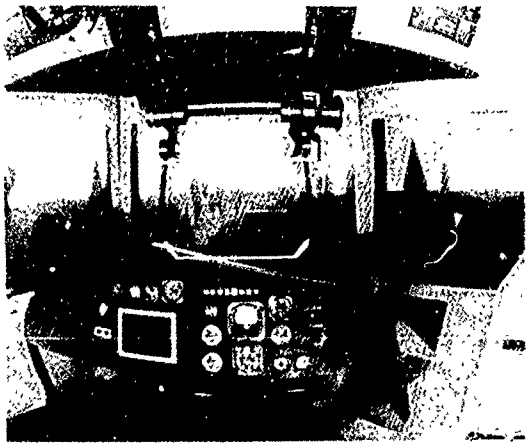
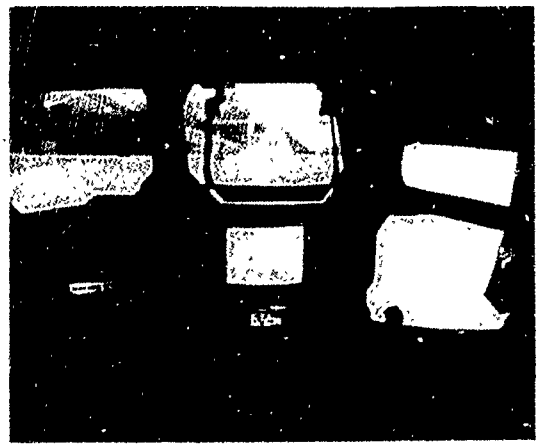


Figure 4.– Interchangeable cab design for VMS.

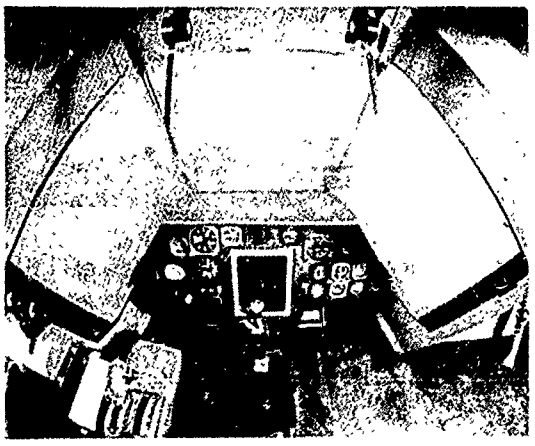


(a) Rotorcraft



(b) Rotorcraft

(c) Transport



(d) Fighter.

Figure 5.– Sample cockpit and visual scene configurations for VMS cabs using CRT monitors.

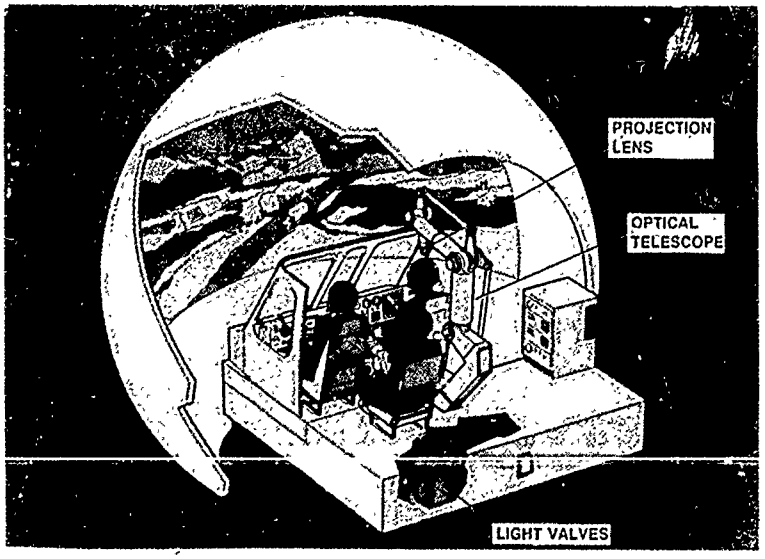


Figure 6.– Dome projection cab for future VMS use.

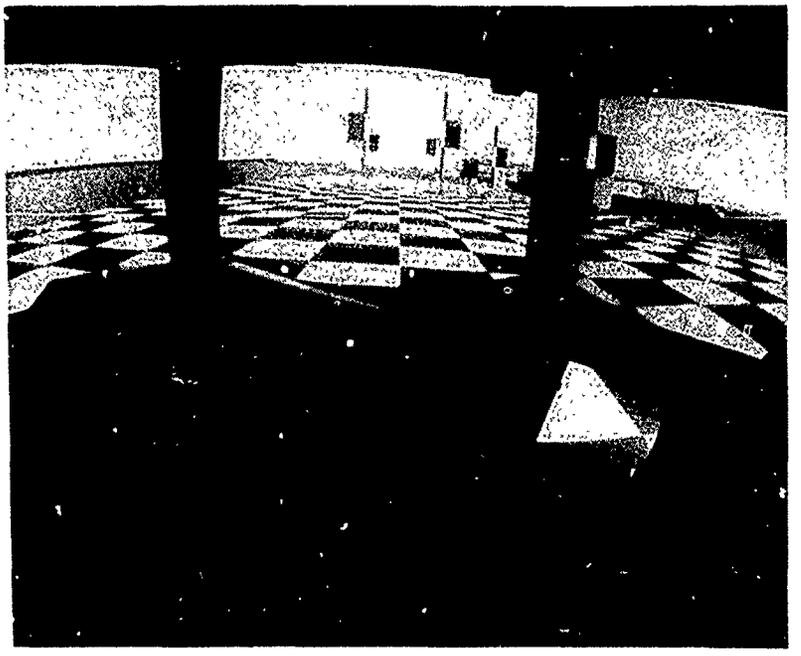


Figure 7.– Computer-generated image view of airfield, with black-and-white checkerboard landing zone, for VMS autorotation experiments.

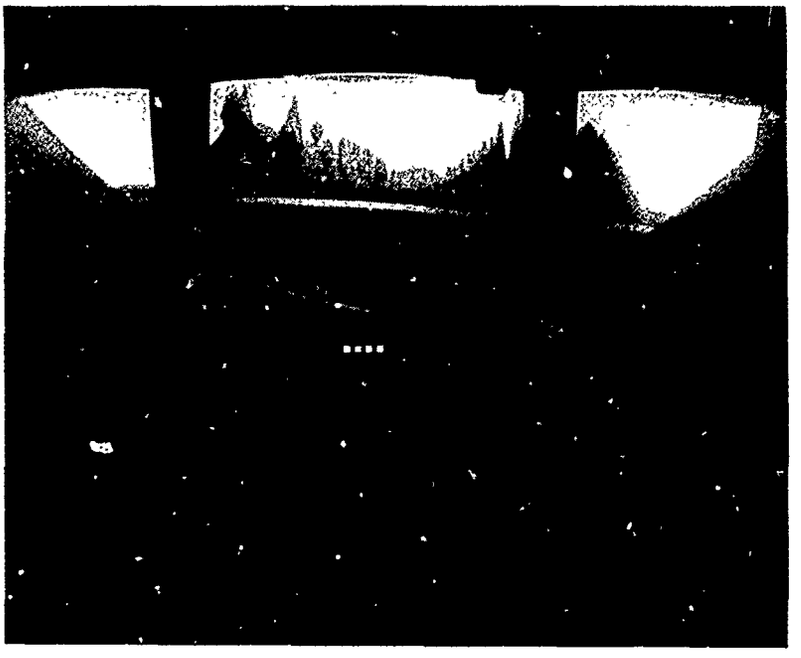


Figure 8.– Computer-generated image view of airfield, with gray-shaded checkerboard landing zone, for VMS autorotation experiments.

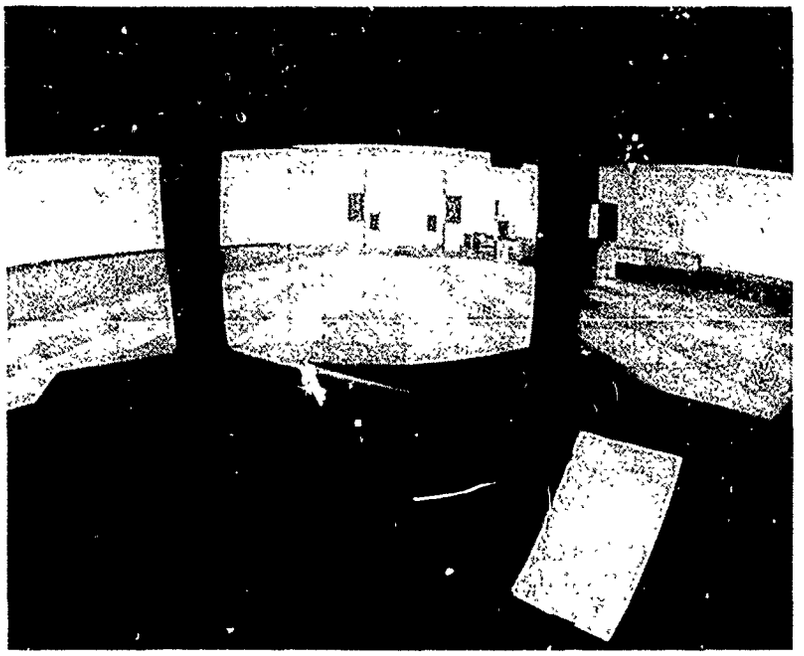


Figure 9.– Computer-generated image view of tree-lined canyon for VMS autorotation experiments.

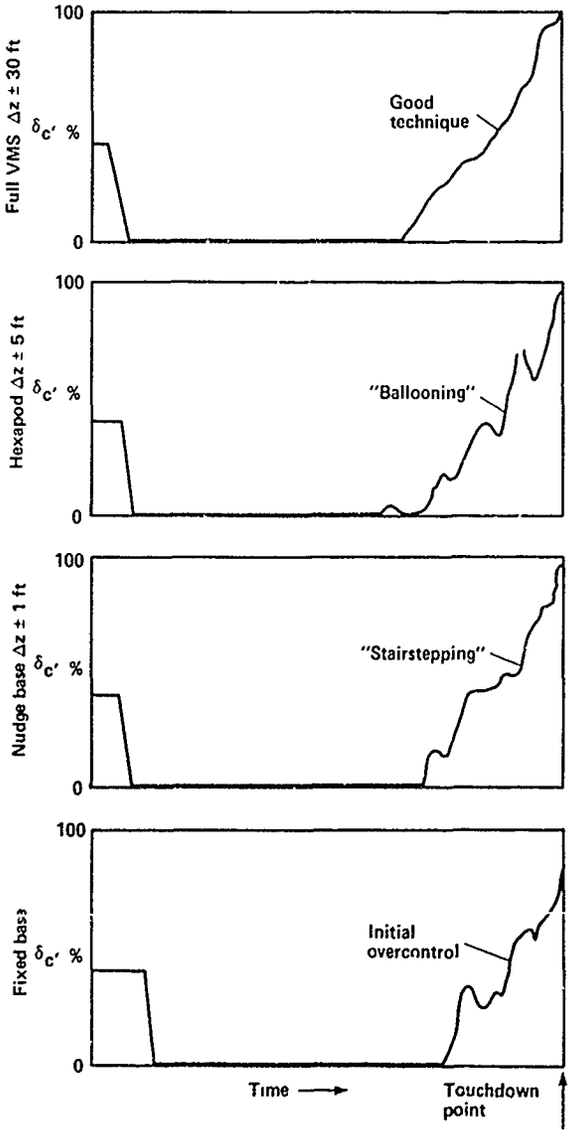


Figure 10.- Time-histories of collective control position during autorotation landings with variations of simulator motion cue levels.

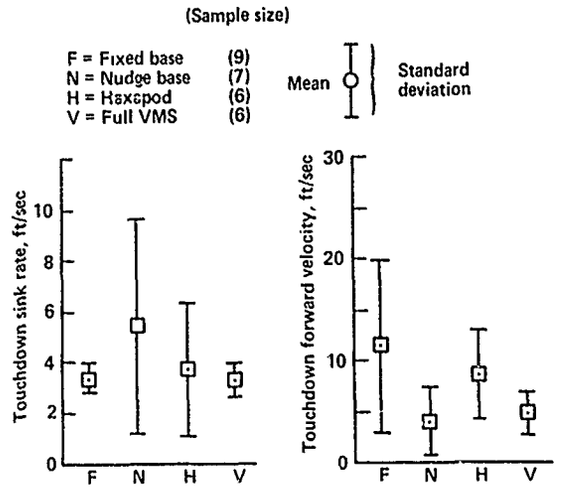


Figure 11.– Autorotation landing performance statistics for VMS experiment: pilot B, 8,000-lb helicopter.

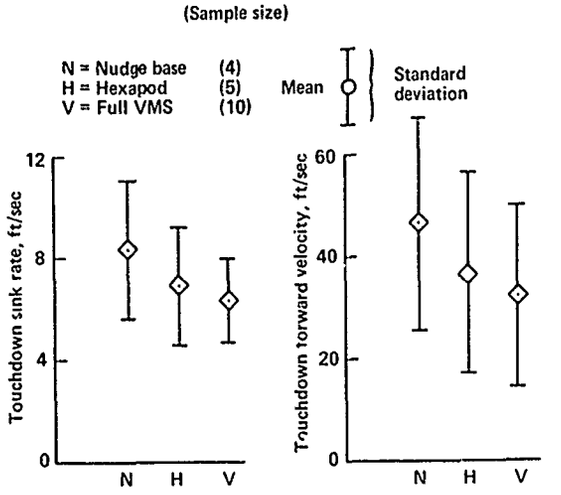


Figure 13.– Autorotation landing performance statistics for VMS experiment: pilot F, 10,000-lb helicopter.

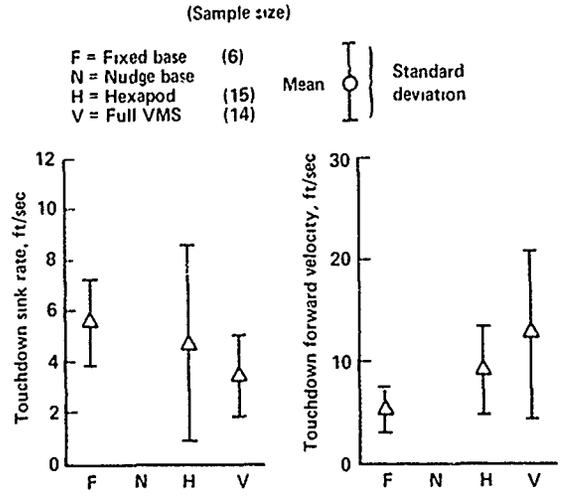


Figure 12.– Autorotation landing performance statistics for VMS experiment: pilot A, 8,000-lb helicopter.

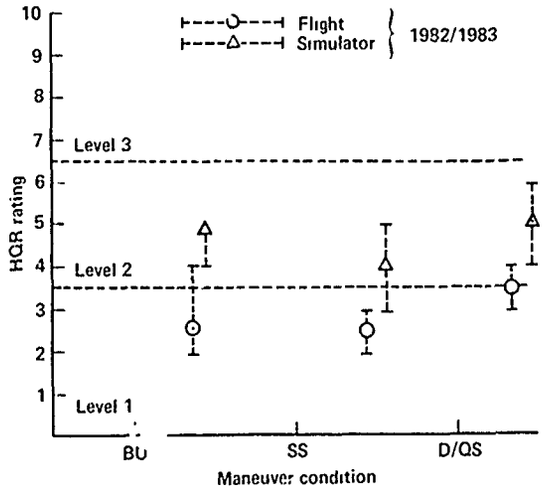
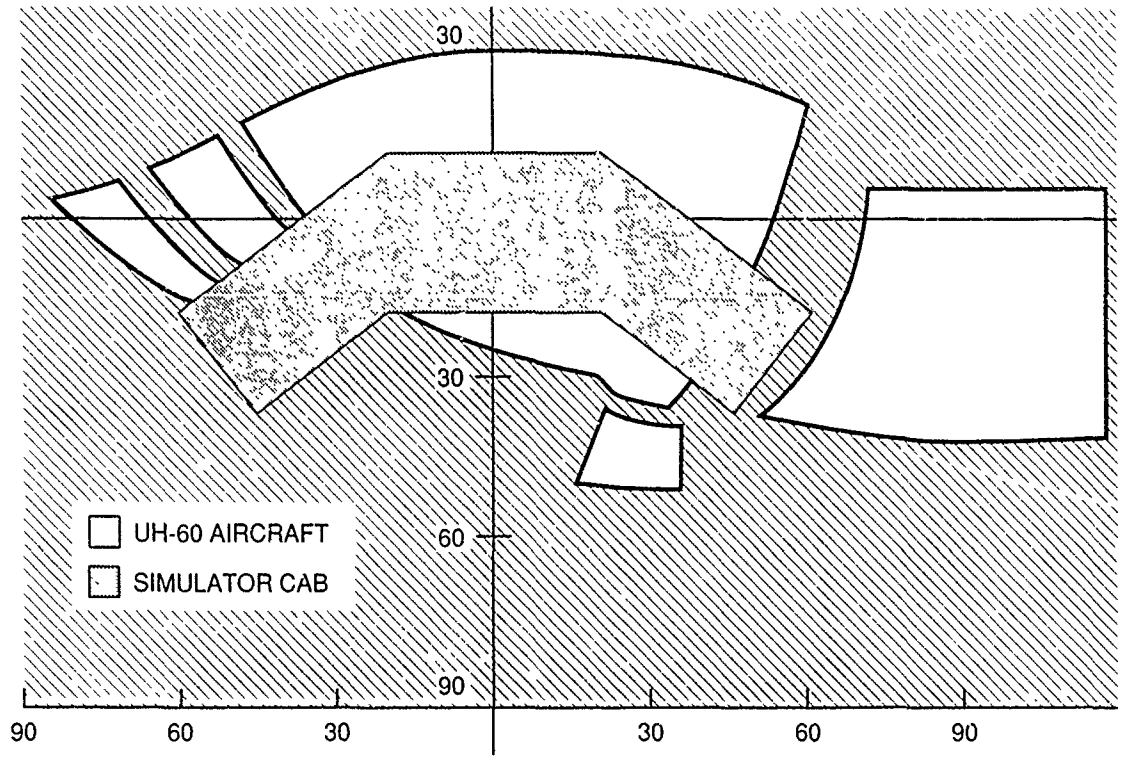
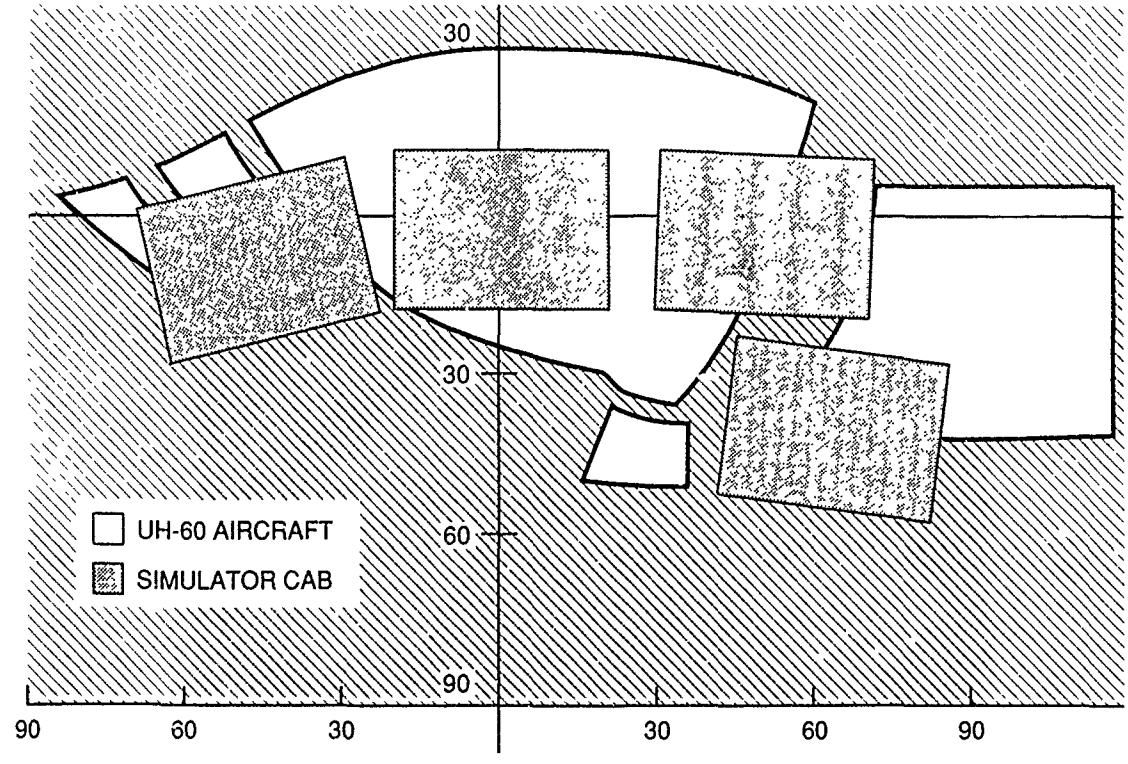


Figure 14.- Comparison of handling-qualities ratings from flight and simulation experiments conducted in 1982/1983, for UH-60 helicopter in three maneuver tasks.



(a) F-Cab (Three window)



(b) N-Cab (Four window)

Figure 15.– Comparison of pilot's field of view between the UH-60 helicopter and two VMS cabs with CRT windows.

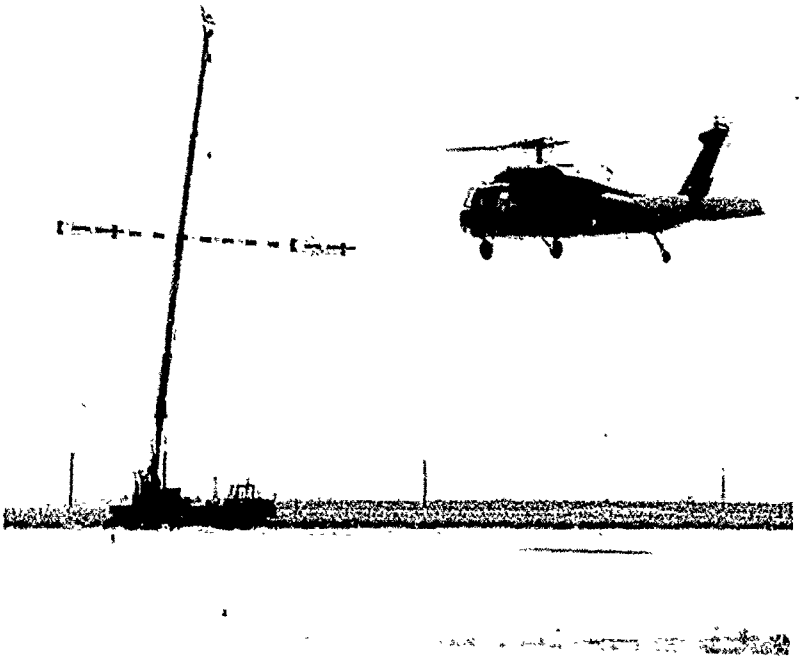


Figure 16.– UH-60 conducting sidestep maneuver against target at Crow's Landing test site.

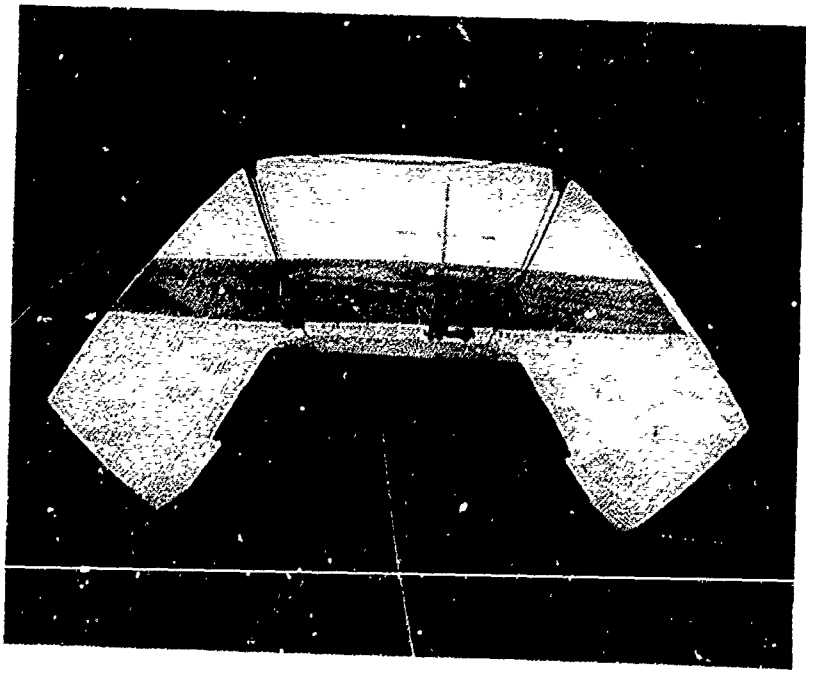


Figure 17.– Pilot view of VMS computer-generated image of sidestep target at Crow's Landing test site.

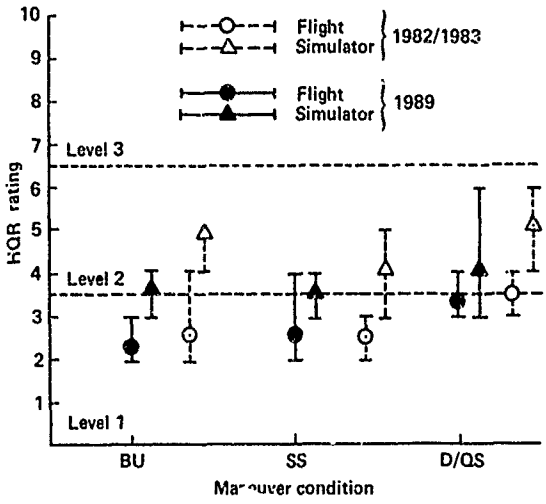


Figure 18.– Comparison of handling-qualities ratings, between flight and simulator experiments conducted in 1982 and 1983, and in 1989, for UH-60 helicopter in three maneuver tasks.

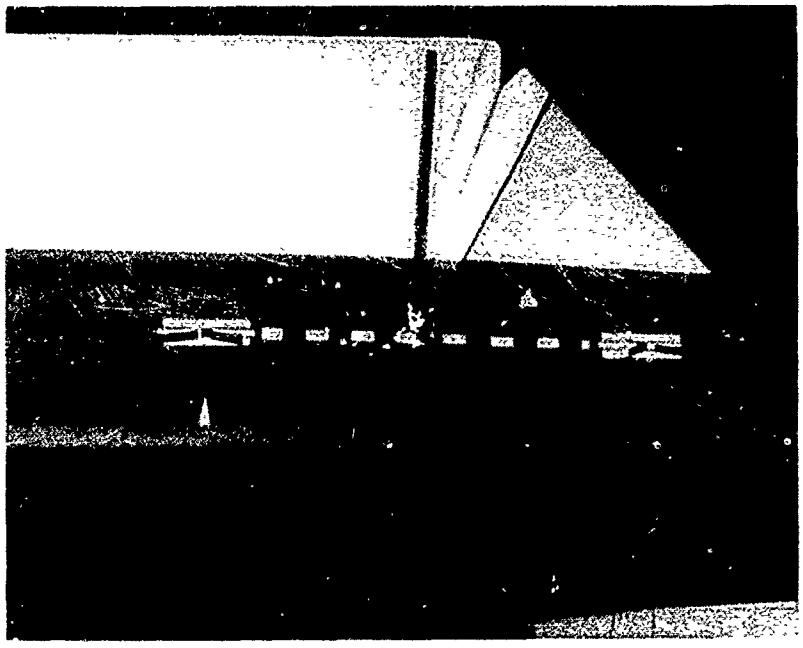


Figure 19.– Close-up view of computer-generated image of target for sidestep maneuver.

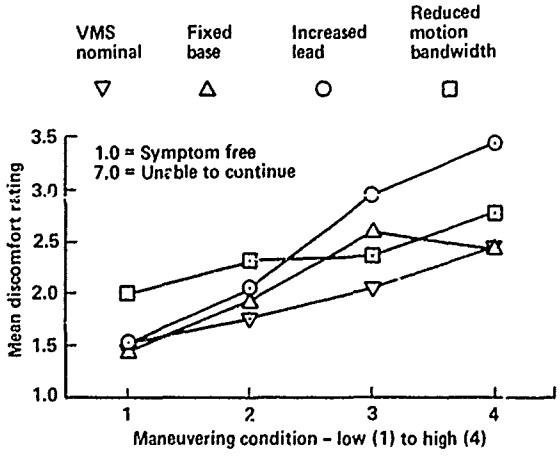


Figure 20.– Effects of simulator motion characteristics and aircraft maneuver task on pilot mean discomfort rating.

RESULTS OF MAN IN THE LOOP SIMULATOR EXPERIMENTS USING AIR-TO-AIR MISSILE MODELS

by

Neill Seavers
MM2 Division
Royal Aerospace Establishment
Farnborough
Hampshire GU14 6TD
United Kingdom

1 INTRODUCTION

Short Range Air-to-Air Missiles (SRAAMS) are an essential part of any modern fighter aircraft's weapon system. The high success rate of Infra-Red (IR) SRAAMS eg AIM 9L 'Sidewinder' class missiles in aerial conflicts during the 1980s eg the Lebanon air war and the Falklands, have made a considerable impact on modern aerial combat tactics and to some extent on aircraft and weapon system designs.

This unclassified paper details some of the experiences and results gained from two man in the loop experimental trials using the RAE Air Combat Simulator. various types of air-to-air missiles and aircraft weapon systems have been employed against a variety of 'threat' aircraft.

2 SIMULATOR DESCRIPTION

The RAE single dome Air Combat Simulator (ACS) was used in all the work discussed below. It consists of a 30 ft diameter dome with a fixed representative fast jet cockpit situated near the dome centre. A Sky/Ground image together with target and missile images are projected onto the interior surface to give the pilot the illusion of participating in air combat.

3 EXPERIMENTAL TRIAL CONDITIONS AND SCENARIOS

The Simulator trials have employed a variety of front line RAF fighter aircraft and representative weapon systems for in-service and possible future systems. The pilot fire control displays consisted of an audio acquisition/lock-on tone for the missile IR seeker head together with an indication of the missile kinematic 'in-range' calculations for various target manoeuvre assumptions. In the first trial, which employed a 'Hawk' class fighter ie one with no radar, the pilot had to employ hs own 'rule-of-thumb' type kinematic missile engagement criteria based on visual inspection of the target aircraft. Two missile employment options were studied: (1) Boresight only IR acquisition, and (2) Helmet Mounted Sight missile head slaving IR acquisition. The HMS system employed various hardware devices in the simulator to determine where the pilot was pointing his head ie the aiming reticule of the helmet sight.

In the case of the second trial, a 'Tornado' class fighter equipped with an air intercept radar and weapon system software models was used in the air combat simulator. The weapon system generated missile 'in-range' fire-control cueing data. This was displayed to the pilot in his Head Up Display (HUD).

An alternate trial condition of the Tornado experiment was to employ a Helmet Mounted Display system integrated to the radar and weapon systems. By employing an LED based array, fire control information was displayed to the pilot when he was looking far outside the HUD Field of View. This enabled the pilot to use the HMD to both cue missile seeker heads to an off-boresight target and then to display radar derived fire control data in the helmet display, when appropriate.

4 TRIAL RESULTS

The simple day fighter trial showed that a helmet aiming reticule system was of some benefit in employing IR type air-to-air missiles. It did enable some measurable advantage to be gained by using the off-boresight potential of missile designs. Not surprisingly this performance gain was linked to the missile gimbal/off-boresight capabilities. The lack of dynamic fire control data resulted in a large number of the missile firings not succeeding due to non optimal firing geometries.

In the second trial the benefits of both (1) an integrated radar/weapon system, and (2) an integrated Helmet Display/radar and weapon system, were immediately apparent. The fire control data, when displayed only in the HUD and tied to the radar system gave a superior missile kill ratio ie number of successful firings/all missile firings compared to a similar non radar equipped fight

In the case where both the radar, weapons system and helmet mounted display systems were integrated the overall combat performance was further improved. The off-boresight target designation and radar/missile lock-up capability together with the dynamic missile fire control data available in the helmet display gave the most optimised air combat performance by all objective criteria.

5 CONCLUSIONS

Weapon system integration of radar, missile systems and helmet mounted display systems is one of the most effective ways to improve the air combat performance of single seat fighter aircraft.

The man machine interface (MMI) is clearly a vital element of this integration and unquestionably a potential single point of failure for the whole system. The use of real-time, man-in-the-loop air combat simulators is an effective approach to explore future MMI concepts and novel fire control strategies integrated with sensor systems and missile concept.

The Development of Avionics-Intensive, Multi-Sensor Cockpits: Simulation Doesn't Always Equal Success

Lt Col C. G. Killberg, USAF
Deputy Commander, 6515th Test Squadron
Edwards Air Force Base, California 93523 U.S.A.

Simulation provides a critical foundation for the design and development of advanced aircraft. Throughout these phases, engineers and pilots use various forms of simulation to predict aircraft performance, determine handling qualities, and evaluate cockpit displays and control mechanizations. Aircrews can "practice" developmental test and operational evaluation missions long before the first aircraft flys, permitting everything from the optimization of flight control laws to the mechanization of the aircrew-avionics interfaces.

Present day wind tunnels and aerodynamic computer models are generally accurate to within a few percentages of actual performance. Much of the time spent in initial performance flight testing is now devoted to confirming the aerodynamic predictions for a new aircraft. For example, during initial testing of the F-15E we have generally found less than 2 percent difference between predicted and actual performance throughout the flight envelope. Testing has been completed on a number of different configurations, including combinations of external fuel tanks, conformal fuel tanks, LANTIRN Navigation and Targeting Pods, and various air-to-ground and air-to-air stores.

Simulators appear to provide a very accurate model of the environment for large transport category aircraft, which operate in comparatively benign conditions. Once flight testing begins, relatively few changes to cockpit mechanization are required.

Although the past decade has brought significant changes to the design of commercial airline cockpits, one may recall that for many years major commercial aircraft manufacturers tried in vain to modernize airline cockpits with significant improvements in instrument and display design. Older airline pilots fought against the improvements because the new designs were unfamiliar. They preferred the cockpit design to which they had grown accustomed. Their philosophy was "if it works, don't fix it." The number of cockpit tasks had not increased significantly nor become more complicated, and there was sufficient space to add new control panels, switches, or displays if needed.

The use of simulation in the design and development of the cockpit man-machine interface for advanced, multi-sensor aircraft is not always as successful. The traditional cockpit design philosophy of one panel for each subsystem, and one function per switch, is no longer feasible. The tremendous increase in the number of sensors and avionics subsystems which must be integrated into the cramped cockpit of a modern fighter make fundamental change in design an absolute necessity.

Certainly simulation has been useful in the development of fighter/attack aircraft with such highly integrated cockpits. Cockpit design mockups, procedures trainers, part-task simulations, and complete state-of-the-art flight simulators are employed to provide an accurate model of the operational environment in which aircraft will operate. This frequently includes night, low level, under or in the weather, using advanced sensors such as Synthetic Aperture Radar (SAR), Forward Looking Infra-red (FLIR), or low light level television, as well as a variety of laser designators and ranging devices.

Despite a extensive investment of time and money in simulation during the development phase, it is virtually certain that the need for a great number of changes to the aircrew-avionics interface will be identified once flight testing begins. It is all too common for experimental test and operational evaluation pilots and system

operators to discover many sub-optimum or workload-increasing mechanizations through all phases of flight testing. They spend a significant portion of their time in debriefings and flight reports explaining why a particular mechanization is difficult or impossible for the aircrew to employ.

The test aircrews write hundreds of service reports recounting such problems and proposing changes to designs which are themselves the result of extensive, and expensive, simulation efforts.

The proposed changes may be minor, involving inconvenient mechanizations which slightly increase aircrew workload. Or significant changes may be required to correct mechanizations that are so egregious they totally inhibit the operation of a sensor or avionics subsystem.

The purpose of this paper is to examine some of the reasons for the failure of simulation to highlight these problems before a highly-integrated fighter flys for the first time.

Two Examples

1 - THE F-16 MSIP COCKPIT

In 1983-1984 the USAF and General Dynamics began flight testing a new cockpit interface, part of the F-16C/D Multi-Staged Improvement Program (MSIP). A major part of the MSIP upgrade is an improved Communications/ Navigation interface. This consists of a data entry keypad, located immediately beneath the head-up display (HUD), and a small data entry display (DED) to monitor communication and navigation input and status. Thousands of man-hours and extensive customer participation went into the design of the interface.

The first developmental test and operational evaluation pilots to fly the new mechanization found it totally unacceptable. There were too many layers of sub-menus. Related functions were accessed through separate sub-menus. Worse, there was no correlation between frequency of task performance and the number of key-strokes or switch activations required.

For example, when the pilot selected TACAN from a master menu, the system would default to a position on the TACAN sub-menu which would allow changing from "X" to "Y" channels. Yet a pilot might not perform this task once in hundreds of flying hours. It took two more switch activations to get to the section of the display where the pilot could change the TACAN channel itself, by far the most frequent reason for accessing the TACAN sub-menu. There were numerous other examples of poor design.

The engineers that designed these less-than-optimum mechanizations may have had a poor understanding of the tasks most frequently performed by aircrews, or how they went about performing them. Yet pilots, both contractor test pilots and Tactical Air Command (TAC) representatives, had seen and approved the mechanization.

They had received briefings from contractor engineers, reviewed documents describing the mechanization, and had flown the mechanization in a simulator. The mechanization had been tested and approved by the customer. Once in the air, however, it was obvious to everyone that the mechanization was seriously flawed.

Considerable effort was required to completely re-design the interface. The same test pilots and operational evaluation pilots who first used the mechanization in an actual aircraft, and were so appalled by the poor mechanization, were constantly consulted during the re-design effort. They spent many hours in contractor simulators evaluating the changes.

The final product took an additional year to develop. It minimizes the number of switch actions required, defaults to the most frequently used selections, and groups related functions together to a much greater degree than the previous design. The new design is user-friendly, quickly accessible even in high-workload situations, and results in very few switch errors.

2 - THE F-15E DUAL-ROLE FIGHTER

The development of the cockpit for the F-15E Dual Role Fighter took a similar path, although a much greater percentage of the cockpit was re-designed. The F-15E was considered a modification to the F-15D. Changes to the cockpit were extensive, however, proceeding in one leap from the "steam gauges" of the F-15C/D to a state-of-the-art "glass cockpit."

The design and development employed greater use of simulation than for the F-16 MSIP cockpit, since the F-15E prime contractor, McDonnell-Douglas, had placed great emphasis on the development of simulation hardware during the preceding decade. Everything from small part-task displays to a complete two-seat cockpit with full visual simulation was used in the development of the F-15E cockpit.

However, in the three years since flight testing began, hundreds of service reports have been submitted by test and operational aircrews on the subject of poor cockpit mechanization. This is not to say that the overall cockpit is poorly mechanized--the aircraft can perform admirably the tasks for which it was designed. But aircrews must sometimes work around an awkward mechanization, and cockpit workload is somewhat higher than desired.

Developmental and operational flight testing have identified the requirement for many changes to the cockpit interface. As testing of additional subsystems and weapons continues, the need for more changes is being established. Unfortunately, production of the F-15E was well already under way when the first aircraft entered flight test. Thus, the required changes will have to be made by retrofit over the next several years.

Tough budgetary constraints mean that only the most significant changes can be made in each software revision. A total re-design, such as the F-16 MSIP cockpit, or a one-time upgrade addressing all problems discovered to date, is not financially feasible.

There was every reason to expect that the extensive use of simulation in the development of the F-16 MSIP and F-15E cockpits would result in a nearly optimum aircrew-avionics interface. Why was this not the case? Certainly some problems resulted from the limitations of the simulation employed, which I shall discuss later. But I believe the main fault lies with the system of evaluation, and in the incomplete or inappropriate use of the simulation available.

Why Simulation Sometimes Fails To Produce Optimum Cockpit Design

1 - THE "COMMITTEE"

Air Force Systems Command requires that the customer, usually represented by a panel of operational aircrews, have extensive input during the design and development of a new weapons system cockpit. Certainly no one knows better the requirements for and intended operational use of a new weapons system or subsystem.

This input is normally made by a cockpit review panel consisting of operationally qualified aircrews who are currently or have recently been assigned to operational units performing the mission envisioned for the new

weapons system. Who better understands the environment under which the system will be deployed? Who better can mentally put himself into the cockpit and imagine how best to mechanize a particular interface?

The shortcoming of this philosophy is that many of the operational aircrews employed to evaluate the initial designs of recent weapons systems, or major modifications to existing ones, have little or no knowledge of the sort of advanced cockpit and display technology being employed. They have no training in human factors, and no knowledge of specifications or regulations pertaining to cockpit design.

Because their experience is usually restricted to one or two older-generation weapons systems, they have no working or even peripheral knowledge of how similar mechanization problems are being solved in other, newer aircraft. The more significant the advance in technology, the more consequential these shortcomings become.

These panels review and approve, on behalf of the customer headquarters, proposed cockpit mechanizations. The members of the panels are chosen by those headquarters. Criteria for selection usually consists of operational experience and, all too often, the individual's *availability*. Since operational units don't like to give up one of their limited number of mission-ready pilots for even a brief period of time, many members are drawn directly from the requirements branch of headquarters staff.

Surprisingly, many have little or no experience in the type of mission for which the aircraft was intended. For example, the F-15E's raison dêtra is air-to-ground, although it retains an impressive air-to-air capability. Yet during the development of the cockpit, an overwhelming majority of voting members of the F-15E cockpit review panel were F-15 air-to-air pilots.

Typically, during the development of a new aircraft, the cockpit review panel meets quarterly at the prime contractor's facility. Members travel from their staff or operational locations, and remain at the contractor site for only three or four days (more during the early stages of development). While at the facility, the panel is bombarded with tens or hundreds of mechanizations to evaluate, most of which they are seeing for the first time.

Subsystem engineers provide briefings on each mechanization in question. The panel discusses the mechanizations and asks questions of the engineers present. The panel then approves the mechanization, recommends changes, requests the contractor develop a different mechanization, or asks for information that is not immediately available. Unless the mechanization is approved or a firm recommendation for change established, the mechanization, in a new and improved form, will be briefed again at the next scheduled meeting.

After a review of new and previous business is completed, the panel will usually adjourn to a simulator to get a hands-on look at some of the mechanizations in question. Often panel members spend only 15 to 30 minutes in the simulator, and can examine each mechanization only once. Review team members are seldom proficient enough to conduct an entire mission profile, nor is sufficient time allotted.

Once the meeting is concluded, panel members will seldom conduct further study of the problem until the next meeting. Simply put, they are too involved performing their regular jobs.

Many major decisions on cockpit mechanization are made during these meetings. These decisions are made by panel member pilots who, by their nature, have very strong opinions and many good ideas. But they have only a limited amount of time to study each mechanization and decide whether it is appropriate or not. And beyond their operational background, they have no training and experience upon which to base their decisions.

2 - PILOT ATTENTION SPAN

It takes thousands of hours to develop the thousands of pilot-avionics mechanizations required for the operation of a modern fighter cockpit. Each mechanization is developed, in isolation, by an individual or small

group who make up the design team for that subsystem. Next, the mechanization is examined by engineers responsible for integrating that subsystem into a larger subsystem, or into a coherent whole.

Sometimes a contractor or customer test aircrew may be asked to review the mechanization of a subsystem, or use it in part-task simulation. These aircrews will also assist in the development of the entire cockpit management philosophy.

Finally, the members of cockpit review teams may consider a particular mechanization for only a few minutes before approving it. During that time, they must listen as the mechanization is briefed by an engineer, try to imagine performing that particular task in an aircraft that has not even been developed yet, consider alternative mechanizations, consider how that mechanization may interact with other subsystems, etc.

An unfortunate aspect of human nature is that aircrews are generally unable, although not consciously unwilling, to completely think through a minor task or mechanization in advance. Once in flight, virtually any pilot can discover and subsequently describe in vivid detail the shortcomings of mechanizations which he approved months or years before. Simulators have done a lot to reduce this phenomenon. But it still seems there is never enough time to evaluate completely a mechanization during development, and always enough time to re-mechanize a bad system once it is discovered in flight test.

3 - EVALUATION BY PARTIAL TASK

Often a simulator is used to evaluate a specific cockpit mechanization by performing only a very limited task. A design group wants an evaluation of a specific sub-mechanization. The test aircrew may be a company test pilot or systems operator, or a visiting operational aircrew.

Simulator time is expensive and limited, and engineering groups from other subsystems are always competing to use the simulator. So to save time, the simulation is accomplished only at a specific point in the sky, with systems already up and running. Then a short evaluation is performed, ending when the limited objective has been achieved.

The limited evaluation may be performed many times, often to evaluate several mechanization candidates. But the start and end points are tightly defined. And the use of other systems, which might have a significant negative impact on the ability to operate the system in question, is avoided. Often, those other systems have not even been developed, or are being re-mechanized themselves. For these reasons it is usually impossible, during the development of a modern fighter aircraft, to perform an end-to-end evaluation of one of the aircraft's multiple missions.

Anyone can perform a task, no matter how poorly mechanized, if he has nothing else to occupy his time or attention. An engineer may design an avionics interface using an organizational philosophy of multiple submenus. The logic of the organization may seem impeccable. He is easily able to accomplish the task in his mind, or on paper, or in a part-task simulation. And the fact that it takes six keystrokes to get to the proper sub-menu seems reasonable, especially if there is some logical progression involved.

The engineer may ultimately review his design with a pilot, either over the phone or sitting in a comfortable conference room. In this atmosphere, the design also seems reasonable to the pilot. That same pilot may later test the mechanization in a part-task or full-up simulator. It is likely that he will examine that mechanization in isolation, or as part of a very limited, highly structured task. And it is possible that, if a deficiency exists, he may still not notice it.

Months or years later, the pilot will perform the same task in a test aircraft in flight. To get to the same point in the sky, he must operate the aircraft from engine start to shutdown. He must activate and monitor all systems necessary for normal flight operations, not just the subsystem of interest. Often, with the additional stress and workload of performing the entire mission, the pilot discovers for the first time that the mechanization is illogical or intensifies workload.

The shortcomings of the mechanization might have been obvious in the simulator, if the time had been allotted, and the aircrew required, to simulate the entire mission profile. Instead of modeling as closely as possible the mission environment, the simulator was used to evaluate, in isolation, only a rigidly-defined aircrew task.

4 - THE LIMITS OF EXPERIENCE

Test crews are often removed by many years from the operational world. Fortunately, some contractor test pilots and weapon systems operators are also members of Air National Guard or Air Force Reserve organizations, and thus keep track of the "real" world and the capabilities and limitations of the average aircrew.

Far too often, however, test crews have so much experience, both in total flying hours and in dealing with advanced technology, that they find it easy to work around mechanizations which will, in operational applications, measurably increase workload for an average operational crew.

In addition, because they tend to test one subsystem at a time, test crews seldom encounter the degree of frustration and high workload which results from operating all systems simultaneously. For this reason, many poor mechanizations are not identified until operational evaluations are performed. Most test aircrews are keenly aware of this phenomenon. They work diligently to maintain an *operational* perspective in their evaluations whenever possible. Even when not the subject of a particular task or evaluation, they will note poor mechanizations which may adversely impact the conduct of tactical operations.

Operational aircrews, on the other hand, are often insensitive to poor mechanization. They are highly competitive, determined to accomplish their mission despite all obstacles. They work hard to adapt to poor mechanizations and complete each task no matter how awkward the interface. It's a matter of personal pride that they can make any system work no matter how poorly designed.

The ultimate example of this phenomenon is the common QWERTY typewriter keyboard. Millions of secretaries type with remarkable speed and accuracy. Few understand that the keyboard they use was purposefully designed to slow them down (to keep from jamming the fragile mechanisms of early typewriters). Even if they knew this, very few would trade the keyboard to which they have become accustomed for one properly optimized for speed. One tends to prefer what one learned first, regardless of the quality of the design.

5 - SIMULATOR SHORTCOMINGS

Sometimes, in spite of the best efforts of all individuals involved, a problem or poor mechanization is not discovered prior to flight test due to some limitation of the simulation itself. Such was the case when the LANTIRN-equipped F-16 was integrated with automatic terrain following.

The F-16 LANTIRN aircraft had been operating with manual terrain following for over 5 years. To use the system, the pilot manually follows a pitch command viewed on the HUD. With the addition of a new quad-redundant digital flight control system, the terrain following (TF) commands can be integrated into the flight control system. This permits fully automatic terrain following, in which the aircraft autopilot controls both the longitudinal and lateral axes.

While the simulator was able to model the flight control system's response to TF commands, it had no motion base. Pilots and engineers were completely surprised by the very aggressive and uncomfortably rough ride produced in the test aircraft. Neither the simulation, nor the hundreds of hours of actual manual terrain following experience, had indicated there might be a problem. When manually following TF commands, pilots had provided just enough lag filtering to smooth out the ride. But when automatic terrain following was engaged, the aircraft flight control system followed the longitudinal TF commands too accurately, with no lag to smooth out the ride.

As a result of flight testing, a new TF mode was developed to provide a somewhat less aggressive and more comfortable ride for use during Automatic Terrain Following.

6 - LIMITATIONS OF SIMULATOR COMPUTER THROUGH-PUT

In my observation, the size and speed of simulator computers has not, in general, kept pace with the increased demands of multiple sensors and displays. This is less a problem with the developmental simulators used by major airframe contractors, where multiple computers may be used for parallel processing when necessary. It is far more noticeable in production training simulators, where moderate to high gain tasks saturate the computational capabilities of the simulator. The resulting time delays can generate divergent pilot induced oscillations in the longitudinal, lateral, and directional axes during high-gain tasks.

Even in developmental simulators, however, lack of fidelity and excessive time delays may result in inaccurate estimates of workload and unnecessary limitations on projected employment of a new weapons system.

7 - THE LIMITATIONS OF VISUAL SIMULATION

There are other examples of simulator shortcomings. It is virtually impossible to generate a realistic FLIR video simulation. It would take an incredibly powerful computer to realistically simulate the detail and dynamically changing resolution of actual FLIR video as it is viewed from a fast moving aircraft at low altitude. This difference is more evident with newer generation digital scan converter FLIR, which have higher resolution than older video multiplexed FLIR. Real-world FLIR cues, which provide pilots with a surprisingly good sense of height and speed, are absent in a simulator.

Actual FLIR video gives the best resolution, and thus most *comfortable* picture, at relatively low altitudes (less than 100 meters). Simulator FLIR video has the same contrast and (low) resolution regardless of altitude. As a result, some pessimistic estimates were made regarding pilot comfort and workload prior to the first night low level flights of the LANTIRN-equipped F-16 aircraft.

The low bandwidth of simulated FLIR video led to the acceptance of inadequate HUD video processors for both the F-16/LANTIRN and the F-15E aircraft. These processors lacked sufficient bandwidth to display the full dynamic range of actual FLIR video. Both looked acceptable when displaying still FLIR video in a laboratory, and both looked acceptable in simulators with their unrealistic (low detail) FLIR simulations.

Once the aircraft flew in the night low-level environment, the shortcomings of the HUDs were quickly identified and significant video processing improvements were required.

8 - SIMULATOR ENCLOSURE LIGHTING

In other instances, the lighting environment of the simulator and its enclosure can generate false conclusions. The simulator used during the development of the F-15E cockpit is mounted in a darkened dome upon which a visual simulation can be projected. During early cockpit review team evaluations of the F-15E cockpit, aircrews were alarmed by reflections from the displays as viewed on the canopy, especially from the rear cockpit. The reflections moved rapidly with the slightest head motion, and were very disorienting.

The Up-Front Controller (UFC) display was thought to be a major cause of the lighting problem. A mechanization was designed which allows either aircrew member to blank his UFC display by pressing the CLEAR push-button twice.

In practice, aircrews are not disturbed by reflections on the canopy during night operations, especially when they are busy performing normal mission tasks. The mechanization used to blank the UFC display, which is of no practical use whatsoever, frequently causes aircrews to unintentionally blank the UFC display when they bump the CLEAR push-button accidentally, or more times than intended.

The use of liquid crystal displays is also difficult to evaluate in a simulator. Displays which appear to have sufficient contrast in a dimly-lit simulator enclosure may be virtually unreadable in the bright sunlight.

Some Modest Proposals

In most cases, the solution to the problems cited lies in an awareness of their existence. A review of the history of previous development programs should broaden the perspective and increase the effort on the part of engineers and aircrew members to more thoroughly visualize the impact of each subsystem interface.

KNOWLEDGE - Where possible, a greater number of operational aircrews need to fly complete mission profiles in simulators before cockpit displays and mechanizations are finalized. They must be thoroughly knowledgeable and proficient in the operation of all systems to successfully evaluate any part of the system. This is expensive and time consuming, since an ad hoc "ground school" must be given, frequently by the same engineers that are busy developing the systems. This can occur months or years before training courses for the system are developed, but it does have the benefit of providing an early start for contractor and customer training personnel.

TIME - Crews must be given sufficient time in the simulator to develop proficiency and become aware of the interaction of various subsystem mechanizations. A quick 15-30 minutes in the simulator per visit is totally inadequate.

BREATHING HARD - Just as it is impossible to improve your physical health without getting your heart rate up by exercising, so is it impossible to evaluate the mechanization of a cockpit without doing so under a workload similar to that of operational conditions. Once partial tasks have been evaluated and a mechanization selected, larger and more realistic mission segments must be simulated by aircrews proficient with the new mechanizations.

BROADER MEMBERSHIP - It is entirely proper that a cockpit review be conducted by the customer, as represented by operationally qualified aircrews. However, the program office must be responsible, as well as responsive, to the customer. On occasion a review panel may make a recommendation that clearly reflects the opinion of only one or two dominant personalities. The program office may adopt the mechanization in question despite the advice of highly trained and experienced contractor engineers and flight test personnel. To reduce this sort of occurrence, the cockpit review team should include members from a broad range of operational and test backgrounds.

EDUCATION - Additional steps should be taken to ensure all members of a cockpit review panel have the education and exposure necessary to review and approve the elements of a cockpit design. Some training in human factors and cockpit design should be provided to those members with no previous training. Before serving on the panel, members should be provided an opportunity to examine a number of other aircraft cockpits and thus experience a variety of solutions to mechanization problems. A visit with current developmental and operational test programs would provide an excellent review of lessons learned from these projects.

STABILITY - Once a cockpit review team is established, and the members provided the necessary training and exposure, the team needs to be kept intact throughout the development effort. Too often, even though the effort is made to educate members of a cockpit review panel, the original members go on to other assignments within one to two years. The newer members do not have the benefit of the education, or the experience, to intelligently evaluate mechanizations that have evolved during the life of the review panel.

ABOUT THE AUTHOR: Lt Col C. G. Killberg is an F-15 Experimental Test Pilot with the 6515th Test Squadron, Edwards Air Force Base. He was previously assigned to the original F-16/LANTIRN test team.

REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's Reference	3. Further Reference	4. Security Classification of Document
	AGARD-CP-473	ISBN 92-835-0578-6	UNCLASSIFIED

5. Originator Advisory Group for Aerospace Research and Development
North Atlantic Treaty Organization
7 rue Ancelle, 92200 Neuilly sur Seine, France

6. Title COMPUTER AIDED SYSTEM DESIGN AND SIMULATION

7. Presented at the Guidance and Control Panel 50th Symposium held at the Altin Yunus Hotel, Cesme/Izmir, Turkey from 22nd to 25th May, 1990.

8. Author(s)/Editor(s)	9. Date
Various	August 1990
10. Author's/Editor's Address	**11. Pages**
Various	402

12. Distribution Statement This document is distributed in accordance with AGARD policies and regulations, which are outlined on the Outside Back Covers of all AGARD publications.

13. Keywords/Descriptors

Avionics
Computers
Control
Guidance
Hardware-in-the-loop
Microcomputers
Navigation
Pilot-in-the loop
Simulation tools
Systems design

14. Abstract

This volume contains the 32 unclassified papers, including the Keynote address, presented at the Guidance and Control Panel Symposium held at the Altin Yunus Hotel, Çeşme/Izmir, Turkey from 22nd to 25th May, 1990.

The papers were presented covering the following headings:
— Computer aided system design;
— Simulation technology for missile applications;
— Simulation technology for aircraft applications;
— Hardware-in-the-loop simulation;
— Systems applications;
— Pilot-in-the-loop simulations.

AGARD Conference Proceedings No.473 Advisory Group for Aerospace Research and Development, NATO COMPUTER AIDED SYSTEM DESIGN AND SIMULATION Published August 1990 402 pages This volume contains the 32 unclassified papers, including the Keynote address, presented at the Guidance and Control Panel Symposium held at the Altın Yunus Hotel, Çeşme/Izmir, Turkey from 22nd to 25th May, 1990. The papers were presented covering the following headings: — Computer aided system design; P.T.O.	AGARD-CP-473 Avionics Computers Control Guidance Hardware-in-the-loop Microcomputers Navigation Pilot-in-the-loop Simulation tools Systems design	AGARD Conference Proceedings No.473 Advisory Group for Aerospace Research and Development, NATO COMPUTER AIDED SYSTEM DESIGN AND SIMULATION Published August 1990 402 pages This volume contains the 32 unclassified papers, including the Keynote address, presented at the Guidance and Control Panel Symposium held at the Altın Yunus Hotel, Çeşme/Izmir, Turkey from 22nd to 25th May, 1990. The papers were presented covering the following headings: — Computer aided system design; P.T.O.	AGARD-CP-473 Avionics Computers Control Guidance Hardware-in-the-loop Microcomputers Navigation Pilot-in-the-loop Simulation tools Systems design
AGARD Conference Proceedings No.473 Advisory Group for Aerospace Research and Development, NATO COMPUTER AIDED SYSTEM DESIGN AND SIMULATION Published August 1990 402 pages This volume contains the 32 unclassified papers, including the Keynote address, presented at the Guidance and Control Panel Symposium held at the Altın Yunus Hotel, Çeşme/Izmir, Turkey from 22nd to 25th May, 1990. The papers were presented covering the following headings: — Computer aided system design; P.T.O.	AGARD-CP-473 Avionics Computers Control Guidance Hardware-in-the-loop Microcomputers Navigation Pilot-in-the-loop Simulation tools Systems design	AGARD Conference Proceedings No.473 Advisory Group for Aerospace Research and Development, NATO COMPUTER AIDED SYSTEM DESIGN AND SIMULATION Published August 1990 402 pages This volume contains the 32 unclassified papers, including the Keynote address, presented at the Guidance and Control Panel Symposium held at the Altın Yunus Hotel, Çeşme/Izmir, Turkey from 22nd to 25th May, 1990. The papers were presented covering the following headings: — Computer aided system design; P.T.O.	AGARD-CP-473 Avionics Computers Control Guidance Hardware-in-the-loop Microcomputers Navigation Pilot-in-the-loop Simulation tools Systems design

— Simulation technology for missile applications; — Simulation technology for aircraft applications; — Hardware-in-the-loop simulation; — Systems applications; — Pilot-in-the-loop simulations. ISBN 92-835-0578-6	— Simulation technology for missile applications; — Simulation technology for aircraft applications; — Hardware-in-the-loop simulation; — Systems applications; — Pilot-in-the-loop simulations. ISBN 92-835-0578-6
— Simulation technology for missile applications; — Simulation technology for aircraft applications; — Hardware-in-the-loop simulation; — Systems applications; — Pilot-in-the-loop simulations. ISBN 92-835-0578-6	— Simulation technology for missile applications; — Simulation technology for aircraft applications; — Hardware-in-the-loop simulation; — Systems applications; — Pilot-in-the-loop simulations. ISBN 92-835-0578-6