AGARD-CP-474

AGARD-CP-474

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD CONFERENCE PROCEEDINGS No.474

Knowledge Based System Applications for Guidance and Control

(Application des Systèmes à Base de Connaissances au Guidage-Pilotage)

NORTH ATLANTIC TREATY ORGANIZATION

AGARD-CP-474

NORTH ATLANTIC TREATY ORGANIZATION

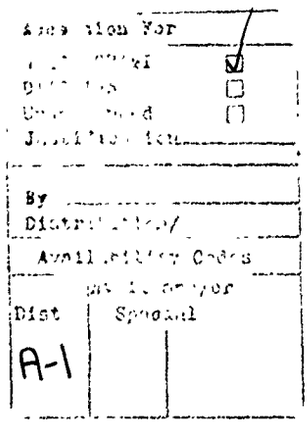
ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT

(ORGANISATION DU TRAITE DE L'ATLANTIQUE NORD)

AGARD Conference Proceedings No.474

Knowledge Based System Applications for Guidance and Control

(Application des Systèmes à Base de Connaissances au Guidage-Pilotage)



Papers presented at the Guidance and Control Panel 51st Symposium held at the Instituto Nacional de Industria, Madrid, Spain from 18th to 21st September 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

Published April 1991

ISBN 92-835-0610-3

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

Theme

The combination of increasing military system and task complexity, in the face of inherent human limitations has set the stage for development of innovative system integration approaches involving the use of knowledge based technology. The field of Artificial Intelligence (AI) is becoming solidified as a science and the technological aspects are developing rapidly. The implications for guidance and control are enormous. Practical applications of AI are critically dependent on advanced architectures, computer processing techniques and integration concepts. Recent advances in digital computation techniques including data base management, represent the core enabling technology necessary for the development of highly innovative design concepts, which will ultimately lead to major new military capabilities. Efficient tactical information management and effective pilot interaction are essential. Pilot decision aiding, combat automation, sensor fusion and on-board tactical battle management concepts offer the opportunity for substantial mission effectiveness improvements. Although real-time tactical military applications are relatively few in number, several exploratory and advanced development efforts are underway. Practical military applications of AI technology are of primary interest. Projected military capability enhancements along with AI limitations were considered. Operational implications, and critical design trade-offs were also emphasized. This symposium provided a timely forum for assessing the overall state-of-the-art of AI applications in the guidance and control area.

A round table discussion to identify application issues and opportunities was held.

Thème

La complexité croissante des tâches militaires et des systèmes d'armes face aux limites inhérentes aux capacités humaines, a préparé le terrain pour le développement d'approches novatrices recourant aux techniques des systèmes à base de connaissances. Le domaine de l'intelligence artificielle s'affirme en tant que science dont les éléments technologiques évoluent très rapidement. Les conséquences pour le guidage et le pilotage sont considérables. Les applications pratiques de l'intelligence artificielle sont directement tributaires des améliorations des architectures, des techniques du traitement des données par ordinateur et des concepts d'intégration. Les progrès récents dans le domaine des techniques de calcul numériques, y compris la gestion de bases de données, représentent le noyau technologique indispensable au développement de concepts hautement novateurs, qui déboucheront, à terme, sur de nouvelles et importantes applications militaires. La gestion efficace des informations tactiques et la bonne interaction pilote-système sont essentielles. L'aide à la décision pour le pilote, l'automatisation du combat, la fusion des capteurs et les concepts de la gestion tactique de la bataille par des moyens embarqués ouvrent la voie à une amélioration substantielle de l'efficacité opérationnelle. Bien que les applications temps réel soient encore relativement rares, un certain nombre de projets de développements, tant exploratoires qu'avancés, sont en cours à l'heure actuelle. Les applications militaires des technologies de l'intelligence artificielle sont d'un intérêt primordial. Les améliorations attendues de l'efficacité des moyens militaires ont été examinées conjointement avec les limitations prévisibles de l'intelligence artificielle. Les implications opérationnelles et les compromis critiques établis lors de la conception des systèmes ont fait l'objet d'une analyse particulière. Ce symposium a été ainsi l'occasion pour faire l'évaluation de l'état de l'art des applications de l'intelligence artificielle dans le domaine du guidage et du pilotage

Une table ronde s'est tenue dans le but de faire émerger les applications potentielles

Guidance and Control Panel

Chairman: Dr E.B Stear
Corporate Vice President
Technology Assessment
The Boeing Company
P.O. Box 3707
Mail Stop 13-43
Seattle, WA 98124-2207
United States

Deputy Chairman: Mr S.Leek
PB 256
British Aerospace
(Dynamics) Ltd
P.O Box 19
Six Hills Way, Stevenage
Herts SG1 2DA
United Kingdom

TECHNICAL PROGRAMME COMMITTEE

Chairman:	Mr James K.Ramage	US
Members:	Dr André Benoit	BE
	Mr Bernard Chaillot	FR
	Dr Heinz Winter	GE
	Dr Bruno Mazzetti	IT
	Professor Pedro Sanz-Aranguez	SP
	Professor John T Shepherd	UK
	Dr Elihu Zimet	US

HOST PANEL COORDINATOR

Mr Carlos A Garriga Lopez
Sener Ingenieria y Sistemas SA
Space and Defence Division
c/ Raimundo Fernadez Villaverde, 65
Planta 20
Madrid
Spain.

PANEL EXECUTIVE

Commandant M.Mouhamad, FAF

Mail from Europe:
AGARD—OTAN
Attn: GCP Executive
7, rue Ancelle
F-92200 Neuilly-sur-Seine
France

Mail from US and Canada:
AGARD—NATO
Attn: GCP Executive
APO New York 09777

Tel: 33 (1) 47 38 57 80
Telex: 610176F
Telefax: 33 (1) 47 38 57 99

ACKNOWLEDGEMENT

The Panel wishes to express its thanks to the Spanish National Delegate to AGARD for the invitation to hold this meeting in his country and for the facilities and personnel which made the meeting possible

Le Panel tient a remercier le Délegué National de l'Espagne pres l'AGARD de son invitation à tenir cette réunion dans son pays et de la mise à disposition de personnel et des installations nécessaires

Contents

L'AVENIR DU CONTRÔLE DE LA NAVIGATION AÉRIENNE EN EUROPE

LE «MUR DE LA CAPACITÉ»

Jacques Villiers

PREMIÈRE PARTIE : LA CAPACITÉ DE L'ESPACE AÉRIEN EUROPÉEN

La première partie de cet article examine donc les caractéristiques spécifiques de l'espace aérien européen des points de vue opérationnel, technique et social.

Elle explique pourquoi l'espace aérien européen est d'ores et déjà plus saturé que celui des Etats-Unis. Parmi les nombreuses causes figure l'hétérogénéité du système européen, telle qu'elle résulte des conditions historiques de son développement.

On montrera cependant que son unification ne pourra pas se faire rapidement et que, ni une telle évolution, ni celle qui pourrait résulter de progrès techniques «au fil de l'eau» ne seront à elles seules susceptibles de permettre un accroissement de capacité suffisant pour satisfaire en quantité et en qualité une demande de trafic aérien en expansion rapide.

Les flux d'information et les flux de tâches finissent par dépasser les potentialités de traitement par des opérateurs humains.

La deuxième partie procède à une analyse approfondie de ces phénomènes de saturation.

L'étude montre que les temps sont proches où il sera possible et indispensable que les calculateurs puissent assister les contrôleurs dans leurs décisions, et plus particulièrement dans celles qui concernent leur stratégie d'action en temps réel.

L'analyse met cependant en évidence que cette évolution se heurte à une difficulté conceptuelle fondamentale due à un phénomène, qui sera dénommé le «mur de la capacité», et qui résulte des interactions entre le libre arbitre du contrôleur et celui du calculateur.

La démarche proposée pour parvenir à franchir ce «mur de la capacité» fait appel à la mise en œuvre de «programmes experts» et suggère un projet dont on décrira le principe et la spécificité

Ce projet sera dénommé FREGATE (Filtre de Régulation Ergonomique de la Gestion des secteurs et d'Anticipation des Tâches Excessives) ; il devrait permettre de faire évoluer d'une manière continue le système actuel vers une automatisation croissante des processus d'assistance aux contrôleurs Ce projet constitue un fil directeur possible pour les études et recherches qui se développent au sein des Etats européens et qui sont fédérées par Eurocontrol et par la CEE.

Il serait impératif que ces travaux débouchent en temps opportun sur la conception d'un système commun pour le début du siècle prochain. Les analyses et propositions du présent article constituent une contribution en ce sens et décrivent une forme éventuelle de leur aboutissement.

La tâche sera longue et ardue, mais les enjeux sont essentiels

CAPACITÉ DE L'ESPACE, CAPACITÉ DES AÉROPORTS

Alors que le transport aérien européen entre dans la phase active de la mise en œuvre de l'Acte unique, les autorités responsables et les compagnies aériennes manifestent une inquiétude croissante concernant la saturation de certains aéroports et du contrôle de la circulation aérienne

Cette préoccupation est d'autant plus justifiée que l'effort de libéralisation se développe dans un contexte de très vive croissance du transport aérien, alors qu'aux Etats-Unis, les premières années de la déréglementation se sont écoulées sur un fond de récession économique.

Or, pour s'épanouir librement la concurrence suppose des infrastructures largement dimensionnées - tel n'est déjà plus le cas en Europe.

Les infrastructures doivent d'ailleurs être d'autant plus facilement accessibles que le libéralisme s'oppose, par sa nature même, à l'optimisation de l'utilisation de leur capacité.

Qui plus est, l'expérience américaine a montré que la capture à son profit des capacités d'infrastructure disponibles constitue pour chaque compagnie aérienne un enjeu stratégique d'autant plus déterminant que la concurrence est plus vive et que cette capacité est plus limitée.

Tel est bien le mécanisme qui a accéléré d'une manière spectaculaire la saturation des aéroports principaux américains Alors que le nombre de mouvements totaux (circulation aérienne en route) n'a crû que de 26% de 1978 à 1987, celui des grands aéroports a augmenté de 64% pendant cette même période; de surcroît, les mouvements sur les grands aéroports ont eu tendance à se concentrer sur des périodes de pointe pour optimiser les possibilités de correspondances (1).

Un phénomène de même nature ne manquera sans doute pas de se produire en Europe où plusieurs pôles aéroportuaires sont déjà proches de la saturation.

Contrairement aux Etats-Unis, il se manifeste cependant en Europe une certaine pression antagoniste due à la montée en diversité et en puissance des lignes secondaires desservant directement les villes de province

En revanche, la libéralisation provoquera une intensification d'un phénomène déjà inquiétant en Europe et beaucoup plus sévère qu'aux Etats-Unis: la saturation du contrôle de la circulation aérienne en route.

Dans les causes des retards, la part respective de la saturation du contrôle en route et de celle des aéroports est souvent mal appréciée par les utilisateurs, dans la mesure où les délais aux décollages imposés par les centres de contrôle en route proviennent, selon les cas, de leur propre saturation ou de celle des aéroports de départ ou de destination.

C'est en général l'ensemble qui est agrégé dans l'estimation de la pénalisation apportée par le contrôle.

Il serait aussi fallacieux d'attribuer la source des problèmes européens aux seules limites de capacité du système aéroportuaire qu'à celles du seul contrôle en route. Toutes deux imposent des contraintes dont les causes imbriquées soulèvent des problèmes de nature très différente et appellent des remèdes spécifiques.

Il serait illusoire de traiter certains problèmes en sous-estimant les autres, dans la mesure où c'est à chaque instant le maillon le plus faible qui limite la capacité totale du système dans son ensemble; les phénomènes de saturation ponctuelle se propagent pendant une longue période dans tout le système: par exemple, la saturation d'un aéroport peut entraîner la saturation du contrôle en route lorsque la situation se débloque et inversement.

On rappellera, pour mémoire, que la saturation des pôles aéroportuaires résulte de la conjonction de facteurs très divers: capacité des pistes et des espaces aériens associés, capacité des aires de stationnement et des aérogares, liaisons ville-aéroport, contraintes écologiques...

Ils soulèvent des problèmes de nature locale, tant pour le développement des infrastructures que pour la fixation de leurs règles d'utilisation , ces dernières, et notamment l'allocation de créneaux horaires, résultent de marchandages bilatéraux sur un fond de droits acquis historiques.

Le présent article se propose de ne traiter que le problème de la capacité du système de contrôle de la circulation aérienne en route

Dans l'état actuel du système européen, et devant l'inquiétude soulevée par les sombres perspectives qui résulteraient de son évolution au «fil de l'eau», un effort de clarification semble en effet nécessaire

Dans une première partie, on étudiera les caractéristiques spécifiques du contrôle de la circulation aérienne en Europe et les améliorations susceptibles de lui être apportées en conséquence. On montrera cependant que, même si celles-ci pouvaient toutes être mises en œuvre, l'augmentation de la capacité qui en résulterait ne serait pas suffisante pour permettre de faire face dans de bonnes conditions à la croissance prévisible du trafic.

Dans une deuxième partie, on étudiera la nature profonde des mécanismes de saturation (le «mur de la capacité») et on proposera un projet original, dont on peut espérer qu'il sera de nature à permettre que le contrôle de la circulation aérienne ne constitue pas dans l'avenir un goulet d'étranglement au développement du transport aérien en Europe.

LA SATURATION DU CONTRÔLE EN ROUTE EUROPÉEN

L'ensemble des pénalisations apportées par l'insuffisance de capacité du contrôle en route et des infrastructures aéroportuaires se chiffre à plusieurs milliards d'écus par an La fluidité du contrôle en route constitue un problème majeur auquel il convient de s'attaquer résolument.

Il serait en effet illusoire d'accroître la capacité des infrastructures aéroportuaires, si le contrôle de la circulation aérienne devait en limiter les potentialités d'utilisation.

Il est donc essentiel de recenser et d'analyser tous les facteurs qui affectent la capacité du contrôle en route européen.

On examinera ainsi successivement :

- l'espace aérien européen,
- l'espace social européen,
- l'espace opérationnel et technique européen

L'ESPACE AÉRIEN EUROPEEN

Si on compare sans nuances l'espace aérien européen à l'espace aérien américain, on ne peut que s'étonner du fait que l'espace européen soit sujet à des saturations alors que le trafic intérieur américain est beaucoup plus intense que celui de l'Europe.

Mais une telle comparaison grossière n'est guère pertinente.

La capacité d'un système de contrôle ne se mesure pas à l'aune du trafic moyen et de l'espace total, mais à celle du trafic de pointe dans les espaces disponibles pour l'écouler, là et quand il se présente

La répartition des axes à grand trafic, de même que la distribution horaire du trafic obéissent de part et d'autre à des lois profondément différentes : l'Europe doit faire face à des pointes de trafic à certains moments et sur certains axes qui figurent parmi les plus fortes du monde.

Le trafic européen se concentre essentiellement pendant la journée : le trafic de nuit est quasiment inexistant en raison des couvre-feux imposés sur les aéroports et de l'absence de trafic intérieur long-courrier comparable au «coast to coast» survolant le territoire américain. De même, les variations horaires et saisonnières sont beaucoup plus marquées en Europe, notamment en raison de l'attrait des Européens pour les voyages de vacances.

De surcroît, l'espace disponible pour acheminer les trafics les plus denses y est considérablement limité par les grandes agglomérations urbaines proches les unes des autres, qui ne peuvent pas être survolées, et par l'espace réservé aux zones de contrôle terminal de leurs aéroports.

Enfin, il existe peu d'espaces à l'écart des grands trafics pour permettre le déploiement des bases aériennes militaires et l'entraînement systématique des équipages.

La cohabitation des deux missions civile et militaire dans un même espace limité ne peut manquer de faire peser de lourdes contraintes sur l'une comme sur l'autre. La pénalisation du trafic civil qui en résulte fait l'objet de suffisamment d'analyses publiques pour qu'il ne soit pas nécessaire de s'y attarder; en revanche, celle que subit le système militaire (en limitations opérationnelles et en coût) est moins connue; l'accroissement rapide du trafic tend à rendre les grands axes de trafic civil de plus en plus difficiles à traverser par les vols militaires; ces derniers doivent d'ores et déjà subir en moyenne 2,2 manœuvres de contournement d'avions civils par heure de vol contrôlé.

L'ESPACE SOCIAL EUROPÉEN

Les conséquences brutales sur l'écoulement du trafic des mouvements sociaux qui se produisent à répétition, ici ou là, constituent à l'évidence la cause la plus fréquente et la plus sévère de limitation de la capacité du système européen.

Dans l'analyse statistique des retards et des contraintes que ce système occasionne effectivement aux compagnies aériennes, il faudrait donc faire clairement la part de ce qui revient à la limite de sa capacité effective, et de ce qui doit être attribué aux conséquences de tensions sociales endémiques.

C'est d'ailleurs précisément dans les périodes de plus forte pression de trafic que se manifestent les mouvements sociaux : c'est pendant ces périodes que les contraintes qui pèsent sur les personnels sont les plus vives (et donc que leur insatisfaction est la plus grande) et que l'impact sur le trafic d'un mouvement de mauvaise humeur ou de grève est le plus marquant.

Les conflits sociaux portent non seulement sur les salaires, mais aussi sur les conditions de travail qui s'aggravent dès que le système approche de sa limite de capacité, notamment si cette limite résulte de l'insuffisance des effectifs mis en place pendant les heures de pointe.

Le problème est universel, mais il est rendu d'autant plus redoutable en Europe qu'un mouvement de grève (ou de réduction intentionnelle de capacité dite «grève du zèle») dans un des pays, entraîne des perturbations dans l'ensemble du système En revanche, une partie du trafic peut être acheminée en contournant les zones stérilisées par une grève.

Ces mouvements s'étalent souvent sur de très longues périodes au cours desquelles l'absence de bonne volonté des contrôleurs entraîne une réduction de la capacité effective du système, sans publicité de ces comportements comme dans le cas d'une grève franche.

C'est ainsi, par exemple, qu'à la suite du protocole d'accord signé le 4 octobre 1988 en France avec les personnels, la capacité du système français est rapidement revenue à un niveau tel qu'elle n'occasionne plus pour l'instant de contraintes lourdes sur le trafic.

Certains espèrent que ces problèmes sociaux pourraient trouver leur solution dans le cadre d'une fonction publique européenne ou dans celui de statuts des personnels dégagés des contraintes inhérentes aux fonctions publiques nationales ou européennes. Il s'agit de problèmes de nature particulièrement délicate : le rythme d'évolution ne saurait dépasser celui des mentalités.

L'exemple américain montre d'ailleurs que la gestion d'un grand système unique n'est pas exempte de graves difficultés sociales il a appelé des mesures radicales qui ne sauraient être données comme exemple attractif aux personnels européens.

L'ESPACE OPÉRATIONNEL ET TECHNIQUE EUROPÉEN

Il est clair que, malgré les efforts d'Eurocontrol, le système européen reste caractérisé par un manque évident d'homogénéité d'exploitation, de planification et de conception. Faut-il s'en étonner ?

Cette hétérogénéité a des racines profondes qui tiennent notamment à un ensemble de causes historiques. Chaque Etat a été amené à développer son propre système en fonction de conditions locales spécifiques : nature et intensité du trafic, organisation des services publics nationaux (civils et militaires), structure et culture sociales, systèmes de formation et de sélection, méthode et style d'approche des problèmes opérationnels et techniques.

De surcroît, la cohabitation des circulations aériennes civile et militaire dans de mêmes espaces implique des coordinations intimes entre les états-majors et les services et aggravent encore le caractère national de chacune des composantes civiles du système européen.

L'EXPLOITATION

Ces disparités concernent d'abord l'exploitation opérationnelle elle-même. Il existe dans ce domaine, plus que dans aucun autre, une «culture professionnelle» affirmée, dont les particularismes affectent profondément les méthodes et les modalités d'exécution du contrôle.

Le contrôle de la circulation aérienne est certes une technique, mais c'est surtout un art pragmatique dont la pratique exige l'acquisition d'un «savoir faire» et le recours à tout un ensemble de réflexes, voire de «recettes» locales; il ne peut pas être considéré comme découlant de la simple mise en œuvre d'une théorie générale. Les hommes, en fonction de leur niveau de recrutement et de leur mode de formation, acquièrent des habitudes de travail dans lesquelles les conditions locales jouent un rôle déterminant.

Rappelons notamment qu'un apprentissage et un entraînement «sur le tas» d'environ trois à quatre ans sont nécessaires pour l'accession à la qualification maximale de contrôle dans un centre à grand trafic.

Dans de telles conditions, on comprendra que le système soit soumis à une très grande inertie d'évolution : tout changement significatif dans les habitudes de travail des contrôleurs implique une modification des réflexes psychosensoriels qu'ils ont acquis par une longue pratique. Or, c'est l'existence de ces réflexes sûrs et rapides qui constitue la source même de la capacité de pointe du système.

C'est ainsi d'ailleurs que, lors d'une mutation d'un centre à un autre, même au sein d'un même système national (voire d'un sous-ensemble de secteurs à un autre au sein d'un même centre), les contrôleurs doivent subir un nouvel entraînement de longue durée.

La spécificité de ce métier est une conséquence de la nature heuristique des processus de contrôle; il en résulte d'ailleurs que l'automatisation même partielle d'un tel système ne pourra pas progresser notablement par la seule mise en œuvre de processus logico-mathématiques, et donc que le recours à «l'intelligence artificielle» - c'est-à-dire la reproduction par des calculateurs de l'expertise et de l'acquis des contrôleurs - constitue la seule voie prometteuse pour l'avenir (2).

Les hommes et leurs outils de travail ne sont donc pas directement interchangeables, dans le court terme, d'un pays à un autre, ni même d'un centre à l'autre, voire d'un secteur à l'autre.

Les conséquences pour l'usager des disparités d'exploitation se concrétisent notamment par le fait que le passage d'une frontière par un avion est souvent moins bien optimisé que le simple passage d'un secteur à un autre au sein d'un système national (coordinations plus lourdes, espace moins bien organisé). Dans cette mesure, la multiplication des exploitations nationales juxtaposées nuit effectivement à la fluidité du trafic.

Il serait cependant fallacieux de conclure de ce seul fait que le contrôle de la circulation aérienne en Europe est pénalisé par l'existence d'un trop grand nombre de centres de contrôle.

On serait plutôt tenté de penser que l'organisation des mouvements d'avions répond plus à des impératifs de lutte concurrentielle qu'à une utilisation optimale des ressources rares constituées par les capacités aéroportuaires et les espaces aériens, et qu'en ce sens, il existe peut-être trop de compagnies aériennes en Europe!

Le choix du nombre optimal de centres de contrôle constitue un problème d'une autre nature; il doit procéder d'une analyse portant sur un grand nombre de paramètres techniques, économiques et sociaux. C'est ainsi, par exemple, que les autorités françaises ont été amenées à répartir progressivement le contrôle en route entre trois, puis cinq centres distincts, ce qui n'a pas nui pour autant à l'efficacité de l'ensemble du système.

A l'heure de la télé-informatique et du Transpac, l'information circule aussi facilement entre des centres géographiquement séparés qu'au sein d'un même centre.

Toutes les positions de contrôle, où qu'elles soient localisées, sont d'ailleurs alimentées en information de base (plans de vol, radar, radiotéléphonie air-sol, téléphone) par voie de liaisons téléphoniques: il y a maintenant bien longtemps que les contrôleurs ne communiquent plus directement entre eux, que dans des circonstances relativement exceptionnelles, même au sein d'un même centre.

L'efficacité globale du contrôle découle en revanche du nombre de secteurs de contrôle entre lesquels est découpé l'espace.

Le nombre de secteurs nécessaires - c'est-à-dire le nombre d'unités de contrôle autonomes coordonnées deux à deux - ne résulte que du nombre maximal d'avions qu'un contrôleur et ses assistants peuvent prendre en charge simultanément dans un environnement donné.

Or, toutes choses égales par ailleurs, ce nombre maximal - de l'ordre de 10 à 15 avions selon la nature du secteur - est le même dans tous les pays ; il est en particulier le même en Europe et aux Etats-Unis.

En revanche, le nombre moyen d'avions qu'un contrôleur achemine effectivement au cours d'une journée ou d'une année peut être très différent d'un lieu à un autre, car il est directement fonction de la répartition du trafic dans le temps. Avec ce seul critère, le système américain peut sembler beaucoup plus «productif» que le système européen, car le trafic y est soutenu d'une manière plus continue. Mais la productivité mesurée par ce critère dépend aussi beaucoup des conditions de travail (nombre d'heures de contrôle effectif par semaine, organisation du travail, flexibilité des horaires, congés octroyés pendant les jours de pointe, etc.).

Sur ce dernier point, le système américain est sans aucun doute plus «productif» que celui de l'Europe en général on notera par exemple qu'en Europe, une partie de l'effectif est autorisée à prendre ses vacances pendant les jours de pointe, alors que, au contraire, les centres américains restent dotés de l'effectif total qui est, de surcroît, invité à effectuer des heures supplémentaires pendant ces périodes les plus chargées.

Il est donc exact que, en moyenne, un contrôleur européen achemine annuellement moins de trafic qu'un contrôleur américain mais le nombre de centres de contrôle n'y est sans doute pas pour grand chose.

Il est exact aussi que la «productivité» des contrôleurs américains (nombre d'avions traités par an) a crû d'une manière rapide aux Etats-Unis (elle a doublé en 10 ans).

En Europe cette «productivité» reste relativement constante en moyenne ; on notera cependant qu'elle a crû brusquement à la suite de la reprise brutale du trafic qui a surpris bien des planificateurs. Il en a d'ailleurs résulté, par le processus décrit ci-dessus, à la fois des contraintes accrues sur les personnels et des mouvements sociaux qui ont amplifié le phénomène de saturation.

En ce qui concerne les coûts salariaux, les conditions sont très différenciées au sein de l'Europe, aussi bien pour les niveaux de salaire (et des charges sociales et fiscales associées) que pour les conditions de travail (nombre d'heures effectives de contrôle annuel, notamment pendant les périodes de pointe de trafic, flexibilité des horaires...).

La réduction de ces disparités ne manquera pas de soulever de graves problèmes et exigera beaucoup de temps et de sages-se, il n'est d'ailleurs pas certain que le coût total du contrôle y trouvera son compte.

L'exemple américain montre aussi qu'un grand système de contrôle unifié est très vulnérable et n'est pas à l'abri de graves problèmes sociaux.

En revanche, l'efficacité d'ensemble de l'exploitation européenne pourrait facilement progresser grâce à l'accroissement des contacts sous toutes leurs formes entre les systèmes nationaux : formation de base commune, échange de contrôleurs et de cadres, approfondissement en commun de tous les problèmes d'exploitation (organisation de l'espace, méthodes de contrôle, outils de travail du contrôleur, coordinations entre centres...).

La situation évolue d'ailleurs favorablement en ce sens. Elle pourrait être largement améliorée s'il était fait appel systématiquement à des mécanismes d'arbitrage pour trancher entre Etats ou centres voisins en cas de conflits opérationnels.

Il serait important que soit élaboré au préalable, en coopération avec les compagnies aériennes, un «Livre blanc» recensant tous les problèmes d'exploitation qui résultent spécifiquement de la disparité des modes d'organisation de l'espace et d'exploitation des différents systèmes de contrôle nationaux.

Il est enfin un domaine où l'action commune s'est déjà imposée à l'évidence, à savoir le contrôle des flux de trafic pour réguler la demande en fonction de l'offre de contrôle disponible et éviter ainsi de saturer les centres de contrôle au détriment de la sécurité et de la fluidité du trafic.

Contrairement à celles du contrôle de la circulation aérienne proprement dite, les relations de coordination de «flow control» ne sont pas uniquement de nature bilatérale de secteur à secteur mais impliquent la prise en compte simultanée de l'ensemble du système (en route et aéroportuaire) et sur un très grand volume d'espace.

On ne peut donc que se féliciter du fait que les mesures soient désormais prises pour mettre en place en Europe un dispositif de «flow control» unifié. Sa mise en œuvre soulèvera peut être des difficultés moindres (mais la preuve n'en est pas encore apportée) que celles qui affectent le contrôle proprement dit, notamment parce que cette pratique n'a pas derrière elle une longue histoire différenciée au sein de chaque Etat européen.

La part des processus logico-mathématiques modélisables est sans doute plus grande que celle du contrôle lui-même, il ne faudrait cependant pas sous-estimer pour autant l'importance du «savoir faire» spécifique qu'ont développé par la pratique les agents chargés de ce travail depuis plusieurs années.

On verra par ailleurs dans la suite du présent article que la régulation efficace des flux peut non seulement réduire les contraintes globales des compagnies aériennes, à capacité donnée du système de contrôle, mais participe d'une manière non négligeable à l'augmentation de la capacité de pointe effective.

LA PLANIFICATION

C'est l'OACI qui élabore les normes et pratiques recommandées en matière de navigation aérienne, tandis que son groupe spécialisé (le GEPNA) énonce les besoins opérationnels spécifiques de l'Europe et coordonne la planification de ses infrastructures et de ses moyens de contrôle.

De son côté, Eurocontrol contribue à l'harmonisation de l'ensemble du dispositif européen et de son développement progressif, et assure la facturation commune des redevances propres à chaque système national.

Les compagnies aériennes participent aux discussions au sein de ces entités.

Mais la planification effective, c'est-à-dire le financement, le choix et l'installation des équipements ainsi que la mise en place des personnels restent sous l'autorité des instances nationales.

Dans ces conditions, on ne saurait escompter que la planification et l'adaptation progressive et harmonieuse du système aux besoins présentent une homogénéité, une cohérence et une efficacité aussi grandes que si elles relevaient d'une autorité effective unique.

Le problème n'est cependant pas aussi simple qu'il pourrait le paraître a priori.

Le pluralisme des modalités nationales d'exploitation opérationnelle qui, on l'a vu, n'est pas aisément ni rapidement réductible, ne permet guère une planification et une mise en œuvre rigoureusement centralisées.

Par ailleurs, les systèmes nationaux sont soumis à des procédures budgétaires propres à chaque pays , certains d'entre eux ont cependant pu alléger les contraintes qui résultent de la rigueur générale qui pèse sur les budgets des Etats, en mettant en place des structures mieux appropriées (budget annexe, établissement public. .) qui permettent, de surcroît, de recourir à des emprunts gagés sur les redevances à venir.

Les Etats rencontrent enfin des difficultés pour l'embauche suffisante et en temps opportun des personnels nécessaires.

Ces contraintes financières sur la planification sont d'autant plus irritantes qu'il ne s'agit que de problèmes de structure administrative et non de financement proprement dit, puisque toutes les dépenses afférentes au contrôle de la circulation aérienne sont, in fine, couvertes intégralement par des redevances appropriées.

Certaines insuffisances et lenteurs d'investissement qui découlent de ces contraintes artificielles, provoquent des points noirs dont l'effet se répercute sur la capacité de l'ensemble du système.

Dans ces conditions, il apparaîtrait souhaitable que les maillons les plus faibles relevant de cette cause spécifique fassent aussi l'objet d'un recensement systématique.

Il serait ainsi judicieux que soit constitué un fonds européen pour procurer les financements complémentaires aux Etats qui ne s'estimeraient pas en mesure de procéder rapidement aux améliorations indispensables ; un tel fonds pourrait être alimenté par des redevances communautaires spécifiques à la charge des compagnies aériennes, comme le sont les redevances nationales.

Par ailleurs, le coût de l'ensemble du système, de même que la rapidité de mise en place de certains matériels, pourraient avantageusement bénéficier, dans certains cas, de recherche/développement coordonnés et d'achats groupés. Une telle politique aurait pour avantage complémentaire d'inciter à la formation de consortiums européens susceptibles d'affermir la position déjà enviable de certains industriels européens dans ce domaine particulier ; elle ne pourrait cependant porter que sur les matériels normalisés (notamment les radars et leurs systèmes de traitement de l'information directement associés, les moyens de navigation et de communication, et certaines visualisations), à l'exclusion des éléments très liés aux spécificités de l'exploitation opérationnelle de chaque pays.

Le cas des calculateurs et de leurs logiciels mérite un examen particulier en raison de la très grande disparité des systèmes nationaux actuels. Il serait souhaitable qu'à l'occasion de chaque renouvellement de matériels, des directives européennes puissent

orienter le choix des calculateurs ou pour le moins de leurs systèmes d'exploitation et des langages évolués utilisés, de façon à préparer l'avenir et permettre notamment l'échange ultérieur de parties croissantes de logiciels

Certains logiciels, notamment de traitement de l'information Radar ou de plans de vol mériteraient aussi d'être normalisés à cette occasion. Il faut cependant se garder de l'espoir utopique de pouvoir remplacer, dans un avenir proche, l'essentiel des logiciels propres aux systèmes nationaux par un logiciel commun, tant sont indissociables les spécificités opérationnelles de ces systèmes et les programmes des logiciels; de surcroît, les logiciels en service ont été profondément influencés non seulement par les différences de spécifications opérationnelles, mais aussi par les processus différents qui ont présidé à leur élaboration.

L'objectif de pouvoir procéder à l'unification des calculateurs et des logiciels du système européen ne peut donc qu'être reporté à un avenir lointain ; il implique une unification préalable de la conception même du système futur.

LA CONCEPTION DU SYSTÈME

Ce problème recèle à son tour de très grandes difficultés.

Il serait en effet impensable qu'un système de contrôle, et notamment la définition des outils de travail et des processus de contrôle, puissent être conçus sans la participation active des contrôleurs eux-mêmes et sans la prise en compte de leurs réflexes acquis ; or, on a montré que ces derniers manquent manifestement d'uniformité.

Il n'en résulte pas pour autant que le système européen soit condamné à rester figé pour toujours dans des particularismes et dans un conservatisme dénués de justification. Mais il en résulte sans aucun doute que les évolutions souhaitables et notamment l'unification du système, ne pourront qu'être progressives: il faut rejeter comme irréaliste l'espoir de pouvoir remplacer le système diversifié actuel par un nouveau système conçu a priori dans le silence des cabinets et des bureaux d'experts ou résultant des compromis «caméliens» (3) de commissions et de comités pan-européens et a fortiori de solutions venues d'ailleurs.

Voilà sans doute bien un domaine où la recherche du «plus petit commun multiple» des systèmes existants pourrait bien se révéler comme le plus grand diviseur des énergies et de l'efficacité.

De grands espoirs peuvent en revanche être placés dans une future et lointaine génération de systèmes de conception nouvelle et commune, dans lequel les automatismes commenceraient à intervenir directement et significativement dans les processus de contrôle proprement dit.

On rappellera que dans les systèmes en service actuellement, ou en cours de modernisation, le traitement automatique ne porte que sur la transmission, la corrélation, la mise en forme et la visualisation optimales des informations pertinentes à chaque instant pour l'exercice du contrôle, à l'exclusion de toute participation à la prise de décision, c'est-à-dire au contrôle proprement dit.

C'est donc cette phase lointaine qu'il faut dès maintenant préparer en commun. Il en est encore temps, car aucune réalisation pratique en Europe, ou ailleurs, n'a encore permis de démontrer sa faisabilité, ni même sa contribution certaine à l'augmentation de la capacité effective de l'espace aérien.

La conception en commun du système hybride contrôleur/calculateur pour le début du siècle prochain devrait donc constituer le grand projet mobilisateur de l'imagination, de la créativité et de l'énergie de l'Europe du contrôle de la circulation aérienne.

Fort heureusement, il existe d'ores et déjà en Europe une importante activité de recherche et d'expérimentation, coordonnée et fédérée dans le cadre du projet PHARE d'Eurocontrol et de la CEE , l'Europe n'a rien à envier en ce domaine aux autres pays développés, même à ceux qui ont le privilège d'exploiter un système portant sur un très grand volume d'espace géré par une autorité unique.

Cet effort porte notamment sur des visualisations nouvelles plus «intelligentes» susceptibles de faciliter le travail des contrôleurs et leur dialogue avec le calculateur (cf. en particulier le projet PHIDIAS du CENA en France).

Des moyens technologiques nouveaux sont désormais disponibles (Radar mono-impulsions, liaisons codées air/sol dites «Radar mode S», calculateurs de navigation des avions dits FMS) A eux seuls, ils ne permettront pas l'accroissement de la capacité de l'espace, si on ne parvient pas à les utiliser dans un cadre d'automatisation du contrôle plus avancée.

Pour y parvenir, il est nécessaire de bien connaître au préalable la nature profonde des mécanismes de saturation de l'espace.

Les nouvelles phases d'automatisation vont imposer de pénétrer dans le domaine encore peu défriché, et d'une immense complexité, de la coopération de l'homme et des calculateurs dans des processus de décision en temps réel.

Il en résulte un problème «de la poule et de l'œuf»: nul ne se presse pour rendre obligatoires ces nouveaux moyens (modes S, FMS) tant que le système de contrôle n'aura pas été automatisé plus en profondeur ; inversement l'automatisation de certains processus de contrôle continuera à marquer le pas tant qu'il n'aura pas été prouvé que des améliorations de capacité pourront effectivement en résulter.

Or, chacun de ces processus demande cinq à dix ans. Pour sortir de ce cercle vicieux, il devient impératif à la fois de rendre obligatoires à court terme dans les espaces congestionnés les nouveaux équipements au sol et à bord et de lancer d'urgence un projet d'automatisation susceptible d'accroître la capacité de l'espace avec les moyens actuels d'abord, pour ouvrir ensuite la voie à des progrès encore plus substantiels dès lors que ces équipements se généraliseront.

LA CAPACITÉ DE L'ESPACE AÉRIEN EUROPÉEN : CONCLUSION

La capacité du système de contrôle européen recèle encore de bonnes potentialités de progrès pouvant résulter des réductions de ses disparités d'exploitation, de planification et de conception.

A elles seules cependant, ces réserves et le rythme de leur mobilisation ne seront certainement pas à la hauteur des besoins nouveaux de capacité qui résulteront de la croissance prévisible du trafic européen.

Les limitations de capacité proviennent en effet de ce que chaque avion et chaque paire d'avions en conflit entre eux, génèrent un flux de tâches dont l'ensemble ne doit à aucun moment dépasser la capacité de traitement que les hommes peuvent assumer.

Parmi ces tâches, figurent non seulement des actions de nature intellectuelle ou matérielle, mais aussi des mémorisations dont l'ensemble ne doit jamais outrepasser les aptitudes du cerveau humain dans le bref temps imparti dont il dispose pour les accomplir.

Au flux d'avions est ainsi associé un flux d'information qui peut lui-même dépasser les limites des possibilités d'appréhension, de mémorisation et de traitement du cerveau humain; en ce sens, il est loin d'être évident que l'accroissement considérable de la richesse des informations qui vont devenir disponibles grâce au mode S soit en lui-même porteur de progrès. Sans l'assistance des calculateurs, ces informations complémentaires pourraient au mieux être mal utilisées et, au pire, contribuer à saturer encore plus les contrôleurs.

LE MÉCANISME DE LA SATURATION DE L'ESPACE

L'ACCUMULATION DES TÂCHES

Lorsque le contrôleur est soumis à une accumulation de tâches qui approche de sa capacité maximale, et a fortiori s'il anticipe que le volume de tâches va augmenter dans les moments qui vont suivre, il tend tout naturellement à accroître ses marges de sécurité et donc à diminuer la capacité effective de l'espace qu'il contrôle

Il cherche en effet à se prémunir à tout prix contre l'éventualité d'avoir à faire face à des accumulations de tâches qu'il ne pourrait pas assumer; il craint aussi que, dans de telles conditions, ses performances de mémorisation soient amoindries et que, en conséquence, les risques s'accroissent d'oublier de traiter certaines tâches et notamment des conflits potentiels non encore résolus entre paires d'avions.

Ce processus explique la brutalité du phénomène de saturation dès lors que le nombre d'avions sur un secteur atteint, ou risque d'atteindre, une limite maximale.

LES «FILETS DE SAUVEGARDE» . LE MAILLON MANQUANT

On comprend, dans ces conditions, l'intérêt qui s'attache, tant pour des raisons de sécurité que de capacité, à la mise en œuvre de «filets de sauvegarde».

Il existe actuellement deux dispositifs de cette nature en amont et en aval du contrôle proprement dit

En amont, on a déjà évoqué ci-dessus l'impact positif sur la capacité, de la régulation des flux d'avions qui prémunit le contrôleur contre un flux de trafic dépassant ses aptitudes à l'assumer en toute sécurité.

En aval, il a été mis en service depuis plusieurs années un «filet de sauvegarde Radar» qui ne constitue pas en soi une aide au contrôle (et qui ne doit pas être utilisé comme telle) mais qui déclenche une alarme ultime avant qu'une collision due à une défaillance du contrôle ne risque de se produire effectivement, et avec un préavis juste nécessaire pour une intervention d'urgence.

Ces deux dispositifs participent donc directement à l'accroissement de la sécurité et indirectement à celui de la capacité par la «sécurisation» des contrôleurs.

Entre ces deux filets de sauvegarde en amont et en aval, il existe un maillon manquant, il s'agirait d'un «filet de sauvegarde» susceptible de prémunir le contrôleur contre l'accumulation excessive à court terme de tâches susceptible, faute d'action par lui-même et en temps utile, de le placer ultérieurement dans une situation inconfortable, voire dangereuse.

Mais, contrairement aux deux «filets» amont et aval, un tel dispositif pourrait non seulement générer des alarmes, mais participer d'une manière directe au contrôle proprement dit dans une de ses composantes importantes: l' ordonnancement des tâches.

Telle est l'idée directrice du projet dont les grandes lignes seront décrites dans la deuxième partie et qui devrait à la fois constituer une aide au contrôle et une sécurisation complémentaire des contrôleurs.

La généralisation des «filets de sauvegarde» sur tous les «filtres» successifs de contrôle (4) aurait ainsi pour effet de transformer l'ensemble du processus de contrôle, actuellement en «boucle ouverte», en un processus plus stable, plus efficace et plus sûr, en «boucle fermée».

LE MUR DE LA CAPACITÉ

On devrait pouvoir en escompter une augmentation effective de la capacité de chaque secteur et donc de l'espace

Dans le cas du contrôle tel qu'il est exercé actuellement, le contrôleur est amené à prendre des décisions «stratégiques» qui anticipent un certain nombre de décisions qu'il sera amené à prendre ultérieurement en fonction de l'évolution de la situation. Le système est stabilisé par le contrôleur lui-même: il est le seul maître du jeu.

Il en serait de même dans le cas d'un contrôle supposé être entièrement automatique et dans lequel le seul «libre arbitre» serait celui du calculateur. Cependant, même si un tel projet était réalisable du point de vue technique, opérationnel, psychologique et social, il n'en resterait pas moins nécessaire d'assurer une longue transition avec le système manuel

Le problème du passage par un système hybride homme/calculateur est donc incontournable.

Or, tout système dans lequel le calculateur sera amené à participer à l'élaboration des décisions soulève un problème redoutable: pour que l'assistance à la décision fournie par le calculateur soit efficace, il faudrait que le calculateur soit informé non seulement des choix retenus par le contrôleur parmi les propositions qu'il lui présente, mais aussi qu'il puisse anticiper l'ensemble des choix que le contrôleur effectuera ultérieurement au fur et à mesure de l'évolution de la situation.

En effet, tout choix nouveau peut impliquer une modification de la situation d'ensemble et obliger ainsi de revenir sur des choix antérieurement effectués. Le système a donc toutes les chances de «pomper», c'est-à-dire de ne pas trouver sa stabilité en raison de la cohabitation de deux libres arbitres (5), même si c'est l'un d'eux (l'homme) qui prend seul les décisions finales.

La transition du système actuel vers un système où l'homme sera assisté dans ses décisions par la machine se heurte donc à un phénomène, instable par sa nature même, qu'on dénommera le «mur de la capacité», par analogie aux changements brusques qui se manifestent lors du passage du «mur du son», à la différence près cependant que le système de contrôle devra demeurer durablement dans une zone de transition.

La faisabilité d'un système assurant cette transition d'une manière autostable et susceptible d'évolution progressive vers une automatisation croissante n'est pas encore démontrée; il n'est pas possible non plus à ce stade de prévoir l'ampleur de sa contribution à l'accroissement de la capacité effective des secteurs de contrôle.

Le projet, dont on trouvera les grandes lignes dans la deuxième partie de cette contribution, n'échappe pas à cette difficulté conceptuelle et à ces incertitudes ; il se caractérise cependant par le fait qu'il propose une stratégie et des moyens spécifiquement imaginés pour tenter de les surmonter.

DEUXIÈME PARTIE : UN PROJET POUR LE PASSAGE DU MUR DE LA CAPACITÉ

LA SPECIFICITÉ DU PROJET

LES BASES THÉORIQUES

Les bases d'un tel projet sont constituées par un ensemble théorique déjà cité ci-dessus en référence

Les résultats encourageants des premières réalisations effectuées sur ces fondements à l'Ecole Nationale de l'Aviation Civile (ENAC) par Leroux et Genthon (1986/87) puis Leroux et Alliot (1987/88) permettent désormais d'envisager des prolongements pratiques et de formuler les grandes lignes d'un projet précis.

LA SPÉCIFICITÉ DU PROJET

Ce projet se caractérise par l'ensemble des caractéristiques suivantes :

a) Le contrôleur reste seul responsable du contrôle

b) Le contrôleur est seul maître de toutes ses actions, y compris de toute manipulation (de clavier ou autre) qui ne saurait donc à aucun moment lui être imposée (donc pas de tâches supplémentaires induites par les processus liés à l'automatisation).

c) Le calculateur effectue toutes les opérations de contrôle en parallèle avec le contrôleur mais indépendamment de celui-ci. Le processus automatique n'est qu'un processus latent en ce sens que, contrairement au contrôleur, le calculateur ne passe jamais lui-même à l'action, ni n'impose ses solutions

d) Le processus automatique ne présente au contrôleur le résultat de son activité parallèle que :

- pour lui suggérer une action spécifique de contrôle qu'il juge immédiatement opportune ou urgente.
- pour lui présenter une situation d'ensemble fondée sur des informations agrégées élaborées par le «processus de contrôle automatique parallèle» et correspondant, à chaque moment et d'une manière aussi réaliste que possible, aux «agrégats d'informations» que le contrôleur construit lui-même par ses propres processus intellectuels et qu'il «balaie» en permanence par la pensée.

e) Toute nouvelle visualisation additionnelle doit obéir au double critère suivant :

- elle doit être gérée automatiquement par le calculateur,
- elle ne doit présenter aucune discontinuité de visualisation sauf lors de l'affichage de nouvelles informations ayant un caractère directement pertinent pour l'exécution du contrôle proprement dit.

f) Afin de faciliter le dialogue contrôleur/calculateur, une telle visualisation cathodique doit servir de support pour des échanges d'information avec le calculateur (désignation tactile ou autre)

Pour pouvoir décrire un tel projet, il est nécessaire de rappeler au préalable les grandes classes de tâches qui sont exécutées par les contrôleurs.

g) Ce projet n'exclut en aucune manière la mise en œuvre d'autres visualisations qui remodèleraient la position de contrôle classique actuelle, mais qui ne sont pas en tant que telles indispensables à la mise en œuvre de la présente théorie.

LE TRAVAIL DES CONTRÔLEURS

CLASSIFICATION DES TÂCHES

Les tâches élémentaires pour l'exécution du contrôle peuvent être recensées de la manière suivante :

a) recherche des nouveaux conflits.

b) «évaluation» de ces conflits dans le cadre de l'ensemble des «conflits en cours» déjà reconnus,

c) décision :

- c1) soit de résoudre immédiatement le conflit,

- c2) soit de différer sa résolution et d'ajouter ce nouveau conflit à l'ensemble des «conflits en cours»,

d) surveillance de l'évolution de chacun des «conflits en cours» en vue :

- d1) de s'assurer qu'aucune situation dangereuse n'est en train de se développer,

- d2) de décider pour chacun de ces conflits en cours le moment optimal pour procéder à sa résolution,

e) choix de la solution et exécution de la résolution des conflits au moment jugé opportun à la suite des processus c1 ou d2,

f) coordination au moment opportun avec les autres secteurs de contrôle.

Chacune des ces tâches élémentaires ressortit à l'une des quatre classes suivantes :

- recherche des conflits (a),

- suivi des conflits (d1),

- ordonnancement des tâches (b,c,d2,f),

- exécution des actes de contrôle proprement dits (e).

Les tâches des trois premières classes sont invisibles par un observateur : elles se déroulent intégralement «dans la tête du contrôleur» et font partie de son «savoir-faire»; les processus intellectuels sous-jacents sont extrêmement complexes et donc difficiles à expliciter et à enseigner a priori.

RECHERCHE DES CONFLITS

La recherche des conflits potentiels relève en principe de calculs mathématiques portant sur l'extrapolation des trajectoires puis sur leur comparaison deux à deux. Les calculs à effectuer sont si nombreux et sont d'une complexité telle qu'ils seraient inaccessibles au cerveau humain travaillant en temps réel et en calcul mental.

Le contrôleur est donc amené à procéder d'une manière heuristique ; la pertinence de ses évaluations et le caractère optimum de ses appréciations sont ainsi entâchés d'un flou d'autant plus important qu'il ne peut évaluer que d'une manière approximative la valeur actuelle de certains paramètres (notamment la vitesse des avions et leur taux d'évolution dans le plan vertical) et a fortiori les aléas qui affecteront la conduite de la suite des vols.

Certaines déclarations de «non conflit» ne peuvent donc pas être considérées comme absolument sûres et définitives.

Elles s'effectuent implicitement, en ne tenant compte que d'une partie de la courbe de distribution statistique des erreurs et aléas et en négligeant les «queues de distribution» au-delà d'une certaine probabilité

Une telle simplification est inéluctable , elle impose en contrepartie une «surveillance» de l'évolution de la situation d'autant plus vigilante que les marges de séparation évaluées s'approchent plus de la limite admissible. Cette surveillance constitue en soi une tâche spécifique et provoque une inquiétude latente ; elle alourdit le travail du contrôleur, et nuit à sa disponibilité d'esprit, un oubli peut donner lieu à incident ou accident.

Pour limiter l'ampleur des combinaisons d'événements à prendre en considération, le calculateur devra aussi procéder d'une manière similaire en «normalisant» les trajectoires pour ne considérer dans un premier stade d'analyse que les événements dont la probabilité d'occurrence dépasse un certain seuil et en éliminant ceux qui pourraient résulter de la conjonction d'évènements peu probables (6).

Mais, que la déclaration de «non conflit» soit humaine ou automatique, le calculateur est en mesure de procéder ensuite à la surveillance permanente de l'évolution effective de la situation , lorsqu'il est amené à réviser un jugement initial, il peut, en temps opportun, attirer l'attention du contrôleur sur le développement d'une situation critique. Il en résultera non seulement un allégement important de la tâche des contrôleurs mais aussi une «sécurisation» qui lui conférera une meilleure disponibilité d'esprit pour les autres tâches. Cette fonction devrait donc concourir de ce double point de vue à l'accroissement de la capacité.

RÉSOLUTION ET SUIVI DES CONFLITS

Lorsqu'un conflit potentiel est détecté, notamment à l'entrée d'un avion dans un secteur, le contrôleur peut procéder immédiatement à sa résolution en demandant au pilote la modification de certains éléments de sa trajectoire (niveau autorisé, vitesse, route...).

Il peut aussi laisser provisoirement les choses en l'état. Il doit alors suivre l'évolution de la situation ; comme dans le cas évoqué ci-dessus, il va de soi que le calculateur peut effectuer automatiquement la tâche de surveillance correspondante.

ORDONNANCEMENT DES TÂCHES

On appellera «ordonnancement des tâches» l'opération qui consiste à prendre en temps réel les dispositions visant à ce que l'enchaînement des tâches à venir se présente de telle manière qu'à aucun moment elles ne risquent de saturer la «capacité» des contrôleurs.

C'est sans aucun doute la classe de tâches qui est la plus essentielle et la plus délicate.

Cette classe de tâches est effectuée par les contrôleurs de manière heuristique ; ses modalités d'exécution n'ont pas encore fait l'objet d'une analyse suffisamment approfondie pour permettre d'expliciter l'ensemble des règles qu'elles mettent en œuvre.

Il y a tout lieu de penser cependant que chaque conflit nouveau et chaque conflit à chaque stade de son évolution est affecté implicitement par le contrôleur d'un «indice» - que l'on appellera son «poids» - qui caractérise ce conflit en lui-même, ainsi que ce conflit dans le contexte de tous les autres «conflits en cours». Ce «poids» reflète d'une manière composite :

- le moment où risque de se produire le conflit effectif,

- la somme de l'attention requise pour la surveillance des autres paires d'avions non encore séparés jusqu'au moment où le contrôleur décidera de résoudre le conflit particul...,
- la réflexion imposée quant au choix du moment et de la solution pour la résolution du conflit,
- la difficulté de l'action d'évitement.

Ce «poids» doit aussi tenir compte du temps requis pour l'accomplissement de ces tâches, ainsi que de l'accroissement de la difficulté due au chevauchement dans le temps de l'ensemble des tâches élémentaires impliquées par l'ensemble des «conflits en cours» ; il résulte surtout de la probabilité que se développe une situation indésirable, par exemple telle que le contrôleur Radar aurait par la suite plusieurs évitements Radar à effectuer simultanément.

Le choix optimal du moment et du mode de résolution de chacun des conflits relève donc bien d'un problème d'ordonnancement des tâches du contrôleur.

De surcroît, le problème est compliqué par le fait que dans le système actuel, cet ordonnancement est effectué à la fois par les deux contrôleurs constituant l'équipe : le contrôleur dit «organique» (ou stratégique, ou de planification. .) et le contrôleur dit «tactique» (contrôleur Radar).

Cette articulation interne au secteur est particulièrement délicate , pour être menée à bien, elle implique la cohérence d'une équipe soudée ayant l'habitude de coordonner son travail d'une manière organisée mais tacite.

Il s'agit du problème crucial dont, à notre avis, dépend l'évolution du contrôle . on peut s'en convaincre en rappelant que la saturation du contrôle ne résulte pas tant de la saturation effective de l'espace que de la saturation par des «pointes d'accumulation de tâches» (et notamment par la probabilité d'avoir à mener simultanément deux évitements Radar) qui dépassent ce que peut exécuter en un temps donné chaque contrôleur ou chaque équipe de contrôle.

Le bon ordonnancement de ces tâches dans le temps constitue donc bien le problème central auquel il convient de s'attaquer.

C'est ainsi que la capacité et la sécurité d'un secteur de contrôle sont directement fonction pour chacun des conflits :

- du choix du moment optimal pour entreprendre sa résolution,
- du choix de la solution optimale, d'une manière telle que soit minimisée la probabilité d'accumulation pernicieuse de tâches à un moment futur quelconque et que soit, si possible, lissée dans le temps la «demande d'attention» des contrôleurs.

C'est en priorité l'assistance à cet ordonnancement des tâches que le projet se propose d'automatiser ainsi que le «filet de sauvegarde» associé, d'où le nom du projet FREGATE (Filtre de Régulation Ergonomique de la Gestion [des secteurs] et d'Anticipation des Tâches Excessives). Ces fonctions nouvelles sont complexes et méritent un examen approfondi.

LA RÉALISATION DU PROJET

LES FONCTIONS

Si le projet implique le fonctionnement en parallèle, et avec un minimum d'interaction, du contrôleur et du calculateur, il n'en reste pas moins que ces deux processus doivent se rejoindre chaque fois que le calculateur est susceptible de rendre un service effectif aux contrôleurs.

Il ressort de l'analyse ci-dessus que l'interface entre le contrôleur qui agit et le programme parallèle de contrôle automatique qui est latent, doit s'effectuer sous deux formes complémentaires :

- l'aide à l'ordonnancement des tâches : AOT,
- l'alerte en cas de prévision par le calculateur du développement d'une situation future indésirable du point de vue de l'accumulation des tâches et à fortiori d'une situation dangereuse : c'est-à-dire un «filet intégral de sauvegarde tactique et organique» (FISTO)

On notera que ce FISTO constitue une généralisation de la notion actuelle de «filet de sauvegarde Radar» mais, alors que ce dernier n'intervient que pour alarmer en cas de danger à très court terme, le FISTO peut alerter pour prévenir de toute situation indésirable du point de vue de l'accumulation des tâches qui risque de survenir à tout moment dans le futur. Contrairement au cas du «filet de sauvegarde Radar» qui impose une action immédiate et rapide du contrôleur, le FISTO n'est nullement impératif, (on montrera qu'il devra être conçu de telle manière qu'il soit de l'intérêt du contrôleur d'en suivre les avis).

A cette fin, il faut que le «programme de contrôle automatique latent» débouche sur une visualisation qui, à la fois, permette l'assistance à l'ordonnancement des tâches et procure le filet de sauvegarde associé. Mais il convient aussi que cette visualisation soit telle que, par simple désignation tactile, elle puisse remplir deux fonctions complémentaires lorsqu'un contrôleur utilise effectivement l'AOT ou décide de répondre aux sollicitations du FISTO, à savoir :

- une aide au choix des solutions (ACS),
- une aide à l'entrée de la solution choisie (AES).

A partir de ces objectifs et de ces principes, il devient possible d'imaginer une visualisation nouvelle susceptible de servir facilement d'interface homme/calculateur pour l'exécution des fonctions qui viennent d'être analysées.

UNE VISUALISATION ORIGINALE

Description de la visualisation

Les besoins recensés ont conduit à imaginer une visualisation d'une nature originale

Les quatre fonctions AOT, FISTO, ACS, AES peuvent s'articuler autour d'une visualisation unique au profit des deux contrôleurs organique et tactique, ou mieux sur deux visualisations identiques placées respectivement à la vue et à la portée de chacun de ces deux contrôleurs.

Elle est de type analogique comme les écrans radar mais à la différence de ceux-ci, chaque point affiché (et l'étiquette qui lui est attachée) ne porte pas sur un avion individuel mais représente toujours une paire d'avions en conflit potentiel. Ces conflits sont classés en fonction du moment auquel ils interviendront si aucune mesure n'était prise entre-temps.

Cette visualisation se présenterait de la manière suivante :

a) Sur un écran rectangulaire horizontal est tracée une ligne médiane horizontale ayant une origine O et servant de gauche à droite d'axe des temps.

b) Sur cet axe sont figurés des points représentant les «conflits d'avions deux à deux non encore résolus». Un avion impliqué dans plusieurs conflits est représenté par plusieurs points

c) L'élongation de ces points sur l'axe des temps représente la durée qui va s'écouler entre le moment actuel et le moment où aurait lieu le conflit effectif si aucune mesure de résolution n'était prise d'ici là.

d) Chaque point est affecté d'une numéro d'ordre et est relié à une étiquette par un tiret de rappel permettant d'identifier l'étiquette correspondante.

Sous des formes optimales à expérimenter (chiffres, couleurs, symbole, épaisseur de tiret, etc), le point et/ou l'étiquette donnent une indication du «poids» du conflit et de son urgence (au sens défini ci-dessus).

e) Chaque étiquette comprend l'indicatif des deux avions concernés.

f) L'écran comprend trois parties :

- la partie centrale utilisée de la manière décrite ci-dessus,
- une partie haute dans laquelle le calculateur trace l'histogramme de la sommation à chaque instant futur de la somme des tâches «pondérées» par leur «poids» respectif qui va découler de la gestion et de la résolution de tous les «conflits en cours»,
- une partie basse servant de clavier d'entrée tactile (ou autre) adapté à chaque instant par le calculateur à la tâche probable que le contrôleur va être amené à effectuer

Par ailleurs, l'étiquette de l'écran Radar classique comprendra, outre les données actuelles, le rappel des numéros des conflits non encore résolus dans lesquels est impliqué l'avion considéré.

Fonctions de la visualisation

a) AOT : le seul examen de la visualisation permet au contrôleur de prévoir sa charge future de travail, de prendre conscience en temps opportun de tout risque de saturation, de ses causes et des mesures à prendre.

b) FISTO . si le contrôleur n'est pas assez attentif à certaines éventualités d'accumulation potentielle des tâches, ou ne s'est pas montré en mesure d'optimiser sa charge future de travail, le calculateur :

- déclenchera une alerte à partir de certains seuils de l'histogramme des tâches futures;
- désignera le (ou les conflits) qu'il estime approprié de résoudre dès maintenant. Cette désignation s'effectuera par exemple en faisant «flasher» le point représentatif du conflit sur l'écran décrit ci-dessus (ainsi que les pistes correspondantes sur l'écran Radar).

On rappelle que la présentation de ces suggestions n'implique pas obligatoirement une action du contrôleur (qui reste libre de sa stratégie) mais constituera sans aucun doute pour lui une aide précieuse. Toutefois, lorsque le calculateur estimera que la situation risque de se transformer en une situation dangereuse à terme, l'alerte pourra se transformer en alarme et devenir impérative

Ce nouveau dispositif proposé n'est pas limité aux quelques secondes qui précèdent un conflit effectif comme dans le cas du «filet de sauvegarde Radar» mais s'étend en un continuum à toute période précédant un conflit potentiel donné. Il constituera un outil fondamental du contrôle proprement dit.

On notera aussi que cette extension de la notion de «filet de sauvegarde» permet de protéger le contrôleur contre toute faute de jugement grave et notamment contre tout oubli d'une situation conflictuelle précédemment détectée, mais occultée par l'exécution d'autres tâches qui retiennent toute son attention.

c) ACS : lorsque le contrôleur décide d'agir pour résoudre un conflit donné, soit de sa propre initiative, soit parce qu'il reconnaît le bien-fondé d'une alerte ou d'une alarme FISTO, il lui suffira de désigner l'étiquette du conflit considéré.

Le calculateur fera alors apparaître dans la partie basse de la visualisation un choix de solutions par ordre de préférence, c'est-à-dire que les solutions seront classées d'une manière d'autant plus prioritaire que leur mise en œuvre lisse mieux la tâche globale moyenne future du contrôleur ou, en cas de besoin, minimise les pointes futures d'accumulation de ces tâches.

Toutes les solutions raisonnables (changements ou limitations de niveau, changement de route, changement de vitesse longitudinale ou verticale, évitement Radar immédiat ou différé, action sur un des avions ou sur l'autre...) peuvent être affichées ; certaines d'entre elles devront cependant être affectées d'un symbole d'interdiction en raison des nouveaux conflits qu'elles sont susceptibles d'induire avec d'autres avions, ou du fait qu'elles sont de nature à déstabiliser l'organisation du trafic telle qu'elle a été précédemment prévue. Parmi ces solutions pourra figurer, dès que cela sera techniquement possible (mode S + FMS), la prise en compte directe de calculateur Sol à calculateur Air du guidage des avions.

Si le contrôleur choisit l'évitement Radar différé, le calculateur surveillera la paire d'avions considérée et rappellera au contrôleur sous forme d'alarme ce conflit non résolu, suffisamment à l'avance pour lui permettre d'assurer en toute sérénité l'évitement Radar. Le

contrôleur est ainsi placé à l'abri de la crainte d'oublier un des conflits Radar, au cas où son attention et ses capacités seraient saturées par l'exécution d'autres tâches.

L'ordre de préférence du classement, de même que les solutions «interdites» peuvent tenir compte de la présence d'avions évoluant actuellement dans un autre secteur, c'est-à-dire que l'optimisation de l'ordonnancement et du choix des solutions couvre un domaine plus vaste que celui auquel s'intéressent les contrôleurs d'un secteur considéré.

C'est ainsi que, de proche en proche, la notion de «secteur de contrôle» pourra perdre sa rigidité actuelle et que le processus de segmentation croissante de l'espace, seule solution actuelle (aux rendements d'ailleurs décroissants) à l'augmentation de la capacité de l'espace, pourra être renversé et, en tout cas, nuira de moins en moins à l'optimisation de l'utilisation globale de l'espace.

d) AES : Lorsqu'un contrôleur décide d'agir sur un conflit, il disposera ainsi d'un choix de solutions. Il lui suffira de désigner du doigt la solution sélectionnée (en principe, mais non obligatoirement celle qui est présentée la première par le calculateur). Cette action aura pour effet d'informer le calculateur du choix du contrôleur (et ceci est essentiel pour assurer la stabilité de tout le processus automatique).

   Mais cette action très simple pourrait avoir de surcroît pour effet :

   - de faire envoyer ces ordres par liaison automatique de données (quand le système sera arrivé à ce stade de développement) puis d'assurer, par la suite, plus ou moins complètement et automatiquement, des manœuvres complexes par couplage du calculateur sol et des FMS des avions ;
   - de faire envoyer éventuellement les ordres correspondants par voix artificielle aux avions non équipés de liaisons codées automatiques;
   - d'informer automatiquement le système militaire associé des autorisations correspondantes, résolvant ainsi un problème lancinant et essentiel pour l'efficacité de la coordination civile/militaire.

e) ACI . la visualisation peut enfin servir d'aide aux processus de coordination intersecteurs et assurer leur «surveillance», à savoir

   - qu'elle peut servir de support intelligent à la coordination intersecteur en ce sens que toute prise de décision influant sur les conditions de transfert intersecteur peut faire l'objet d'un message au secteur suivant (si le contrôleur suit les propositions du calculateur, les conditions de transfert qui ont été élaborées par ce dernier de manière telle qu'elles soient acceptables par le secteur suivant);
   - qu'elle peut servir de «filet de sauvegarde de coordination» en attirant l'attention du contrôleur sur tout avion s'approchant des limites du secteur et n'ayant pas encore fait l'objet d'une coordination avec le secteur suivant.

f) En conclusion, on voit que la visualisation proposée constitue à la fois une aide directe au contrôleur pour l'ordonnancement de ses actions (dont il garde une complète maîtrise) ainsi qu'une protection efficace contre tout oubli ou distraction.

On peut espérer que le potentiel de l'équipe de contrôle sera largement accru par le «calculateur contrôleur virtuel automatique» (le troisième homme fictif) à l'image des pilotes qui peuvent se concentrer sur les tâches qu'ils sont les plus aptes à effectuer et garder toutes leurs potentialités à leur optimum, lorsqu'ils sont déchargés de toutes les tâches de surveillance et d'exécution, que les pilotes automatiques (voire le FMS) peuvent effectuer sans leur concours.

Cette visualisation serait aussi un instrument idéal de coordination entre le contrôleur «organique» et le contrôleur Radar : elle constitue en ce sens une aide précieuse à la coordination intrasecteur.

Cette visualisation serait aussi particulièrement précieuse au moment de la relève d'une équipe par une autre lors des changements de quart. elle synthétise en effet à tout instant la vie actuelle et future du secteur. C'est dire l'efficacité qu'on est en droit d'en espérer en situation normale, et l'état de détente des contrôleurs qu'on peut en escompter· ceux-ci seront non seulement assistés dans l'ordonnancement de leurs tâches, mais pourront se consacrer sans appréhension à la conduite d'une seule tâche à la fois (notamment la conduite d'un évitement Radar) sans être agressés, d'une manière permanente et latente, par la crainte d'oublier un autre conflit ou d'avoir laissé se développer une situation qui pourrait devenir difficile à maîtriser par la suite.

LA DIFFICULTÉ CONCEPTUELLE

On peut se demander dans quelle mesure ce projet, dont les lignes directrices ont été décrites ci-dessus, pourrait se révéler de nature à permettre de surmonter la difficulté conceptuelle fondamentale qui s'oppose au franchissement du «mur de la capacité».

Le calculateur ne peut tenir les visualisations à jour et ne peut proposer et classer les solutions que s'il est tenu informé de toutes les décisions effectivement retenues de proche en proche par les contrôleurs.

Mais il doit aussi pouvoir faire des hypothèses réalistes sur les décisions que le contrôleur sera amené à prendre dans le futur. Pour pouvoir réduire le champ du probable voire du possible, le calculateur devra lui-même connaître les «règles de l'art» du contrôle (les contrôleurs ne décident pas n'importe quoi!).

A cette fin, il faudra programmer le calculateur à l'image de l'expertise des contrôleurs et donc expliciter au préalable avec l'aide des contrôleurs eux-mêmes, l'ensemble complexe des règles qu'ils appliquent dans chacune des multiples situations qui risquent de se présenter, et des poids d'angoisse qu'ils attribuent à l'appréhension des tâches à venir.

Il y a tout lieu de penser que le développement de ces «programmes experts» soit possible, mais on sait déjà qu'ils comprendront un nombre considérable de cas et de «règles» (plusieurs milliers) et qu'ils ne pourront être perfectionnés que par un long processus auquel les contrôleurs doivent être intimement associés.

Les techniques dites de l'intelligence artificielle et les méthodes modernes d'explicitation des règles que les contrôleurs appliquent sans en avoir une conscience explicite, ouvrent des voies très encourageantes comme le montrent en particulier les premiers résultats des travaux de Leroux à l'ENAC.

LE PASSAGE DU MUR DE LA CAPACITÉ : LES BOUCLES VERTUEUSES

Plus et mieux le calculateur influencera les choix des contrôleurs, plus le «champ de l'incertain» pourra être réduit (l'expérience seule pourra montrer dans quelle mesure les choix préférentiels du calculateur seront effectivement retenus par les contrôleurs) et, en conséquence, plus les choix proposés seront réalistes et plus le système sera stable. Plus les contrôleurs estimeront pouvoir se laisser influencer par le calculateur, meilleure sera l'efficacité du calculateur et plus grande la satisfaction des contrôleurs.

Le projet recèle ainsi des «boucles vertueuses» qui devraient faciliter son introduction et permettre sa montée en puissance : tout accroissement d'efficacité élémentaire accroît le service rendu, qui lui-même tend à son tour à accroître la faisabilité et l'efficacité du projet dans son ensemble.

Il en est de même de l'information du calculateur des décisions effectivement retenues par les contrôleurs: les contrôleurs auront intérêt à faire connaître leurs décisions au calculateur pour rendre celui-ci plus efficace dans l'ensemble des services qu'il lui rend. C'est ainsi que, plus le calculateur sera efficace dans l'élaboration de choix, plus il y a de chance que ces choix soient suivis par le contrôleur.

Or l'entrée dans le calculateur de la solution choisie est conçue pour se faire d'une manière particulièrement aisée pour les contrôleurs, leur contribution se bornant à la désignation de la solution retenue parmi les choix proposés. Lorsqu'ils estimeront pouvoir ainsi informer le calculateur de leurs décisions, les contrôleurs seront largement gratifiés :

a) amélioration de la pertinence des solutions proposées par le calculateur,

b) suivi automatique des autorisations de contrôle,

c) surveillance automatique des conflits, alertes et alarmes FISTO,

d) coordination automatique intra- et inter-secteurs,

e) information des militaires, etc.

Il existe donc dans le système plusieurs «boucles vertueuses» qui devraient faciliter son introduction et son amélioration de manière itérative et qui feront que :

- plus les contrôleurs se serviront du système, plus ils seront directement gratifiés, donc plus le système sera efficace pour les contrôleurs,
- plus le système sera efficace pour les contrôleurs, plus ils s'en serviront.

La boucle a des chances de se boucler ! Tel est le pari du projet.

Il convient de noter que, si le processus d'analyse de la situation par le calculateur est «à l'image» de celle qu'effectue le contrôleur, le calculateur disposera de moyens plus puissants que ceux du contrôleur pour les traiter exhaustivement, ainsi que sur une zone dépassant celle d'un seul secteur individuel.

De son côté, la «surveillance» de la situation et notamment des conflits non encore résolus constitue un processus purement logico-mathématique exempt du flou heuristique des processus de décision.

UN PROJET MOBILISATEUR POUR LES ANNÉES 90

Ce projet constitue le prolongement pratique des objectifs et de la théorie esquissés dans l'annexe 7 du rapport des Sages (7). Il ne devrait pas rencontrer d'obstacles majeurs de nature psychologique ou sociale.

Si son processus itératif tenait ses promesses, il constituerait une voie possible d'augmentation de la capacité de l'espace (dont on sait qu'elle est essentiellement limitée par l'accumulation des tâches plutôt que par le manque d'espace proprement dit), de diminution des stress des contrôleurs et corrélativement d'augmentation de la sécurité globale

Il présente le mérite de permettre de se greffer sur le système actuel et notamment son logiciel et ses sorties (strip papier ou cathodique, visualisation Radar et digitatron). Il pourrait bénéficier cependant du remodelage de la position de contrôle par des visualisations plus intelligentes (cf. PHIDIAS par exemple).

Il ne nécessite pas impérativement de liaisons de données automatiques air/sol pour son introduction ; il se justifie en lui-même, mais le projet une fois implanté, permettrait de tirer tout le profit de cette novation technologique, notamment le couplage automatique aux FMS.

Cependant, ce projet, malgré son très grand dépouillement et sa simplicité apparente (notamment en ce qui concerne l'addition de la seule nouvelle visualisation), exige un logiciel aussi important et complexe qu'un logiciel qui assurerait le contrôle entièrement automatique.

Il suppose de surcroît non seulement une connaissance des tâches de contrôle dans leur aspect théorique, mais encore une expertise approfondie des règles de l'art et du comportement effectif des contrôleurs face à toutes les situations élémentaires (et aux combinaisons de situations) susceptibles de se présenter; les programmes experts à développer touchent notamment à l'évaluation des «poids» des différents types de conflits et des solutions envisageables.

Un tel projet ne peut donc aboutir que sous forme de projet mettant en œuvre toutes les composantes du contrôle : ingénieurs, contrôleurs, électroniciens ; en ce sens il est mobilisateur et générateur de cohérence et, on l'espère, d'enthousiasme interne. Il devrait être conduit comme un «grand projet».

La répartition des tâches élémentaires doit cependant être suffisamment souple pour ne pas neutraliser la créativité essentielle à la maîtrise d'un domaine encore largement dominé par l'incertain.

Il va de soi, comme l'expérience passée l'a toujours confirmé, que les études experts qui seront menées dans le cadre de ce projet présenteront un intérêt considérable pour bâtir parallèlement un instrument d'enseignement programmé (notamment d'enseignement de l'ordonnancement des tâches et d'élaboration et de choix de solutions), instrument qui devrait permettre des progrès substantiels dans les durées et les qualités de l'enseignement ainsi que dans l'adaptation des contrôleurs aussi bien au système actuel qu'au nouveau système. Les contrôleurs seraient ainsi instruits sur les mêmes bases que celles qui présideront à l'élaboration du système de l'avenir.

La formation et les études pour le développement du système iraient de pair et se valoriseraient mutuellement. L'unification à terme du système européen y trouverait son compte. On peut espérer que les études et recherches fédérées par Eurocontrol et par la CEE et qui se développent d'une manière très satisfaisante au sein des Etats européens, déboucheront en temps opportun sur la conception du système commun du début du siècle prochain.

L'auteur souhaite que les analyses qui précèdent puissent apporter leur contribution à la définition d'un fil conducteur commun à ces travaux et que ses propositions puissent être considérées comme une des formes possibles de leur aboutissement.

De son côté, l'Ecole nationale de l'aviation civile (ENAC) a déjà reçu un contrat du Centre d'étude de la navigation aérienne français (CENA) pour développer sous forme de maquettes probatoires certains éléments essentiels de ce projet.

La tâche sera longue et ardue, mais les enjeux sont essentiels.

NOTES

(1) Cf. «Pour une politique européenne du transport aérien». J. Villiers in *ITA Magazine* n° 56 et 57.

(2) Voir «L'intelligence artificielle dans le contrôle de la circulation aérienne». J. Villiers in *ITA Magazine* n° 43, mai/juin 1987.

(3) Ce néologisme un peu scabreux évoque les chameaux dont on dit qu'ils ne seraient qu'une version de cheval... conçue par un Comité !

(4) Pour la notion de «filtres» successifs, voir «La Méthode des filtres», J. Villiers in Revue de l'Institut de Navigation, n° 61, 1er trimestre 1968.

(5) Il va de soi que le «libre arbitre» du calculateur ne résulte que de celui du concepteur de son logiciel.

(1) Cf. «La méthode des filtres» in *Revue de l'Institut de Navigation*, n° 61 (1968), déjà citée.

(2) Rapport de la «Commission de réflexion sur le service public de la navigation aérienne», janvier 1985.

THE PILOT'S ASSOCIATE
Exploiting the Intelligent Advantage

by

Douglas L.Holmes
Lockheed Aeronautical Systems Company
Thousand Oaks, CA, United States

John P.Retelle, Jr
Lockheed Missiles and Space Company
4500 Park Granada Bvd, Calabasas CA 91399-0310, United States

The Pilot's Associate program has provided a series of technology demonstrations of the potential of integrating intelligent systems and artificial intelligence technology into modern avionics systems. The Defense Advanced Research Projects Agency and the United States Air Force have provided funding and program management to determine the potential increases in mission effectiveness from such a system The Pilot's Associate effort pursued by Lockheed and its partners has produced not only prototypes for advanced systems, but also new insights into the nature of the systems themselves as well as new approaches for quickly producing software for these systems The rapid prototyping methods that have been utilized have also provided the ultimate consumers - the pilots - with significant awareness of the operation of the Pilot's Associate, and with many opportunities to improve the requirements for such a system. This paper describes the evolution of Lockheed's Pilot's Associate System approach leading to the current system configuration Also described are some lessons learned from managing a large software development team assembled to produce an unprecedented system

Introduction

A steadily increasing emphasis on the military pilot's role as tactician is a consistent theme in the evolution of Western tactical aviation. This challenging role for these human combatants is being made even more difficult by extensions in the range and search volume of on-board sensors coupled with dramatically enhanced weapons and aircraft performance. The new aircraft system capabilities have provided individual pilots with the means and attendant responsibility to exercise effective control and pursue important tactical goals across a large portion of any potential aerial battlefield Yet these same advances in weapons system capabilities, matched to some degree by improvements in air and ground based enemy systems, create extraordinary demands on the skills of aircraft pilots as tacticians and threaten to overwhelm aircrews in the future.

The problem of coping with these demands on the pilot's powers of observation, planning and decision making, frequently summarized in the catchphrase "situation awareness", has long been recognized by the operational community as a central issue in assuring combat effectiveness. Yet the essential nature of this complex, context-dependent question has confounded efforts to complete a satisfactory analysis of the problem. Some early efforts to resolve the problem with avionic systems, displays and software and hardware enhancements to sensors have resulted in only marginal improvements, or may have actually worsened the situation. In 1986, the U.S. Defense Advanced Research Projects Agency (DARPA) initiated the Pilot's Associate program to investigate an artificial intelligence approach to the comprehensive problem[1, 3].

A Description of the DARPA Pilot's Associate Program

The Pilot's Associate program is one of several technology application efforts in DARPA's Strategic Computing Program. The technical goal of the Strategic Computing Program is the development of advanced computing technology, including intelligent systems, new processor architectures and speech and vision capabilities, to address critical military problems. The Pilot's Associate program was created to apply the technologies of real-time, cooperating knowledge-based systems for exploring the potential of artificial intelligence to improve mission effectiveness and survivability of advanced fighter aircraft.[1,3]

The Pilot's Associate program is planned as a five-year, two phase effort, administered for DARPA by the United States Air Force Wright Research and Development Center (WRDC). Following a pre-contract feasibility phase and demonstration (Demo 1) with numerous contractors, the first three-year contracted program phase called for two non-real-time demonstrations (Demo 2 and Demo 3) of a limited functionality Pilot's Associate system in a laboratory development environment. The second two-year phase will culminate in a system evaluation and demonstration (Demo 4) of a full-function, multi-mission Pilot's Associate in a real-time full mission simulation environment. Two independent development teams, being lead by Lockheed Aeronautical Systems Company and McDonnell Aircraft Company, have approached the implementation of the Pilot's Associate from different directions.[2] This paper describes the experience of the Lockheed team.

In March 1988, Lockheed conducted a very successful Demo 2 of a prototype Pilot's Associate that met or exceeded expectations for the system at that stage of development. The system at that time was narrowly focused on an air-to-air mission scenario; it ran at approximately six times real-time (a thirty minute mission was completed in roughly three actual hours of simulator time). This demonstration, viewed by over 200 pilots, scientists and engineers, was hosted in a laboratory cockpit simulation driven by one VAX and six Symbolics computers. It featured a real-time replay of a simulated two-ship air intercept mission against an enemy force of fighters and bombers, and a live laboratory run of the same mission to allow interaction and "what if..." speculations and system responses. Among other results, the demonstration con-

firmed the feasibility of producing cooperating knowledge-based avionic systems, and indicated that such systems can be engineered to run in real-time, and illustrated the operational utility of a Pilot's Associate.[1]

The second major demonstration of the first phase, Demo 3, was held in November 1989. It featured an enhanced prototype version of the Pilot's Associate performing in a new and more demanding "force protection-escort" mission, and a prototype version of the complementary Mission Support Tool for pre-mission planning. Demo 3, which was similar in format to Demo 2, demonstrated a maturing intelligent system with enhanced functionality. This faster prototype Pilot's Associate ran at approximately three times real-time and continued to foreshadow significant operational value for the eventual airborne system. The success of this demonstration motivated continued funding and support for the second phase of the program which is currently in progress. Phase two will result in a more robust, real-time prototype Pilot's Associate in early 1992. It includes a major demonstration, Demo 4, at the end of the phase, and an operational evaluation of the system in a full mission simulation facility by a cadre of current tactical pilots.

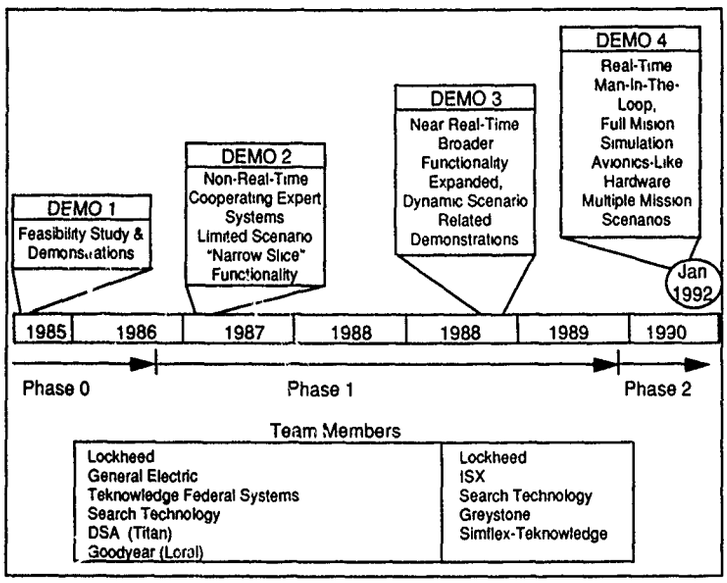


Figure 1: The Pilot's Associate Program

Lockheed's Pilot Associate

To date, the principal effort has been focused on the design and development of a series of evolving software prototypes of the Pilot's Associate which could eventually lead to a fieldable avionics system. In phase one, development was accomplished in a variety of LISP environments on Symbolics machines. In the current phase, the system is being developed in C++ and will run on specialized, advanced "avionics-like" hardware.

Lockheed has divided the Pilot's Associate system into five subsystems that roughly correspond to a top-level functional partitioning of the tactical air combat domain. This somewhat arbitrary but pragmatic division of responsibilities resulted in a loosely-coupled heterogeneous system in which each subsytem was distinctly implemented according to an approach considered appropriate to the particular subdomain.

A key aspect of this configuration is the adoption by the Lockheed team of a common planning language known as the Plan and Goal Dictionary. This language, which originated as a directed graph used in system design, allows the rapid promulgation of planning information, often in different formats, to multiple subsystems. Initially, a sixth subsystem, called the Mission Manager, existed to provide a global blackboard as a central repository for active plans and goals. As the Pilot's Associate matured, these functions were decentralized into the various subsystems and eventually, the mission manager subsystem disappeared entirely.

The five remaining subsystems are: the Pilot-Vehicle Interface, Situation Assessment, Systems Status, Tactics Planner, and Mission Planner. Two of the subsystems, Situation Assessment and Systems Status, fall into the functional category of "assessment", and two others, Tactics Planner, and Mission Planner may be categorized as "planning" while the fifth, Pilot-Vehicle Interface provides the human-centered focus necessary to support the pilot's decision-making requirements through an intelligent, intuitive interface.

Figure 2 illustrates this allocation of functions. The figure shows data coming to the system in the form of briefed information about the mission, threats and planned tactics, sensor data, aircraft state and status data, and inputs from the pilot's control devices. The output includes com-

mand and display control data to aircraft systems and information to be exported to cooperative platforms. The Pilot's Associate is designed to both filter the flow of this data to display only data relevant to the current mission, or required by the pilot, and, to add value to the data. The system function, then, is to reason about and report what the data means with respect to the mission plan, and to suggest and specialize response plans for situations as they arise.

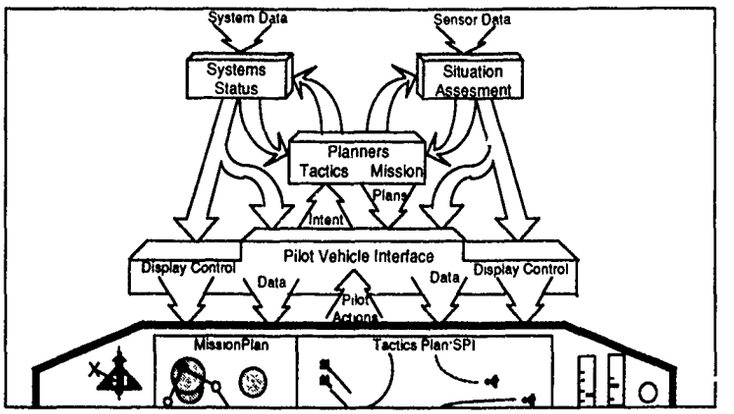


Figure 2: Pilot's Associate Concept

Provided below are brief descriptions of the five major subsystems of the Lockheed team's Pilot's Associate.

Pilot-Vehicle Interface

This subsystem is responsible for explicit and implicit communication with the pilot and interface to the cockpit. The design integrates interface management, an adaptive aider and an error monitor to accomplish both assessment and planning functions It evaluates pilot activities and mission events in order to interpret and "understand" the pilot's actions. The subsystem then responds by configuring displays and control devices, evaluating workload and performance, detecting errors, and adaptively aiding the pilot in the execution of his tasks to optimize his situation awareness and allow him to focus his attention on important or critical events Finally, the system provides a display that meets the expectations of the pilot in specific situations and that is sensitive to his personal preferences and techniques.

Pilot intent determination is an essential function of the Pilot-Vehicle Interface subsystem. By monitoring pilot actions as well as the mission context, the system is able to compare pilot actions or specific commands to a model of the pilot's plans and goals The subsystem can then determine, without tedious or distracting explicit input, the intentions of the pilot that will guide the behavior of the remainder of the Pilot's Associate subsystems.

System Status

This subsystem monitors and analyzes onboard systems to determine the current aircraft state and to evaluate current system capabilities. Any detected malfunctions anywhere in the aircraft are evaluated to determine the degree of degradation of the overall system capability and assess the impact of the degradation with respect to current and proposed mission plans. Where possible, the system generates remedial plans and coordinates them with plan generators in other subsystems. All Pilot's Associate functions are sensitive to and focused by the current mission state. Thus, for example, a minor malfunction in a redundant system will not command inappropriate resources in the heat of battle, but, plans that require the use of some degraded system at a later time in the mission will be constrained, adjusted or not proposed.

Situation Assessment

The Situation Assessment subsystem monitors events external to the aircraft to provide combat situation analysis. It combines stored mission data with data from the sensor suite and cooperative sources to provide context-sensitive information to the pilot and mission state information to other subsystems. For example, the target value can be assessed with respect to both mission objectives and ownship status. Also, threat data are assessed based on "inferred" lethality and intent and projected or planned ownship activities.

The Situation Assessment subsystem combines a need to react to unexpected objects and events with the query-driven function of interrogating the external space with respect to current or proposed plans, and in response to pilot requests. This requires that the subsystem not only reason about the situation in a data-driven way, but that it is also capable of focusing its attention on the data that are relevant to the current tactical plans. The Situation Assessment function includes methods for reasoning with uncertainty, allowing preliminary conclusions to be drawn based on imperfect or suspect data, and then be confirmed as contributing data is available and inferences become more certain.

Mission Planner

The Mission Planner creates and maintains a takeoff-to-landing mission profile, including routes, resources, and time constraints, which establishes the context for all Pilot's Associate reasoning. Beginning with a start and end point, and available fuel and other resources, it generates a flight profile, including route, that trades mission performance and exposure to known threats with consumption of resources. The resulting space-time description of the route, including projected threat exposure and expected consumption budgets, is then posted for other subsystems to reason about. The Mission Planner also functions as a higher level planning resource for the Tactics Planner, providing minimum risk trajectories to support both offensive and defensive tactical plans.

Tactics Planner

The Tactics Planner subsystem reasons about threats and targets for attack or evasion, and tasks the Situation Assessment subsystem to monitor certain high interest target data tracks. The Tactics Planner does not invent tactics. It recommends tactics suitable to the actual situation, but which are selected from the pilot's own pre-briefed tactics or or from a library of stored tactics The selected tactic is then customized to the geometry and timing of the existing situation. Approved tactics are monitored and adjusted, or repealed and replaced, in response to enemy counter-moves.

The Tactics Planner maintains a set of viable response plans to situations that unfold as the mission progresses. It is based on a skeletal planning scheme in which a hierarchy of plan elements is maintained. Each plan element carries its own knowledge base of when it should be invoked, how it is specialized, what other plan elements are related, what actions that are necessary for execution, and the conditions for failure or completion. This data structure permits the system to support both sequential and simultaneous plans, and allows for multiple plans to be considered and maintained as ready options until the mission situation dictates the selection of an exclusive option. The heirarchial structure permits the invocation of general plans even before precise details of the situation are known. Moreover, the distributed knowledge base allows a plan to be modified piecewise--at the plan element level--without destroying the larger structure of the plan The Tactics Planner also includes a set of reflexive responses to urgent situations to support rapid response to previously undetected threats

Lockheed's Program Approach

From the outset, it was recognized that, the engineering methodology used to develop the Pilot's Associate could be as important as the resulting system. No similar large-scale real-time intelligent system development had previously been attempted, and the degree to which previous experience with expert systems, coupled with conventional avionics software, would transfer to the Pilot's Associate was unknown. The program presented some additional difficult challenges. There was, for example, no sufficiently large concentration of experienced knowledge engineers in Lockheed or any other aerospace company to accomplish the program. Also, the operational domain of the 1995-era fighter aircraft system was itself largely conjectural, since neither the host weapons system--an advanced fighter aircraft--nor its employment concept existed at that time.

A number of aggressive and novel development techniques were adopted to cope with these challenges and were flexibly applied as the program evolved. Over time, this collection of techniques for developing intelligent systems has evolved into a systematic process. It is generally accepted that much of the success of the program, certainly with respect to the team's achievement within cost and schedule, can be attributed to this resulting development method. It is anticipated that the process will reach a degree of maturity in the current phase of the Pilot's Associate that will allow its generalization and application to the full life cycle development of a broad class of intelligent associate systems.

Structured Rapid Prototyping

A key aspect of the methodology is the use and function of rapid prototyping. Early design analyses and knowledge engineering efforts indicated that the conventional "waterfall" software development method, dependant on and flowing from a concise set of well-defined system requirements, could probably be modified and augmented for the Pilot's Associate program. The absence of any comparable precedent system suggested that it would be necessary to include frequent evaluations in the form of system-level prototypes to assess progress and to keep the program on track.

The very circumstances that recommend an intelligent avionics system seem to make it impractical to analytically define a precise set of requirements that specify the system in a conventional manner. The chosen program approach was to describe an initial, relatively small, set of hypothetical requirements and test and expand them with a series of prototype systems. Results flowing from each prototype would then form the basis for a new set of requirements and subsequent prototype. This general process had been successfully used in the development of many experimental expert systems. To make it suitable for a larger scale engineering program, it was necessary to structure the prototype cycle, adding a defined set of top-level objectives and a formal sequence of development activities, as depicted in Figure 3, to each planned three-month cycle.

The structured rapid prototype cycle has proven to be an invaluable feature of the development process. The incremental system prototypes have become much more than progress checks and debugging testbeds, they represent the principal expression of the actual

system concept. Thus, experience gained in the implementation of each prototype provides both guidance and motivation for refining and illustrating requirements and design for succeeding prototypes. Often, they have also illuminated significant new requirements or subtle prerequisites to existing ones. They serve as a primary vehicle for communication within the team and with outside observers. The prototypes also add to the knowledge engineering process, serving as a springboard and focus for knowledge acquisition activities. Of particular interest is the insight the prototypes provide to the domain experts of the potential employment of the system and how such a capability would affect their actions.

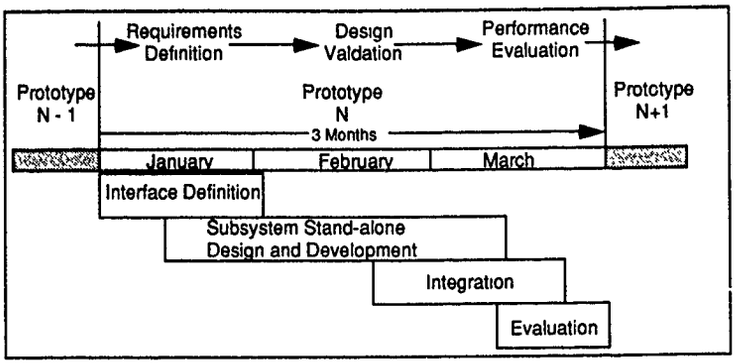


Figure 3: Typical Rapid Prototype Cycle

Advisory Boards

Two Advisory Boards were formed to provide objective advice and support from distinct perspectives. The Technical Advisory Board is comprised of seven consultants with national reputations in various aspects of Artificial Intelligence systems development. This board reviews, at one subsystem per quarter year, each major element of the Pilot's Associate system. By this process of "defending the program thesis" on a regular basis, the program managers were able to minimize technical risk, exploit opportunities in parallel research and identify potential problems before they become significant.

The Operational Task Force is comprised of former operational experts (both USAF and US Navy fighter pilots) with recent experience, now in industry, who review the operational aspects of the program. This group serves as a liaison with the operational community and ensures that their expressed concerns are appropriately addressed. It also serves to focus knowledge acquisition activities and acts as a forum for resolving differences of opinion among experts and developing a coherent body of expert knowledge for inclusion in the Pilot's Associate

From a program management perspective, the advisory boards have been helpful in finding a useful balance between immediate development concerns and the long term view of the program leading to transition of the technology into mission aircraft.

Distributed Team Concept

Recognizing that there was not sufficient technical ability in the emerging field of artificial intelligence within even a large aerospace corporation like Lockheed, a relatively large team of engineers was assembled from several companies located throughout the United States. By distributing technical leadership and responsibility for subsystem design and development across all the participants, Lockheed has fostered a cohesive, goal-oriented team. In several cases, a company leads development in one subsystem and supports work in others. This arrangement promotes both enthusiastic "ownership" of the work and accountability among the team members and stimulates technology sharing and cooperation. In general, the quarterly prototype integration sessions have provided an adequate forum for personal interactions among the engineers and government program participants. Other communications have been managed largely by electronic mail (Arpanet and a Lockheed dedicated computer network) and conventional telephones.

Evolution of "Intelligent System" and "Associate System" Concepts

An important early product of the of the Pilot's Associate Program was a more refined view of the nature of intelligent avionics systems in general, and of aircrew associate systems in particular. This maturing view has greatly influenced both our understanding of the potential application and value of such systems, and the process by which we develop them.

Significant Characteristics of Intelligent Avionics Systems

Intelligent avionics systems are distinguished from expert systems, or aggregations of expert systems by several important characteristics. The first, and perhaps most dramatic difference, is the scope of the application domain. Expert systems typically focus on specific problems or sets of related problems within a single well-bounded, narrowly defined domain. Intelligent avionics systems, conversely, are developed to cope with a wide range of dynamic (in space and time)

problems, general in nature and later specified by actual mission context, that span the complete operational domain of the intended user. This broader application necessarily emphasizes domain-specific problem solving techniques and strategic knowledge. It forces engineers at every level, from design to systems implementation, to comprehend and capture significant aspects of underlying domain theory and to produce a flexible, adaptable product capable of coping with a myriad of novel situations not explicitly considered in the development process.

A second significant distinction between expert systems and intelligent avionics systems lies in the operating environment and, in particular, in the manner in which mission data are acquired by the system. Expert systems are often "consulting" systems, typically designed to operate in a relatively benign environment. Operating knowledge is derived through a dialogue with the user that depends exclusively on his ability to acquire and to some degree, interpret information about the problem. Intelligent avionics systems are intended to operate in demanding performance environments--fighter aircraft cockpits in combat conditions, for example--that may not present reliable opportunities for such a dialogue, and that may render such a procedure infeasible. Therefore, intelligent avionics systems must obtain much of their operating data directly from a sensor suite, that may include the full set of external and internal sensors aboard the host platform, without human intervention or interpretation. The Pilot's Associate, for example, incorporates knowledge and strategies for obtaining required information, as well as the ability to control sensors and an awareness of the consequences of those actions. Intelligent avionics systems must also incorporate the means for clear, unambiguous and reliable communication with the human operator in environments that obviate more traditional single channel input/output devices. This important feature of intelligent avionics systems plays a large role in reducing operator workload and in producing a speedy, efficient man-machine interface

Additionally, the demanding operational environment implies a robust system that functions dependably with incomplete, noisy or even false data. It adds requirements for a system that can operate on rugged host processors and that can be used to advantage by average or even entry-level operators. Considerable care is required in the development of the user interface and in the inclusion of internal "sanity checks" to accommodate the typical errors of novice users

Finally, intelligent avionics systems are distinguished from expert systems in their degree of interaction with other electronic or mechanical systems Conventional expert systems are often stand-alone systems. Intelligent avionics systems are conceived as an element of a larger, integrated electronic system. And as such, a primary role of an intelligent system is to provide functional integration of various electronic systems in order to bring the fullest capability of the total system to bear on particular problems, in the context of actual circumstances. That capability is principally derived from the particular ability of intelligent avionics systems to be aware of the actual situation and to understand and reason about objectives

Significant Characteristics of Associate Avionics Systems

In a similar manner, our view of associate avionics systems, as a particular class of intelligent avionics systems, has evolved considerably. Initially, an associate system was seen as an essentially self-contained, independent system that performed a well-defined set of operations in response to specific requests to support a particular set of human activities.

We discovered that this, seemingly reasonable, perspective of the system was inconsistent with some important realities of human nature and the operational environment. In particular, two salient operational requirements forced a revision of our view of the Pilot's Associate and, consequently, of associate systems in general.

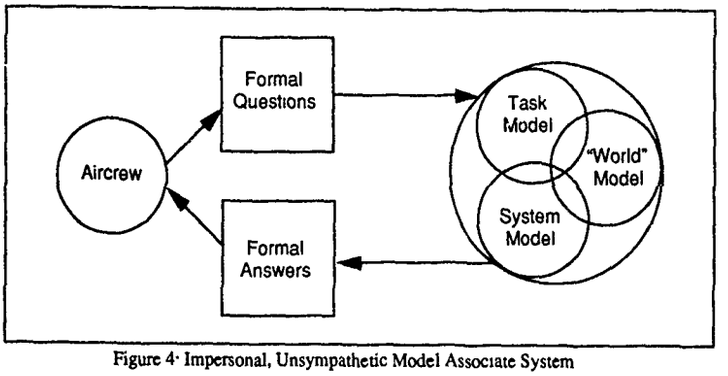


Figure 4· Impersonal, Unsympathetic Model Associate System

The first environment, shown schematically in figure 4, resulted from the tactical requirement to devise a system that would function and produce results in a fashion that would be predictable to the associate human (i.e., the pilot), but unpredictable to his adversary. We quickly realized that the system must consistently furnish the pilot with results that are believed to be "reasonable" (i.e. that are predictable to the pilot) or they would be ignored. At the same time, it is critically important that those results do not force a stereotyped set of predictable actions that yield substantial advantages to opponents.

The second operational environment, shown in figure 5, flows from the exigencies of the performance requirement which demand an extremely efficient and precise human interface, one

that frequently depends on implicit communication. Often decisions must be made and actions taken in time periods too short for a two-way exchange of information. In other situations, physical or tactical constraints impede verbal or manual communication. In these cases, critical communications are embedded in overt pilot actions or his lack of action. Typically, correct interpretation of such actions by an associate system depends on the understanding of an established plan for these circumstances.

Consider, for example, the situation of a fighter pilot in an hypothetical future conflict, equipped with weapons effective at ranges greater than thirty nautical miles, and a multi-spectral sensor suite effective at perhaps twice those ranges, confronting a similarly equipped opponent. Given the large detection volume of such sensors, and the projected density of air traffic in future conflicts, it seems likely that the pilot will frequently encounter situations in which the number of possible threats or targets within or nearing critical range exceed the number of air-air missiles on board (and probably the combined total of weapons in a tactical element). The pilot will then be required, in a very short time, to make a series of decisions that balance mission objectives and risk to the group and himself, that consider complex rules of engagement, resources and future engagements, and that frequently depend on subtle tactical clues and fine detail. When the situation is further complicated by even simple enemy tactical maneuvers and countermeasures, the decision process quickly reaches a level that may require decision-aiding systems for the pilot. The crucial quality of such assistance in a given situation, whether provided by a human crew member or an electronic system, is the correct anticipation of the pilot's needs that results in the timely provision of the "right" information or initiation of some particular action to support the pilot's planned response to the situation In these cases, "right" is highly individualistic. Some pilot's focus on some specific data, others rely on an awareness of the whole situation; some pilots are bold and aggressive; others cautious and cunning. A "standard" set of information or actions that forces the pilot to adapt his concept of action to some other view of the situation is distracting at best and frequently detrimental Even worse is a process that mechanically results in the same response to every situation and may provide a substantial tactical advantage to the enemy

A highly personalized tactical system, that incorporates the pilot's personal plans and responses to a variety of anticipated situations, and that can further specialize them according to the pilot's own rules for specialization, offers a solution to the dual problems of providing an efficient interface and tactical variation. This is shown in figure 5. The current implementation of this concept for the Lockheed team's Pilot's Associate is realized by means of a ground-based "Mission Support Tool" that allows the pilot to customize selected parameters of the airborne Pilot's Associate to his personal preferences, and to incorporate his own plan for the current mission. Thus, the pilot maintains the responsibility for planning the mission and selecting or deriving the primary, secondary and contingency tactics for a number of expected situations. The Pilot's Associate is configured as an instrument to support the precise execution of those plans, when they are appropriate, or to select a tactic from the pilot's own stored library of tactics when the pre-selected briefed tactics are unsuitable. In the heat of combat, the pilot will not be forced to try to make sense of some novel scheme invented by a machine; instead he will only be presented with precise specializations of plans, appropriate to the actual situation, that he analyzed, selected and briefed with his wingmen for the mission.

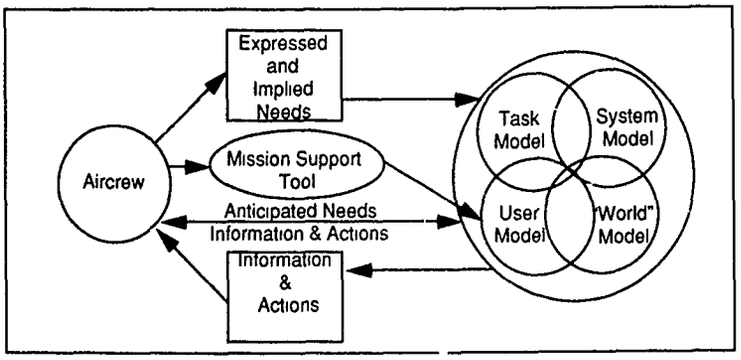


Figure 5: Personal, Aggressive Support System Model

The feasibility of this concept, which is strongly endorsed by the team domain experts and our Operational Task Force, was illustrated with the Tactics Planner subsystem in the most recent formal system demonstration in November on 1989 In that case, the engagement tactics for the demonstration mission were developed on an early prototype Mission Support Tool and subsequently loaded into the Pilot's Associate for successful use in the demonstration.

Summary

The Pilot's Associate effort pursued by the Lockheed development team has produced not only prototypes of intelligent avionics systems, but also new insights into the nature of these systems, and, new techniques for designing and developing software for these and similar systems. The structured rapid prototyping methods that have been utilized have also provided the ultimate consumers--tactical pilots--with significant awareness of the design concept and operational value of the Pilot's Associate, and with many opportunities to improve the requirements for such a system. It is believed that this fusion of operator input, through the

knowledge engineering process, and intelligent systems technology will result in an avionics system that will provide authentic situation awareness of all factors relevant to the assigned mission and provide dramatic improvements in mission effectiveness and aircraft survivability. The fielded version of the Pilot's Associate will eventually allow access to, and flexible employment of, every aircraft system, placing a floor, not a ceiling, on the pilot's performance, providing him the crucial edge in future air combat.

References

1. Broadwell, Martin, " Pilot's Associate, The Intelligent Edge, " paper presented at Loktek X, Lockheed's yearly public technical symposium, Washington, DC, November 1988

2. Lizza, Carl, "Pilot's Associate: A Perspective on Demonstration 2," AIAA Computers in Aerospace Conference, Atlanta, GA, 1989

3. "Strategic Computing, New Generation Computing Technology: A Strategic Plan for its Development and Application to Critical Problems in Defense," Defense Advanced Research Projects Agency, October 1983.

SCI^3 : UN SYSTEME A BASE DE CONNAISSANCES D'AIDE A L'INTERPRETATION D'IMAGES INFRAROUGES A BORD D'UN VEHICULE

D. MORILLON, T. CONTER et M. DE CREMIERS,
SAGEM. Division Recherche en Automatique
Centre d'Etudes de Pontoise - 95301 OSNY
Tél : 30 30 92 80

&

L.LEFORT et F GERMAIN
ITMI
ZIRST Chemin des prés - 38240 MEYLAN
Tél : 76 90 33 81

Cette communication fait suite aux travaux réalisés dans le cadre des marchés DRET N° 85/400 et 87/433.

Résumé : Sur tous les véhicules militaires, avions, drônes, bateaux, blindés et robots terrestres, l'imagerie infrarouge s'impose chaque jour de plus en plus en raison de l'information supplémentaire qu'elle apporte de jour comme de nuit, par tout temps et en toute discrétion.
Un système d'aide à l'interprétation de ce type d'images a été étudié Il utilise notamment des connaissances de navigation, de perception en infrarouge et de mise en œuvre de traitement d'image pour faciliter l'interprétation en permettant la prise en compte optimale des données présentes à bord : carte numérique du terrain, système de navigation inertielle, informations météorologiques. Ces travaux ont été concrétisés par la réalisation d'une maquette en laboratoire.

Mots cles : Système à Base de Connaissances, Interprétation d'Images, Navigation Inertielle, Cartographie Numérique, Infrarouge.

Abstract : Infrared sensors are becoming more and more essential on all military vehicles : aircrafts, RPVs, boats, tanks and land robots, because of the extra information they provide, day and night, in all weather conditions and with absolute discretion.
This paper presents a system designed to help in the interpretation of infrared images. It uses navigation, infrared sensing, and image processing knowledge to make interpretation easier, allowing optimal use of onboard data : digital maps, inertial navigation system, weather information. This work has been validated on a lab prototype

Keywords : Knowledge Based System, Image Interpretation, Inertial Navigation, Digital Maps, Infrared.

1. INTRODUCTION

L'éventail des systèmes optroniques en production ou en cours de développement couvre les trois grands domaines : terre, air, mer. D'un point de vue opérationnel, sur appareil piloté, deux besoins au moins apparaissent avec l'utilisation d'images infrarouges :

- Aide à la lecture des images par la fourniture au pilote d'une image avec incrustation, par imagerie de synthèse, des résultats de l'interprétation,
- Aide ou automatisation du recalage par désignation d'amer.

L'étude présentée ici, qui a été concrétisée par la réalisation d'une maquette laboratoire, a démontré la faisabilité et l'intérêt opérationnel d'un système à base de connaissances d'aide à l'interprétation d'images infrarouges à bord d'un véhicule. Cette interprétation d'images infrarouges (8µm -12µm), se fait sur la base de trois types d'informations :

- localisation et attitudes du véhicule fournies par un système inertiel de navigation,
- informations géographiques (planimétrie et altimétrie),
- informations sur les paramètres régissant le comportement thermique des éléments des scènes à interpréter.

En pratique, à bord d'un véhicule militaire, l'interprétation d'images comporte deux fonctions qui correspondent à deux problèmes assez différents d'un point de vue cognitif :

- Une fonction de **Navigation**, où l'on cherche à retrouver dans le paysage les éléments les plus caractéristiques portés sur les cartes et que l'on pense visibles compte tenu de la position estimée. Cette démarche se caractérise par un processus de raisonnement descendant et un domaine de connaissances bien délimité (univers restreint dans lequel les symboles à identifier sont en nombre limité et connu)
- Une fonction d'**Observation**, où l'on analyse l'image afin d'y détecter des menaces pouvant être des éléments annoncés mais non localisés (colonne de blindés en déplacement..), voire des éléments totalement imprévus (batterie de DCA,...). Cette approche conduit à un processus de raisonnement ascendant ou, autrement dit, une démarche guidée par les indices trouvés dans l'image. L'univers des connaissances à employer est ici largement ouvert, du fait de la méconnaissance du type et du nombre de symboles à rechercher

Dans cette étude, l'attention a été focalisée sur le mode **Navigation** et un cas d'étude a été utilisé, celui d'un avion d'armes volant à basse altitude.

2. STRUCTURE DU SYSTEME

Une caractéristique fondamentale du problème étudié est l'aspect très descendant des processus, c'est-à-dire du raisonnement de la carte vers l'image soit encore du modèle vers les données. Le terrain survolé étant en effet connu, le problème est de se situer dans cet univers.

Afin de présenter l'architecture du système, prenons un exemple :

◊ ***A un instant donné, on sait que l'on se trouve à un endroit donné et que l'on regarde dans une certaine direction ; que peut-on voir ?***

Un module d'**Extraction de Carte Active** a pour rôle de dresser la liste des éléments portés sur la carte et visibles a priori compte tenu de la position et des attitudes connues inertiellement. Pour ces éléments visibles du paysage, que nous nommerons ici **Symboles**, il détermine une apparence géométrique a priori et une fenêtre de recherche dans l'image.

◊ ***On voit un château d'eau et une route, lequel des deux rechercher en premier ?***

Un module **Navigation** va décider de l'ordre dans lequel la recherche des symboles sera effectuée. Il applique pour cela des connaissances de Navigation

Ce module va également devoir gérer la reconnaissance des différents symboles d'une même scène ou d'un même symbole à plusieurs instants successifs.

◊ ***Peut-on en savoir plus sur le château d'eau ?***

Un module de **Dérivation** va prendre en compte les conditions de vol et tout particulièrement météorologiques pour compléter le modèle du symbole en introduisant une description radiométrique de celui-ci Les connaissances appliquées portent donc sur la perception en infrarouge à basse altitude.

◊ ***Comment rechercher ce château d'eau ?***

Lors de la première phase de l'étude, nous avions pensé décrire les symboles en termes d'indices [CRE 88]. Les essais menés sur la première maquette ont montré que ce type de décomposition était un peu trop rigide et ne correspondait pas à l'expertise de traitement d'image. Celle-ci consiste plutôt à se ramener à une description plus abstraite de l'objet, sur les 2 aspects géométrie et radiométrie et choisir dynamiquement les traitements les plus adaptés à partir de la donnée de cette seule description abstraite. Il est donc apparu une notion de filières de traitements d'image (contours, régions, morpho-math...) permettant de résoudre spécifiquement certaines classes de problèmes.

La recherche d'un symbole est donc réalisée par l'application de plusieurs filières successives sous le contrôle dynamique d'un module **Identification**. Les connaissances appliquées à ce niveau sont strictement du domaine de la vision

◊ ***Comment mettre en oeuvre la filière contours choisie par le module Identification ?***

Un module de **Contrôle Vision** est chargé de décomposer la filière retenue en opérateurs élémentaires, de les paramétrer s'il y a lieu, de les faire s'exécuter, de contrôler le résultat de leur application par des mesures. Les connaissances appliquées à ce niveau sont également du domaine de la vision.

◊ ***Enfin, qui va faire les opérations sur l'image ?***

Un module d'**Opérateurs Vision** va exécuter ces opérations. Il est basé sur une bibliothèque générale de traitements d'image.

◊ ***Dernière question : comment prendre en compte les résultats de traitement d'image ?***

Il faut un mécanisme de remontée des informations jusqu'au niveau du module Navigation. Les opérateurs de vision retournent au contrôle vision des éléments permettant à celui-ci d'apprécier l'efficacité de leur application et d'avoir un contrôle véritablement adaptatif. A la fin, le dernier opérateur remonte un ou plusieurs indices que le Contrôle Vision remonte directement à l'Identification. Le module Identification, à son tour, après avoir activé une ou plusieurs filières, remonte au module Navigation les localisations dans l'image correspondant à ces symboles.

On remarquera sur la figure 2.1 les trois étages du traitement .

. Traitements géométriques
. Contrôle intelligent de l'Interprétation : Les modules Navigation, Dérivation, et Contrôle Vision ont comme point commun la programmation déclarative
. Traitements d'image

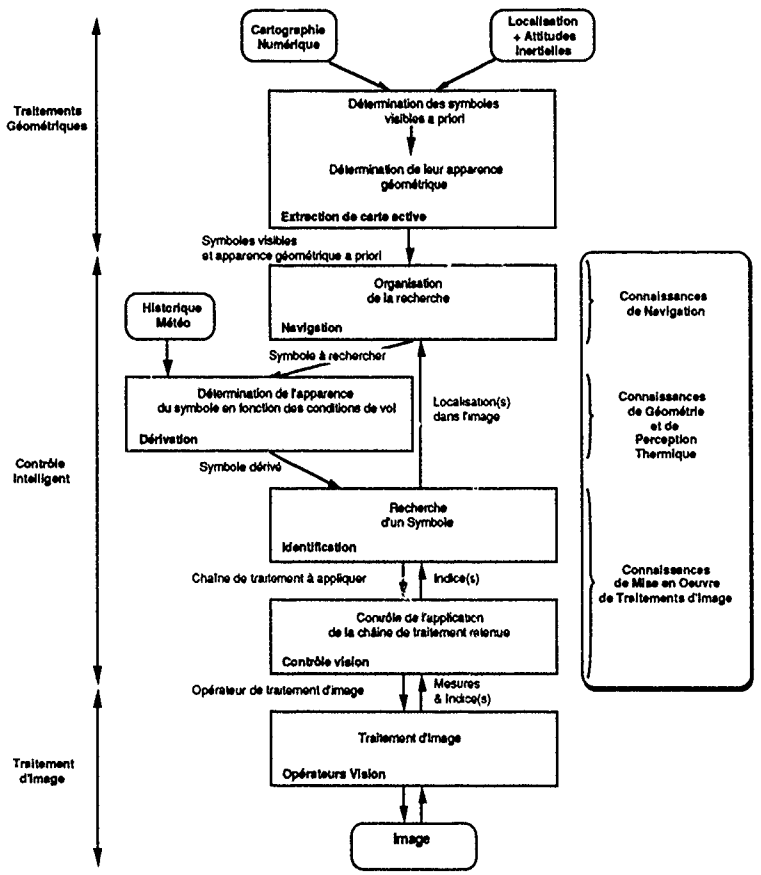


FIGURE 2.1 **ARCHITECTURE FONCTIONNELLE**

3. MODELISATION DE L'UNIVERS D'INTERPRETATION

3.1. INTRODUCTION

L'univers d'interprétation est l'ensemble des informations nécessaires au fonctionnement du système.

Il s'agit d'une part des données d'entrée du système :
- les informations cartographiques,
- les conditions thermiques,
- les informations inertielles,
- les images.

et d'autre part des résultats de l'interprétation (hypothèses de reconnaissances de symboles).

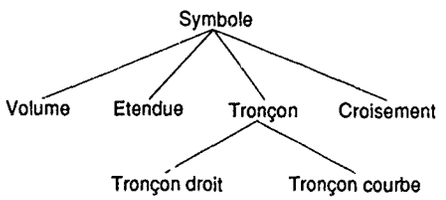
Nous ne détaillerons ici que les informations cartographiques et la représentation des résultats de l'interprétation

3.2. LES INFORMATIONS CARTOGRAPHIQUES

La consultation de pilotes d'une part et de spécialistes en traitement d'images d'autre part nous a permis de définir une liste de symboles dont la recherche présente un intérêt significatif. Nous nous sommes ainsi limité aux seuls éléments à la fois utiles pour la navigation et potentiellement détectables par des techniques de traitement d'image.

Pour modéliser et manipuler les symboles, une approche objet a été utilisée, car elle offre un moyen d'implémentation direct de concepts, tout en constituant un environnement de programmation efficace. Cette représentation porte en elle une part importante de la connaissance liée au problème, de par la structuration des données et les mécanismes d'héritage qu'elle permet de réaliser.

Les symboles ont donc été décrits sous forme d'une taxinomie qui repose sur le concept de **Symbole** :



Les travaux menés en vision ont fait ressortir que l'expert en traitement d'image faisait plutôt abstraction de l'objet à rechercher et sélectionnait ses opérateurs en fonction d'un modèle plus abstrait décrivant simplement le symbole sous ses aspects dimensionnels et radiométriques. La représentation du symbole comporte donc des champs décrivant ses caractéristiques dimensionnelles et radiométriques apparentes dans l'image et constituant une **description abstraite du symbole**

La structure qui permet de représenter les informations géographique est la carte symbolique, ou **S-Carte**, qui inclut la planimétrie (liste de symboles) et l'altimétrie (modèle numérique de terrain GMEDTN)

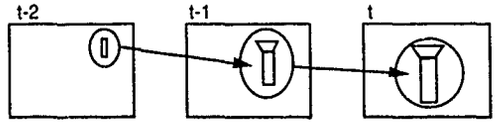
3.3. LES RESULTATS DE L'INTERPRETATION

Pour réaliser la première reconnaissance d'un symbole, on peut utiliser la position donnée par l'inertie pour déterminer les éléments géométriques permettant la recherche du symbole (position dans l'image, forme,...).

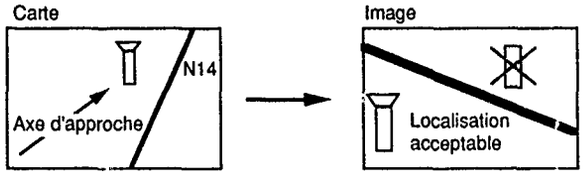
Cette première reconnaissance va nous fournir une ou plusieurs localisations possibles dans l'image correspondant à autant d'hypothèses de reconnaissances du symbole.

Comment exploiter ces résultats pour faciliter la recherche d'autres symboles et discriminer les différentes remontées ?

1) Sur une séquence en adoptant des modélisations de déplacement dans l'image utilisées en poursuite de cible pour vérifier si tout au long de la séquence on arrive à poursuivre chaque écho initial. En principe, si l'on se rapproche du symbole, il va grossir et sa discrimination par rapport aux artefacts deviendra possible et seule une hypothèse devrait subsister.



2) On peut alors utiliser un raisonnement géométrique qualitatif : "Compte tenu de l'axe d'approche, le château d'eau de Castelviel est obligatoirement vu à gauche de la N14 ; donc, comme on pense avoir reconnu la N14, parmi les deux localisations trouvées pour le château d'eau seul l'écho 1 est une solution possible"



Aucune de ces deux solutions n'est pleinement satisfaisante, soit parce qu'elle ne permet pas d'utiliser le résultat d'une reconnaissance pour faciliter et diagnostiquer la reconnaissance d'autres symboles, soit parce qu'elle n'autorise pas la prise en compte de l'aspect "séquence d'images". Nous avons donc préféré une solution qui tire profit de la disponibilité d'une centrale inertielle à bord du véhicule.

Le principe retenu dans l'application consiste à considérer chaque interprétation comme une observation de l'erreur de la navigation inertielle de l'appareil. A chaque interprétation correspond donc une hypothèse de localisation réelle de l'appareil - obtenue en corrigeant la position inertielle à partir de l'observation réalisée. Lorsqu'une interprétation génère plusieurs possibilités de localisation d'un amer dans l'image, cela correspond à plusieurs hypothèses de localisation de l'appareil.

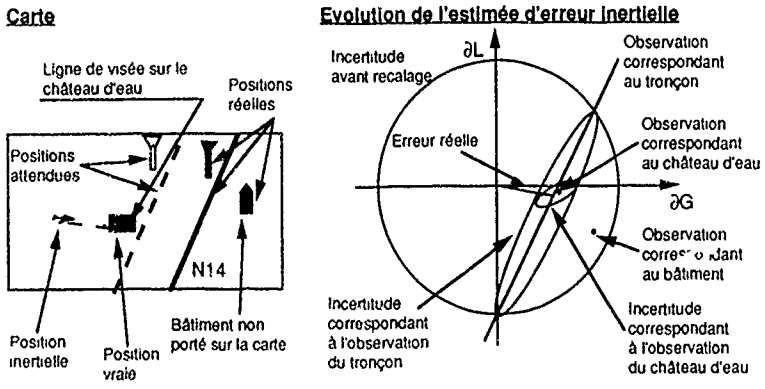
Ce principe a été implémenté grâce à un filtre de Kalman dont le vecteur d'état a deux composantes : ∂L, l'erreur inertielle de latitude et ∂G, l'erreur inertielle de longitude.

La gestion des hypothèses de localisation permet la prise en compte de remontées vision donnant plusieurs localisations possibles pour un symbole recherché.

Reprenons maintenant l'exemple fourni pour le raisonnement géométrique. Dans le champ de vision, nous avons un tronçon de route nationale, un château d'eau et un bâtiment non porté sur la carte.

Supposons que l'on se recale tout d'abord sur la route, l'incertitude sur la position se réduit de façon ellipsoïdale autour de l'erreur estimée avec le grand axe dans la direction du tronçon.

Si l'on cherche ensuite à détecter le château d'eau, en utilisant l'incertitude réduite, seule la localisation correspondant véritablement au château d'eau sera acceptée, l'artefact provenant du bâtiment sera rejeté, n'étant pas compatible avec l'estimée d'erreur correspondant à la reconnaissance de la route.



Cette représentation des résultats de l'interprétation permet de propager le bénéfice de la reconnaissance d'un élément, à la fois dans la scène courante et dans les scènes suivantes, car le résultat de l'interprétation est exploité par rapport à la référence inertielle et non directement dans l'image.

4. DESCRIPTION DETAILLEE DU SYSTEME

4.1. MODULE D'EXTRACTION DE CARTE ACTIVE

Le module d'Extraction de Carte Active a pour rôle de déterminer quels sont les symboles visibles, quelle est leur apparence géométrique dans l'image (position, forme, dimensions, orientation) et dans quelle fenêtre de l'image ils pourront être recherchés. Il reçoit pour cela du module Navigation les informations de situation de l'appareil sous forme d'une localisation intégrant :

- la position (L, G, Z) et les attitudes (Cap, Roulis, Tangage), fournies par la navigation inertielle,
- une estimation de l'erreur inertielle courante, fournie soit par la navigation optimale (hypothèse de départ du système), soit par le système d'interprétation lui-même, à partir de reconnaissances de symboles déjà effectuées.

Il renvoie à ce même module Navigation la liste des symboles visibles, avec leurs modèles abstraits renseignés du point de vue des caractéristiques géométriques apparentes dans l'image. Cela permet de constituer une image prédictive de la géométrie des symboles dans l'image.

4.2. MODULE NAVIGATION

Le rôle du module Navigation est la gestion globale de l'interprétation, à savoir décider quels sont les symboles intéressants à reconnaître et exploiter les résultats de leur recherche dans l'image

En entrée, le module Navigation reçoit la carte active, i.e. la liste des symboles supposés visibles compte tenu de la localisation optimale de l'appareil Il reçoit également l'ensemble des hypothèses de localisation courantes.

En sortie, après avoir recherché tous les symboles utiles, le module retourne une ou plusieurs hypothèses de localisation (de l'appareil), en principe affinées par rapport à celles reçues en entrée.

Le module Navigation assure ce rôle au travers de deux fonctions principales :

Elaboration d'une interprétation complète

Le principe d'élaboration d'une reconnaissance basé sur le concept de Localisation est le suivant (voir figure 4.1) :

1) On prend le symbole le plus intéressant de la première scène et on le recherche en fonction de la localisation inertielle optimale. On obtient une ou plusieurs hypothèses de localisations.
2) On continue ainsi pour tous les symboles de la première scène puis ceux de la scène suivante (dans le cas ou l'on souhaite analyser une séquence) en prenant en compte à chaque fois toutes les hypothèses de localisation. Il se constitue ainsi un **graphe de localisations.**

Pour chaque nouvelle localisation dans le graphe de localisation, on calcule un facteur de certitude global selon une formule dérivée de la théorie des croyances (Dempster Shafer)

L'interprétation est terminée lorsque le graphe ne comporte plus qu'une branche, avec un taux de confiance et une précision de localisation estimée jugés suffisants.

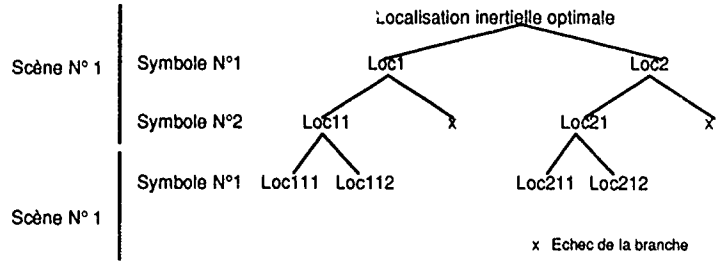


Figure 4.1 **GRAPHE DE LOCALISATION**

Choix des symboles

Les connaissances de choix des amers intéressants pour les recalages à vue sont détenues par les pilotes et navigateurs. Notre système, bien que basé sur une recherche des symboles par traitement d'image, utilise sensiblement les mêmes connaissances. Celles-ci peuvent se traduire sous la forme d'un ensemble de critères décorrélés deux à deux (surface apparente, longueur, confondabilité,...).

Le choix des symboles est réalisé en deux étapes :

1) Sélection des symboles globalement acceptables, c'est-à-dire satisfaisant un minimum chaque critère - implémentée par un mécanisme procédural de filtrages successifs sur chacun des critères.
2) Tri de ces symboles retenus. Face à ce type de problème les algorithmes de tri sont impuissants, sauf à élaborer un critère global basé sur une somme pondérée de tous les critères élémentaires. Le problème d'un tel "amalgame" est de ne pas permettre une réelle prise en compte de chaque critère. Par exemple un symbole répondant bien à tous les critères sauf un pourra passer devant un autre symbole simplement bon mais sur tous les critères, ce qui n'est pas forcément souhaitable. Le choix s'est donc porté sur des techniques d'aide à la décision [ROY 87] qui se sont révélées tout à fait adaptées et correspondant bien à l'esprit "Intelligence Artificielle" de programmation déclarative et de manipulation de l'imprécis

4.4. MODULE DE DERIVATION DU SYMBOLE

Le module Dérivation a pour rôle d'établir un modèle dérivé du symbole en fonction des conditions de vol rencontrées. Il s'applique tout particulièrement à la description radiométrique du symbole, la description géométrique ayant déjà établie en grande partie par le module d'Extraction de Carte Active. Il permet, en fonction des conditions rencontrées, d'instancier ou de modifier les champs de description thermiques du symbole.

Ce complément d'information est nécessaire pour affiner le choix des filières de vision ainsi que le paramétrage des opérateurs qui seront appliqués par la suite

C'est un système de production dont la base de règles contient les connaissances relatives à l'influence des conditions de vol sur l'aspect du symbole et dont la base de faits est constituée, entre autres, par le symbole à dériver et les conditions météorologiques passées et présentes.

L'expertise a été fournie par des spécialistes des caméras thermiques et par analyse statistique d'une base données d'images constituée dans le cadre du projet.

4.5. LE MODULE OPERATEURS VISION

Le rôle du module Opérateurs Vision est de mettre à la disposition du module Contrôle Vision un ensemble d'opérateurs de traitement d'images permettant la détection des symboles dans les images. Ces opérateurs sont de 3 types :

- opérateurs de pré-traitement, transformant une image en une autre image,
- opérateurs de traitement, transformant une image en un ensemble d'indices visuels,
- opérateurs de post-traitement, transformant un ensemble d'indices visuels en un ensemble réduit de ces mêmes indices ou en un ensemble d'autres indices visuels de plus haut niveau.

Nous ne détaillerons pas ce module plus avant, nous rappellerons simplement deux concepts très importants dans le système :

. **Le modèle abstrait du symbole** (géométrique et radiométrique)
C'est à lui que se ramènent les traitements de façon à permettre une "représentation standard" des résultats, autorisant ainsi leur évaluation, tri et comparaison.

. **La représentation de l'imprécis**
Pour traduire l'imprécision liée au modèle dérivé sur la base duquel sont réalisées toutes les recherches, une structure d'intervalle [Mini, Probable, Maxi] a été utilisée. Malgré sa simplicité ce principe s'est révélé suffisant.

L'ensemble des opérateurs développés est présenté dans la figure 4.2 sous forme de graphe des enchaînements possibles.

4.6. MODULE IDENTIFICATION

Le module identification est en charge de la planification dynamique et du contrôle de la recherche de symbole dans l'image. L'identification d'un symbole donné se fait par agrégation de résultats issus de recherches par traitement d'image conduites intégralement par le module "Contrôle Vision". Chaque recherche correspond au suivi d'un chemin dans le graphe d'opérateurs avec a priori un certain nombre d'alternatives locales (voir figure 4.3).

Chaque sous graphe correspondant à un ensemble cohérent d'opérateurs permettant de mettre en évidence une certaine apparence de symbole est appelé **Filière.** Dans le cadre du projet, sept filières ont été développées.

Le module assure trois fonctions que nous présentons successivement ci-après.

Choix de filière

Il est assuré par un système de production dont les règles comportent en partie gauche des conditions sur l'apparence du symbole et en partie droite des prédicats indiquant l'intérêt des différentes filières dans ce contexte. Chaque prédicat se traduit par un vote recueilli au niveau de chaque filière dans une structure à cinq champs (de non satisfaisante à très satisfaisante).

Quand tous les votes ont été réalisés, on analyse les résultats pour éliminer les filières non satisfaisantes et classer par ordre d'intérêt "a priori" celles qui sont retenues.

Activation et succession des filières

La communication avec le niveau inférieur "Contrôle Vision" se fait à chaque décision d'application d'une filière, sous forme d'activation du module contrôle vision pour son application ; en retour le module reçoit aucune, une, ou plusieurs positions possibles, avec pour chacune un taux de confiance "Vision". Le module lance ensuite la filière suivante ou arrête la recherche.

Evaluation finale des remontées

L'application de plusieurs filières successives et leurs résultats sont pris en compte dans le calcul d'un facteur de confiance retourné au module Navigation. Ce calcul s'appuie sur la comparaison des résultats des filières par rapport à l'hypothèse de vision de départ, mais aussi sur la comparaison des résultats entre les différentes filières.

4.7. MODULE CONTROLE VISION

Le module Contrôle Vision est activé par le module supérieur Identification d'un Symbole, dans le but de confirmer ou d'infirmer la présence d'un symbole, sous la forme d'une demande d'application d'une filière, correspondant à la recherche d'un indice visuel donné et tenant compte de la connaissance a prion du symbole contenue dans les champs géométriques et radiométriques constituant son modèle abstrait.

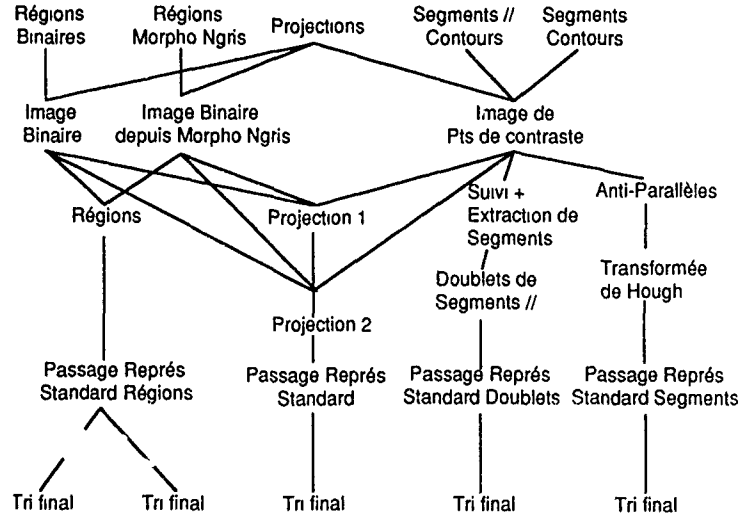


Figure 4.2 **GRAPHE DES OPERATEURS DE VISION**

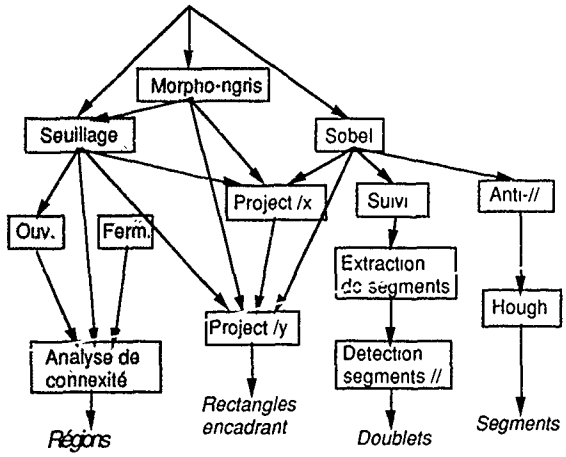


Figure 4.3 **EXEMPLE DE FILIERE**

La conduite d'une filière Vision consiste donc en la génération d'une séquence d'opérateurs de traitement d'image.

Le Contrôle Vision appelle le niveau inférieur de Traitement d'image de bas niveau par des activations successives d'opérateurs vision avec la donnée de tous les paramètres nécessaires à ces opérateurs.

Le module contrôle vision réalisera donc plusieurs tâches :
- détermination des chaînes d'opérateurs possibles et choix de la plus pertinente,
- guidage de l'application des opérateurs, notamment le choix de leurs paramètres,
- intégration des mesures intermédiaires réalisées par les opérateurs,
- test des conditions d'échec de la recherche après chaque opérateur.

Le concept de Macro-Opérateur

Le **Macro-Opérateur** (noté MOP) est la structure de base du module Contrôle Vision. Fonctionnellement, un MOP traduit une entité en termes de traitement d'image [THO 88]. Ainsi, on considère différents niveaux de MOP allant de l'opérateur vision bas niveau à la chaîne de traitement complète.

La notion de MOP est liée à celle de mesure Ainsi, dans le cas général, on regroupera sous le terme de MOP la donnée d'une séquence d'opérateurs bas niveau (éventuellement un seul) et d'un ensemble de mesures à effectuer a posteriori. Ces mesures ont pour but de juger, dans la mesure du possible, du résultat de l'application du MOP courant, et éventuellement de guider le choix et les conditions d'application des MOPs ultérieurs.

La génération dynamique de plan

Le choix d'une filière revient à sélectionner une succession de sous-buts qui constituent autant d'étapes dans la réalisation du but principal trouver la localisation d'un symbole dans une image.

La façon dont est réalisé chaque sous-but est déterminée dynamiquement en fonction du contexte. En effet, soit ce sous-but est lui même redécomposable, soit il est réalisable par une seule transition dans le graphe des opérateurs de vision. Cette prise en compte différée des choix locaux d'opérateurs permet de tirer le meilleur profit des résultats issus des mesures intermédiaires.

On distinguera donc deux types de MOP :
- **les MOP terminaux** correspondent de manière univoque à une séquence "indivisible" d'opérateurs vision bas niveau,
- **les MOP non terminaux** se décompose, de manière unique ou non, sous forme d'une séquence de MOP terminaux ou non terminaux.

5. MAQUETTE DE DEMONSTRATION REALISATION - EXPERIMENTATION

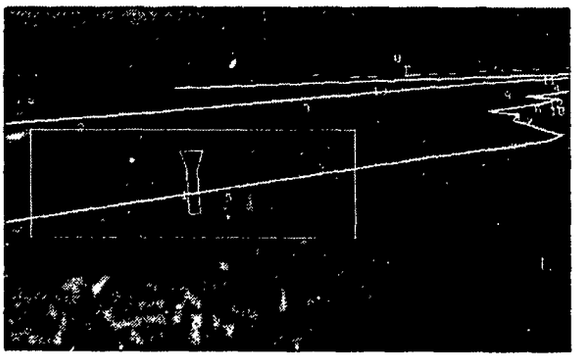
5.1. INTRODUCTION

La maquette de démonstration, objet du marché, a représenté un travail considérable réalisée sur machine Lisp SYMBOLICS 3650 :

- plus de 20000 lignes Lisp entièrement originales,
- plus de 90000 lignes de C dont moitié d'originales pour la librairie de traitement d'image,
- saisie, recalage et documentation de plus de 60 scènes,
- expérimentation du système sur ces scènes.

5.2. EXEMPLE D'INTERPRETATION DE SCENE

Nous allons, dans cet exemple, interpréter une scène semi-urbaine. Le système commence par déterminer les symboles visibles et leur apparence a priori. Il les trie et ne retient que 2 symboles intéressants . le château d'eau d'Aimargues et un tronçon de route. Il part sur l'identification du château d'eau avec une fenêtre de recherche correspondant à la localisation optimale (Figure 5.1).

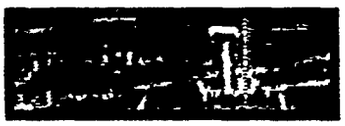


Après dérivation du symbole courant, on effectue une phase de mesure sur l'image afin de compléter le modèle radiométrique.

On ordonne et trie les filières pour n'en retenir que deux :

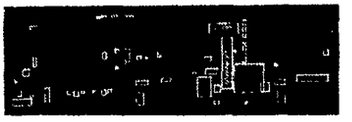
- une approche morphologie mathématique / analyse de connexité,
- une approche projection de points de contours.

On entre dans le module Contrôle Vision avec la 1ère filière, macro-opérateur non terminal, qui se décompose en un 1er macro-opérateur réalisant un "chapeau haut de forme" suivi d'un seuillage adaptatif pour mettre en évidence les zones blanches



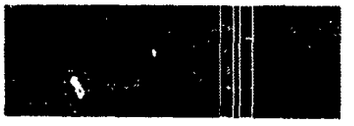
Le 2ème macro-opérateur non terminal est ensuite décomposé et la branche "ouverture morphologique" est retenue :

Le 3ème macro-opérateur, terminal, est une analyse de connexité suivi d'un tri qui nous fournit une seule localisation dans l'image :



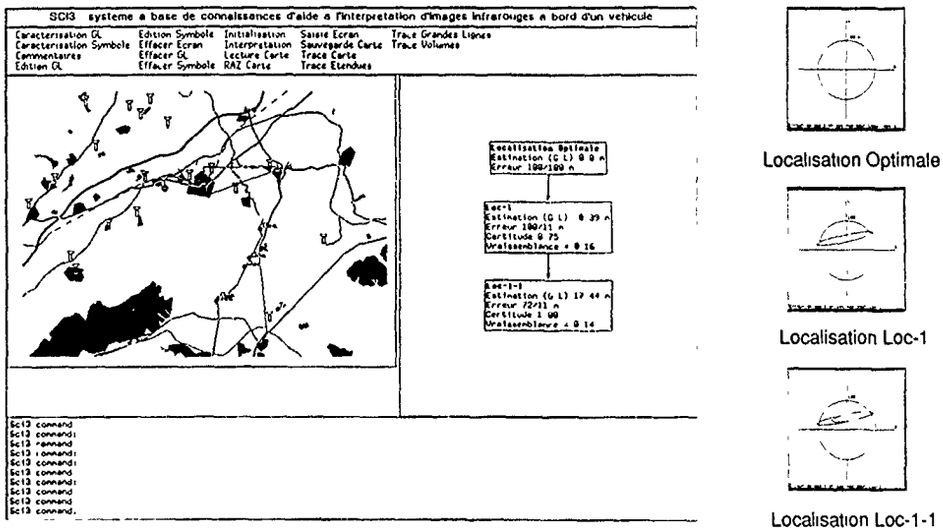
On remonte au module Identification qui lance en confirmation la 2ème filière disponible dont la première opération est une extraction de points de contours verticaux :

On projette ensuite et on retient deux pics d'une largeur correspondant à la largeur attendue du château d'eau :



On effectue dans chaque tranche une projection horizontale et on recherche les pics pouvant correspondre au château d'eau :

On remonte ces 2 localisations à l'Identification qui va fusionner avec les résultats de la détection et ne fournir qu'une localisation. On remonte cette localisation fusionnée au module Navigation qui élabore l'hypothèse de localisation Loc-1 (planche 5 1) avec une estimée de l'erreur de ∂G = -8 m ∂L = 39 m pour une erreur introduite de 50 m sur chaque axe (indiquée par le point dans la fenêtre de présentation de l'estimée d'erreur de position). L'estimée de l'erreur est une ellipse allongée (petit axe = 11 m) dans la direction de la ligne de visée avion - château d'eau. C'est sur cette base que va être élaborée la fenêtre de recherche du Tronçon de route.



Localisation Optimale

Localisation Loc-1

Localisation Loc-1-1

FIGURE 5.1
CARTE ET GRAPHE DE LOCALISATION - EVOLUTION DE L'ESTIMEE DE L'ERREUR INERTIELLE ET DE L'ERREUR D'ESTIMATION

6. CONCLUSIONS ET PERSPECTIVES

Une réalisation complexe

Tant par les techniques qu'il emploie, que par les connaissances mises en œuvre dans les modules "intelligents", le système est d'une très grande richesse. La complexité résultante a entraîné de lourds développements mais elle était nécessaire pour résoudre le problème posé. Il est intéressant de souligner que les techniques de programmation, fonctionnelle et surtout déclarative qui sont une des composantes de l'Intelligence Artificielle ont contribué à la réussite du projet autant que les techniques "cognitives".

Des techniques innovantes

Dans chacun des domaines pré-cités, des solutions intéressantes ont été retenues, nous mettrons ici l'accent sur deux d'entre elles. La variabilité de l'apparence tant géométrique que radiométrique des symboles a conduit à définir une représentation abstraite de leur apparence dans l'image. Sur la base de ce modèle il a été possible de réaliser des fonctions de tri, comparaison de résultats de filières de traitement d'image, pourtant de natures très différentes. Ceci a été réalisé, par des mesures de distance entre modèle et résultats, rendues possibles grâce à cette représentation. La représentation des résultats de l'interprétation sous forme d'hypothèses d'erreurs de position inertielle du porteur, facilite leur exploitation à fins de confirmation pour la reconnaissance d'autres symboles que ce soit dans une même image ou à des instants distincts

Des résultats prometteurs

Dans sa configuration laboratoire actuelle, la maquette permet de reconnaître plusieurs types d'amers dans des images FLIR basse altitude : tronçons droits de voies de communication, superstructures telles que tours, châteaux d'eau et autres bâtiments isolés. Le taux de bonne détection est à l'heure actuelle de 80%. L'échantillon d'images utilisé reste cependant limité par rapport à l'ensemble des conditions que pourrait rencontrer un système opérationnel et il y a lieu d'être encore prudent sur ce point.

Des sujets de recherche

La maquette réalisée, est aujourd'hui un point d'appui pour des développements intéressants :

- Le mode Observation, très intéressant d'un point opérationnel et théorique, n'a pas encore été étudié. Sa réalisation suppose l'ajout de fonctions de poursuites dans l'image pour détecter les véhicules et la mise en place d'une interprétation contextuelle (lieu et vitesse de déplacement) pour leur identification
- Dans son fonctionnement actuel le système pourrait être amélioré grâce à des capacités d'apprentissage "en opération", notamment au niveau de la prévision de l'apparence thermique des symboles pour laquelle on pourrait utiliser le résultat de reconnaissances déjà réalisées.

Des applications potentielles

En plus d'applications aéroportées, le système pourrait être utilisé sur véhicule terrestre, autonome ou non, pour des applications de recalage et de suivi de route. Une prise en compte de risques de masquages plus élevés et de modèles plus complexes au niveau de l'apparence des symboles dans l'image, amenant des traitements d'images plus délicats, seraient les principaux points à développer pour ce type d'application. Enfin, des applications en exploitation d'images satellites ou de reconnaissance aérienne sont également envisageables.

BIBLIOGRAPHIE

[CRE 88]
Un système Expert d'Aide à l'Interprétation d'Images Infrarouges à bord d'un véhicule
M. DE CREMIERS, D. MORILLON,
Actes de la Conférence Spécialisée "Intelligence Artificielle & Défense"
Huitièmes Journées d'Avignon, Juin 1988

[CRO 87]
Mathematicals tools for Representing Uncertainty in Perception
J.L. CROWLEY & F. RAMPARANY
Proc. of the 1987 AAAI Workshop on Spatial Reasoning and Multi-Sensor Fusion
St. Charles, Illinois, USA, October 1987

[FAU 71]
Navigation Inertielle Optimale et Filtrage Statistique
P. FAURRE
DUNOD 1971

[FOR 84]
Contextual Analysis of Tactical Scenes
A.V. FORMAN Jr, P.J. ROWLAND, W.G. PEMBERTON
SPIE Vol. 485 Applications of Artificial Intelligence (1984)
pp 189-197

[GAU 84]
La Thermographie Infrarouge
G. GAUSSORGUES
Collection Technique et Documentation
Lavoisier 1984

[GIL 83]
A Model Driven System for Contextual Scene Analysis
J.F. GILMORE A.J. SPIESSBACH
SPIE Vol. 432 Application of Digital Image Processing (1983)
pp 262-268

[GIL 85]
Military Applications of Expert Systems
J.F GILMORE
Congrès "Les Systèmes Experts et leurs Applications"
AVIGNON 1985
pp 251-321

[GON 87]
Une Société de Spécialistes à Niveaux Multiples pour l'Interprétation de Signaux
Y.GONG & J.P HATON
AFCET Reconnaissance des Formes et Intelligence Artificielle (1987)
pp 245-257

[HAV 83]
Representing Knowledge of the Visual World
W. HAVENS & A. MACKWORTH
IEEE Computer (1983)
pp 90-91

[KIM 88]
A Context Dependent Automatic Target Recognition System
J.H. KIM, D.W. PAYTON, K E OLIN & D.Y. TSENG
SPIE Vol. 485 Applications of Artificial Intelligence (1984) pp 2-7

[ROY 85]
Méthodologie Multicritère d'Aide à la Décision
B. ROY
Collection Gestion-Editions ECONOMICA

[SHE 85]
The Impact of Artificial Intelligence on Airborne Maritime Reconnaissance
J. SHEPARD & R.J SCOTT.WILSON
AGARD N°380 The Impact of VHIC on Radar Guidance and Avionics Systems, pp 39-1 à 39-13

[THO 88]
OCAPI : a Monitoring Tool for the Automatic Control of Image Processing
M. THONNAT & V. CLEMENT
Processing Procedures, 12th IMACS World Congress On Scientific Computation, Paris, France, July 1988

[WHI 87]
Pilot's Associate Approach to Integrating Distributed Expert Systems
T.C. WHIFFEN
NAECON 87

PATH GENERATION AND EVALUATION FOR A MISSION PLANNING EXPERT SYSTEM.

Federica LUISE
Danilo DABBENE

AERITALIA, Defense Aircraft Group
Artificial Intelligence Laboratory
Corso Marche 41
10146 Torino
ITALY

ABSTRACT

The aim of this paper is to describe AERITALIA's (Defense Aircraft Group) experience with the problem of path generation and evaluation for a multitarget air to ground planning system working in a limited geographic scenario.

Since in the same planning operation we have to satisfy many goals (each goal is a mission target) we have to use techniques of splitting problems into simpler ones as it is not possible to use satisfactorily heuristic guided search (like A*). Although these algorithms give an optimal solution for each goal, this doesn't ensure that the union of the solutions that are optimal will be the best for the whole set of goals.

The mission is decomposed into sub-missions, and each one of them is solved finding all the solutions satisfying the forced constraints (e.g. maximum leg length, maximum turn number). All the solutions so generated become a new search space to be used to find out the final best solution.

Every sub-mission is represented by a task (a process) that generates all the partial solutions (partial paths) that will be combined in an upper stage by a higher level task (mission task) in order to generate the whole set of global solutions.

Currently all the tasks are independent and are managed by a task-scheduler which simulates the behaviour of a multiprocessor architecture carrying out the task synchronization by an event generation and waiting mechanism. A few simple primitives allow the tasks to generate events and wait for them (e.g. waiting for the end of a task computation).

Every sub-mission task builds each path as a collection of atomic paths (point-to-point) which may be shared by different solutions of the task. The mission tasks will combine the subtasks results, generating a graph whose roots are all possible paths and whose leaves are the atomic paths (a leg). Afterwards the best ones

among these paths are chosen using a bayesian evaluation system that traverses the path graph calculating the main features (e.g path length, dangerousness etc.) and sending the values calculated in the leaves to the roots.

The evaluator is a graph which has path feature evaluation nodes and bayesian combination nodes whose function is to combine evaluations in one single value. The criteria used concern risk exposition, path recognizability, consumption, altitude profile, turn width, success probabilty on target. The evaluation graph is run time configurable to make possible the use of all the criteria or any subset.

The developed prototype has been tested by our pilots and the results are described below.

The system has been developed on a lisp machine Explorer II using Lisp and Flavors and later ported on a Symbolics 3630.

Introduction: The planning problem

The generation of a flight plan for a ground attack mission in visual flight is a process that finds a path joining a sequence of way points which connect the take off airfield with the landing one through the planned targets of the mission.

A mission planner used in the field must be able to generate for each mission an optimal flight plan (or a set of plans) with respect of a certain set of criteria. Pilots must always have the possibility to choose their flight plan from a set of plans (in order to avoid always having the same plan for the same mission). Besides the solutions, the mission planner must provide information about the probability of success of the mission, risk of the route, consumption, etc.

A set of constraints must be taken in account to build the paths that will become flight plans. These constraints will be used in the generations of plans together with a set of criteria for evaluating the quality of the plans themselr. Many of these criteria and constraints are based on pilots experience. When pilots have to plan a mission they use theirflight experience and knowledge of the territory they must flight over. They also use their knowledge of the aircraft performance, personal strategies and attack modalities.

Nowadays the task of preparing flight plans is performed by pilots who are the best experts in this field (and obviously the most direct interested in the plan's quality). An automatic planner must generate plans at least as good as those of pilots (better if possible) so it must take into consideration the same knowledge that pilots use in the problem resolution and that they have acquired during their flight experience.

The path generation and evaluation: an analysis from the knowledge point of view.

A detailed analysis of the planning problem (if not exhaustive) is necessary to better understand the problems one can meet in the development of an automatic mission planner. This analysis has been possible only observing how a pilot plan a mission. Between the different elicitation techniques the most useful was to ask pilots to "think aloud" during a series of sessions. The records of these sessions allowded a better understanding of the way in which pilots work and to gather the first kernel of knowledge, later gradually increased. Even some flights were useful to effectively understand the pilots point of view. During these flights pilots explained to us in the field their problems, the knowledge involved and its use.

In the introduction we asserted that to plan a mission means to determine a path joining a set of way points so as to connect the take off airfield with the landing one through the defined targets of the mission.

The way points to be connected must satisfy a set of requirements that restrict the choice possibilities in the definition of the paths. These requirements are: the probability of a correct identification of way points during flight according to the arrival direction and the constraint to obtain paths in certain lenght range with limited and not too frequent turn angles (geometrical route constraints).

Moreover it is also necessary to take account of the orographic information of the territory to be flown over and of the possibility of finding threats on the route. Thus it is necessary to find an optimal path according to different points of view (length, threatness, probability of correct identification of the way points, etc.), some of them can contrast each other.

Some evaluation criteria depend on the average value of certain characteristics of the calculated path (e.g. the identification of the way point), others depend on the complete value of the path (lenght, threatness etc.). These evaluation criteria are calculated in different ways: this fact influences the generation of the route. In fact it means that we cannot use heuristic search algorithms in the space of the states (i.e. algorithm of the A* class (6)) because they require incremental evaluation functions in order to be used with a certain advantage.

Moreover it is also necessary to take into account tactical criteria particularly in choosing the optimal push up point on the target. This kind of choice depends especially on the target type (bridge, highway junction etc.), on the armaments and on the attack type (toss, lay down, etc.). Finally, the flight plan must take into account the temporal constraints (i.e. time on target).

So the problem to build a flight plan for a mission becomes the problem of defining paths linking the focal points of the mission (take off, targets, landing) so as to satisfy some constraints and to be optimal according to certain requirements.

At the first glance one could think to reduce the problem into subproblems (e.g. from the take off base to the first target, from this last to the next and so on till the landing base) and then to resolve separately these subproblems and unify the results to find out the complete solutions.

In general this is not possible because constraints and optimality requirements are used in every point of the route, so it is not possible to guarantee that they are respected in the joining points of the subproblem solutions.

In other words, in our case the simple "sum" of local optimal solution doesn't guarantee an overal optimality. In some cases this is possible, it depends on the mission characteristics. For instance, a reiterated attack allows to leave the target from any direction so the geometrical constraints on the route falls down and it is possible to obtain a global optimum.

The developed prototype

The developed prototype (currently it works on a Symbolics 3630 and is implemented in Common Lisp) is composed of four main modules.

- an object oriented data base containing all the necessary information to represent the territory (orography, way points, towns, roads, rivers and so on) and the tactical scenario (missiles, artillery, radars with their operative range). Furthermore the data base contains heuristic information: every way point has an identifiability diagram (given by pilots) and targets have attack diagrams pointing out goodness factor of the arrival direction on the target according to orography, attack type, target type. For each class of target (bridge, airfield etc.) there is an attack diagram drawn from the information given by pilots. These diagrams are instantiated to the specific cases using functions that take into account the local data (orientation, position etc.).

- a module exaustively generating all the permissible paths. This module uses an algorithm that combines the reducing problem into subproblem techniques with a breadth first algorithm on a graph. The problem is decomposed in tasks (from take off to the first target ... from the last target to landing): for each task every path satisfying the geometrical constraints on route (maximum number of turns, maximum turn angle, etc.) is found. In the end the task that generates the complete paths combines all the partial paths (given by the previous tasks) removing the solutions that doesn't verify the constraints in the conjunction points. The final solutions are then evaluated and ranked so to be able to choose the best ones.

Obviously such an algorithm doesn't have real time performance, but we had different reason in choosing this approach for the time being.

First: an analysis of the space of the problem states space guaranteed that although it is computationally demanding, the algorithm was executable in acceptable time and that its complexity (exponential according to the way point number of the route) had, in our case, a finite and computable maximum limit.

Second: this algorithm becomes a workbench to gather, test and evaluate the necessary knowledge to evaluate the paths quality. In fact, having all the acceptable solutions, it is possible to verify the better ones, together with the rejected ones: in this way it is possible to verify with the experts if any potential good solution has been rejected and to controll the quality of the knowledge introduced in the system.

Finally, this is the basis on which it is possible to develop other algorithms, not exaustive and almost real time, testing the quality of the found solutions and the difference from the theoretic optimal solution.

- an evaluation module based on a bayesian evaluation network reconfigurable to work with different criteria sets.

- a batch excution and evaluation module that runs a set of benchmark tests. This module allow to to execute benchmark to evaluate different algorithms from the point of view of the solutions quality.

These four module form a "dedicated shell" in which it is possible to study and to evaluate the best approach to the problem and on which it is possible to insert different planning algorithms and to evaluate their quality "on the field".

Below there are some consideration got from the first uses of this shell.

Search algorithm to generate plans

The spectrum of search algorithm extends from brute force techniques, which use no knowledge of the problem domain, to knowledge intensive heuristic search.

The problem of the brute force algorithms is their complexity that can lead to unacceptable execution time if the state spaces is not well limited. Their strength is that they guarantee not only to find always the best solution but also to find other solutions, if there are, good enough to be used. In this case pilots have the choice.

The heuristic approach uses domain knowledge intensively to guide the search. This may be in the form of cost functions and of estimating functions if a search algorithm is used (A* (6), bidirectional search BS* (4) etc.), or in the form of strategies if more complex planning tecniques are used.

In our case, search algorithm based on heuristics showed some problems: first of all some cost functions are not monotone so optimality of the solution is not guaranteed (in our experiments with a bidirectional search algorithm BS*, about 30% of the best solutions were not really the best solutions compared with the hexaustive approach). Secondly these kind of algorithm were studied to compute mainly the best solution, not a set of the best solutions. Moreover, even if we modify these algorithms to obtain the n best solutions we are not guaranteed that these solutions are effectively different each other, and pilots consider very important the possibility to choose between a set of best solutions. In most cases these algorithms find very similar solutions (e.g. they differ only for a way point sometimes near to those of the other route) and don't allow a significant choice. Obviously the exaustive algorithm too suffer from the same problem, but having all the acceptable solution, it is possible the search of a different path.

Finally, most of the classic search algorithms are optimal to find the best path between two points, but don't consider the case in which one has to connect any number of points: so they must be used together with reduction problem techniques. Furthermore, this is not enough, in fact even in this way we are not guaranteed to find the best solution, so we must also use more sophisticated mechanism to allow reasoning tecniques as the "hypothesize and test".

The development of a more sophisticated planning system that use knowledge to generate strategies to plan seems promising. The fundamental point here is the way in which the system "sees" the map. In fact pilots are able to read the map globally and to extract elements (valley or hills to hide to radars) useful to generate problem solving strategies: in other words pilots find out "macro paths" that try to refine respecting constraints. Contrarily classic heuristic search algorithm are able to read the map locally, near the point they are expanding: so it is not possible to extract global strategies. Performance and flexibility of the first approach are remarkable: for instance it is possible to use different strategies in different situations. However this is an approach still in the theoric phase and it requires a very strict interaction with pilots and expecially an interdisciplinary approach in the development phase: for instance a module to "read the map globally" may require image interpretation techniques.

The system role in the knowledge elicitation

The use of the system as a task oriented shell was particularly useful in knowledge acquisition, allowing a dialog between system and expert who is able to evaluate the difference between his solutions and those offered by the system. In this way we obtained an effective reflection on knowledge, more and more useful in respect to that obtained in theory.

Pilots interaction with the system is made by a classic multi-window interface, using mouse and menus. The interface allow a synthetic map representation on which way points, cities, rivers, highways etc. and the tactical scenario (feba and threats) are displayed (see fig. 1 at the end of the paper). It is possible to visualize on this map the different solutions (fig. 2): in this way pilot has an output similar to that commonly used. The interface allows some functions to edit the tactical scenario (create and remove of threats, feba definition), to introduce all the necessary data to define the mission (take off, landing, targets, weapons, attacks etc) and to introduce the pilot's own solution. This last is a very useful option as it allows to verify the knowledge used in the evaluation of the system solutions directly on the pilot one, verifying immediately its quality.

The possibility to modify quite quickly parts of the bayesian network used for the evaluation (e.g. changing certanty factors) allows a tight pilot-system interaction and makes knowledge refinements easier.

The direct interaction expert-system, helped by a knowledge engineer who may (when possible) correct the system, allows the expert to get involved much more in the system development and in the work team. In fact in this way of work, the expert can see his words to become quickly part of the system.

On the other hand, there are some unfavourable aspects in this approach: for instance there is the risk to focus the attention on the interface development (easy use, good display, etc.) instead of working effectively to gather knowledge This is a real risk because the time required to develop a good interface is remarkable, its complexity loads on the whole system performance (e.g page faults) and this process tends spiral.

The qualitative tests of the prototype and their evaluation

Tests were executed on a limited geographic scenario (about 200 x 250 km) using the exaustive algorithm. The tests objective was to verify the correctness and consistency of the data necessary to generate the solutions and to control their acceptability and quality. As we were not interested in verifing the algorithm performance, the algorithm used (clearly not real time) and the limited scenario didn't affect the tests results.

Tests followed the modalities described below:

- Definition, together with pilots, of three tactical scenario (one only with the feba, the others with SAM-6 and SAM-7 missiles and anti aircraft artillery)

- Definition, together with pilots, of a meaningful test set in order to cover the different possibilities (take off, landing, different type of targets, different number and kind of attack) The test set was applied to the three tactical scenario.

- Pilots were given a map with the same information stored in the system, in order to work in the same situation as the system (pilots uses a lot of information to plan their mission, some of them as the power transmission lines are drawn on the aeronautical chart but aren't stored in our system).

- Pilots were asked to plan the mission of the test set alone. At the same time the system planned the same missions.

- Pilots were asked to evaluate the best solutions proposed by the system and the system evaluated the solutions proposed by pilots. The system works in an exaustive way finding every possible solution and choosing the best ones: so from now on when we say "different solutions" proposed by pilots we intend "different from the best solutions proposed by the system".

Sometimes pilots found different solutions, in spite of the exaustive way of work of the system. This is because the system finds every possible route satisfying the geometrical constraints, pilots instead may find solution lightly out of the admissible ranges. We developed a module for the introduction of calculate solutions specially for this cases, in order to undergo them to the system evaluation.

- Analysis of the results. We consider the quality level given by the pilots and by the system, and we extract the points of the routes meaningful different. In fact sometimes there were equivalent points near each other so in this case the choice and therefore the difference in the route wasn't important. It was particularly important to evaluate the difference in the choice of the push up and escape point. These last kinds of difference involve different approach to the target and therefore an inconsistency between the tactical knowledge used by pilots and that of the system.

From the tests analysis it results that pilots often gave different solutions to each other. One of the pilots who made the tests was the leading expert for the whole project development: as we thought the system found solutions nearer to the leading expert ones. This shows the subjectivity of part of the system knowledge necessary for this kind of work, but this is also a demonstration of the flexibility of the artificial intelligence techniques in representing and using not well formalized and highly subjective knowledge.

In any case, as regards the quality of the solutions, no one was substantially different from the solution proposed by pilots. Pilots evaluated some of the system solutions better than theirs and didn't consider any system solution decidedly inferior.

A more detailed analysis showed that 40% of the push up points and 50% of the escape points corresponded with those of the reference pilot, while the proportion became lower with other pilots. On average the increment of the number of targets corresponds to an increment of the difference. Sometimes the system revealed a lack of fuel, but repeating the same mission with more fuel, the system found meaningful better solutions. In the cases in which the push up points in the system and pilots solutions were different, the respective single evaluation (of the push up point) was equivalent (see fig. 3 at the end of the paper).

As regard the quality of the overall solutions, the opinion was good. There was only one critique: the system often passes nearer the threats than pilots do. Pilots prefer to avoid threats, passing far from them even if they are sure of threats location: especially when weather is not good and visibility is poor it becomes very important to be sure to get out of the operative range of the threat. Some of the differences were due to this consideration. In any case this is an information about the necessity to proceed to another and more intense phase of knowledge acquisition.

In general the main needs verified during the execution of the test set were of three kind:

- the need to improve the geographic data base (both in extension and resolution) in order to have a better agreement with the operative use of the system.

- the need to give more flexibility to the system in order to release part of the constraints when it is necessary

- the need to continue the process of knowledge acquisition about tactical aspects in order to cover a greater number of attack modalities and to consider a greater number of type of target.

Future works.

Future works will carry a transformation of the prototype from a dedicated shell to a power and engineered version to be used on the field.

The two main items will be:

- the tuning and engineering of all the modules that were developed with essential functionalities as they were necessary but not crucial to the system.

- the development of the final planning module, after the test of other search algorithms and more and new refined planning tecniques.

Amongst the first kind of modules, the geographic data base will be one of the more interested by the growth: it will be able to treat more and more data both because we will enlarge the territory and because the information will have a greater resolution level. Moreover, it will be necessary to develop the kernel of a DBMS (data base management system) in order to manage data stored on files, reducing the use of the virtual memory (actually widely used) and optimizing the response time.

As regard the second point, we will gather and organize new and deeper knowledge especially about the tactical aspects (attack modalities according to the diffrent type of target and of weapons, run in direction selection etc.). It will be very important to determine the main strategies that pilots uses in the generation of flight plans and tha way in which the system can "read the map from a global point of view".

The results obtained lead us to consider useful the followed approach to the development of a ground station to plan the operative mission.

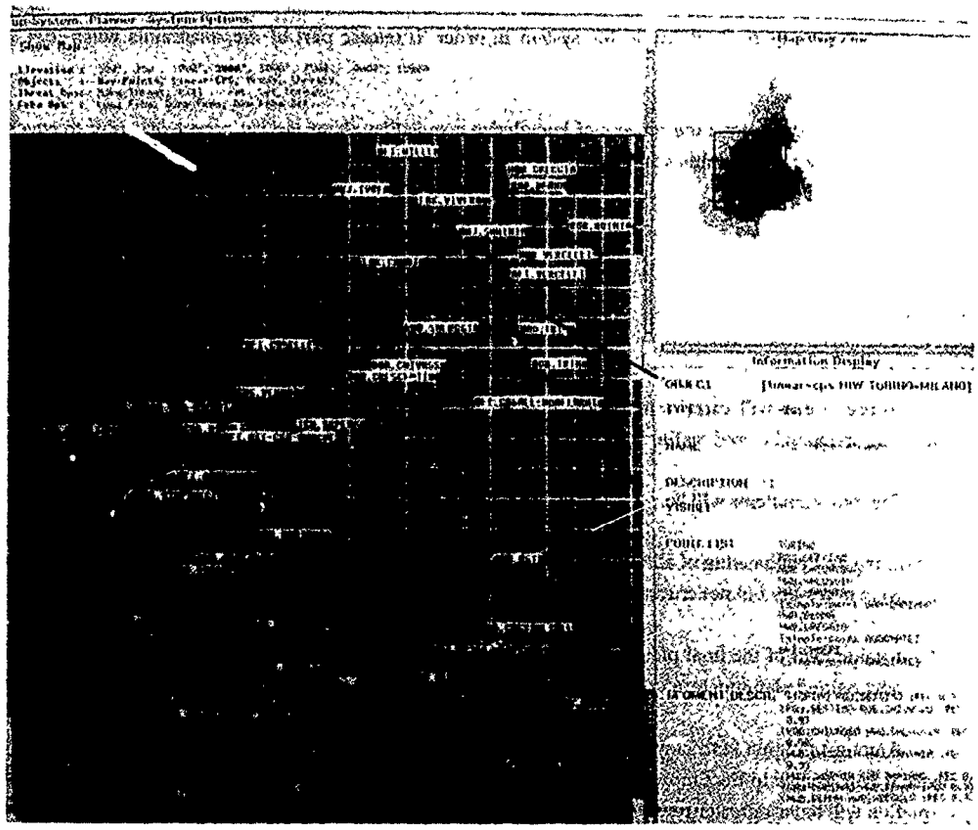


Fig. 1 . This photo shows the Marples interface. The entire territory is shown in the right upper window, the little square indicates the part of the territory displayed in the big window on the left. Every square of the map (in the big window) represents a 5x5 km zone. It is possible to represent the territory with different elevation thresholds (3000 ft. in the photo). The red zones are zones higher than the choosen threshold, the green zones are lower. In this photo way points, highways (red lines), rivers (blue lines), roads (black lines) are displayed.

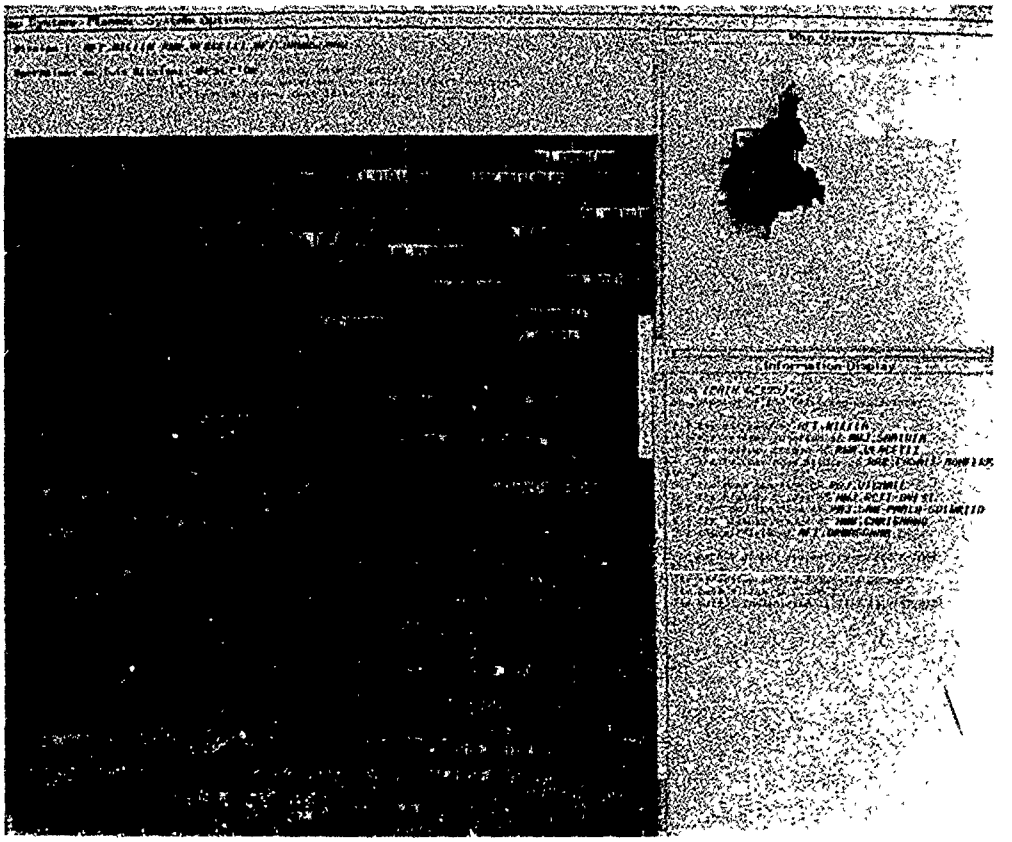


Fig. 2 . The line crossing the entire territory (right upper window) represents the FEBA, the right side of the FEBA is the enemy one. The brown zones in the big window are dangerous because beyond the FEBA. Moreover, a threat is set in right lower part of the window; the dangerous zones included in the operative range of the missile are orange. In this photo two different routes are displayed: with the threat the best route is the pink one (the lower route), without the threat the best is the black one.

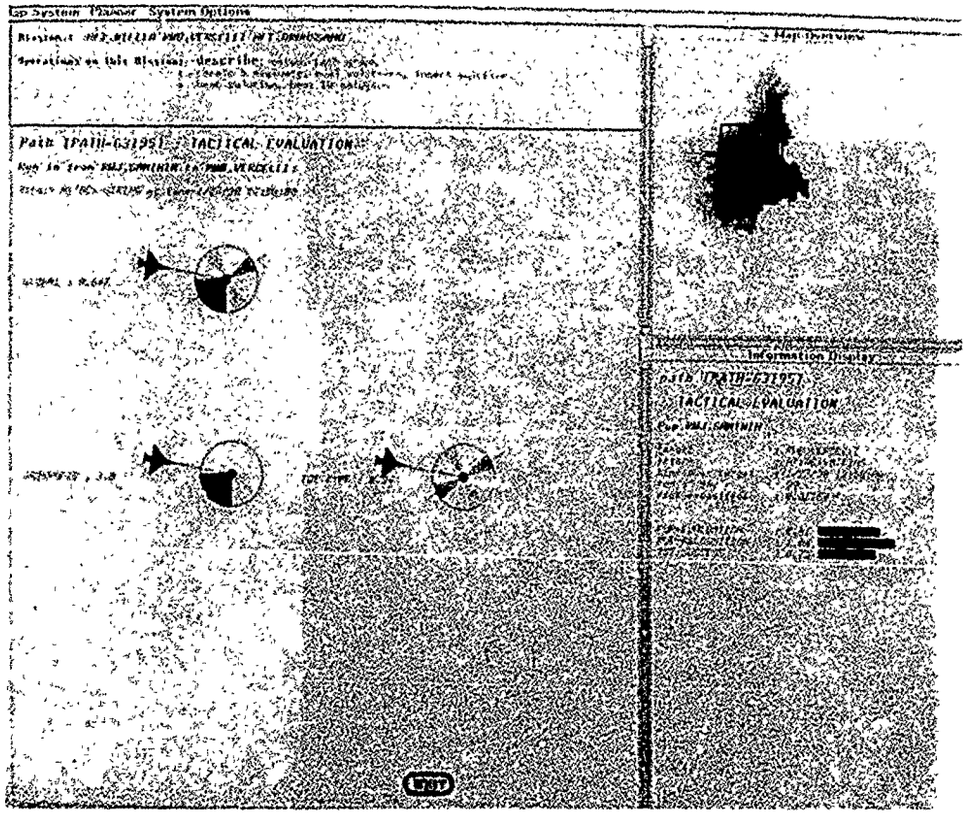


Fig. 3 . In this photo the tactical evaluation of a push up point is dislayed. The push up point is the railway junction of Santhia (rwj.santhia) and the target is the railway bridge of Vercelli (rwb.vercelli). The overall evaluation is shown in the Information Display window and depends upon the push up recognizability and upon the probability of success on target. Moreover this last depends on other factors: one is the goodness of the arrival direction on target that depends on the target type and on the orography near the target (see central window tgt-type and orografic)

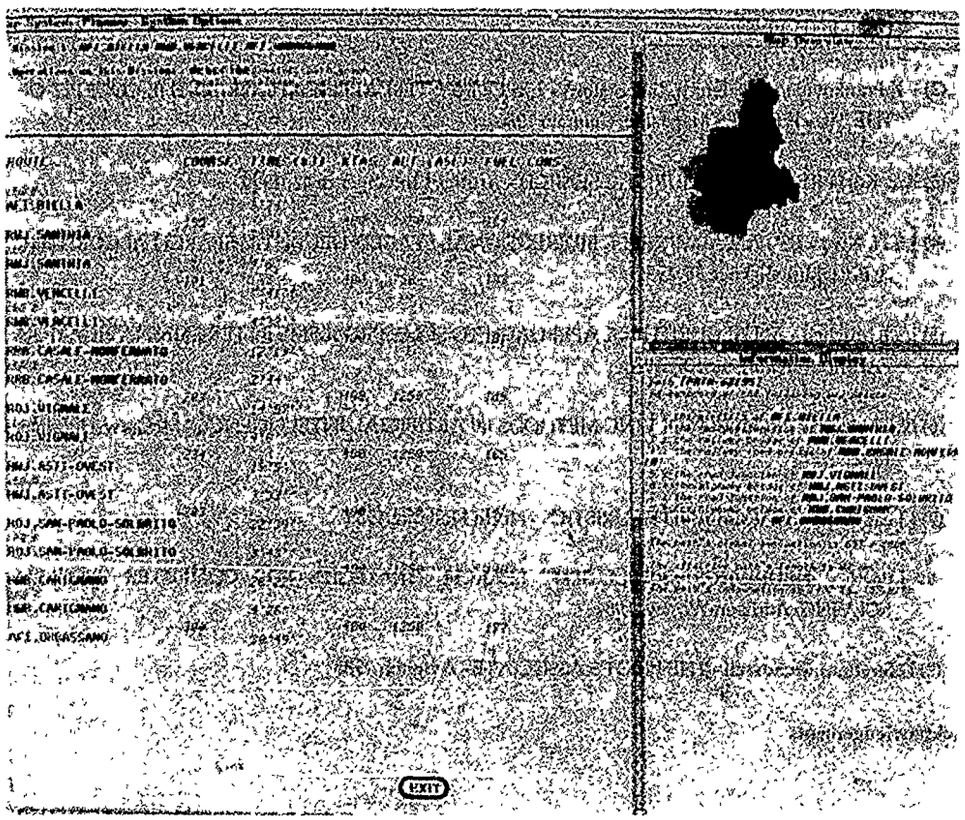


Fig. 4 . This is the folder of the best route described on the right window.

BIBLIOGRAFIA

(1) P.Gallo,D.Dabbene, F.Luise, P.Giordanengo - EXPERT SYSTEM FOR PILOT ASSISTANCE: THE CHALLENGE OF AN INTENSIVE PROTOTYPING. Applications of Artificial Intelligence VII, Orlando (FL) 1989

(2) - K.Frankovich, K.Pedersen, S.Bernsteen - EXPERT SYSTEM APPLICATIONS TO THE COCKPIT OF THE '90s , IEEE AES Magazine, Gennaio 1986

(3) R.E. Korf - REAL TIME HEURISTIC SEARCH - Artificial Intelligence 42, 1990

(4) J.B.H. Kwa - BS*: AN ADMISSIBLE BIDIRECTIONAL STAGED HEURISTIC SEARCH ALGORITHM. Artificial Intelligence 38, 1989

(5) J.S.B.Mitchell - AN ALGORITHMIC APPROACH TO SOME PROBLEMS IN TERRAIN NAVIGATION - Artificial Intelligence 37, 1988

(6) N.J. Nilsson - PROBLEM SOLVING METHODS IN ARTIFICIAL INTELLIGENCE - McGraw-Hill, Inc., 1982

(7) J.Pearl (edt) - SEARCH AND HEURISTICS - North Holland, 1983

(8) J.Pearl - HEURISTICS, INTELLIGENT SEARCH STRATEGIES FOR COMPUTER PROBLEM SOLVING. Addison-Wesley, 1984

(9) D.Smith,M.Broadwell - THE PILOT'S ASSOCIATE - Avignon 1988

Acknowledgements

We wish to acknowledge for their precious contribution Mr. N. Bragagnolo, chief test pilot at AERITALIA (Defense Aircraft Group), our leading expert during the whole project development and Mr. C Calzoni, test pilot at AERITALIA (Defense Aircraft Group).

KNOWLEDGE-BASED COCKPIT ASSISTANT FOR IFR OPERATIONS

R. ONKEN

Universität der Bundeswehr München
Institut für Systemdynamik und Flugmechanik
Werner-Heisenberg-Weg 39
D-8014 Neubiberg

SUMMARY

A knowledge-based cockpit assistant for IFR (Instrument Flight Rules) operation is presented, aimed at improvement of situation assessment and performance increase by computer aids for flight planning and plan execution. Here, situation assessment also includes monitoring of the pilot's own activities. The modular system structure is described as well as the individual system modules. The cockpit assistant was tested in a flight simulator by professional pilots under realistic IFR-scenarios. The concept of the test design as well as test results are presented. The system design goals are mainly confirmed by these results.

1. INTRODUCTION

IFR operation is a standard for commercial air traffic in order to warrant arrivals on schedule under adverse weather conditions. This is also true for a substantial portion of the general aviation traffic. However, lack of visual cues from outside the airplane as well as increased automation and complexity of cockpit instrumentation can result in overdemanding loads on the pilot. This leads to a significantly higher number of accidents in IFR operation, in particular for single-pilot-IFR flights [1,2,3].

Investigations on the cognitive behavior of humans [4,5] have revealed that electronic cockpit assistance for the pilot has a good chance of becoming effective for

- situation assessment
- planning and decision making and
- plan execution

by complementing human capabilities and not taking away from him all challenges.

On the basis of this formal body of knowledge of the user needs a cockpit assistant, called ASPIO (assistant for single-pilot IFR operation), has been developed and implemented in a flight simulation facility, which is described in the following.

An extensive test program has been conducted in order to provide data of how performance improvement, workload balancing and pilot acceptance can be achieved by the aforementioned cockpit assistant functions.

2. STRUCTURE OF COCKPIT ASSISTANT

Conceptually, from the very beginning of the development, considerable emphasis has been put on taking into account the interdependence effects between the cockpit assistant , the pilot, the ownship systems as well as the air traffic and atmospheric environment, looking for the single-pilot application in the first place. Consequently, the cockpit assistant has got three main interfaces with its environment, i.e. with ATC (Air Traffic Control), with the pilot and with the other onboard systems of the aircraft (fig.1).

With regard to the ATC interface, a digital two way data link is posited. This assumption results in ATC instructions being directly fed into the assistant system and, at the same time, being aurally presented to the pilot.

The pilot interface makes excessive use of speech communication in either direction. A speaker dependent speech recognition system in single-word mode was used for pilot messages towards the assistant. The vocabulary of about 140 words and word strings, in accordance to civil aviation terminology, covers the

- terminology used by the pilot for flight guidance management (e.g. descend to flight level ...)
- command inputs towards the cockpit assistant
- command inputs into other aircraft systems
- numbers
- instantiations for airfields and navigational aids.

A syntax structure of several levels is implemented to ensure sufficient reliability and promptness of recognition (fig.2). Synthetic speech is used for speech output, with different voices for different categories of assistant messages.

More complex information like comprehensive flight plan recommendations is presented visually.

The aircraft interface is established by a data pool which contains all aircraft relevant data about

- flight status (including control variables such as autopilot settings)
- radio navigation settings
- radio communication settings
- status of aircraft subsystems (e.g. engines).

There are three main functional blocks according to the specified functions for planning (including situation assessment), plan execution and monitoring. The planning function is provided by the automatic flight planner (AFP) module. The plan execution function is broken down into submodules, i.e. the model of the pilot-flying (MPF), the automatic pilot-not-flying (APNF), which is more or less comprising what is to be executed by the co-pilot in a two-man cockpit, and the autopilot (AP). These modules, together with the module for plan execution monitoring (MON), are described in the following.

Automatic Flight Planner (AFP)

For every flight, a flight plan (including destination) must be issued before take off. This flight plan could be worked out by the AFP, but it will usually be prepared by means of other facilities and will be taken for the AFP initialisation.

On the basis of the actual flight plan, the AFP performs a rigorous evaluation of the elements of the current situation and its future projection, which might reveal that the actual plan is no longer executable. Causes could be actual or anticipated deviations from the flight plan and such events as new ATC instructions not in accordance with the flight plan, adverse weather conditions etc. For instance, this evaluation will activate the planning function by means of a problem solving algorithm for the selection of an alternate destination and corresponding generation of a new flight plan.

For the representation of the necessary AFP knowledge in the IFR domain, the method of problem reduction is used (fig.3). This representation can be used for both the situation evaluation and the planning itself. For some of the planning decisions, fuzzy criteria are used following the theoretical foundations of fuzzy sets in [6].

The AFP planning results are presented to the pilot as recommendations, and if not corrected by the pilot, these planning results replace former flight plan commands for plan execution.

Model of pilot-flying (MPF)

On the basis of the flight plan as generated by the AFP and acknowledged by the pilot, the MPF can perform automatic management of flight plan execution. The rather coarse command structure of the flight plan of the AFP is broken down by the MPF into action sequencies like

- speed reduction
- flaps actuation
- gear down
- final check
- etc.

for the flight phase of the final approach.

These sequencies follow the regulations of piloting and ATC, which are extensively proceduralized. These procedures, being automatically performed by the MPF, cannot be different from what the pilot is supposed to put into effect. Therefore, this module can also be construed as a model of the pilot, of which the output can be utilized for pilot monitoring purposes, too.

The MPF module can be considered as almost exclusively rule-based. This rule base is essentially structured by the goal/task hierarchy of the overall flight task. The tasks (or respective rules) are either pertinent to flight phases or are dependent on event occurrences, represented by scripts and productions, respectively.

Automatic pilot-not-flying (APNF)

The APNF represents the effector module of the cockpit assistant together with the autopilot (AP) module. This module comprises all functions usually performed by the co-pilot in the conventional two-man cockpit. Among these functions are instrument setting flap and gear setting, ATC communication, checklist execution and monitoring (the latter is directly passed on to the monitoring module MON). There are also standard callouts as usually delivered by the co-pilot in the conventional two-man cockpit. The APNF performs these callouts via speech messages. The APNF can also be directly tasked by the pilot with respect to navigational calculations or requests about flight-relevant information (radio stations, alternates, etc.).

There are two modes of tasking the APNF. This is done either by the pilot or by the cockpit assistant itself, depending on where the responsibility of flying the aircraft is placed. The pilot may hand over this responsibility to the cockpit assistant in the same way as he can pass it on to the co-pilot in a conventional two-man cockpit. In this case, the MPF module will accept the role of tasking the APNF and will make use of the effecting capabilities of the APNF module as well as of the AP, as described above. The APNF also sets the AP modes and the command values. If the pilot has control, the AP may or may not be engaged according to the pilot preference.

Monitoring Module (MON)

This module works on the basis of the actual outputs of the pilot, i.e. the resulting flight status, and the MPF outputs. Deviations from the flight plan can be detected by comparing these data during flight. It thereby comprises an important segment of the situation assessment task. This module distinguishes three categories of monitoring functions:

- Monitoring of flight progress ('nodal attention checks') in order to be aware of the arrival at any subgoals of the flight plan
- Monitoring of health status of aircraft systems
- Monitoring of pilot actions and their compliance with those required by the actual flight plan

In figure 4, the normal mode of operation is depicted which is mainly investigated. For this mode, which was favored in the first place, an extensive test program is conducted. Figure 4 shows the potential pilot inputs into the cockpit assistant and the data flow with regard to the monitoring function. The advices or warnings, as generated by the MON, are tansferred via the speech interface. This is one mode among a number of other possible ones which can be distinguished by the degree of charging the cockpit assistant with tasks otherwise performed by the pilot. In case of pilot incapacitation the ASPIO functions can be used for autonomous emergency flight.

3. IMPLEMENTATION AND EVALUATION EXPERIMENTS

For the experimental investigation the cockpit assistant is implemented in a flight simulator facility with a one-seat fixed base cockpit (fig.5). Besides aircraft dynamics (6-degrees of freedom model of the HFB 320), autopilot (AP), radio navigation systems and wind characteristics are simulated in this facility. The pilot interface includes computer-generated outside vision, artificial stick force as well as speech input and output in order to closely match the speech dialogue in a two-man cockpit environment. As an option, instead of pressing keyboard buttons, the speech system can also be used by the pilot for all kinds of inputs into the aircraft systems. The functions of the cockpit assistant are implemented in a VAX and a PS/2 computer. Some implementation parameters are given in table 1.
The experiments were aimed at proving enhancements in overall system performance, also with regard to safety, pilot workload balancing and pilot acceptance under the most possible realistic conditions. For this purpose, the following criteria were investigated:

- Accuracy of flight
- Pilot errors in situation assessment and system operation
- Duration and quality of planning processes
- Pilot workload
- Pilot acceptance

The pertinent parameters were evaluated for a number of test runs of IFR-approaches. These test runs were organized in the way as shown in table 2. Three different IFR scenarios [7] were developed which consisted of standard situations as well as of unusual events and emergency situations. Scenario 1 is exclusively standard. Therefore, it was possible for this scenario that all pilots flew both test runs, with and without cockpit assistant (CA). However, in scenario 2 and 3 planning and decision-making is triggered by unforseeable events. Each pilot could only have one test run in either of these scenarios, either with or without cockpit assistant. A total of nine professional pilots, everyone with a great amount of IFR flight experience, have performed test runs for the three different scenarios. A great amount of experimental data has been provided. Some of the evaluation results are presented in the following.

As an indicator for flight accuracy, the airspeed deviation was considered to be most appropriate for evaluation. Figure 6 shows typical time histories for scenario 1. Figure 6a illustrates the deviations occurring without the cockpit assistant, as opposed to figure 6b, which shows much smaller deviations for the case of activated cockpit assistant. Both flights are performed by the same pilot. The evaluation results for the standard deviation for all pilots, as shown in figure 7, demonstrate a highly significant improvement in flight accuracy for the flights with activated assistance functions. For scenario 2 the results were even better, for scenario 3 not quite as good as for scenario 1.

Pilot errors could be directly and unambiguously detected. One can say that, by definition, no major deviations from normal flight could be observed for the flights with activated monitoring function of the cockpit assistant. However, without cockpit assistant, pilot errors were detected, also more serious ones.

The time needed for planning and decision-making was determined by the difference in time between a triggering ATC instruction, which also demanded for a reply about the pilot's further intentions, and the pilot's reaction. In figure 8, the pilot's decision time without support by the cockpit assistant, following an unexpected ATC message in scenario 2, is shown in comparison to the time elapsed until pilot reaction for acknowledgement of the flight plan recommendation of the cockpit assistant was detected. The planning and decision task was to check for an alternative approach procedure under consideration of weather and airport data, after a message was received about the breakdown of the instrument landing system at the destination airport. The differences are obvious. In addition, the pilots stated in the debriefing session that all decisions recommended by the cockpit assistant, made sense.

The pilot workload was assessed by means of subjective rating and the introduction of a secondary task. For subjective rating, the SWAT-method (Subjective Workload Assessment Technique [8]) was used. As a secondary task, periodical tapping [9] was specified. The results show a slight reduction of workload, but without significance.

Pilot acceptance was evaluated from pilot statements in a questionnaire which was furnished by the pilots after their test flights. Among other ratings, the technique of semantic differential [10] was used, as shown in figure 9. The median values show positve mean reactions, i.e. good acceptance by the pilots. The neutral overall assessment for the component "not distracting/ distracting" was explained by the pilots by certain lack of familiarization in system handling, in particular with respect to the specific speech recognition system used for these experiments.

4. CONCLUDING REMARKS

A knowledge-based cockpit assistant for piloting tasks in IFR aviation can offer great benefits in monitoring, planning and decision-making for complex situations. The system can rapidly offer recommendations to the pilot without getting 'tired' or

loosing information. The recommendations are derived by evaluation of alternative solutions against objective criteria. The presented system specially is intended for the single-pilot IFR operation. However, most of its essential functional elements can be applied for the two-man cockpit as well. The system is able to understand the situation on the basis of knowledge about facts and procedures of the piloting task environment and the actual data about the flight status and pilot actions. The situation can be evaluated with regard to disturbing impacts on the flight plan and, if necessary, it derives a revised flight plan as a recommendation to the pilot. It also can serve the pilot for plan execution tasks in a similar way as the co-pilot is working in the two-man crew. Speech dialogues as known from the two-man cockpit remain essentially unchanged because of the integration of a pilot interface with speech input and output.

The flight simulator experiments for testing the cockpit assistant under realistic conditions, including the air traffic environment, lead to the conclusion that this kind of system actually can complement human capabilities and is accepted by the pilots. Flight accuracy was significantly improved, adverse consequences of pilot errors were eliminated and also planning and decision- making was improved. It could be shown that pilot workload was slightly reduced, although the pilot interface was not optimized at this stage of development.

5. REFERENCES

[1] Harris, D.F.; Morrisette, J.A.
Single Pilot IFR Accident Data Analysis, NASA Contractor, Rep. 3650, 1982

[2] Chambers, A.B.; Nagel, D.C.
Pilots of the Future: Human or Computer?, Communications of the ACM, Vol. 28, No. 11, 1985

[3] Schänzer G.
Sicherheitsphilosophien im Luftverkehr, Mitteilungen der TU Braunschweig, Heft I, 1989

[4] Rasmussen, J.
Skills, Rules and Knowledge; Signals, Signs and Symbols, and other Distinctions in Human Performance Models, IEEE-SMC-13, No. 3, 1983

[5] Rumelhart, D.E.; McClelland, J.L.
Parallel Distributed Processing, MIT Press, Cambridge, 1986

[6] Zadeh, L.A.
Fuzzy Sets, Information and Control, Vol. 8, 1965

[7] Dudek, H.-L.
Die Szenarien zur Bewertung des Pilotenunterstützungssystems ASPIO, Interner Bericht an der UniBwM, Institut für Systemdynamik und Flugmechanik, 1989

[8] Schick, F.-V.; et al.
Validation of the Subjective Workload Assessment Technique in a Simulated Flight Task, DFVLR-Forschungsbericht 89-01, 1989

[9] Michon, J.A.
Tapping Regularity as a Measure of Perceptual Motor Load, Ergonomics, Vol. 9 No. 5, 1966

[10] Bergler, R. (Hrsg.)
Das Eindrucksdifferential, Verlag Hans Huber, Bern, 1975

[11] Onken, R.; Dudek, H.-L.
Assistant for Single-Pilot IFR Operation (ASPIO), IJCAI Workshop on Integrated Human-Machine Intelligence in Aerospace Systems, Detroit, 1989

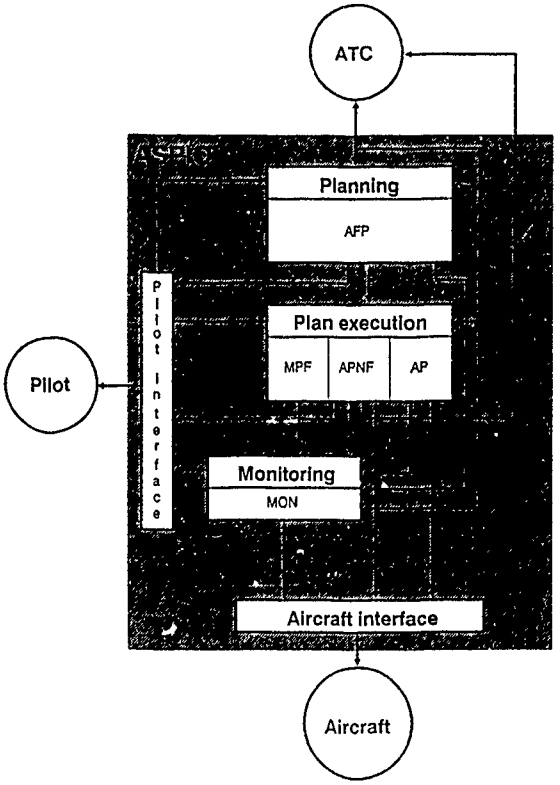


Figure 1: Structure of cockpit assistant

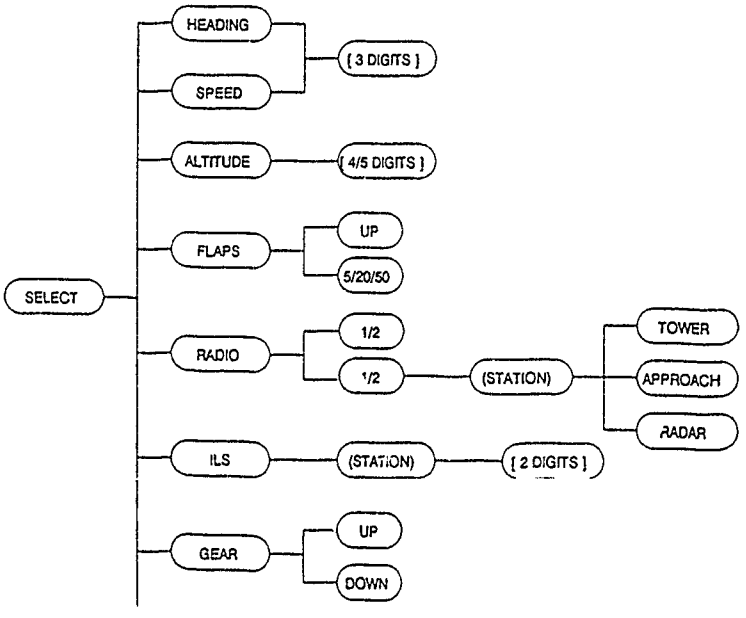


Figure 2: Part of syntax structure for speech recognition

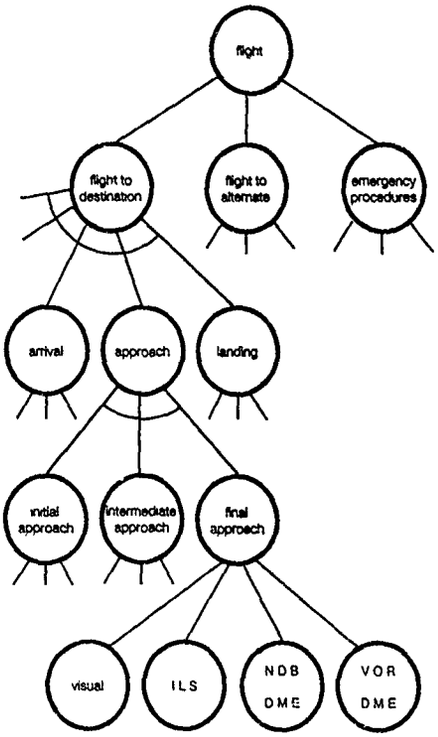


Figure 3: Structure of problem space

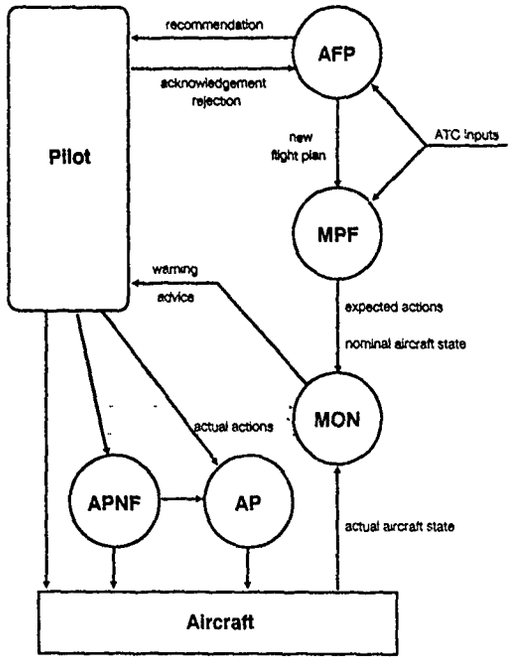


Figure 4: Normal mode of operation

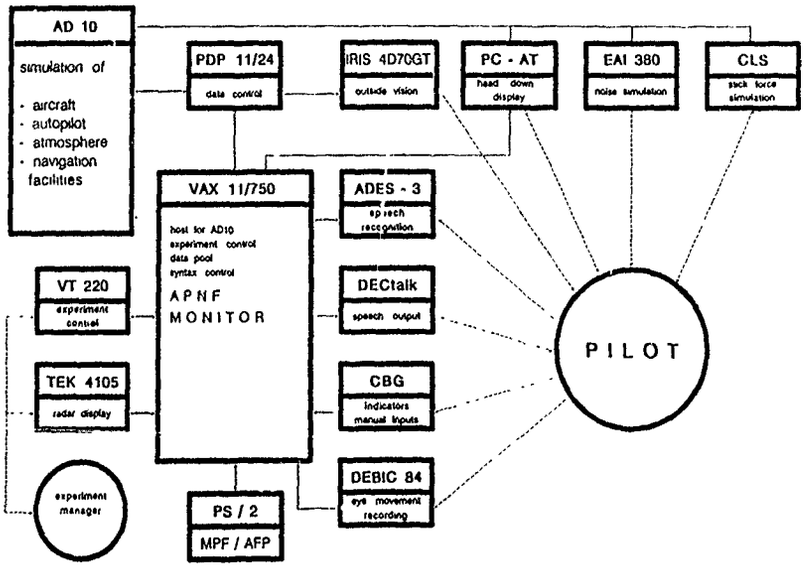
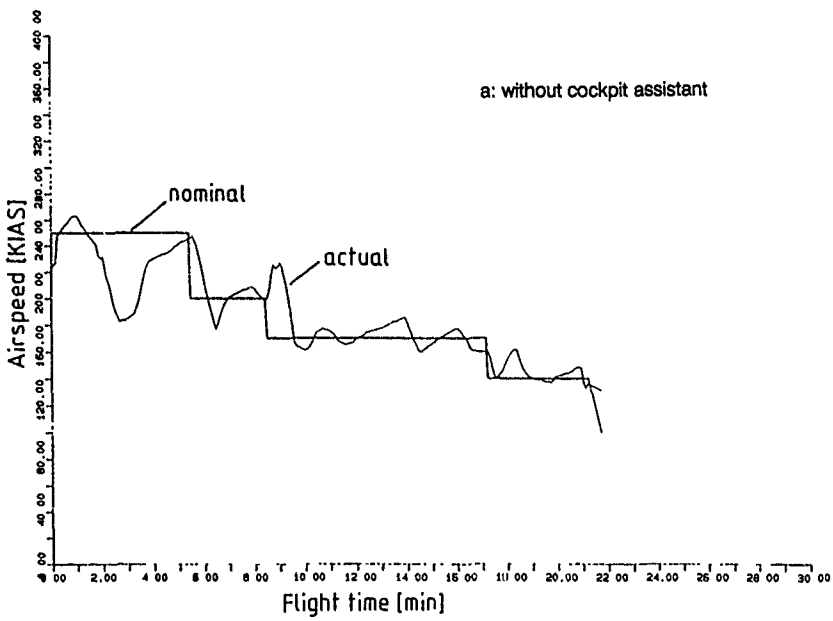


Figure 5: Flight simulation facility



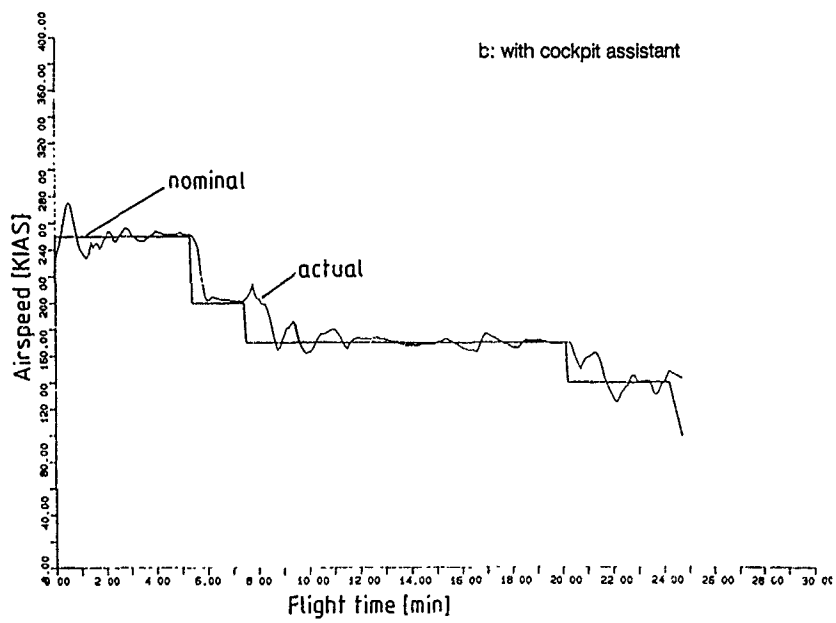


Figure 6: Typical time history of airspeed

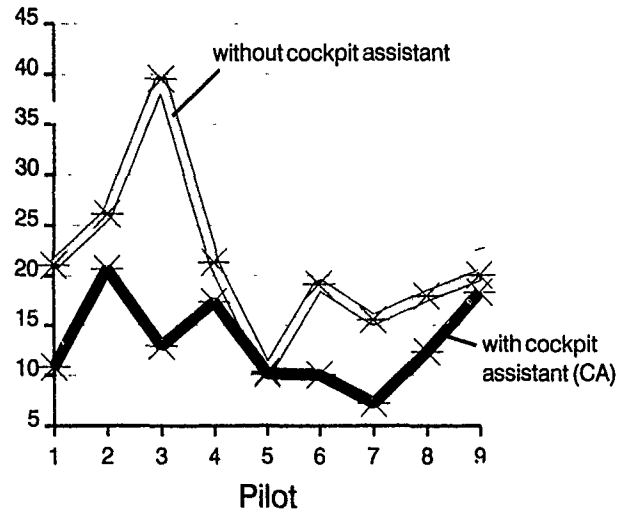


Figure 7: Standard deviation of airspeed for different pilots

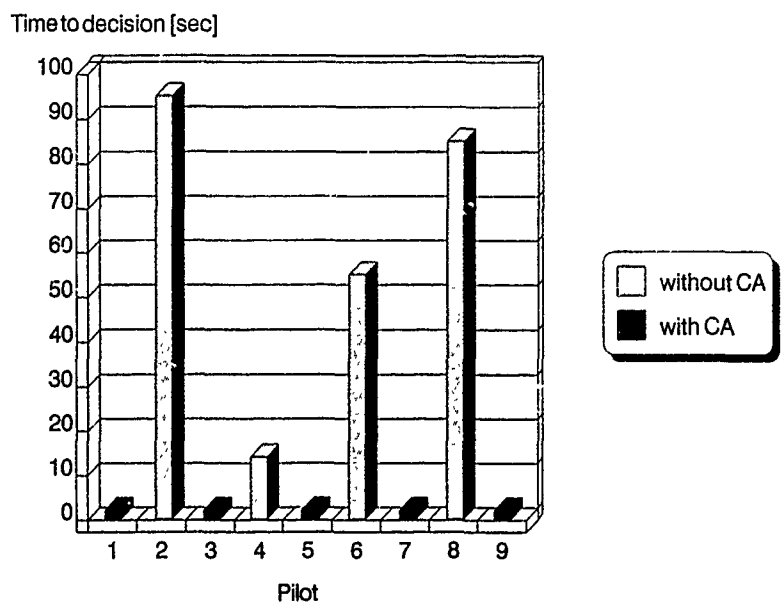


Figure 8: Time to decision, scenario 2

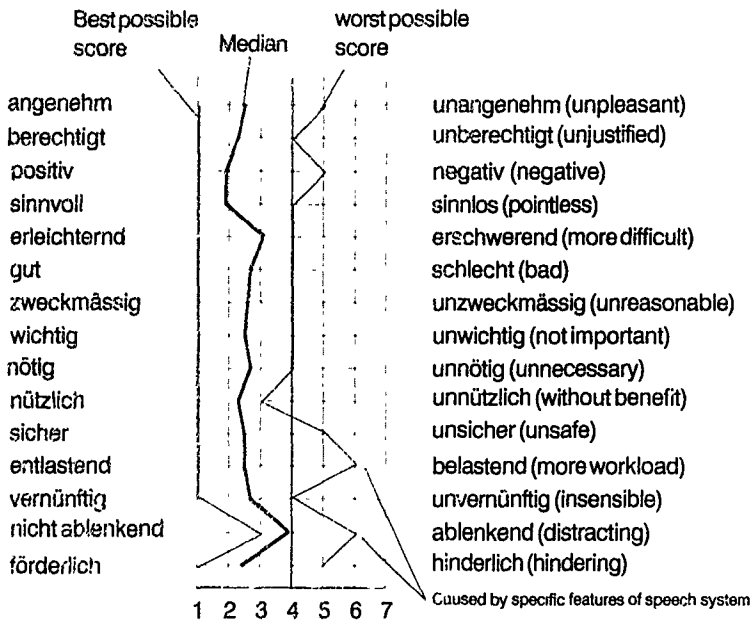


Figure 9: Questionnaire result for the semantic differential "The cockpit assistant is..."

	Realized in	Computer	Rules	Lines Source Code	Program Size (KByte)
AFP	C	PS/2 - 60 (UNIX)	1040	9440	330
MPF	C	PS/2 - 60 (UNIX)	850	5300	100
APNF	Fortran	VAX 11-750 (VMS)	470	2340	82
MON	Fortran	VAX 11-750 (VMS)	720	4110	57
ADES syntax control	Fortran	VAX 11-750 (VMS)	1080	4660	44

Tabelle 1. System implementation parameters

Pilot	Scenario 1		Scenario 2		Scenario 3	
	without CA	with CA	without CA	with CA	without CA	with CA
1,3,5,7,9	X	X	-	X	X	-
2,4,6,8	X	X	X	-	-	X

Tabelle 2: Organisation of experiments with the scenarios 1, 2 and 3

UN SYSTEME D'ACQUISITION AUTOMATIQUE DE CIBLES

par

Philippe Valery
Thomson-CSF Division Electronique de Missiles
23—27 rue Pierre Valette
92242 Malakoff Cedex
France

RESUME

Cet article présente un système de détermination automatique de silhouette d'un véhicule sur imagerie infra-rouge.
Cette reconnaissance s'appuie sur l'utilisation explicite d'un modèle de l'objet à identifier. Le formalisme objet, mis en oeuvre pour représenter le modèle permet de dépasser le cadre d'une simple modélisation géométrique et photométrique par l'introduction, dès la phase de conception du modèle de connaissances sur le fonctionnement du système.
Le processus de reconnaissance est fondé sur un schéma général de prédiction-vérification d'hypothèses et exploite un ensemble de primitives de type "régions homogènes" afin de réaliser l'interprétation de l'image. Enfin, un mécanisme d'interaction entre le module d'interprétation et le processus de segmentation bas-niveau permet de contrôler la phase d'amélioration de la silhouette en utilisant les informations du modèle.

INTRODUCTION

Un des axes principaux du développement des performances des systèmes d'armes modernes concerne l'amélioration de leurs capacités sensorielles.

L'identification automatique ou semi-automatique d'objectifs constitue l'une des fonctions principales des systèmes de veille, de reconnaissance, ou bien des autodirecteurs de missiles.

Le système présenté dans cet article se place plus particulièrement dans le cadre de cette dernière application. Dans le domaine des missiles autonomes tirés à distance de sécurité, l'identification de l'objectif devient une nécessité, tout particulièrement dans le cas d'une cible mobile pour laquelle un simple guidage inertiel ne saurait être suffisant.
La fonction d'acquisition automatique de cible s'avère également primordiale afin de mettre en oeuvre des traitements de résistance aux contre-mesures.

L'objectif du système RUBIS (RUle-Based Identification System) est la détermination automatique de la silhouette d'un véhicule sur imagerie infra-rouge. L'extraction de silhouette s'inscrit dans un cadre plus large concernant l'ensemble de la fonction d'acquisition automatique d'objectif.

Le concept d'acquisition d'objectifs de petite taille (aéronefs, véhicules terrestres, cibles pour lesquelles aucune préparation de mission n'est effectuée) recouvre un ensemble de fonctions de complexité croissante :

- la détection ;
- la reconnaissance ;
- l'identification.

La détection de cibles consiste en une recherche d'objets dans l'image pouvant constituer des objectifs potentiels. Les critères de discrimination pris en compte lors de cette étape sont relativement rudimentaires et les algorithmes de traitement des images, pour la bande spectrale infra-rouge, sont généralement fondés sur une extraction de points chauds et le pistage de ceux-ci.

La reconnaissance de cibles a pour objectif la classification des cibles potentielles suivant leur catégorie et le rejet des fausses alarmes et des leurres.

L'identification des cibles constitue la phase ultime du processus d'acquisition et a pour finalité la détermination, pour chaque objet, de son type.

La détermination de silhouette consiste à localiser précisément l'objectif en extrayant parfaitement sa forme dans l'image. Cette opération est effectuée sans utiliser de connaissances a priori sur l'échelle et sur l'orientation de l'objet dans l'image.

RUBIS met en oeuvre des techniques issues du domaine de l'Intelligence Artificielle afin de réaliser une segmentation initiale de l'image en régions puis d'effectuer l'interprétation de cet ensemble de primitives en utilisant un modèle du véhicule à identifier.

Le processus d'interprétation suit un schéma général de prédiction-vérification d'hypothèses en utilisant les éléments remarquables de l'objet pour émettre une hypothèse de vision (point de vue, échelle, position de l'objet dans l'image). La phase de vérification tente ensuite de confirmer ou d'infirmer cette hypothèse par identification précise de chaque élément constituant le véhicule.

L'affinage progressif de l'interprétation peut conduire à la définition de requêtes de recherche particulières de régions dans le cas d'objets manquants ou mal segmentés lors de la partition initiale. Cette interaction du module d'interprétation avec les traitements de segmentation permet d'améliorer sensiblement les performances de RUBIS.

Nous présenterons successivement le module de segmentation, le processus d'interprétation ainsi que le schéma de définition des modèles puis nous donnerons des résultats quantitatifs obtenus sur une base d'une trentaine d'images FLIR 8-12 μm représentant un blindé sous un ensemble d'attitudes variées.

LA SEGMENTATION EN REGIONS

La tâche de ce module consiste à fournir au processus d'interprétation un ensemble de primitives suffisamment précises pour entreprendre la mise en correspondance entre les indices visuels extraits de l'image et les objets ou parties d'objets du modèle de référence.

Le choix des primitives de type région est justifié par l'application. Il s'agit en effet d'identifier des véhicules relativement mal résolus dans l'image ; les dimensions de leur silhouette n'excédant pas quelques dizaines de pixels. L'utilisation d'un mode d'interprétation fondé sur des primitives géométriques (segments de droite) ne saurait convenir car la précision des primitives est trop faible. Pour un segment de longueur égale à 10 pixels, une erreur de positionnement de 1 pixel sur chaque extrémité conduit à une erreur de 11° sur l'orientation du segment. Par ailleurs, la répartition des gradients d'intensité sur la surface de la cible ne rend pas toujours compte des différentes parties du véhicule et les segments de contour obtenus n'ont pas d'homologues dans un modèle de type CAO.

La définition des cibles en termes d'assemblages de régions de luminance homogène permet d'atteindre la robustesse indispensable à notre application. Toutefois, l'utilisation d'une représentation en régions de l'image ne fournit pas suffisament d'informations pour être en mesure de calculer précisément l'orientation de la cible. Dans le cas des objectifs mobiles de dimensions réduites, cette information n'est pas nécessaire car la connaissance approximative de l'attitude de l'objet dans l'image est suffisante pour localiser le point d'impact optimal.

Le module de segmentation est divisé en deux ensembles principaux. D'une part, un traitement de segmentation en régions fondé sur une approche de division récursive et d'autre part une base de règles de production qui analysent localement l'ensemble des régions et déclenchent l'application des actions correctrices de division ou de fusion.

Les traitements de segmentation initiale sont fondés sur deux principes importants :

- un seuillage multiple des niveaux de gris ;
- une approche récursive des divisions.

Ce processus est initialisé par la recherche d'un seuil sur l'image initiale. Le seuil est appliqué sur l'image et une première partition en régions connexes est déterminée. Pour chacune des régions ne satisfaisant pas le critère d'uniformité, l'ensemble des opérations de division est réitéré selon un processus récursif.

La détermination du seuil pour une région est fondé sur la méthode de KOHLER [1] qui exploite l'histogramme des transitions moyennes de la région. Cet histogramme est déterminé de la façon suivante :

Pour tout couple (p,q) de pixels adjacents tels que L(p) < L(q)

si L(p) < n < L(q) N(n) = N(n) + 1
C(n) = C(n) + Min(L(q) - n, n - L(p))

H(n) = C(n) / N(n)

où L() est la fonction luminance.

La valeur de seuil S correspond au niveau de gris pour lequel H(S) atteint son maximum.

Le seuil sélectionné correspond à la maximisation du nombre de transitions moyennes qu'il met en évidence.

L'approche récursive permet de déterminer et d'appliquer les seuils localement et de minimiser ainsi les effets de propagation des segmentations obtenues par application de seuils globaux.

L'analyse des segmentations obtenues montre que des défauts subsistent. Ces imperfections peuvent être classées en trois catégories principales :

- fragmentation excessive des régions : une zone ayant une existence sémantique (représentation d'un objet physique) est divisée en de nombreuses régions qui n'ont aucune réalité physique ;

- phénomène d'"overmerging" : certaines régions recouvrent partiellement un objet ainsi qu'une partie de l'environnement ;

- erreur de placement de la frontière d'une région : ceci est essentiellement dû au profil du gradient au voisinage de la frontière des zones homogènes. Ce profil n'est pas parfait, et cela occasionne des erreurs sur l'estimation de la position du maximum du gradient.

Ces défauts ont deux causes essentielles :

- le critère de segmentation est mal adapté à la zone traitée (ceci est particulièrement vrai sur les images présentant à la fois des zones homogènes et des zones texturées) ;

- l'analyse de l'image est trop globale et ceci conduit à la détermination de paramètres de segmentation localement mal adaptés.

Pour pallier ces inconvénients, nous avons choisi d'améliorer la segmentation initiale par des traitements contrôlés par des bases de règles. Les prémisses de ces règles de production ont pour but la détection d'une configuration intéressante afin de déclencher dans la partie action un traitement adapté.

La mise en oeuvre d'un dispositif de contrôle de ces traitements est nécessaire afin d'éviter que les actions de division et de fusion ne se succèdent indéfiniment sur une portion de l'image. Ce système de segmentation a en effet un comportement non-monotone dont il est impossible d'assurer la terminaison a priori. Nous détaillerons donc la méthode de contrôle que nous avons mise en place et qui est basée sur la répartition de paramètres d'état sur chaque objet manipulé par le système.

La représentation des primitives de la segmentation est réalisée dans un formalisme objet. Deux classes principales d'objets sont utilisées par le système de segmentation bas-niveau :

- la classe REGION
- la classe RELATION D'ADJACENCE

Chaque région est représentée par une liste d'attributs et par son extension iconique (liste des pixels qu'elle contient). Nous distinguons trois types d'attributs principaux : les attributs photométriques (liés à la répartition des niveaux de gris sur la surface de la région), les attributs géométriques et les attributs morphologiques (liés à la forme de la région).

Outre les informations propres à chaque région, nous sommes amenés à utiliser au cours du déroulement du processus de segmentation des informations sur les liaisons entre les régions lorsqu'il s'agit de déterminer les couples de régions adjacentes candidates à une fusion. Nous avons donc choisi de représenter les relations d'adjacence entre deux régions par un objet du système qui possède des attributs sur la frontière entre les

deux régions (longueur du périmètre commun, moyenne, valeurs extrêmes et écart-type du gradient le long de la frontière).

Les règles de segmentation sont organisées en paquets correspondant au type d'action qu'elles appliquent. Le premier paquet regroupe les règles de fusion. Au nombre de quinze, ces règles effectuent des fusions selon des critères photométriques ou morphologiques.

Les critères photométriques font intervenir principalement , la luminance moyenne de chaque région, une information de texture (mesurée à l'aide des attributs de répartition du gradient calculés sur l'intérieur de la région pour éviter les défauts liés à la proximité de la frontière de la région).

Les critères géométriques et morphologiques sont fondés essentiellement sur l'aire des régionc (tout particulièrement sur les rapports d'aires entre deux régions adjacentes candidates à la fusion), et sur des attributs liés à la forme des régions qui permettent de détecter des configurations particulières comme , par exemple, l'existence de régions particulièrement fines (par le calcul des moments d'inertie, du ratio entre l'aire de la région et l'aire de son érodé etc...). La règle suivante est un exemple de cette utilisation des attributs :

Deux régions fines adjacentes
dont l'une est peu étendue
dont les axes principaux d'inertie sont parallèles
dont la longueur de périmètre commun est faible
sont fusionnées

Deux modes principaux de division sont mis en oeuvre dans RUBIS. Le premier mode consiste à activer sélectivement sur une seule région la segmentation par seuillage fondée sur la méthode de KOHLER. Cette méthode de segmentation est activée par les conditions suivantes :

- grande dynamique de niveaux de gris ;

- l'existence de plusieurs modes sur l'histogramme en niveaux de gris de la région ;

- la présence de gradients assez élevés dans cette région.

Le second mode de division est appelé "Division Morphologique". Cette action consiste à diviser une région selon un masque obtenu par ouverture morphologique. Cette opération permet d'isoler les zones fines ou les irrégularités de forme apparaissant sur la région et d'effectuer l'opération de division en isolant ces zones. Ce traitement a pour but de supprimer les zones de rétrécissement d'une région. Ce phénomène est généralement occasionné par une discontinuité du gradient très locale (souvent limitée à 1 ou 2 pixels) le long de la frontière d'une région. La simplification de la forme des régions constitue un gage de réussite pour le processus d'interprétation qui aura à effectuer une mise en correspondance entre des régions issues de l'image et les objets du modèle qui sont souvent décomposables en formes relativement simples. Cette division est activée sur des régions de compacité (aire / périmètre^{2}) faible.

Le contrôle du processus de segmentation est guidé par les quelques principes suivants :

- Maîtriser les phénomènes d' "oversplitting" et d'"overmerging" ;

- Assurer la terminaison du processus de segmentation ;

- "Prévenir plutôt que Guérir" : Il s'agit, d'une part, d'éviter des actions catastrophiques qui remettraient en cause le fonctionnement du système et d'autre part de préparer la mise en correspondance en isolant dès la phase de création des régions des objets qui paraissent caractéristiques.

Le contrôle du processus de segmentation peut être envisagé à deux niveaux. L'approche globale consiste à effectuer une évaluation sur l'ensemble des régions afin de calculer un critère de satisfaction vis à vis du traitement de segmentation. Cette évaluation globale a pour but d'orienter la statégie de segmentation à haut-niveau qui consiste à privilégier un type particulier d'actions sur une zone de l'image. Cette stratégie de haut-niveau doit être accompagnée d'un mode d'application. Ceci revient à déterminer précisément le séquencement des déclenchements des règles en associant à chacune d'elles la ou les primitives de l'image à modifier. Cette forme de contrôle a été mise en oeuvre par Nazif et Levine [2], [3], [4], [5], [6] qui ont développé un algorithme de contrôle

fondé sur la théorie des ensembles flous.

La mise en oeuvre d'un système de contrôle global conduit à de nombreux calculs afin de remettre à jour régulièrement le critère d'appréciation de la segmentation. Les principales caractéristiques prises en compte lors de cette évaluation sont les suivantes :

- l'uniformité de chaque région ;
- le contraste inter-régions.

L'approche adoptée dans le système RUBIS consiste à délocaliser la fonction de contrôle sur chaque objet manipulé par le système à base de règles. Chacune des régions de la segmentation comporte dans sa liste d'attributs un attribut d'état qui permet de noter les actions admissibles sur cette région. Ainsi cet attribut peut prendre les valeurs suivantes.

- libre : pour indiquer qu'il n'existe aucune restriction

- indivisible : cette valeur signifie que la région ne peut être divisée. Il existe deux variantes pour cette caractéristique qui concernent la méthode de division (une région peut être indivisible par seuillage ou par division morphologique ou totalement indivisible) ;

- gelée : aucune action n'est admise sur la région. Ceci signifie que la région est indivisible et qu'elle ne peut être fusionnée avec aucune de ses régions adjacentes.

Par ailleurs, nous avons la possibilité d'interdire une fusion particulière pour un couple de deux régions adjacentes en notant cette caractéristique sur un attribut d'état de l'objet représentant leur relation d'adjacence.

Cette organisation du contrôle permet d'implanter aussi bien une stratégie globale que locale. Dans RUBIS, nous nous sommes limités à l'implantation d'une stratégie locale monotone. En effet, l'affectation d'une valeur à l'attribut d'état d'une région n'est pas remise en cause lors du processus de segmentation en régions excepté pour remplacer sa valeur courante par une valeur plus restrictive. Ainsi une région dont l'état est "morpho-indivisible" peut passer à l'état "indivisible" mais en aucun cas à l'état "libre". Cette utilisation des attributs d'état, associée à la diminution des intervalles de tolérance des critères de segmentation assure la convergence du processus de vision bas-niveau.

Cette approche ne conduit pas à une partition de l'image optimale vis à vis des critères d'évaluation énoncés par Nazif et Levine mais il s'agit là d'un compromis entre un résultat acceptable et une charge de calcul excessive.

Par ailleurs, l'ensemble des régions obtenu est destiné à être interprété par le processus de vision haut-niveau qui utilise un modèle explicite de l'objet à reconnaître. Au cours de cette interprétation, la partition en régions initiale ne sera plus simplement évaluée par rapport à un ensemble de critères indépendants du contenu de l'image mais vis à vis d'un modèle de l'objet. Celui-ci sera utilisé pour émettre des hypothèses de resegmentation locale afin de rechercher des objets manquants ou de préciser les limites des objets identifiés afin d'améliorer les performances d'identification.

Il est extrêmement ambitieux et même illusoire de chercher à construire un traitement général de segmentation indépendant de la finalité de l'exploitation qui sera faite de l'ensemble des primitives extraites de l'image.

Dans le cas de RUBIS, il n'est pas primordial de traiter parfaitement les zones à très forte texture car elles représentent généralement des zones de l'environnement naturel sur lesquelles aucun traitement d'interprétation n'est effectué.

L'unique traitement des régions texturées consiste à détecter l'existence de la texture par une mesure de répartition des gradients. La région est isolée en utilisant l'attribut d'état, afin d'éviter qu'une action de division ne soit effectuée car cela conduirait à un nombre très important de régions et à la saturation totale du système. Ceci constitue la meilleure illustration du troisième principe "Prévenir plutôt que guérir".

Dès la phase de segmentation, des informations ainsi que la connaissance de la finalité du système sont nécessaires afin d'optimiser les performances du module de vision bas-niveau. Le schéma de contrôle utilisé dans RUBIS a permis d'implanter via un ensemble de méta-règles des critères de gel de certaines régions présentant des caractéristiques discriminantes vis à vis de l'objectif de RUBIS, à savoir "l'identification de véhicules". Les régions de luminance et de compacité élevées sont isolées même si leur aire en fait des candidates à la fusion. Ces régions représentent des éléments chauds du véhicule ou de l'environnement (leurres, feux, etc...) et constituent des germes pour le système d'interprétation qui initialise le schéma de prédiction-vérification

d'hypothèses sur ce type d'objets remarquables.

L'attribut d'état est également utilisé comme compte-rendu des actions de division d'une région. Si un mode de division ne donne aucun résultat sur une région, cette information est conservée via l'attribut d'état, ce qui permet d'éviter une invocation de cette action sur cette même région alors que l'issue est vouée à l'échec.

L'ensemble de primitives obtenu à l'issue de la phase de segmentation n'est en aucun cas optimale ("Un objet, une région") et les traitements de haut-niveau doivent être capables de réaliser la reconnaissance malgré les défauts. Le choix des seuils et de la stratégie pour le processus de bas-niveau ne saurait éliminer un type de défaut : en effet, la suppression totale du phénomène d'overmerging conduirait à un nombre prohibitif de régions tout particulièrement sur des zones à forte texture mais, par ailleurs, l'overmerging est extrêmement préjudiciable au processus d'interprétation qui débute l'identification du véhicule à l'aide des informations déduites de quelques régions supposées représentatives d'un objet du modèle.

Ceci montre la nécessaire collaboration entre les différents niveaux d'un système de vision artificielle lorsque le domaine d'application est relativement vaste (scènes naturelles, conditions de prises de vue variables, environnements complexes, masquages, contre-mesures etc...).

LA MODELISATION DES CIBLES

L'identification des cibles est réalisée, dans le système RUBIS, par un module de mise en correspondance entre un modèle du véhicule, dont le type est supposé connu, et l'ensemble de régions mises en évidence lors de la phase de segmentation bas-niveau.

La réussite de cette étape de reconnaissance est extrêmement dépendante de la qualité du modèle fourni. L'évaluation de cette qualité peut être réalisée vis à vis des critères suivants :

- la précision et la finesse de la description ;

- la robustesse de la modélisation qui rend compte de son adaptabilité à des conditions d'observations variées (point de vue, qualité des images) ;

- la facilité de mise en oeuvre du modèle par le processus d'interprétation.

Nous avons donc cherché à définir, dans RUBIS, un cadre de modélisation peu contraignant, robuste, et capable de prendre en compte l'aspect tridimensionnel.

L'approche retenue est un modèle multi-bidimensionnel. Cela signifie que la modélisation d'un véhicule est composée d'un certain nombre de modèles élémentaires de vues bidimensionnelles du véhicule selon plusieurs orientations caractéristiques. Compte tenu de la résolution des images et du champ couvert par la cible sur l'image , nous nous sommes limités à cinq orientations caractéristiques correspondant à une vue de face, une vue de profil, une vue d'arrière, une vue de trois-quart face et une vue de trois-quart arrière.

Chacun de ces modèles élémentaires est décomposé en un certain nombre d'objets qui sont les représentations d'une partie du véhicule selon l'orientation du modèle élémentaire considéré. Ainsi, le modèle élémentaire d'un blindé de type AMX30 selon une vue arrière sera constitué de six objets :

- deux pots d'échappement droit et gauche ;
- deux chenilles droite et gauche ;
- la vue arrière du chassis ;
- la vue arrière de la tourelle.

Le modèle du véhicule est représenté dans un formalisme objet en utilisant les notions d'héritage afin de construire une arborescence pour chacun des éléments constituant cette cible. Cette structure arborescente regroupe l'ensemble des vues d'un même élément du véhicule selon l'ensemble des orientations décrites dans le modèle.

Cette hiérarchisation de la description d'un objet consiste à regrouper au niveau le plus élevé de généralité l'ensemble des caractéristiques communes à toutes les représentations de l'objet dans les niveaux inférieurs. Par exemple, quelque soit l'orientation du véhicule sur l'image, le pot d'échappement, s'il est visible, sera représenté par une ou plusieurs régions de luminance élevée ; cette caractéristique commune sera conservée au niveau le plus élevé de l'arborescence. Cette hiérarchie est décrite en termes de classe et de sous-classes afin de tirer pleinement parti de la notion d'héritage inhérente à la représentation orientée objet.

Il existe deux types de critères pour caractériser un objet :

- Les critères dits "intrinsèques" car leur vérification ne fait intervenir que des attributs propres aux régions concernées par cet objet. Il s'agit par exemple de la luminance, de la compacité, de l'élongation.

- les critères appelés "relations de voisinage" qui décrivent les conditions que l'objet doit vérifier vis à vis des objets voisins.

La modélisation des véhicules comporte donc une double structuration ; d'une part une arborescence sur chacun des éléments et d'autre part des relations de voisinage entre les représentations de ces éléments [7].

Le souci de robustesse qui a guidé la tâche de modélisation nous a conduit à introduire dans la modélisation, d'une part la notion de granularité variable, et d'autre part à définir dès la conception du modèle des hypothèses de gestion des échecs pouvant survenir lors de la mise en correspondance.

La notion de granularité variable est introduite afin de rendre compte dès la phase d'établissement du modèle des phénomènes de pertes de résolution suivant la distance de la cible au capteur. La perte de résolution a deux causes principales :

- le nombre de pixels apparaissant à la surface de la silhouette de la cible qui est une conséquence directe de l'éloignement de l'objet ;

- la fonction de transfert de l'optique qui conduit à une diffusion apparente de la cible dans l'image, à une perte de netteté des détails et à des phénomènes de fusion entre les zones proches de luminosité peu différente.

D'après ces observations, il apparaît clairement que le modèle d'un véhicule situé à longue distance ne peut pas être déduit du modèle d'une vue à courte distance par simple réduction d'échelle. L'introduction de plusieurs modèles correspondant à une même orientation du véhicule permet de prendre en compte ces différents aspects tout en définissant, dès la conception du modèle, des hypothèses de gestion des échecs du processus d'interprétation. Un exemple de cette organisation est fourni par la modélisation du char de type AMX30 "vue arrière". Sur une image de grande résolution, on peut distinguer les régions correspondant aux deux pots d'échappement ainsi qu'une région légèrement moins lumineuse représentant le chassis. Dans le cas du même véhicule vu selon la même orientation à une échelle beaucoup plus faible, il est rarement possible de distinguer ces trois régions qui n'en forment plus qu'une seule. La prise en compte de ce type de phénomène lors de la phase de construction du modèle fournit au processus d'interprétation plusieurs alternatives de raisonnement. Ceci évite d'avoir à envisager des mises en correspondance entre une fusion de plusieurs objets du modèle et d'une seule région de l'image ou bien de chercher à diviser une région de l'image sans information a priori sur le mode de division à mettre en oeuvre.

L'INTERPRETATION

Le module d'interprétation est fondé sur un schéma général de mise en correspondance entre les objets du modèle et l'ensemble des primitives extraites de l'image. La mise en correspondance est effectuée par un processus de prédiction-vérification d'hypothèses. Chaque identification d'un élément du modèle donne lieu à la création d'un objet instance de la classe représentant l'élément reconnu. Cette instance stocke la ou les primitives correspondant à l'élément. La construction d'instances du modèle au fur et à mesure de la reconnaissance permet de gérer plusieurs interprétations de façon parallèle.

Dans le cas le plus général d'une mise en correspondance entre un modèle tridimensionnel et une image bidimensionnelle, l'hypothèse de vision est un sextuplet de valeurs (a,e,d,X,Y,τ) correspondant respectivement aux deux angles d'azimuth et d'élévation de la ligne de visée dans le repère cible, à la distance capteur-but, à la position du centre de la silhouette et à l'orientation de la silhouette dans l'image. Dans notre application, nous avons négligé l'angle τ en faisant l'hypothèse que le capteur est stabilisé en roulis et que le sol est suffisament horizontal pour supposer τ = 0. Par ailleurs, nous n'avons modélisé qu'un ensemble restreint d'orientations de la ligne de visée par rapport à la cible :

- vue arrière (a = - 90°, e = 0°)
- vue de face (a = 0°, e = 0°)
- vue de profil (a = 90°, e = 0°)
- vue de trois-quart arrière (a = - 45°, e = 0°)
- vue de trois-quart face (a = 45°, e = 0°)

Nous n'avons jamais tenu compte dans notre modélisation de l'angle e. Les modèles

construits pour les cinq orientations privilégiées peuvent être utilisés sans aucune modification pour des angles e < 20 °.

Le paramètre d de l'hypothèse de vision est remplacé dans notre système par l'échelle E du véhicule sur l'image.

La prédiction d'hypothèses consiste à utiliser certaines primitives de l'image et à tenter de les mettre en correspondance avec des objets du modèle. De chacune de ces tentatives, sont déduits les paramètres d'une hypothèse de vision à savoir une attitude de l'objet, une position approximative de sa silhouette et une première estimation de l'échelle.

Ce processus de prédiction est fondé sur des objets particuliers du modèle appelés "Ilots de confiance du modèle" (I.C.M). Ces objets seront utilisés comme des germes sûrs afin d'émettre des hypothèses de vision. Ces objets doivent posséder les propriétés suivantes :

- ils sont identifiables sans connaissance a priori su l'échelle et l'orientation de la cible dans l'image ;

- ils sont discriminants afin d'orienter le choix du modèle bidimensionnel à utiliser ;

- ils présentent, si possible, des caractéristiques suffisamment distinctes par rapport aux objets voisins afin d'être isolés par le module de segmentation en régions. Ceci permet d'obtenir comme représentation de cet objet dans l'image une seule région facilement identifiable à partir de laquelle il est possible d'obtenir une première estimation de l'échelle du véhicule dans l'image.

La représentation des ICM dans le modèle fait appel à la notion d'héritages multiples. Une classe ICM est créée et est l'une des classes antécédentes des classes représentatives des objets qui jouent le rôle d'ICM.

Les objets à forte émissivité thermique constituent, dans le cas de notre application, d'excellents ICM. Ainsi, pour un blindé de type AMX30, les pots d'échappement et les chenilles (particulièrement vues de face ou d'arrière) constituent les principaux ICM de la modélisation.

Le déroulement du mécanisme d'interprétation débute par une recherche de toutes les régions de l'image vérifiant l'un des critères de sélection des ICM ; ces critères étant regroupés au niveau de la classe ICM. Pour chacune des primitives ainsi identifiées, sera tentée une mise en correspondance avec l'une des classes "filles" de la classe ICM et ainsi de proche en proche jusqu'à une classe "feuille" de l'arborescence.

Chaque mise en correspondance est évaluée par un calcul de qualité qui tient compte d'une part de l'adéquation entre les attributs propres des primitives et les caractéristiques intrinsèques de l'objet du modèle et d'autre part de la présence ou de l'absence des éléments voisins décrits dans le modèle. La propagation descendante des mises en correspondance est accompagnée d'une remise à jour de l'indice de qualité en tenant compte de la valeur trouvée au niveau précédent et des nouveaux critères vérifiés au niveau considéré.

Lorsqu'une hypothèse est émise, le processus de mise en correspondance est activé pour chacun des objets appartenant au modèle choisi. Une phase de prédiction permet de limiter la zone de recherche de cet élément grâce aux informations déduites de l'identification de l'ICM.

L'architecture de modèle permet de maintenir, parallèlement, plusieurs lignes de raisonnement au cours d'une interprétation. Il est parfois impossible de conclure entre deux interprétations (par exemple : vue de face ou vue de trois-quart face d'un même véhicule) mais le système indique tout de même les éléments correctement reconnus même si leurs caractéristiques ne sont pas exactement semblables à celles correspondant à une alternative unique.

Lorsque plusieurs ICM d'un même véhicule sont simultanément visibles sur une image, plusieurs interprétations cohérentes sont obtenues. Dans ce cas, un mécanisme de propagation de contraintes est mis en oeuvre. Cette opération est réalisée en appliquant les contraintes de voisinage issues de l'identification d'un ICM sur l'interprétation réalisée en utilisant le second ICM comme germe.

Le stade ultime du processus d'interprétation concerne l interaction entre le module de vision haut-niveau et le système de segmentation bas-niveau [8]. Cette interaction poursuit deux buts : d'une part affiner et préciser les limites de la silhouette et d'autre part traiter les cas d'échecs (absence apparente d'un objet).

Cette amelioration consiste à appliquer localement des opérations de division en choisissant la zone d'effet grâce à l'interprétation obtenue par le processus de haut-

niveau.

L'ensemble des objets correctement identifiés dans l'image est utilisé afin de calculer la position précise d'un certain nombre d'axes caractéristiques du modèle élémentaire utilisé. A ce modèle est associé un prototype constitué de l'ensemble des positions standard des axes pour une échelle de référence. La combinaison des axes de référence et des axes déterminés dans l'image conduit au calcul d'une fenêtre de recherche pour chaque objet du modèle élémentaire. La position et la taille de chaque fenêtre sont extrêmement précises car ce calcul prend en compte simultanément toutes les informations issues de l'identification de plusieurs objets.

Chaque fenêtre est utilisée pour déterminer la liste des régions susceptibles de représenter l'objet concerné. Un traitement de division est appliqué aux régions ayant une intersection avec la fenêtre mais également une partie de la surface située hors de la fenêtre. Pour chacun des fragments mis en évidence par l'opération de division, le système cherche à les associer à un objet du modèle. L'opération est réitérée jusqu'à l'identification de l'objet cherché ou bien jusqu'au constat d'un échec qui signifie soit que l'objet est masqué soit que l'aspect sous lequel il apparait n'est pas modélisé.

L'utilisation du modèle pour émettre des requêtes de segmentation bas-niveau permet de minimiser les imperfections inévitables de la partition initiale en régions et d'améliorer de manière significative les performances du système.

L'évaluation quantitative des résultats a été réalisée sur une base d'une trentaine d'images représentant un blindé sous un ensemble d'attitudes variées. Cette évaluation a été menée en comparant la silhouette expérimentale déterminée par RUBIS avec la silhouette théorique extraite interactivement par un opérateur.

Deux critères d'appréciation sont pris en compte. Le coefficient Ci qui représente le pourcentage de surface de la silhouette théorique correctement inclus dans la silhouette expérimentale. Le second coefficient Cd mesure le pourcentage de surface de la silhouette expérimentale n'appartenant pas à la silhouette théorique.

Les résultats ont été évalués avant et après la phase de resegmentation. Le coefficient Ci égal à 71% avant resegmentation passe à 78% après cette étape. Le coefficient Cd passe de 7% à 5%. Sur la plupart des images traitées, le mécanisme de resegmentation accroit la performances dans des proportions très faibles de l'ordre de 1% mais dans certains cas difficiles d'environnement structuré, cet affinage améliore sensiblement la précision de la silhouette (jusqu'à 20 % d'augmentation de la valeur de Cd). Ceci peut avoir des conséquences importantes sur la précision de localisation du point d'impact optimal.

Le système RUBIS a été développé sur une station de travail de type SUN 3/260 en langages C et Common Lisp. L'ensemble des règles pilotant le module de segmentation en régions a été implanté à l'aide du générateur de système expert KIRK et le langage objet utilisé pour l'ensemble du projet est le produit FLAME. Ces deux outils fondés sur Common Lisp sont deux produits développés par THOMSON-CSF.

CONCLUSION

La maquette actuelle de RUBIS a permis d'une part de prouver la faisabilité d'un système autonome de reconnaissance de cibles. Elle a constituée un banc d'essai pour un certain nombre de méthodes et d'outils issus du domaine de l'intelligence artificielle qui fournissent au concepteur un environnement de développement souple et évolutif afin de tester et de prototyper rapidement les algorithmes.

Les résultats obtenus sur la base d'images de test sont probants car RUBIS réalise dans la plupart des cas une détermination de silhouette correcte malgré la présence de fonds complexes. Ces performances ont deux raisons essentielles : l'approche multi-critères de la segmentation en régions et la tolérance du module d'interprétation vis à vis des défauts de segmentation grâce à la collaboration des deux niveaux du système par l'émission de requêtes de segmentation à partir du processus de mise en correspondance.

RUBIS a en outre un potentiel d'évolution important car il constitue un cadre général pour supporter des développements ultérieurs tant au niveau de la segmentation (utilisation du mouvement par exemple) qu'au niveau de l'interprétation (stratégies de recherche, extension vers un nombre plus importants de types de véhicules).

La prise en compte de l'aspect temporel semble constituer l'axe de développement le plus prometteur. Ceci peut se traduire dans le processus de segmentation par l'utilisation du mouvement (segmentation du champ des vitesses) ou en propageant les résultats obtenus pour l'image N sur l'image N+1 afin d'affiner ainsi la segmentation

au lieu de réinitialiser le processus pour chaque nouvelle image. Dans le processus d'interprétation, la mise en oeuvre d'un modèle d'évolution temporelle de l'aspect de la cible dans l'image devrait permettre un élagage considérable de l'arbre des solutions pour les mises en correspondance et augmenter significativement les performances tant du point de vue de la précision que d'un point de vue efficacité, point fondamental dans la perspective d'une application embarquée.

BIBLIOGRAPHIE

[1] R. KOHLER
A segmentation system based on thresholding
Computer Graphics and Image Processing - Vol 15 - 1981 - pp 319-338

[2] M.D. LEVINE, S.I. SHAHEEN
A modular computer vision system for picture segmentation and interpretation
IEEE PAMI - Vol 3 - Sep 1981 - pp 540-556

[3] M.D. LEVINE, A.M. NAZIF
An optimal set of image segmentation rules
Pattern Recognition Letters (1984) pp 243-248

[4] M.D. LEVINE, A.M. NAZIF
Rule-based image segmentation : a dynamic control strategy approach
CVGIP - Vol 32 - pp 104-126 - (1985)

[5] M.D. LEVINE, A.M. NAZIF
Dynamic Measurement of computer generated image segmentations
IEEE PAMI Vol 7 - Mar 1985 - pp 155-164

[6] A.M. NAZIf, M.D. LEVINE
Low-level image segmentation : an expert system
IEEE PAMI - Vol 6 - n°5 - Sep 1984 - pp 555-577

[7] J.H. CONNELL, M. BRADY
Generating and generalizing models of visual objects
Artificial Intelligence - Vol 31 - (1987) - pp 159-183

[8] C.A. KOHL, A. HANSON, E. RIESEMAN
Goal-directed control of low-level processes for image interpretation
Proceedings : DARPA Image Understanding Workshop Fev 87 - pp 538-551

REMERCIEMENTS

La présente étude a été soutenue par la Direction des Recherches Etudes et Techniques (contrat DRET n°88/350).

Constraint Management Requirements for On-Line Aircraft Route Planning

Uwe Teegen
Deutsche Forschungsanstalt fur Luft- und Raumfahrt (DLR) e.V.
Institut fur Flugführung
Postfach 3267
D - 3300 Braunschweig
Federal Republic of Germany

Summary

In the future, the cooperation of pilot and controller will change. Technical advances contributing to this change are a more intelligent airborne Flight Management System (FMS) and a datalink connecting the FMS and Air Traffic Control (ATC). Against this background, concepts for an Experimental Flight Management System and its human-centered system design approach are described. Combined with a basic scenario of future Air Traffic Management (ATM) the requirements for an airborne constraint management subsystem are developed. The fundamentals of aircraft route planning and system operation including considerations on the interaction between constraint management and man-machine-interface are discussed and an on-line algorithm for aircraft route planning is presented. The paper also describes the present state of a software prototype and the soft- and hardware employed.

1. Introduction

Flight Management Systems have become an essential part of civil aircraft operation in the last decade. Increasing research efforts on automation regarding future Air Traffic Management (ATM) reveal opportunities to improve the support of the air traffic controllers by utilizing airborne resources. Future FMSs and ATM systems will have a datalink module and thus enable direct data communication between airborne and ground equipment. Undoubtedly, such a datalink combined with planning and decision support for the air traffic controller will increase system performance. Flight precision as well as exploitation of airspace will increase considerably. Combined with increasing 'intelligence' of the supporting system the human operator demands additional explanation on how machine-generated proposals are generated and what the implications are on system operation and aircraft safety. However, the changing role of the human operator in this environment probably introduces new errors (Wiener [1]).

In the past numerous papers have been published outlining fundamentals of a future 'intelligent cockpit'. References [2] to [5] present typical examples. The paper by Rouse, Geddes and Curry [2] provides a detailed summary regarding functional design and architecture of the man-machine-interface in case of a complex system. It is chosen as a basic reference. In this context detailed formatting aspects of information to be displayed as well as selection of appropriate operator input devices are of minor interest. However, the design of an operator support system requires a wholistic view to create an efficient overall system. System architecture, algorithm design, data structures and appropriate selection of task-related information in total can provide high performance and acceptance (Hacker [6]). Simply spoken, the questions - what is needed, how can it be achieved and which realisation features fit human capabilities - depend on each other and have to be answered nearly simultaneously to provide a profound design approach.

The paper concentrates on this comprehensive aspect of system design. The aircraft route planning task of the pilot as well as the constraint management subsystem are discussed making direct reference to the man-machine-interface. An ATM scenario and a FMS conceptual design proposal pointing to future needs are presented for the definition of functional requirements as well as for the implementation of a software prototype.

2. Flight Management Concept

Prerequisite for the implementation of an aircraft route planning algorithm is the FMS. Therefore, the conceptual model of an experimental FMS currently designed as well as a scenario for future integrated ATM are described.

2.1 The Conceptual Model of an Experimental Flight Management System

The following description refers to the Conceptional Model of an Experimental Flight Management System defined by the working group 'Experimental Flight Management System (EFMS)' within the "Programme for Harmonized Air Traffic Management Research in

EUROCONTROL (PHARE)" [7,8]. The participating European Research Institutions - BFS, CAA, CENA, DLR, NLR, RAE, RSRE and EUROCONTROL - specified an experimental system, which aims at high flexibility to serve as a research tool for datalink, air traffic management, profile prediction and optimisation as well as man-machine-interface studies and experiments. A major goal of the programme will be the realisation of an ATM demonstrator including real aircraft operation in an advanced experimental environment.

This system proposal may serve both as an example for an advanced conceptual design of a future FMS and as basis for an experimental implementation

The system consists of the eight modules shown in figure 1 performing the following functions:

- Man-Machine-Interface
  Translates objects of the EFMS Network into Man-Readable Messages and vice versa

- Constraint Manager
  Maintain Active and Secondary Constraint List (i.e. an extended issue of the Flight Plan)
  Respond to Constraint Edits
  Respond to Activate Secondary Constraint List Command

- Profile Prediction and Outer Loop Guidance
  Construct the Active and Secondary Flight Profile
  Generate and Broadcast the Outer Loop Guidance Vector

- Data Base Manager
  Extract Data from the Data Base following to a query and respond to the initiating element
  Update Data Base

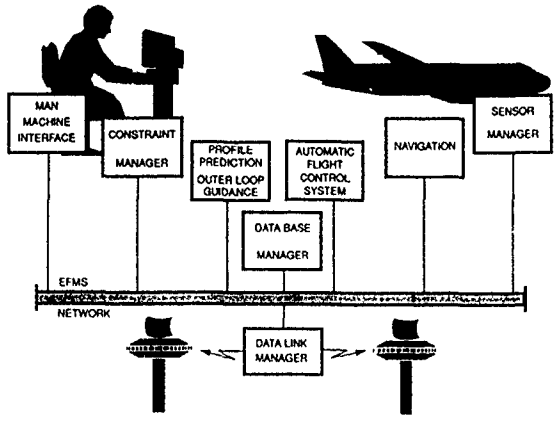


Figure 1 - EFMS Functional Elements

- Data Link Manager
  Gateway between the Air-Ground Datalink Network and the EFMS Network

The remainder provides the connection between the EFMS network and aircraft-specific environment:

- Automatic Flight Control System (AFCS) Interface
  Generate Implementation Specific Guidance Commands for Output to AFCS

- Navigation System
  Interface between specific Navigation Equipment and EFMS Network
  Selection and Management of Navigation Sensors
  Autotuning and Data Conditioning

- Sensor Manager
  Interface between Aircraft specific Sensors and EFMS Network
  Data Conditioning, Filtering, Interpolation

Before proceeding to the details of system operation two definitions are formulated [8]:

A *constraint* consists of a waypoint combined with attributes specifying position, altitude, speed, time and a constraint type. Optional attributes such as maximum speed or altitude limitations may be added if required. The resultant *constraint list* may be characterised as an extended issue of a *flight plan*.

The *profile* or *trajectory* is computed from the constraint list, meteorological data and aircraft performance data. Combined with lateral and vertical deviation limits a 'tube' is obtained. Time constraints result in the conception of a 'bubble' moving through the 'tube'. 'Bubble' and 'tube' represent the contract between aircraft and ATC. The 'bubble' denotes the manoeuvre space of the aircraft in which the aircraft is authorised to optimise its own trajectory. The trajectory then is the basis for computing the *outer loop guidance vector*, which is fed into the AFCS.

Strategical planning operation

The operation sequence for strategical route planning is initiated by the pilot with the input of waypoints, thus creating a flight plan which is the basis of the constraint list completed by default values and any limitations from the database. The flight profile generation module computes from this data, meteorological forecast data,

ATC constraints received via datalink, aircraft performance data and navigation data an optimized trajectory. The trajectory is displayed to the pilot. If it is appropriate the pilot may request a clearance for that trajectory. ATC will return the take-off time and - if necessary - a number of additional constraints. If the ATC constraints do not conflict the airborne constraint list, the pilot may activate the trajectory. If not, the pilot has to adhere to the ATC constraints and to initiate another run of the flight profile generator to get an appropriate trajectory. The clearance request is repeated. The described operation is iterative and will be repeated until an agreement between pilot and controller is achieved. Then, ATC will return the trajectory and deviation limits, which define the 4-D 'tube' in space.

Alternatively the pilot may wish to alter constraints. This initiates an identical negotiation process between the pilot and the controller, but will include the essential ATC constraints of the current flight plan.

Tactical operation

Tactical control commands such as 'holding', 'step climb', 'descent to' or 'direct to' require immediate action by the pilot Then the pilot has to enter modifications of the constraint list, start the profile generation process and subsequently to initiate the negotiation process with ATC.

At present, different definitions of the time span referring to tactical control are discussed. Here, a time span of about 30 minutes ahead is assumed. Short term tactical control in case of emergency conditions will be handled most probably via R/T-communication.

2.2 Air Traffic Management Scenario

The future ATM scenario proposed by a GARTEUR working group outlines a concept of operation [9] which is generally characterized by computer assistance of the controller, a datalink between aircraft and ground equipment as well as suitably equipped aircraft, i.e. FMS and even 4-D navigation capability.

Three distinct levels of datalink communication between aircraft and ATC are of importance in this respect:

The background level is characterised by an automatic exchange of information without human intervention. This level includes general aircraft data, such as aircraft call sign, aircraft type, equipment identification etc., and initialises the datalink communication process. Subsequently, exchange of dynamic data at fixed time intervals predominates datalink communication. Regarding the downlink, position as well as meteorological data with reference to time and aircraft position will be included. The uplink communication may contain broadcast weather data, ATC constraints and runway data.

The strategical level represents pilot or controller initiated exchange of long-term related strategic planning information.

The tactical level requires an immediate response by pilot or controller.

Both strategical and tactical level refer to the trajectory negotiation process where the tactical level assumes the higher priority.

Figure 2 illustrates the data types and datalink communication [9]. Corresponding to these basic items ATC is in a position to

- perform collective optimisation to resolve conflicts between individual preferred profiles, where clearances take the form of a rigourously defined 'tube' in space,
- exploit improved navigation in case of suitably equipped aircraft by direct use of aircraft position data or because the aircraft can comply with clearances unaided,
- maintain a database providing current meteorological data in a three dimensional grid within the geographical region of interest.

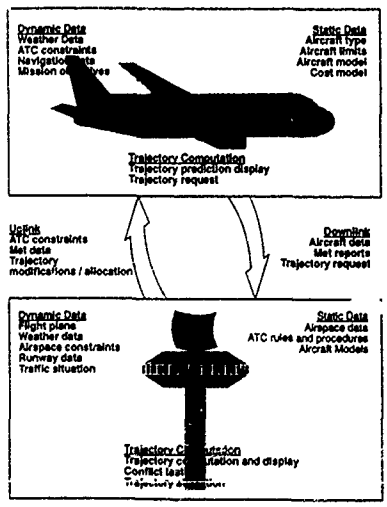


Figure 2 - Data Types and Datalink Communication

With suitably equipped aircraft as well as ground-based optimisation procedures air traffic density could be raised to maximum capacity without reducing safety demands.

Poorly equipped aircraft limit the maximum capacity because they have not the capability to operate in a comparable and narrow 'tube'.

3. Functional Requirements for Constraint Management

Constraint Management represents a central function of the FMS. Therefore, functional requirements for constraint management relate to the operation of the entire FMS, regarding internal constraint handling and including man-machine-interface aspects. A wholistic approach is needed.

3.1 System Operation

The constraint management module gathers the constraint inputs of the pilot and creates the constraint list which is the basis for subsequent data processing within the profile prediction module. The result is displayed to the pilot and - in case of pilot's agreement - will be sent to ATC via the datalink. Further constraints added by ATC will restart the process of editing the constraint list by the pilot. After activation of the constraint list and the inherent trajectory computation, the pilot supervises aircraft performance and the progress of flight with regard to aircraft state, navigation state and outer loop guidance. A constraint list modification may result in a considerable increase of workload, since supervision of the currently active and modified constraint list has to be done in parallel.

Even such a simplified description of the pilot task irrespective of a possible work-sharing between the two crew members asks for some assistance for the pilot.

Assistance for route planning can be achieved by means of airline-defined routes created off-line and stored in the database or by implementing a route planning algorithm for on-line use. In the first case each airline employs a set of tailored, airline-specific routes. The second case is just a suggestion for the further development of the FMS, but both solutions must be consistent.

A current restriction for system operation is the limited number of constraint lists handled by the FMS To solve this problem future systems must contain several constraint list buffers. On the one hand this requires a complete set of buffer manipulation commands such as "COPY", "DELETE" and others. On the other hand priority values for the operational sequence have to be introduced to the system: A tactical ATC message as well as the subsequent modification of the active constraint list demand a higher priority than the route planning process of the next flight leg, which, in turn, affects the profile prediction process.

The most frequently addressed FMS module will be the database management unit. The performance characteristics of this module will influence the total system performance considerably. The database contains navigation data, aircraft performance data and meteorological data. The source of the navigation database which is provided according to ARINC standards [10]. An off-line, ground-based procedure will be needed for transformation of the navigation database into a form suitable for efficient database access.

These features increase the system complexity considerably. Efficient system handling by the pilot, however, demands a low complexity of the system model within the man-machine-interface. Therefore, future systems will include operator models and aiding units providing task queuing and method selection dependent on pilot workload assessment [2].

COMMAND	NAVIGATION OBJECT IDENTIFIER	ATTRIBUTE(S) / VALUE
INSERT	o	departure, destination, begin_of_flight_plan, end_of_flight_plan, via, define
DEPARTURE DESTINATION	airport	
BEGIN END	airport, navaid, waypoint, pilot-defined-waypoint	
VIA	o	
DEFINE	name	coordinates
DELETE		
CHANGE	o	altitude, speed, climb, descent, time
ALTITUDE SPEED CLIMB DESCENT TIME	navaid, waypoint, pilot-defined-waypoint	value
DIRECT_TO	navaid, waypoint, pilot-defined-waypoint	
HOLDING	navaid, waypoint	turn-direction

o airway, SID, STAR, Approach Route, navaid, waypoint, pilot-defined-waypoint

Figure 3 - Constraint List Editor Commands

3.2 Internal Constraint Handling

The essential function of the constraint management module is represented by the 'constraint editor'. Modification of the constraint list requires a set of editor commands optionally supplemented by arguments, i.e. attributes and values for a detailed definition of the particular constraint. Figure 3 outlines the command set as well as necessary argument specifications.

There are only few basic commands needed ("INSERT", "DELETE" and "CHANGE") which are to be combined with a navigation object identifier as well as optional attributes

and values. In case of different attribute modifications rather long command sequences may occur. From an ergonomic point of view it seems convenient to extend the command set by including attribute modification commands, e.g. the attributes 'via', 'altitude', 'speed' etc. combined with an identifier and a value. Because of their operational importance, two tactical commands "DIRECT-TO" and "HOLDING" are included, although they actually may be considered as attribute representations of the basic commands "DELETE" and "INSERT".

Other candidates for a command set extension are 'LATERAL-OFFSET' and 'AVOID'. However, they are not included because they represent future research topics to achieve an efficient utilisation of computer resources of ground-based and airborne equipment.

Besides the constraint editor function a buffer manipulation command set has to be realised. It includes commands like COPY, KILL or DELETE, CONCATENATE; CHANGE BUFFER, LIST BUFFER and ACTIVATE, all of them to be combined with one or two buffer identifiers.

The complete command set supported by the constraint management module represents a hierarchy. Buffer manipulation commands are comparable to operating system commands of a computer, and the constraint list editor is directly related to text editing. Different command input formats, if identified as unambiguous, shall be acceptable for the constraint management module, to facilitate the development of different internal models of the command hierarchy. Even the combination of different commands shall be allowed.

A general, self-evident requirement regards constraint identification. To end up with a consistent constraint list the constraint management software must be able to discriminate between different kinds of constraints in cooperation with the database management module A rather simple example may be the differentiation between airways and waypoints enroute and in the terminal area. Pilot-defined waypoints as well as airline specific routes or even system-generated routes and waypoints have to be classified appropriately.

A formal error-checking is performed by the man-machine-interface module, e.g. a check whether the command is complete. An entry without a navigation object identifier or an attribute change without a value make no sense. The constraint management module then will perform error-checking with regard to the entry in cooperation with the database management, e.g. whether the object identifier is available in the navigation database or whether attribute modifications are allowed. Airway direction, altitude or airspace restrictions may lead to a rejection of an attribute modification by the system.

To sort the constraint list entries, the constraint management module needs the information of starting and endpoint or departure and destination, if the planned route is a complete flight leg. By means of the route planning algorithm the constraint list can be constructed without inserting any waypoint in-between. The input sequence of waypoints - even if the pilot inserts additional waypoints to guide the algorithm - is of no importance.

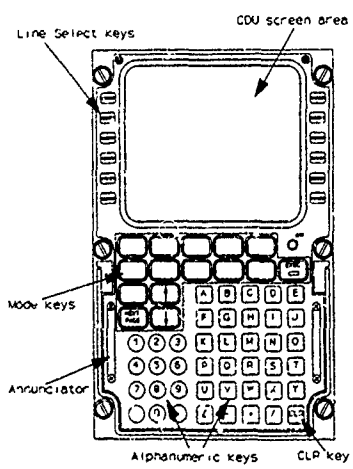


Figure 4 - Control Display Unit (ARINC 739)

With the constraint list completed the constraint management module may initiate the profile prediction process. This process may be started automatically even if the pilot continues entering additional waypoints. The system has to verify whether such additional inputs are already contained in the constraint list or - if not - whether they represent an increase of costs and therefore lead to a rejection of the additional waypoint. Waypoint entries by means of the "VIA" command must not be refused by the system, but represent essential points of the planned route.

3.3 Man-Machine-Interaction

The I/O-device for current FMSs is a Control Display Unit (CDU). which has been standardized by ARINC Specification 739 [11]. Figure 4 shows this CDU referring to the EFMS project [12]. This man-machine-interface system is characterized by a display of 14 lines of alpha-numeric text including title line and scratch pad, 6 line select keys on each side of the display, 15 function keys and a set of alpha-numeric input keys. Map/chart information is directed to the navigation display (ND) of the Electronic Flight Instrument System (EFIS). Modes of operation allow the selection of different information contents and output formats for the navigation display. The EFIS control panel includes the selector for several display modes (rose, arc, plan), the range selector and five optional data display switches. For each of these selectors only one function can be activated at a time.

However, alphanumeric input requires a lot of time and, unfortunately, often results in errors, even in uncritical situations [1]. An advanced control display unit will have to include hardware features such as graphic display and pointing device for an easy combination of graphical objects with desired functions [13,14].

For flight plan generation and aircraft route planning, respectively, graphic objects denote map/chart information requiring a rather high resolution and respectable screen size. The requirements for a vertical situation display are even less, but the vertical planning supervision demands a comparable pilot 'playground', i.e. a graphic display design and an adequate pointing device. Whether this pointing device is optimally represented by a rollball or a touch screen can not be answered satisfactorily yet.

A page-oriented display layout as realized in the current display design of the CDU may be useful because of the diversity of information contents. Probably, the applied information structure of the current CDU refers directly to the pilots' cognitive structure [15]. A graphic display can realise such a structure by means of an information overlay technique, comparable with slides containing an identical basic information pattern, but different detailed information contents layed over each other. Nevertheless an alpha-numeric display of the constraint list, initialisation data and aircraft performance will be necessary. On the one hand it facilitates 'a change of the third figure behind the decimal-point', on the other hand it increases pilots' acceptance.

Another function made available by the control display unit is the access to the map information access for the displayed region or range. The planning pilot will need this information about navigation objects not contained in the constraint list, but located close to the planned track. Besides that, the pilot must be able to change the range or reference coordinates. These requirements have to be supported by the database management module.

These considerations refer to a separate control display unit including a graphic display, pointing device and keyboard. The navigation display of the EFIS may remain unchanged in principle with just minor modifications. The plan mode may be removed and operation may be simplified by discarding the optional data selector, if 'essential information' can be brought up automatically. In general, the navigation display can be used to visualize the active constraint list as well as tactical commands from ATC.

4. Aircraft Route Planning

Search strategies applied on route planning represent a classic item of Artificial Intelligence (AI) exemplified by the famous 'travelling salesman' problem, the solution of which identifies the shortest route between several cities permitting him to visit all cities, but only once (Nilsson [16]). Other applications of AI search strategies regard robot movement in dynamic environment [17] as well as aircraft route planning for military missions [18,19].

The problem domain of civil aircraft routing is slightly different from these applications. Intermediate waypoints, airway segments and terminal area procedures have to be developed first, even the number of intermediate waypoints is unknown, when starting the search. Nevertheless, search strategies are applied to all these planning problems, only optimisation criteria and problem space description will differ.

Route Planning Algorithm

The task 'Aircraft Route Planning' is performed in a two-phase cycle. It is initiated as soon as the system has knowledge of the start/begin (or departure) and the end (or destination) of the pre-planned flight. Normally a waypoint, navaid or airport is required for the definition of 'start/begin'. The starting point may be represented by the current aircraft position or a nominated route waypoint.

Within the first phase distance and track of the direct route between 'begin' and 'end' are calculated. If there are more than these two points available, e.g. in case of a 'via airway' command entering all airway defining waypoints, an identical calculation is performed for each waypoint in relation to the start-point. Then, the waypoints are sorted to distance. With a criterion based on distance and track calculation, waypoints being far-off are deleted. Additionally, waypoints occurring twice in the list, for instance, in case of airway intersections, are identified and reduced to a single list entry.

The first phase is characterized as mainly passive. Some basic calculations combined with database queries are performed to collect as much information as possible about the list entries. The pilot is not informed about the results in particular.

The second phase is dominated by active search. The waypoint list, created within phase one and probably incomplete, represents the initial state of search. Beginning with the first list entry (starting point or departure) a number of adjacent waypoints on the actual and each intersecting airway is used for the creation of a search tree. Additionally, those waypoint entries of the initial waypoint list for which route

discontinuities can be stated are included. The number of adjacent waypoints denotes the search depth.

An accumulative cost index based on track angle deviation between adjacent waypoints, and compared to the direct track between start/begin and end of the planned flight, leads to the selection of the next waypoint on the route.

If the algorithm detects a significant track angle change for the waypoint sequence, which is described by the previous waypoint, the actual waypoint and the selected optimal next waypoint, it will try to connect them directly, thus deleting the intermediate waypoint entry.

The underlying search strategy may be classified as an informed best-first search [19] supplemented by heuristic elements. If a graphic visualisation of the process is available, the sequentially performed search and waypoint evaluation can be easily understood.

In the following, two examples will provide some explanation on the search process and the verification procedure.

Verification examples

Aircraft route planning is performed under a number of different conditions, which refer to different constraint list editor commands and operational aspects, respectively. Presently, twelve different scenarios have been realised to investigate planning algorithm and optimisation criteria. Two of them, representing actual flight conditions will serve as examples.

Figure 5 illustrates the current route of the aircraft, with the upcoming waypoint to be reached, WPBEG as starting point, the intermediate waypoints ABC, WPDEF and GHI and the endpoint LMN. The algorithm attempts to connect the waypoints ABC and GHI directly, thus creating the route description or constraint list, respectively, as WPBEG, ABC, GHI, LMN.

A direct connection as shown between LMN and WPSTU will be rejected, because the algorithm has knowledge of the restricted area.

Example A - Command input: END WPSTU

With this command input the endpoint attribute of LMN is overwritten. Within a constraint list just one 'end' is allowed.

Phase 1: The waypoints of the current constraint list and the new list entry WPSTU are sorted to the distance from the starting point, resulting in the sequence WPBEG, ABC, GHI, WPSTU, LMN. The last waypoint is identified as far-off and deleted.

Phase 2: There is only one connection existing between WPBEG and ABC, which then is chosen as optimal. In the next step several alternatives are identified: ABC - WPDEF, ABC - GHI, ABC - OPR representing the first layer of the search tree, and, additionally, WPDEF - GHI, GHI - WPSTU, OPR - WPSTU in case of two layers activated. A criterion, based on track angle deviations at the distinct waypoints, compared to the direct track between WPDEF and WPSTU, is applied to the different alternatives, selecting the optimal waypoint. The procedure results in the optimal waypoint sequence WPBEG, ABC, OPR, WPSTU.

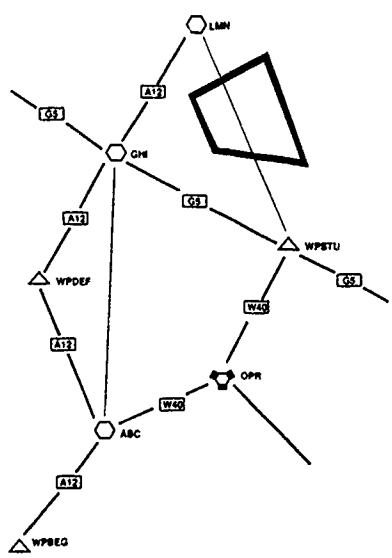


Figure 5 - Route Planning

Example B - Command input: INSERT G5

Phase 1: The waypoints GHI and WPSTU, defining the airway in the specified region, represent new entries. The waypoint GHI occurs twice, because it is already part of the active constraint list. It is reduced to a single entry. Sorting the waypoints to the distance from the starting point, results in the initial waypoint list WPBEG, ABC, GHI, WPSTU, LMN.

Phase 2: Compared to example A, an identical search is performed. But here, it results in another waypoint sequence, characterising the optimal route. It is

defined by WPBEG, ABC, GHI, LMN. The waypoint WPSTU is identified as not optimal and will be deleted. In consequence, the command input by the pilot is ignored. A message is generated to inform the pilot.

Problems to be solved

A command input VIA G5 instead of INSERT G5 - example B - forces the algorithm to include at least two subsequent waypoints with the airway attribute G5. The result will be the waypoint sequence WPBEG, ABC, OPR, WPSTU, GHI, LMN. Such solutions, far-off the optimal route, result in failure states of the algorithm, in which the algorithm needs assistance. By means of heuristics, i.e. rules of thumb, these problems can be resolved in future.

Besides aspects regarding the search algorithm, presently a distinction is made regarding enroute and terminal area planning task, because of different data representations for enroute airways and terminal area routing [10]. A common internal representation has not been realised yet. In general, the route planning procedure is identical for both enroute and terminal area.

5. Software Prototype Realisation

The implementation of the aircraft route planning algorithm requires the simulation of constraint management, database and man-machine-interface. The simulation may be reduced to those functional elements directly related to the route planning task.

Regarding the constraint management module, the command set of the constraint list editor as well as some basic buffer manipulation commands to facilitate parallel work on different constraint lists had to be realised. The central element of this simulation is represented by the route planning algorithm and some basic error-checking which enables the algorithm to cope properly with incorrect data input. Additionally, a navigation database had to be installed providing data access for the constraint management and for the route planning task. The third element to be realised is the interface between man and machine. The first lay-out was centred on the present CDU design simulated on a computer terminal. These requirements represent the initial system design approach for "rapid prototyping". Objectives of the system simulation were the development and investigation of the route planning algorithm, the structure of the database contents and the interaction between the different elements.

The selection of software tools was strongly influenced by the route planning algorithm, which required handling of list structured objects and recursive application of procedures. To tread new paths, a Common Lisp Programming Environment on a PC/AT was chosen for realisation. Figure 6 illustrates the current system. Because of problems regarding accuracy and general run-time in the initial development phase, the calculation of coordinates, distances and tracks was realised with a Pascal program. Data exchange between Lisp and Pascal program was provided by a PC-Assembler routine. The map display (plan mode) was implemented on an IRIS workstation with a program written in C.

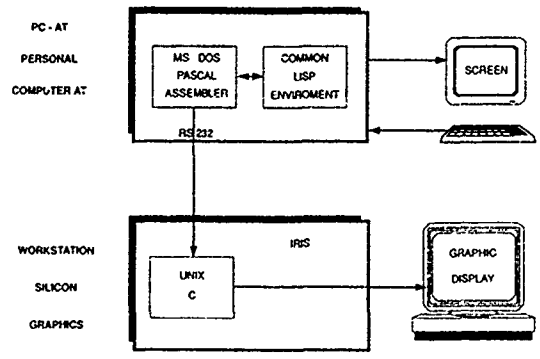


Figure 6 - Current Realisation of Constraint Management Module and Route Planning Algorithm

Currently the system is reorganised because of run-time and data storage problems. Finally, the programs written in Lisp will be transferred from the PC to a workstation, with the man-machine-interface module realised as a separate software module.

References

[1] Wiener, E.A., Beyond the Sterile Cockpit, Human Factors, 27(1),1985, p.75-90.

[2] Rouse, W.B., Geddes, N.D. and Curry, R.E., An Architecture for Intelligent Interfaces: Outline of an Approach to Supporting Operators of Complex Systems, Proceedings of the National Aerospace and Electronics Conference, Dayton (OH), May 19-23, 1986 (NAECON 1986), Vol. 3, p. 914-920.

[3] Lambert, R.E., Cockpit Information Management Through an Intelligent Pilot/Vehicle Interface, AIAA/AHS/ASEE Aircraft Design, Systems and Operations Conference, Seattle, WA, 31.7.-2.8.1989

[4] McNeese, M.D., Humane Intelligence - A Human Factors Perspective for Developing Intelligent Cockpits, Proceedings of the National Aerospace and Electronics Conference, Dayton, OH, 19.5.-23.5.1986 (NAECON 86).

[5] Ellis, B., Towards the Unmanned Cockpit, in: Knowledge Based Concepts and Artificial Intelligence: Application to Guidance and Control, AGARD Lecture Series 155, AGARD- 3-155, August 1987.

[6] Hacker, W., Software-Gestaltung als Arbeitsgestaltung, in: Fähnrich, K.-P. [Hrsg.], Software-Ergonomie, München 1987 (In German).

[7] CENA, DLR, NLR, RAE and EEC, A Conceptional Description of an Experimental FMS for Air Traffic Management Research, Issue 1.2, September 30th, 1988 (Internal Document).

[8] CENA, DLR, NLR, RAE and EEC, USER REQUIREMENTS DOCUMENT for the Development of an Experimental Flight Management System for Air Traffic Management Research, Version 3, 13-Oct-1989, (Internal Document).

[9] Garteur Action Group FM (AG) 03, A Conceptional Model of a Future Integrated ATM System and Novel Functional Requirements for a Future Flight Management System, Garteur TP 49 and 50, RAE Working Paper FM WP (89)004, 1989.

[10] ARINC, Navigation System Data Base, ARINC-Specification 424-7, August 3, 1987, Aeronautical Radio Inc.

[11] ARINC, Multi-Purpose Control and Display Unit, ARINC-Specification 739, May 15, 1986, Aeronautical Radio Inc.

[12] van Dorp, A.C., User Requirements Document for the Control Display Unit of the Experimental Flight Management System, August 1990(Internal Paper).

[13] Johannsen, G., Neue Entwicklungen bei Mensch-Maschine-Systemen, automatisierungstechnik at, Vol.. 35, Oct. 1987, p. 385-395 (In German).

[14] Wickens, C.D., Haskell, I. and Harte, K., Ergonomic Design For Perspective Flight-Path Displays, IEEE Control Systems Magazine, June 1989, p. 3-8.

[15] Roske-Hofstrand, R.J. and Paap, K.R., Cognitive Network Organisation and Cockpit Automation, 3.Symposium on Aviation Psychology, April 22-25, 1985, Columbus, Proceedings p. 71-78.

[16] Nilsson, Nils J., Principles of Artificial Intelligence, Palo Alto, 1980.

[17] Slack, M.G. and Miller, D.P., Route Planning in a Four-Dimensional Environment, Proceedings of the Workshop on Space Telerobotics, NASA-CR 184687, Vol. 3, p. 41-47.

[18] Lizza, C.S. and Lizza, G., Pathfinder - A Heuristic Approach to aircraft Routing, Proceedings of the National Aerospace and Electronics Conference, Dayton (OH), May 20-24, 1985, Vol. 2, p. 1436-1443 (NAECON 85).

[19] Wilber, G.F., Strategic Route Planning Using Informed Best-First Search, Proceedings of the National Aerospace and Electronics Conference, Dayton (OH), May 23-27, 1988, Vol. 3, p. 1137-1144 (NAECON 88).

PLANNING AND PLANNING MANAGEMENT FOR AUTONOMOUS AND SEMI-AUTONOMOUS VEHICLES

by

Milton B. Adams and Robert M. Beaton

The Charles Stark Draper Laboratory, Inc.
555 Technology Square, Mail Station 4E
Cambridge, Massachusetts 02139

ABSTRACT

Planning systems for autonomous and semi-autonomous vehicles can be viewed as performing very high level guidance and control functions. The structure of hierarchical planning systems parallels that of the management and decision-making structure of many organizations where long-term, less detailed skeletal plans are developed at the highest levels of the hierarchy, and near-term, fully detailed plans are developed and then implemented by the lower levels. Mission planning systems can be decomposed into planning algorithms and planning management functions. Together, they provide the onboard, intelligent decision-making required to plan and execute the nominal mission and modify mission activities in response to unforeseen events, with little or no reliance on human intervention. ***Planning Algorithms*** produce plans of actions that best achieve stated mission objectives within specified mission constraints for a given mission environment. ***Planning Management Functions*** control the execution of plans, monitor the progress of the plans currently being executed, decide if replanning is necessary and, if so, define the revised planning problem to be solved by the planning algorithms. In this paper, an architecture for a mission planning system is presented with descriptions of the associated planning algorithms and planning management functions. Implementation considerations for architectures in that class are discussed, and an approach to stochastic modeling of the planning process implied by such architectures is proposed.

1 INTRODUCTION

Planning algorithms and planning management functions together provide the onboard, intelligent decision-making that allows an autonomous vehicle to plan and execute its mission and to modify its activities in response to unforeseen events, with little or no reliance on human intervention. Given a well-defined planning problem (i.e., objectives, constraints and knowledge of the mission environment) ***Planning Algorithms*** produce a plan of actions that best achieves stated mission objectives within specified constraints for a given mission environment. ***Planning Management Functions***[1] monitor the progress of the plans currently being executed to decide if replanning is necessary and, if so, define the revised planning problem to be solved by the planning algorithms. In effect, planning management functions *manage* both the planning process and the execution of plans, acting as the interface between the mission planning algorithms and other onboard systems such as fault detection and isolation, redundancy management, sensing and image recognition systems, navigation and control. Much of the effort invested in automation for autonomous vehicles during the past decade has been in the development of planning algorithms for a variety of mission applications. In contrast, little effort has been made toward formalizing the development of the associated planning management functions that are a prerequisite to successfully fielding planning algorithms in operational autonomous and semi-autonomous vehicles

The complexity of mission planning and planning management problems has led to the use of hierarchical decompositions to make those problems manageable [1]. At the C.S. Draper Laboratory, hierarchical planning and scheduling architectures have been developed in several problem areas including: layered control architectures for autonomous submersibles [2]; hierarchical planning management architectures for autonomous aircraft [3]; and hierarchical planning and scheduling architectures for railroad traffic control [4]. For each of these activities the design objective has been to develop an architecture which allows the complex planning and scheduling problem to be decomposed into manageable pieces. The architecture must support the flow of information (commands, resource requests, and status reports) throughout the system that is required to support the orderly execution of the planning process. Mission and trajectory plans are developed within the hierarchy to optimize a selected objective function (e.g., minimize lethality, fuel or time or maximize mission accomplishment) subject to constraints that must be satisfied by the plans (e.g., allocations on mission time, survivability or weapons use).

Within the planning system, the level of detail to which mission activities must be planned by a given planning algorithm is determined by spatial and temporal planning horizons. Over a short planning horizon actions must be planned in great detail. In contrast, it is unnecessary, even futile, to plan actions that lie beyond a short planning horizon at the same high level of detail. Thus, there is a natural decomposition of the planning function into a *hierarchy of planners*, wherein the planning horizon decreases and the level of detail at which actions are planned increases as one moves from higher to lower levels of the hierarchy. In this paper, our discussions will be presented in the context of an aircraft mission planning problem. For that example problem, at the highest level of the hierarchy (*Mission* level) skeletal plans of the entire mission are constructed. At intermediate levels (*Route/Activity* levels), near-term actions that are consistent with the strategic plan are planned in greater detail. Finally, at the lowest level of the hierarchy (*Flight Safety* level), commands are generated for vehicle subsystems (e.g., sensors, vehicle propulsion and control systems) that execute the mission and ensure vehicle safety. Figure 1 illustrates the plans generated for the aircraft example by a hierarchy consisting of three levels. The uppermost segment of that figure depicts the potential mission objectives set forth in the mission requirements as well as threat regions that are known *a priori* and zones within which vehicle travel is restricted. Below

1 Planning Management Functions include what are traditionally referred to as Mission Management Functions The term planning management is employed to focus attention on their role in managing the onboard mission planning

that are illustrated the plans generated by the three levels of planning. These are discussed in more detail in succeeding sections

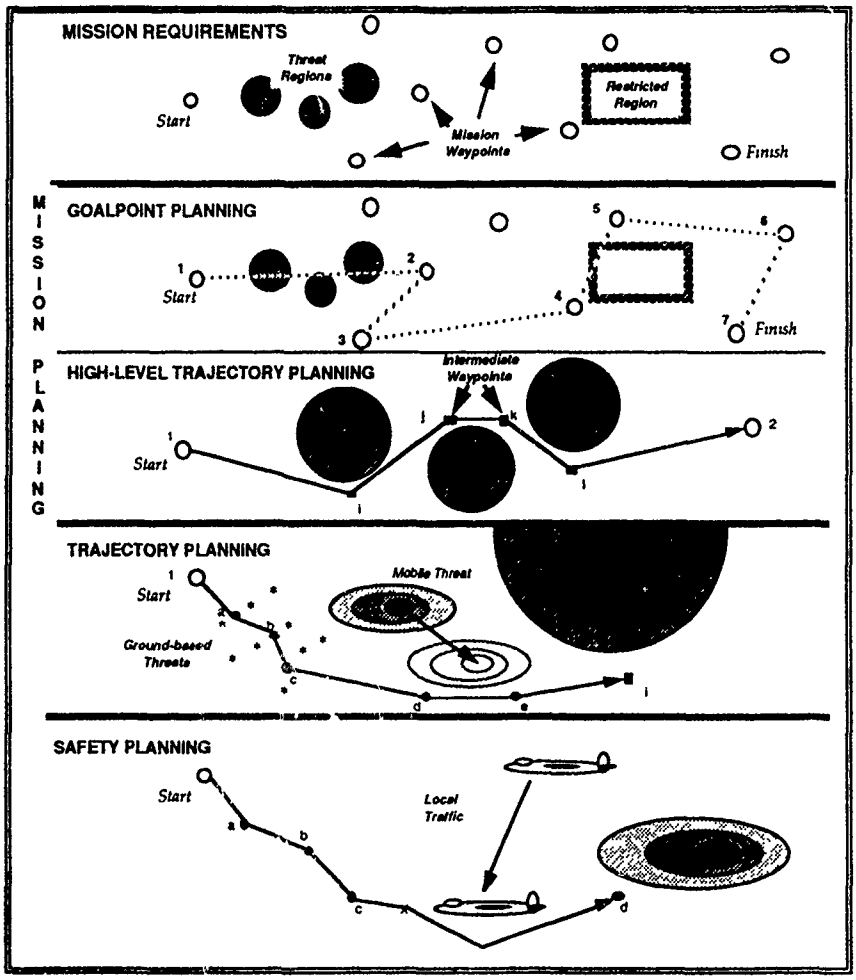


Figure 1 Planning Hierarchy Example

This paper addresses the design and development of hierarchical planning systems for autonomous and semi-autonomous vehicles with an emphasis on the *planning management functions*. Section 2 discusses the hierarchical decomposition of the planning problem and the relationship between planning and planning management. Section 3 presents a system architecture for implementing a hierarchical planning system and defines the planning management functions that must be performed in such a hierarchy. Some considerations in implementing the planning algorithms and planning management functions are addressed in Section 4. Section 5 discusses the differences between the process of producing a plan and the plan produced and proposes a modelling approach for the dynamic planning process represented by the proposed hierarchical decomposition. Finally, Section 6 closes the paper with a summary and some concluding remarks.

2 Hierarchical Decomposition of the Planning Problem

The hierarchical approach to decision-making decomposes a large problem into a number of separate levels or subproblems, thereby reducing the overall complexity, ideally with little reduction in overall system performance. The top level of the hierarchy plans the entire mission using coarse or abstract models. As one progresses downward within the decision-making hierarchy, the spatial and temporal scope of the decision-making functions decrease and the level of modelling detail increases. The selection of the individual subproblems, the specification of their local performance criteria and the modelling of their interactions are all part of an integrated approach to developing a hierarchical decomposition. An important goal in this selection is to maintain a balance in the complexity of decision-making effort across all levels of the hierarchy.

The planning system described in this paper consists of a hierarchy of *planners* which collectively cooperate to solve mission planning problems. Each planner consists of a number of components: planner-specific data, models, reasoning mechanisms (algorithms) and control mechanisms (management functions). The levels of abstraction of each of these planner components vary from level to level within the hierarchy, but must be self-consistent at any given level. The control mechanisms not only control the planning algorithms at each level of the hierarchy but also control the flow of information and control among the various planners in the hierarchy. This ensures that the planning process proceeds in an orderly fashion. Thus, each planner within the planning system (i.e., the hierarchy of planners) provides planner-specific outputs to and accepts planner-specific inputs from other planners within the planning hierarchy.

2.1 Planning vs. Management

As described above, there are two principal functional elements contained within each level of a planning system hierarchy: planning algorithms and planning management functions. The **planning management functions** are responsible for three classes of control subfunctions: controlling plan execution, managing interactions among adjacent levels of the planning hierarchy and implementing replanning decision logic. The **planning algorithms** are responsible for developing plans whenever the planning management functions have decided that replanning is appropriate. The development of individual planning algorithms is addressed elsewhere (see, for example [5,6]) and is outside of the scope of this paper: However, the interfaces between the planning algorithms and planning management functions and their mutual interactions are addressed here. Each of the planning management subfunctions is described in Section 3. The following basic questions must be answered when designing a decision hierarchy:

- How many levels are required?
- What should their temporal horizons be?
- How should decision-making be partitioned within a level?
- What constraints and objectives should be passed from level to level?
- What happens when any given decision-making unit cannot meet its constraints?

There are no well founded, systematic procedures for addressing these design questions, although research continues in an attempt to synthesize one. Empirical evidence suggests that system performance is not overly sensitive to the gross structural form of a hierarchy, e.g., the number of levels and how they are partitioned. The difficulty lies in the design of the planning and management functions that are embedded within a hierarchy. These design issues are the focus of this paper.

2.2 Hierarchical Decomposition of Automated Mission Planning

To make our discussions concrete, a specific hierarchical decomposition of the vehicle mission planning problem has been selected. The selected planning system hierarchy has three levels: the *mission level*, the *route/activity level* and the *flight safety level* (see Figure 1). The nature of the plans and the planning algorithms that are used to generate plans at each of these levels are described below. The planning management functions that manage the execution of those planning algorithms are also described in a subsequent section.

Two types of waypoints are referred to in the discussions of planning algorithms: mission waypoints and intermediate waypoints. *Mission waypoints* are associated with mission objectives and are specified as inputs to the planning system. *Intermediate waypoints* are waypoints that are generated at the discretion of the planning algorithms. They are used to define the trajectories between selected mission waypoints. Intermediate waypoints can be added to or deleted from the mission plan by the planning algorithms.

2.2.1 Mission Planning Level

At the mission planning level, the selection and ordering of a subset of mission waypoints that make up the nominal mission plan and the definition of the trajectories at a low level of resolution between the mission waypoints [5,6] are performed. The mission planning problem is decomposed into a *high level trajectory planning problem* and a *goalpoint planning problem*, each with an associated planning algorithm A by-product of the plans generated at the mission planning level is a resource allocation spanning the complete mission that is consistent with the overall mission resource budget.

2.2.1.1 High Level Trajectory Planning Algorithm

The high level trajectory planning algorithm generates near-optimal (within specified constraints on time, energy and survivability) trajectory segments between mission waypoints. In a premission planning mode, the high level trajectory planning algorithm generates a set of trajectory segments linking all pairs of mission waypoints. During a mission, some or all of these trajectory segment s may be updated at the discretion of the planning management functions to accommodate changes in the expected mission environment (e.g., weather and threats). In our implementation, trajectory segments are generated by a modified A* search over a predefined set of candidate mission waypoints [6]. The A* search methodology has been selected because it provides a computationally efficient mechanism for making trade-offs between solution optimality and search time. In generating trajectory segment s between mission waypoints, the A* search employs lethality models for anticipated threat encounters and energy consumption models that account for vehicle airspeed, altitude, altitude rate and winds. The high level trajectory planning algorithm accommodates (1) moving air traffic (multiple vehicles), (2) winds and weather systems, (3) ground-to-air threats, (4) specific criteria for waypoint capture and (5) corridors bounding trajectory segments between specified mission waypoint pairs. The trajectory segments, together with estimates of the costs that would be incurred in pursuing them, are stored in a trajectory buffer for subsequent use by the goalpoint planning algorithm.

The A* search methodology makes it possible to obtain multiple solutions (i.e., multiple segments between a given pair of mission waypoints), each corresponding to a different optimization criterion and/or set of constraints. This

provides the goalpoint planning algorithm described below with a set of trajectory segments that trade off time, energy and lethality in different ways.

2.2.1.2 Goalpoint Planning Algorithm

The function of goalpoint planning is to use the high level trajectory segments and their associated cost (resource usage) information to construct a mission plan. This is done by selecting and ordering the subset of mission waypoints (objectives) and associated trajectory segments that maximizes the utility of the mission plan under specified constraints. In our implementation, the goalpoint planning algorithm is initialized with a seed plan, typically a degenerate plan consisting only of the origin and final destination of the vehicle. This plan is iteratively improved through a number of modification cycles [5,6]. During these modification cycles, the best plan found during all previous cycles is stored in a *best plan buffer*. Within each cycle, the plan representing the initial condition for that cycle is modified in accordance with a set of heuristics. The output of this process, the mission plan consisting of the selected mission waypoints and associated high level trajectories, is placed in the *flight plan buffer*, which contains the plans that are generated at all levels of the hierarchy.

2.2.2 Route/Activity Level

The route/activity level of the hierarchy is responsible for filling in details in the plan generated at the mission level. Two classes of plans are generated at this level: more detailed routes between the intermediate waypoints that are specified in the the high level trajectory plan segments and *activity plans* required to accomplish the mission objectives associated with each of the mission waypoints specified by the goalpoint plans. The objectives, resource allocations, timelines etc. for generating these route and activity plans are all contained in the plan generated at the mission level. These plans have a shorter spatial and temporal horizon than the plans generated at the mission level. In order to perform planning at this level, the mission environment must be described at a higher level of detail than is required at the mission level. A more detailed description of the mission environment is obtained by fusing onboard sensor data with a priori map data.

A brief description of a route planning algorithm is given below. Since the nature of the activity planning algorithms will vary significantly as a function of the designated mission objective (e.g., reconnaissance, supply, ground attack, interdiction, etc.), no activity planning algorithm is described here.

2.2.2.1 Route Planning Algorithm

The primary function of route planning at this level of the hierarchy is to plan the vehicle's flight path at a higher level of detail than is provided by the intermediate waypoints of the high level trajectory segments specified at the mission planning level. The route planning algorithm produces detailed plans, consisting of a set of intermediate waypoints that are consistent with both the resource allocations established by the mission plan and the vehicle's performance envelope. The inputs to the route planning algorithm are the mission plans stored in the flight plan buffer and the current best estimate of the state of the mission environment supplied by the information fusion function. The output consists of route plans connecting a pairs of mission waypoints or intermediate waypoints defined by the high level trajectory plan. The route/activity planning algorithms[2] are required to generate plans that: (1) adhere to flight corridors and mission specifications, (2) avoid local storm systems and adverse winds, (3) avoid collisions with local air traffic, (4) avoid collisions with the ground, (5) account for vehicle system failures or damage and (6) specify the utilization of sensor and payload systems.

In our implementation, route plan generation is accomplished by a near-optimal A* search similar to that used by the high level trajectory planning algorithm of the mission planner. A major difference between the functions of the route/activity and mission planners is the information used by each in generating plans. The mission planner uses *a priori* and communicated map data. Because the route/activity planners generate plans incorporating greater detail and for shorter temporal horizons than the mission planner, the route planning algorithm requires both real-time sensor data and a priori data for generating its plans.

2.2.3 Flight Safety Level

During the execution of the nominal mission plan, events that have not been anticipated in formulating the nominal mission plan and which threaten the safety of the vehicle may occur. The planning hierarchy accommodates these events at the lowest level of the hierarchy (see Figure 2) where near-term flight safety plans are generated. These plans take precedence over any other plans in the near term.

The primary function of the flight safety planning algorithm is to provide a plan, over a very short temporal horizon, that is guaranteed to meet survivability requirements. Secondary emphasis is placed on time and fuel optimization. The fact that the flight safety planning algorithm is invoked is an indication that more reasoned approaches could not be used to obtain a plan in time to accommodate the current safety threatening situation. Inputs to the flight safety planning algorithm are contained in the flight plan buffer (the mission plan, route plan and possibly an active flight safety plan) and the current best estimate of the state of the local mission environment. As with the route planning algorithm, the flight safety planning algorithm depends heavily on fused real-time sensor data for knowledge about the state of the world. The detail of the flight safety plan is comparable to or higher than that of the route planning level. The flight safety planning algorithm produces a plan that is executable within the vehicle's performance envelope and that ensures a flight path that is within an acceptable risk of vehicle loss. The output may either be a complete flight plan to the next intermediate waypoint generated or a plan whose temporal horizon allows sufficient time for more reasoned planning to regenerate the trajectory to the next intermediate waypoint.

---

[2] More than one algorithm may be required to meet these requirements

2.3 Planning Management

Whereas planning algorithms generate the specific sequences of actions that define a mission plan, *planning management functions* monitor the current situation for events that may require a replanning response, determine the nature of that response by defining the inputs to the planning algorithms, and control the interactions among the several levels of planning. The planning management functions play the role of an executive that:

(1) monitors progress relative to the execution of the current mission plan and the vehicle's ability to continue to successfully execute the current plan,

(2) monitors the current mission situation for changes in the state of the vehicle or mission environment that may necessitate a replanning response

(3) monitors any changes in mission objectives or constraints that may necessitate a replanning response

(4) determines the nature of the replanning response by selecting the appropriate planning algorithms and defining their inputs

(5) controls the interactions among the several levels of the planning hierarchy

Note that is conceivable to implement the monitoring functions described in items 1-3 within the information fusion function; however, this is undesirable because the monitoring functions are mission related and go beyond the information fusion tasks of estimating the state of the vehicle and its operational environment. The design and *distributed* implementation of these planning management functions are outlined below.

3 System Architecture

The multi-level, distributed architecture depicted in Figures 2 and 3 has been developed to implement the planning algorithms described in Section 2 and their associated planning management functions. Within this architecture, the total planning management function is partitioned into modular sub-functions and distributed throughout the hierarchy. Each level is self-contained and has its own set of planning algorithms.

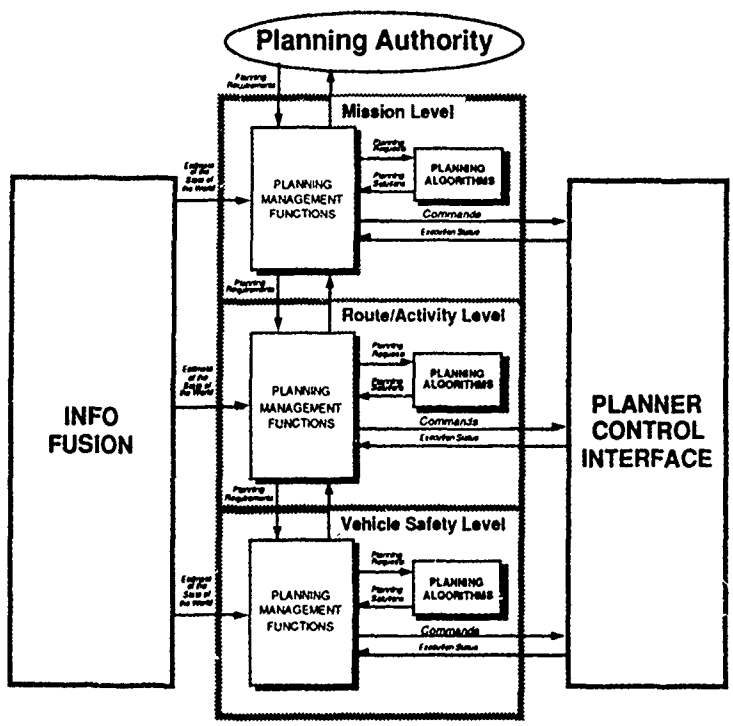


Figure 2. System Architecture

The management functions at each level make decisions based on three sources of inputs: the *mission manager databases*, which contain information about the state of the aircraft, the state of the environment and the mission objectives and constraints; the *flight plan buffer*, which contains the current plans developed at each level of the planning system; and *direct communication* from managers at adjacent levels, or, in the case of the mission manager, direct communication from a planning authority specifying changes in planning requirements.

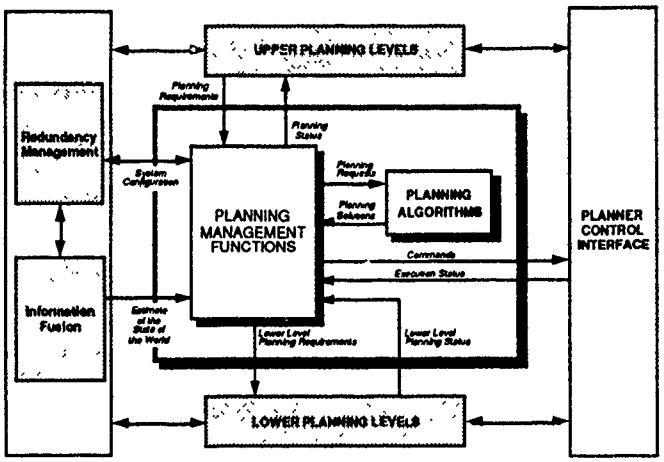


Figure 3. Generic Level of a Planning Hierarchy

The modular, distributed implementation of the planning management functions allows easy modification of one level of management without affecting the whole system. In addition, while each level of management performs similar functions and has similar interfaces, the modular structure allows each level to be tailored for planning activities specific to that level.

3.1 Management Functions

The primary role of the planning management functions is to control the planning process at each level of the hierarchy. In order to achieve this objective, four classes of management functions must be implemented for each level of the hierarchy:

(1) Plan execution functions
(2) Plan monitoring functions
(3) Replanning decision functions
(4) Inter-level coordination functions.

Each of these is discussed in detail below.

3.1.1 Plan Execution Functions

Communication between the planning system and the vehicle guidance and control functions, sensors and payload is made through the *planner-control interface* block in Figure 2. The plans developed by the planning algorithms at any of the planning system levels contain two kinds of information. One kind represents descriptions of activities that are to be executed by the guidance and control functions, sensors or payload. The other kind of information defines requirements for lower levels of planning. The content of the plan defining guidance and control, sensor or payload activities must be translated by the planner-control interface into formats that are understood by the subsystems. Once translated, these activities are executed by the appropriate subsystem, and the progress of that execution is monitored. Both the current progress of plan execution as well as the impact that the current progress may have on the vehicle's ability to successfully accomplish activities planned for the future are monitored. These and other monitoring functions performed by planning management are discussed below.

3.1.2 Plan Monitoring Functions

3.1.2.1 Monitoring Plan Execution

Associated with each mission plan is a measure of its expected accomplishment, referred to here as its utility, and its expected profile of resource utilization. Thus, the progress of the execution of the current plan can be measured along two dimensions: accomplishment and resource utilization. The actual *level of mission accomplishment* is compared with the expected level of mission accomplishment. The *actual resource utilization* is compared with the planned profile. If the expected accomplishment or the actual resource utilization differs significantly from what is expected, then planning management may initiate replanning. If the expected resource utilization exceeds the available onboard resources, then replanning will produce a new plan that can be accomplished within the available resources. If the actual resource utilization is lower than that predicted by the usage profile, then replanning might be invoked in an attempt to exploit surplus resources to produce a plan of greater utility than that of the current mission plan. Part of the design of the planning management function is the selection of the decision thresholds that are used in the comparisons between the actual values and the predicted values. Mechanisms must be established for communicating the results of plan monitoring to provide status information to superior levels of the planning hierarchy.

3.1.2.2 Monitoring Mission Objectives and Constraints

Any changes in either the mission objectives (mission waypoints) or explicit mission constraints are made through a user interface with the pilot/crew in a piloted vehicle or through communications from a higher planning authority for an autonomous or semi-autonomous vehicle. Changes of this type will require modifications to the mission requirements database and will result in an associated request to replan. These changes will have the most significant impact on planning at the mission level. In addition, it may be desirable to have the ability to externally override the planning functions, allowing the mission plan to be controlled manually. This capability will be useful during initial flight demonstrations of autonomous planners since it provides a mechanism for manual control as well as for testing specific flight plans, trajectories and vehicle maneuvers. This point is further elaborated in the section *Levels of Autonomy*.

3.1.2.3 Monitoring Vehicle and Environment Status

Planning management functions are required in order to monitor the status of both the vehicle and the mission environment in order to detect any changes that may affect the vehicle's ability to successfully execute the current mission plan. This monitoring is event-driven and is based on information received from the information fusion function and its redundancy management sub-function. In our implementation, the decision to initiate replanning in the face of events of this type is made by comparing the a priori plan utility with the value of expected utility conditioned on the given event.

The two primary classes of changes in vehicle status that may triggers a re-evaluation[3] of the current mission plan are failures and damage to the vehicle. For redundant systems, a failure may result in an immediate loss of performance as well as in reduced subsystem reliability (e.g., one channel of a triplex system goes down). Here, performance refers to the vehicle's dynamic capability (i.e., performance envelope), sensing capability or payload system's capability. Reduced system reliability implies a lower probability of having the capabilities required to successfully accomplish future mission objectives. The planning management functions employ an onboard system reliability model to predict the probability of the future availability of specific vehicle subsystems given the current failure status of those subsystems. These reliability predictions are incorporated into the computation of the expected level of accomplishment of candidate mission plans For example, a decision to abort a mission would depend significantly on the number of redundant elements that have been lost due to failures or damage and the impact of the failures or damage on system reliability.

Features of the mission environment that may impact the expected utility of the current plan are weather (winds/storms), threats and local air traffic Timely information regarding these features of the mission environment must be supplied by the information fusion function.

Once a *significant* change in the state of the vehicle or environment has been detected by the planning management functions, one of several events may occur: (1) replanning may be immediately initiated, (2) the plan may be re-evaluated in the face of the detected change and on the basis of that re-evaluation a decision to replan can be made or (3) replanning at that level may be delayed because it has been decided that a lower level planner should first respond to the change or it is has been observed that a lower level planner has independently begun to respond to that change.

3.1.2.4 Onboard Vehicle Performance Models

Onboard vehicle performance models are used in evaluating candidate plans and in detecting degraded vehicle performance. When failures, damage, or other potential performance degrading events occur, these models must be updated. Performance monitoring and modeling take place over both short and long time scales and at various levels of abstraction. The updated models developed over long time scales are used to decide whether mission replanning may be necessary; and, if so, the updated models are used in building the new plan and allocating resources across the time span of that new plan. Modeling over a shorter time scale is used in emergency situations to evaluate the immediate impact of unidentifiable failures on the performance of the vehicle and its subsystems These may be simple input-output models that provide approximate response of the vehicle to specific commands.

The success of performance modeling requires appropriate interaction with mission planning/management. For example, in order to identify or update models of vehicle performance, specific vehicle maneuvers may be required in order to identify failed actuators or to "excite" the proper vehicle modes to identify and model the performance degradation due to ineffective aerosurfaces.

Qualitative vs. quantitative models

Examples of areas where performance monitoring and updating are important from the point of view of both fault tolerance and mission planning/management are:

- Vehicle fuel consumption rate as a function of flight condition
- Vehicle turn / climb / descend capability
- Vehicle speed response
- Sensor subsystems performance
- Payload subsystems capability

3.1.3 Replanning Decision Functions:

Given inputs from

(1) **superior levels** regarding plan requirements,
(2) **subordinate levels** regarding planning and plan execution status,
(3) the **planner control interface** regarding plan progress and
(4) **information fusion** regarding the state of the vehicle and the mission environment,

[3] The evaluation will be based primarily on the expected utility of the current plan given the current estimates of the state of the vehicle and environment.

the planning management functions at each level must decide whether to continue to pursue the current plan that has been developed at that level or to replan. Given that a decision has been made to replan, inputs must be developed for the planning algorithms. In effect, those inputs define the problem to be solved by the planning algorithms. Included in those inputs are the constraints or resource allocations that must be satisfied by the plan (e.g., allocations on mission time, fuel, survivability, and weapons use) as well as the objective function that should be optimized in developing the plan (e.g., minimize lethality, minimize fuel, minimize time and maximize effectiveness). In cases for which there are multiple objectives, the management functions must establish the relative importance among those objectives. Finally, given that replanning must be performed in real time (i.e., soon enough), the management functions must decide how much time is available for planning and how the onboard computational resources should be allocated in generating a new plan.

3.1.3.1 Reconfiguration / Recovery from Failures

Planning management plays a joint role with redundancy management in reconfiguration and recovery following failures. For a given phase of a mission, the best configuration of the aircraft's hardware and software depends on the objectives that are to be accomplished during that phase. In addition, recovery from failures can be aided by the lowest level of the planning hierarchy in that the lowest level specifies short term flight trajectories requiring only the most stable flight modes immediately following a significant failure event.

In the event of a failure, the primary objective of redundancy management is to configure the vehicle to ensure stability and safety of flight. Having insured that, redundancy management provides the planning system with information regarding the status of onboard systems and the range of vehicle capabilities that are possible given that status. The planning systems must account for that achievable range in developing plans. Given a formulated plan, the planning system apprises redundancy management of the associated vehicle capability requirements, and redundancy management configures the non-failed vehicle subsystems in a manner that best realizes those capabilities.

For aircraft operations, it is important to distinguish between two types of failures that require redundancy management . One type consists of those failures that impact flight safety. These types of failures (e.g., actuator failures, computer failure) represent a threat to immediate safety of the aircraft. The second type consists of those failures that impact mission effectiveness. These types of failures (e.g., payload failures, sensor failures) do not pose a threat to the safety of the aircraft but they may impact the mission. For safety related failures redundancy management using standard techniques. However, for mission related failures, redundancy management must be performed in the context of the current mission situation which requires interaction with the planning management functions.

3.1.3.2 Control the Generation of Plans

Each of the monitoring functions described earlier ultimately provides information regarding whether or not to initiate replanning. *Metaplanning* literally translates to "planning about planning" and here refers to the way in which the process of generating a new plan is controlled. Some of the elements of metaplanning relate to determining the level(s) of the hierarchy at which replanning should be performed, deciding which planning algorithm to employ at a given level, and establishing the time allotted to generating a new plan. The last section of the paper proposes a method for modelling the planning process that provides insight into how to formulate a "control law" to insures stability of the planning process.

Planning management must be able to distinguish between small departures from the current mission plan that can be corrected by making small perturbations to that plan (e.g., changes in commanded airspeed) at the route planning level and significant departures that can only be corrected at the highest level of the planning hierarchy. For example, metaplanning may initially restrict the mission planning level by requesting that it first attempt to resolve any problems through modifications to high level trajectory plans before attempting complete replanning of the mission waypoint selection and ordering. Discriminating between small and significant departures from plans generated at all levels of the hierarchy is a principal metaplanning function. For small departures, the planning algorithms may be directed to initiate replanning using the current plan, potentially reducing the time required to find a good new plan. For large departures, replanning may begin with a default "empty" plan, requiring the new plan to be completely rebuilt.

3.1.4 Inter-Level Coordination Functions

In a hierarchical system, higher levels provide guidance to lower levels. In the planning system, the plans generated at higher levels contain both constraints and objectives that serve as requirements for planning at lower levels (see Figure 2). Planning management is responsible for translating that information into requirements for lower levels. Conversely, each subordinate level receives requirements from its superior, and those requirements must be properly interpreted for input to the planning algorithms at the subordinate level. In addition, each subordinate level must provide information to its superior regarding the status of the execution of the current plans generated at that subordinate level. The primary flow of information within the hierarchy planning is that requirements drive the planning process from higher levels to lower levels, while the changes in the operational environment drives the planning process from lower levels to higher levels.

3.2 Implications for the Design of Planning Algorithms

3.2.1 Isolating the Planning Algorithms

One of the primary objectives of the design of the architecture depicted in Figures 2 and 3 has been to make a clear distinction between reasoning and control mechanisms, i.e., between planning and management functions. The architecture has been defined in a manner that provides clean interfaces between the planning algorithms and the planning management functions. The major benefit of this approach is that it simplifies the already difficult task of designing the planning algorithms by ensuring that management issues need not be addressed by the planning algorithms themselves. In addition the planning algorithms are packaged in a modular fashion that facilitates the

insertion of new algorithms and allows one planner to call another (see Section 4) without consideration of potential management side effects.

3.2.2 Time to Plan

In order for the hierarchy to perform effectively in real time, the time allocated to a given planning algorithm for a specific planning task must be established by the planning management functions. Constraints on time to plan may be defined at a given level or may be passed down from a higher level. In addition to the objectives and resource constraints that define a planning problem, each planning algorithm also requires an initial state as an input. Indeed, every plan must begin at a point in both space and time that is reflected in its initial condition. Choosing that point in the future implicitly bounds the time available to plan. Thus, the planning management functions at each level are responsible for coordinating the choice of an initial condition on the planning process with the planners at other levels. Furthermore, the planning management functions must decide which planning algorithms may be most appropriate given the available time to plan. Establishing the time to plan and associated planning initial conditions is a non-trivial decision-making problem, but one that is crucial to the overall performance of the planning system.

3.3 Levels of Autonomy

The *level of autonomy* of an automated planning system can be measured in terms of the frequency of operator interaction with that system and the complexity of those interactions. The more frequent and complex the interactions, the lower the level of autonomy. A vehicle capable of operating at several levels of autonomy provides an effective mechanism for ***testing and evaluating*** the performance of autonomous planning and decision-making functions without incurring the risks of completely autonomous operations. It also provides the capability for initially ***operating*** an autonomous vehicle at a low level of autonomy, while facilitating a gradual transition to the ultimate objective of complete autonomy. The planning management functions serve as the interface between the onboard, automated systems and the human operator (see Figure 4).

In order to design planning management functions that permit operations at multiple levels of autonomy, it is useful to partition the planning management functions in a way that is natural to human decision-makers. The execution, monitoring and replanning decision-making functions described in the preceding sections represent such a partitioning. The information flowing into those partitions and the types of decisions they produce as outputs will be the same whether the decisions are made autonomously or by a human. In the case of the operator-in-the-loop, the presentation of the input information to the human decision-maker is provided through an appropriate user-interface. Partitioning the decision-making in this manner provides the additional benefit of making the autonomous decision logic easier to test, debug and evaluate, since the human tester can participate directly in the tests via the interface designed for person-in-the-loop operations.

The design of the planning management functions should be structured so as to most easily accommodate the three potential modes of operation described above: autonomous, operator-in-the-loop and test[4]. The hierarchy shown in Figure 2 shows a *planning authority* interacting with the planning system only at the highest level. In general, interactions may be required at any of the levels shown in Figure 4, especially during testing of the system. The hierarchy and its planning management functions should be designed from the outset to accommodate the expected variety of interactions and modes of operation.

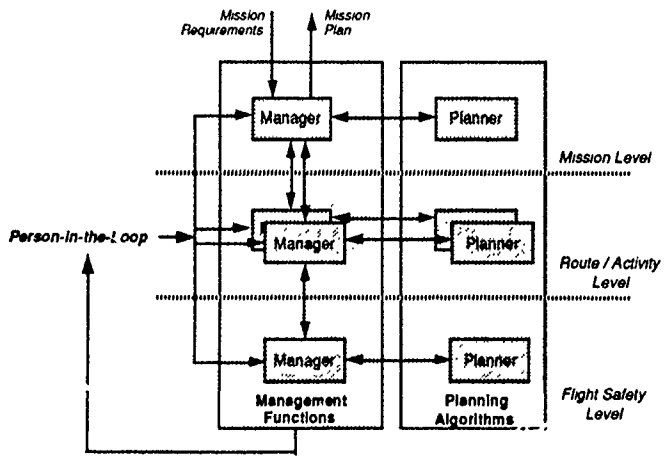


Figure 4 Person in the Loop

[4]Note that an operator-in-the-loop test mode will be required even for autonomous systems

4 Implementation Considerations

The preceding has focused on the requirements for the planning management functions. In this section we discuss important issues that must be considered when implementing the planning management functions in a hierarchical planning system.

Below we describe a standardized software framework within which the planning management functions and planning algorithms at each level of the planning hierarchy can be implemented. Establishing such a framework will not only simplify the design process, but will also make it easier to expand the system design at some future time. For example, for the aircraft planning problem discussed here, we have identified a hierarchy containing two levels of trajectory planning (referred to here as high level trajectory planning and route planning). If in the future it were decided that three levels of trajectory planning were preferred, the third level could be added with much less effort if a standard framework is used for each level.

The standardized implementation described here for a particular level of the hierarchy contains three basic functions (see Figure 5):

- ***A Manager*** : The component which contains all of the planning management functions discussed previously in the paper.
- ***One or More Planners*** : The planners generate solutions to planning problems that are defined by the manager and/or other planners.
- ***A Manager/Planner Interface***: This component of the framework provides a mechanism for controlling the flow of information between management functions and planning functions both within and across levels of the hierarchy.

The details of the design of the software within each component need not be identical across levels of the hierarchy. What is important is to standardize the information and control flow in and out of the components.

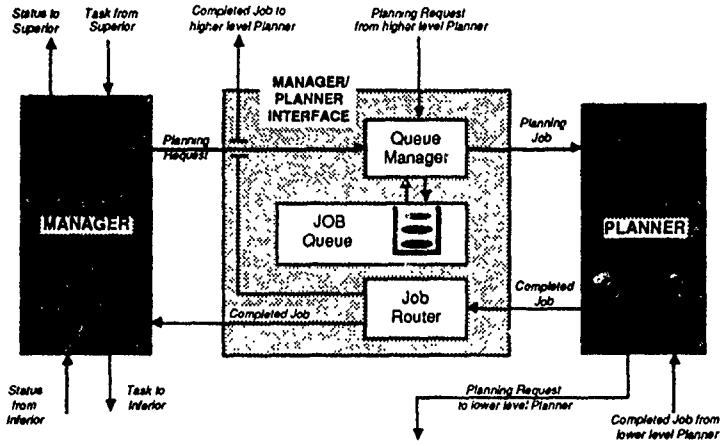


Figure 5. A Generic Level of the Mission Planning Hierarchy.

4.1 The Manager

Consider the following example. The mission planner creates a flight plan that consists of a number of mission waypoints and associated high level trajectory segments. Prior to the execution of a new trajectory segment, the mission planning level passes the new segment to the route level manager as a task. The task includes the destination waypoint and the operational resource allocations. The route level manager directs the route planning algorithms to create a detailed route plan, to execute the route plan, monitoring the execution of that portion of the flight plan, and to replan as the situation warrants. In situations where a manager cannot produce a plan that satisfies the constraint and resources allocated by its superior, it reports back to the higher level manager. For example, if the route level manager cannot produce a satisfactory route within the specified resource allocations, it reports this situation back to the mission level manager. The mission level must reallocate the available resources to meet the current shortfall identified at the route level or reformulate the planning problem.

When the manager at a given level determines that the effectiveness of the plan that it is currently executing has been reduced or when the manager identifies an opportunity to improve the effectiveness of the current plan, the manager generates the appropriate job and sends it to the queue manager.

4.2 The Manager/Planner Interface

To support the flow of information throughout the hierarchy, it is desirable to standardize the structure of that information. This can be accomplished by employing a data structure referred to here simply as a "Job". A *job* identifies a planning problem that some element of the hierarchy (either a manager or another planner) has defined. A job is a mechanism for packaging a planning problem into a standardized form that can be communicated throughout the hierarchy. Each job is made up of seven elements:

- *ID*: A unique ID that allows the hierarchy to keep track of the different jobs in the system.
- Creator: The manager or planner that created the job. This information is used by the job router to know where the results of the job should be directed when it is completed.
- Planner: The planner to which the job is to be sent. This information is required to define the destination for a job when there is more than one planner within a level of the hierarchy.
- Priority: A value that indicates the relative importance of the job.
- Start-Up Time The time the job was created. The start-up time, together with the job's priority, is used to determine the job's location in the job queue.
- Problem Description: The complete set of inputs to the planning algorithm including objectives, constraints, resource importance factors and the time available to plan.
- Problem Solution : The place holder used to store the solution to the planning job when the planner has completed its task.
- Status: The status of the job (e.g., on the queue, executing).

Jobs are initiated by the manager. They are deposited in a job queue within the Manager/Planner Interface according to a priority and time spent in the queue. When a job reaches the top of the queue, it is executed by the planning algorithm and the results (*problem solution*) are sent back to the requesting manager or planner via the job router. At any point during the execution of a job, the manager can direct the queue manger to interrupt or terminate the job. Planning algorithms can also create jobs for planning algorithms at lower levels of the hierarchy. These jobs get handled in an identical manner although the information is passed across levels of the hierarchy An example of a planning algorithm generating a job occurs in the mission planner. In order to determine the best ordering of the mission waypoints, the goalpoint planning algorithm must know the approximate cost of flying between pairs of mission waypoints. It obtains this information by sending a job to the high level trajectory planning algorithm.

4.3 The Planner

The system architecture depicted in Figure 2 has been developed to eliminate the need for planning algorithms to treat management issues. One concern when orchestrating the execution of the planning algorithms is the need to balance the computational load of the system so that planning is completed across all levels "just in time". Even though the management functions perform the command and control associated with the hierarchy, it is expected that in most cases, the bulk of the computational requirements will come from the planning algorithms. It is therefore particularly important that the planning algorithms be designed to perform their functions within a specified availability of the computational resources of the system. One method of achieving this goal is for the management functions to impose "time to plan" limitations on the planning algorithms during operation.

5 Topic for Future Research: Modeling the Planning Process

In this section, we propose to model the process of formulating a mission plan (either the complete initial plan or an updated plan created in response to an unexpected event during mission execution) as a discrete event stochastic process [7]. By modelling the planning process in this way, one can gain insight into how to develop the control law for such a process, that control law being embodied in the planning.management decision-making functions described earlier. Considerations in creating such a model are discussed below in the context of the initial plan generation process. Generalization to the replanning process is discussed at the end of the section. It should be noted that we only introduce out approach to modelling the planning process and suggest avenues for further research into this modelling and control problem.

The process of generating an *initial* plan begins at the highest level of the hierarchy. The objectives and constraints (resource allocations) that define the planning problem at each level successively "flow down" from superior to subordinate levels. Ideally, each subordinate level creates a plan that "best" accomplishes the objectives set forth by the superior within the allocated resources and specified time-to-plan. The metric (or metrics) by which the "best" plan is measured is also established by the superior[5]. This provides a mechanism to insure that the overall measures of mission accomplishment remain consistent across planning at all levels. If sufficient resources are allocated for accomplishing the objectives defined for each successive subordinate level, then planning will be successful on the first try at every level and the total time to generate a complete plan (that is, across all levels of the hierarchy) is a ***deterministic*** quantity and is simply the sum of the times required for generating plans across all levels. Unfortunately, since each superior level must use models to "predict" the expected resource requirements for its subordinate(s), those predictions will not always be sufficiently accurate to ensure this ideal solution. Specifically, it will often happen that the subordinate at some level will discover that the resource allocations predicted and passed down by its superior are insufficient to create a feasible plan, much less an optimal one.

Of course, the ideal flow down solution could be achieved if each superior level always provided a large resource margin relative to its predictions. Specifying generous resource allocations,can result in a plan of reduced overall mission accomplishment relative to the maximum accomplishment possible using all available resources,. Providing too little margin in resource allocations can result in a planning deadlock due to subordinates being unable to find feasible plans. As illustrated by the following example, the tradeoff between generous and tight resource margins is one that can be addressed by the proposed model of the planning process.

Consider two cases wherein the resource allocations provided by the superior are inconsistent with the plans that can be developed by the subordinate using higher fidelity models: (I) the subordinate requires more resources for a feasible solution and (II) a good plan requires far fewer resources than predicted by the superior. In case I, the subordinate must

5 At all levels, secondary measures of plan merit relating to resource usage should also be considered The relative weighting of one type of resource versus another (e.g., fuel vs. survivability) is also established by the superior level

send a request back *up* to the superior for either more resources or a reduction in the scope of the expected objectives to be planned for at the subordinate level. Along with this request the subordinate may also send the best plan (or plans) that could be produced for the stated objectives and the *additional* resources required to pursue that plan. In this case, the superior must decide how to reallocate resources across its larger (relative to the subordinate) temporal and spatial horizons, possibly relaxing objectives, in order to make subordinate level planning feasible. That reallocation implies a change in the superior's plan, potentially abrogating the planning effort already expended at other levels and thereby delaying the time to complete the planning process. Alternatively, the superior could pass the problem further up the hierarchy to have the resource inconsistency addressed at a yet higher level. Of course, the higher in the hierarchy that a problem is resolved, the greater the extent of lower level planning effort that may be obviated, further increasing delay in completing the planning process.

In case II, the subordinate returns its best plan along with an indication of the associated resource surplus. The superior may choose not to reallocate the surplus, with the consequence that a plan of potentially higher levels of accomplishment may be sacrificed (other considerations regarding resource surpluses are discussed later in this section).

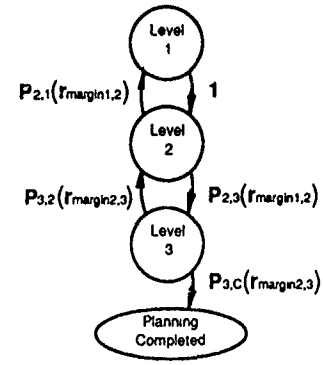


Figure 6 Planning Process State Transition Diagram

One element of the stochastic modelling problem is that of approximating the impact on the planning process of the two effects of resource budgeting, namely, resource margins that are too tight can lead to an unstable planning process while resource margins that are too generous can lead to an overall plan of significantly reduced effectiveness. Thus, one of the models that must be developed is of the probability that a subordinate will be unable to create a feasible plan that satisfies the superior's resource margins. In the simplified **planning process state transition diagram** in Figure 6, that probability is denoted by $P_{j,i}(r_{margin\ i,j})$ with Level i as the superior and Level j as the subordinate and with $r_{margin\ i,j}$ a measure of the resource margin provided by Level i for Level j's objectives. The probability distribution for the total time to complete planning is computed as a function of the times to generate a plan at each level and the probabilities of the planning process transitioning either up or down in the hierarchy. Clearly, this model is a gross simplification in that the time to plan at any level is specified by other planning management functions which would also require modelling. However, even simple approximate models may provide significant insight into the impact of the selection of the resource margins. Finally, the stability of the overall process is a major concern and it is anticipated that this approach to modelling will be useful in assessing stability and in quantifying stability margins.

In addition to the planning process, there is a stochastic element to plan execution that lends itself to modelling. That is, the degree to which a plan is susceptible to failure in the face of unforeseen events is partially a function of resource margins. That is, when margins are tight and additional resources are required for near-term accommodation of unforeseen events, the long-term plan may become inviable due to a lack of sufficient resources. Again, there is a trade off between liberal resource margins that provide the reserves required to absorb the impact of unexpected events and tight resource margins that are set in order to allocate as much of the available resources as possible to accomplishment of valued objectives. Replanning is initiated when the plan fails. If the expected rate of plan failure (the rate of initiating replanning) is sufficiently high, then there may be insufficient onboard computational resources to support that amount of (re)planning. Thus, this additional effect of resource margins must also be accounted for in the design of the planning management functions.

6 Summary and Conclusions

An architecture for mission planning systems for autonomous or semi-autonomous vehicles has been described. That architecture has been described in the context of a specific planning system hierarchy designed for aircraft mission planning problems. However, the discussions here apply equally across a spectrum of space, air, land and undersea mission planning problems. A distinction has been drawn between the algorithms that create plans and the decision-making functions that control the process of creating and executing the plans: the planning management functions. A detailed discussion of the decision-making functions that comprise planning management has been presented and a standardized framework has been proposed for implementing some of the elements of the planning architecture associated with planning management. Finally, an initial approach to developing a stochastic model of the planning process has been proposed. Models of this type will be useful in helping to design the decision-making functions that control the planning process.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the contributions of their colleagues James Harrison, Richard Hildebrant and William Weinstein in formulating the approach to planning and planning management described here. They are especially indebted to the many helpful suggestions provided by Mr. Harrison on several drafts of this paper.

REFERENCES

[1] Albus, J. S., Lumia, R., and McCain, H., "Hierarchical Control of Intelligent Machines Applied to Space Station Telerobots," IEEE Transactions on Aerospace and Electronic Systems, vol. 24, No. 5, September, 1988.

[2] Loch, J., Waller, E , Bellingham, J.G., Beaton, R. and Triantafyllou, M., "Software Development for the Autonomous Submersible Program at MIT Sea Grant and Draper Laboratory," Proceedings 6th Int. Symp. on Unmanned Untethered Submersibles Technology, pp 25-32, 1989.

[3] O.L Deutsch, M. Desai, and L. A. McGee, "Far-Field Mission Planning for Nap-of-the-Earth Flight", Proceedings of the American Helicopter Society Meeting, Cherry Hill NJ, Oct 13-15, 1987.

[4] W.W. Weinstein, M.B. Adams and R R. Hildebrant, "Traffic Flow Control System," CSDL Technical Report, June 1988, The Charles Stark Draper Laboratory, Inc, Cambridge, MA.

[5] M. B. Adams, O. L. Deutsch, and J. V. Harrison, "A Hierarchical Planner for Intelligent Systems," Proceedings of SPIE - The International Society for Optical Engineering, Vol 548, Arlington, VA, April 9-11, 1985.

[6] R.M. Beaton, M.B. Adams, and J.V. Harrison, "Real-Time Mission and Trajectory Planning", Proceedings of the 26th IEEE Conference on Decision and Control, Los Angeles, CA, December, 1987

[7] X -Y. Rien and Y.-C. Ho, "Models of Discrete Event Dynamic Systems," IEEE Control Systems Magazine, Volume 10, Number 4, pp. 69-87, June 1990.

INTELLIGENT REAL-TIME KNOWLEDGE BASED INFLIGHT MISSION MANAGEMENT

George F. Wilber
Boeing Military Airplanes, MS 9H-84
P.O. Box 3707
Seattle, WA 98124-2207
(206)394-3195

SUMMARY

This paper describes the problems and issues of developing a tactical mission manager. It discusses development aspects of intelligent real-time avionics, and outlines an efficient real- time AI methodology and implementation for the development of the intelligent systems. It also outlines advanced software development techniques and provides an overview of related Boeing research efforts.

INTRODUCTION

Tactical combat missions are becoming more complicated due to improved enemy tactics, equipment, communications, and training. Advanced aircraft performance, and sophisticated subsystems and sensors increase aircrew work load. The missions are more dynamic due to the introduction of highly mobile threats, real time intelligence updates, and advanced battle management and control interfaces To successfully carry out a modern tactical combat mission requires the aircrew to synthesize, filter, comprehend, and respond to masses of sensor, threat, mission, and aircraft systems data in real time. Only a combat aircraft equipped with a flexible, real time, adaptive mission planning and execution capability can perform effectively in this environment

The automated inflight mission manager provides real-time mission planning and replanning capabilities, facilitates increased aircrew situation awareness, monitors aircraft and mission status, analyzes key mission situations and problems, and provides the aircrew with intelligent courses of action. The mission manager is an intelligent avionics system. Intelligent systems differ from conventional systems, primarily, because they must make their own decisions. Conventional avionics systems spend most of their time processing algorithms. Intelligent avionics systems spend a great deal of effort processing complex knowledge or logic paths These logic paths are usually based on expert derived experience based knowledge.

A great deal of the complexity of intelligent software is contained in the reasoning and decision making logic. The algorithms still contain the bulk of the code and perform most of the processing functions.

High order programming languages provide effective constructs for developing algorithmic computer code. However, only limited facilities are available to encode the complex logic definitions. Intelligent avionics systems must process large amounts of logic to make their decisions. When the logic can not be broken down hierarchically, software complexity is greatly increased. The code becomes very complex, incomprehensible, and unmaintainable.

Artificial Intelligence (AI) knowledge based techniques were developed to allow problem domain knowledge to be acquired from experts and efficiently encoded for automated computer processing, reasoning, and decision making. This allows the development of systems that efficiently encode the knowledge obtained from experts, and provides the facilities necessary to carry out the automated reasoning process. The problem with current knowledge based systems is that they generally do not function on avionics computers, in embedded environments, or in real time.

The embedded avionics environment contains some specialized characteristics that distinguish it from many conventional AI environments. These differences make it necessary to develop specialized architectures and software that exploit the idiosyncrasies of embedded systems. When exploited, these characteristics allow efficient development of very intelligent systems. In fact, they bound the AI problem, allowing processing shortcuts which increase efficiency and reduce AI processing overhead.

Developing an automated inflight mission manager requires the formulation of knowledge based software development techniques, methods, and tools. Development environments must be built that will allow average skill level computer programmers to develop, test, integrate, and validate the intelligent software. Run-time environments must be developed that provide low overhead, real-time, knowledge driven software processing. These systems must use conventional programming methods, practices, engineers, languages, and facilities to create hybrid intelligent AI and conventional avionics systems.

AUTOMATED INFLIGHT MISSION MANAGER

The objective of the Automated Inflight Mission Manager is to enhance the attack aircraft's capabilities for the 1990's and beyond, emphasis has been placed on the ground attack role of tactical, strategic and unmanned aircraft. Enhanced mission capabilities include: a paperless cockpit, where the combat mission folder is contained within the aircraft avionics; automated control of an enhanced sensor suite to improve target search, detection, and identification; intelligent threat assessment and countermeasure selection to minimize risk to the aircraft; automated mission planning and replanning to provide mission adaptability and flexibility; and management of advanced controls and displays to minimize aircrew work load and enhance situation awareness The inflight mission manager internalizes the hard copy mission folder used by aircrews today, and represents it as a paperless electronic mission folder which can be changed and updated inflight by the aircrew. The following capabilities are required of the automated mission manager:

1) An intelligent system to synthesize, prioritize, process, and display mission events in real time to the aircrew. The system will provide intelligent mission problem resolution for aircrew decision actions, enhance weapon system survivability, and increase the probability of mission success.

2) An automated inflight capability to plan and replan mission activities in response to degraded aircraft performance, unforeseen events and defenses, and changes in: targets, threats, routing, timing and air refueling.

3) An interactive system that incorporates aircrew and communications inputs, resident data bases, algorithms and sensor data for computing and optimizing inflight re-routing, threat avoidance and weapons application. The system recommended actions will be presented to the aircrew in a time critical action priority sequence.

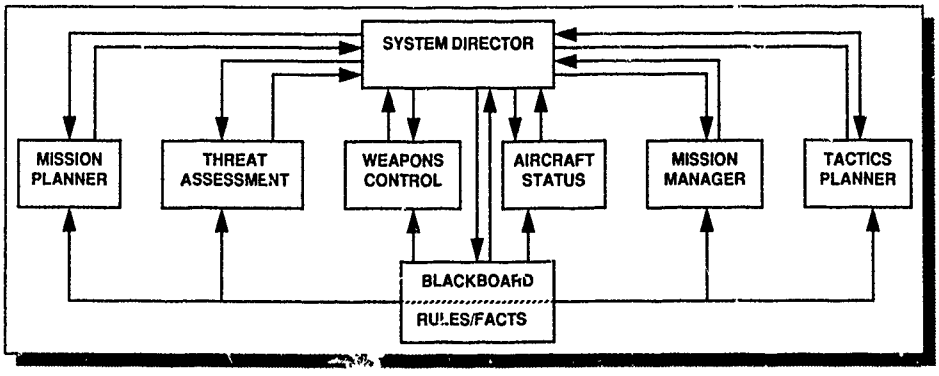
Mission manager problem inspection reveals that mission management is actually a collection of individual management problems. These problems are separate and distinct in nature, function, and task, but are highly dependent on each other for information and control. The major task in outlining an acceptable approach to developing a mission manager is defining the individual management problems and their inter-relationships.

Individual mission management problems cover a wide variety of functions that require vastly different solution processes. Examples of individual problems are: checklist management, aircrew interface, mission planning, sensor control, threat assessment, countermeasure selection, and aircraft status updating and monitoring. Each of these problems is unique in its nature. Some can be solved with standard algorithmic programming techniques, others require automated reasoning techniques and some demand combined reasoning and algorithmic solutions.

Control of multiple mission and avionics systems and processes requires an intelligent system director. The system director is an intelligent controller that controls and focuses system processing and resources that by scheduling and invoking system functions. The system director handles the communications between the individual mission manager problem solvers and accomplishes two major tasks. See figure 1.

The first system director task is to activate the mission plan, and carry out the mission. The mission plan is the contract between the aircrew and the automated mission manager system. It contains all information needed to successfully carry out the mission. This is by far the most important task the system director performs. Even on a perfect mission where no abnormal events occur the mission manager is essential. It controls aircrew displays, ensures correct procedures are known and followed, and monitors, analyzes and filters mission and aircraft status to provide the aircrew with the best possible situation awareness. The system director performs this task by sequencing through the mission plan and activating applicable processes.

The second system director task is to analyze the global world state and look for anomalies that may preclude executing the mission as planned. This involves monitoring the aircraft, weapons and mission status. The system director may invoke mission replanning, threat avoidance, or other mission manager processes to modify the mission plan. The system director performs this task by looking



- *The inflight mission manager is a hierarchy of cooperating systems*
- *System director controls and monitors the individual systems*
- *Global data is stored in the blackboard structure, available to all systems*

FIGURE 1: MISSION MANAGER ARCHITECTURE

for distinctive mission situations which are passed to the system director from the individual mission manager component processes. The system director then takes the appropriate responses by scheduling required mission manager functions to be executed.

The mission manager performs crucial mission tasks which require the real-time analysis and interpretation of numerous dissimilar pieces of information. It performs autonomous or semi-autonomous reasoning and decision making, and filters vast quantities of data to reduce operator workload and to ensure that correct decisions are made and correct actions taken. It must be adaptable to changing requirements both during mission execution and over the entire system lifecycle. This is especially crucial because it contains large amounts of heuristic user defined logic.

INTELLIGENT AVIONICS SYSTEMS

The automated inflight mission manager is an example of an intelligent avionics system. Reasoning and decision making functions must be embedded directly in it to provide the level of competence necessary to accomplish the complex missions. The system must function autonomously or semi-autonomously and should reduce not increase aircrew workload.

Intelligent avionics systems differ from conventional avionics systems, primarily, because they must make decisions based on user defined logic statements. Conventional systems spend most of their time processing algorithms. They process only enough logic to drive the algorithms, and little of this logic is very complex.

Conventional avionics software is complex, difficult to understand to maintain. A contributing factor is that the logic and algorithms are woven together forming complex object relations.

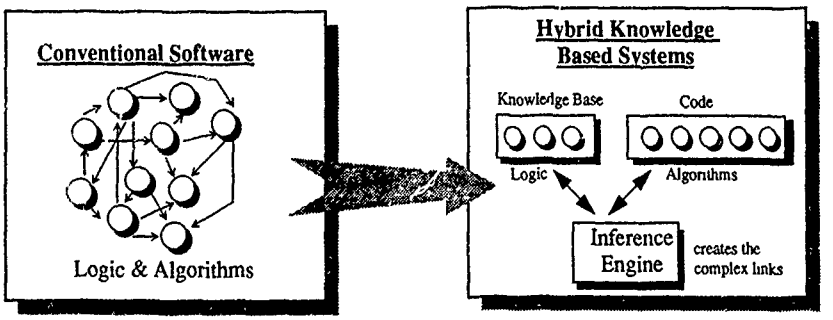
Programs that contain a lot of user defined logic or knowledge must be flexible and adaptable to constant change. This is because their problems have not been completely defined. They contain too many paths to enumerate and evaluate all of them. The large number of paths and our inability to effectively enumerate them is what creates the need to use AI techniques.

Conventional programming languages provide effective constructs for developing algorithms. They effectively handle sequential processing, iteration, recursion, function calling, and data abstraction. They contain only a limited capacity to handle complex logic definitions. Smart systems must process large amounts of logic to make the decisions necessary to drive the algorithmic processing. Logic that can not be broken down hierarchically generates very complex software. Usually numerous flags are created and added as the problem is refined, making the code more complex, incomprehensible, and unmaintainable.

AI knowledge based techniques were developed to allow problem domain knowledge to be acquired from experts and encoded for automated reasoning and decision making. Knowledge based technologies provide the framework for real-time embedded avionics processing. Knowledge based systems use special high order language constructs to define the logic. These constructs are called productions rules. They reduce software complexity by pulling the logic out of the code and putting it in the knowledge base.

Pulling this complex logic out of the code reduces the complexity of the code by eliminating the portion of the code that represents the complex logic. Complex logic coded in Ada and other conventional languages, is usually made up of numerous conditionals, flags, and switches, which are very difficult to understand and maintain. The rules in the knowledge base are user defined and are modular. They are automatically linked to each other and to the algorithmic code by the inference engine during program execution. This allows the development, visualization, and maintenance of the software at a higher level of abstraction, and leaves the actual creation and maintenance of the logical links to the inference engine. See figure 2.

Most AI systems do little algorithmic processing, they concentrate on knowledge processing. Embedded systems require hybrid conventional and AI solutions, where most of the complexity is contained in the reasoning and decision making logic, while the algorithms contain the bulk of the code, perform most of the processing functions, and expend most of the processing resources



- *Separates the logic from the Algorithmic Code.*
- *The Algorithmic Code becomes less complex, more modular, and discrete.*
- *Express the logic at a higher level of abstraction in the knowledge base.*

FIGURE 2: KNOWLEDGE BASED APPROACH

Effective smart avionics software must be developed by combining features of two separate technologies: (1) conventional software engineering techniques, which provide powerful languages, standards and practices; and (2) Artificial Intelligence (AI) techniques, which provide autonomous reasoning and decision making capabilities. These two technologies must be effectively combined to provide a "hybrid software engineering framework" that can support high quality, cost effective, and maintainable software for smart avionics systems. Much work has been done each of these areas, but little has been accomplished in efforts that combine all of them. To develop intelligent real-time embedded systems required us to pull together techniques from each of these areas and develop a methodology that facilitates efficient software development and supports our mission needs.

AN ARTIFICIAL INTELLIGENCE METHODOLOGY

The methodology selected to solve the mission management problem uses distributed, parallel, cooperating expert systems. These expert systems are functionally separate and communicate using message passing techniques. Each individual expert system requires

only basic expert system capabilities. All that is needed is an efficient and relatively simple production rule driven inference engines that support english like rule parsing, interface to a data base, and provide procedural and priority attachment facilities.

Declarative mission knowledge is contained in production rules. Standard high level programming language functions and procedures form the basis of the low level algorithmic code. These procedures will be invoked and controlled by the rules, providing a dynamic yet simple control structure.

The system is data or knowledge driven. The inference engine invokes the appropriate software functions as determined by the state of existing facts. Thus, the world state determines what software is invoked. External systems and sensors directly assert facts and the resultant system states drive the inference process. Inferencing ensures that appropriate mission activities are accomplished at the right time. The system continuously provides its best solutions to the existing mission management problems. This AI methodology allows the efficient combination of information from diverse sources.

The inferencing techniques operate on sets of production rules. Production rules are associated with individual system facts and algorithmic procedures. Inferencing for or about a fact results in the firing of the rules associated with that fact. Facts are organized in hierarchical, frame like, graph structures. Inferencing techniques control heuristic traversal of the structures. As a result of the inferencing, fact values are updated and procedures are invoked. Procedures can be called directly using procedural attachment to the rules, or indirectly as side effects of the inferencing. Execution and control of these procedures is the primary objective of the mission manager.

The system director is implemented as an enhanced expert system that functions similar to an AI blackboard controller. The system director controls a global data base or blackboard, which contains the mission plan and current aircraft and mission status. The system director controls the individual expert systems or knowledge sources directly by analyzing key blackboard entries. It invokes specific knowledge sources to schedule events, analyze the blackboard, or perform high level mission functions. The knowledge sources consist of expert systems, production rules or procedure calls to system functions.

The primary communication medium between the mission manager processes is the blackboard or global data base. All communication between the experts and algorithmic procedures is through this structure. There are three major data bases in the blackboard.

1) The Aircraft Status Data Base contains a hierarchy of facts that describe current aircraft, equipment, and weapon status It contains information on: switch positions, equipment modes, fuel status, and aircraft degradations.

2) The Mission Status Data Base contains a hierarchy of facts that describe current mission status. It contains information on current mission phase, mission plan, mission priorities, mission objectives, rules of engagement, and lists mission actions as complete or pending

3) The Mission Plan contains all knowledge required to carry out the mission. It describes mission phases, priorities, flight plan, sensor control planning, and all scheduled mission and aircrew tasks, switch actions, and procedures. It serves as the contract between the aircrew and the automated mission manager Knowledge of the current mission plan provides the aircrew with an understanding of mission manager plans and actions. Mission plan changes can only be made through the system director and with appropriate aircrew coordination.

AI ARCHITECTURE IMPLEMENTATION

The architecture implemented to solve the real-time inflight mission manager problem uses only basic knowledge based techniques distributed into an efficient cooperating parallel environment. This knowledge based software consists of three major elements. data base, rule base, and inference engine.

The Data Base

The data base or fact base contains all currently known facts. Facts are similar to programming language variables with additional attribute that they are implicitly linked together and to the rules via the knowledge base. Facts may be of a variety of types, such as: integers, reals, floats, enumerations, or strings. Facts are declared by stating a name and type. Multiple values for each fact may exist through multiple fact instantiations, and the facts may be grouped into composite structures (records).

The Rule Base

The production rules contain the logic and knowledge for the specific problem being solved. They define specific aspects of the problem and what to do under those circumstances. They are modular and discrete and provide a kind of relational approach to logic definition. The rules contain premises and conclusions. The premises define the context under which the conclusions should be executed. The premises contains facts and possibly constants, function calls, or algorithmic expressions. Rule conclusions contain commands to invoke rules or procedures or update fact values. Facts updated in the conclusion may trigger more rules to fire.

The knowledge base contains all of the systems declarative knowledge. Facts and rules are both declared in the knowledge base file. See figure 3.

The Inference Engine

The inference engine is the software that drives the automated reasoning process. The inference engine parses the knowledge base, links the facts to the rules, and performs automated run-time inferencing. It is the key element in the automated reasoning process. The inference engine must be efficient and should impose as little overhead as possible.

The knowledge base parser reads the fact definitions and interprets production rules. It creates efficient data structures that allow fast access to the rules and facts. The parser links the rules, facts, and procedures together by creating a list of rules for each fact. A fact's

```
FACTS
        counter_radar           enumerated_fact (true, false)
        current_radar           enumerated_fact (gun, SAM)
        radar_mode              enumerated_fact (tracking, locked_on, firing)
        aircraft_altitude       real_fact
        minimum_altitude        real_fact

RULES
    Rule_1
        IF      current_radar is SAM & radar_mode is tracking
        THEN    ASSERT counter_radar
    Rule_2
        IF      counter_radar & aircraft_altitude <= minimum_altitude
        THEN    UPDATE radar_countermeasure TO jam
    Rule_3
        IF      counter_radar & aircraft_altitude > minimum_altitude
        THEN    UPDATE aircraft_altitude TO minimum_altitude
```

- *A sample knowledge base illustrating chaining. When Rule_1 is fired, the fact counter_radar is updated. This triggers the premises of Rule_2 and Rule_3 to be tested. If the result of testing a rule's premise is true, then its conclusion is fired, causing other facts to be updated, and in turn other rule premises to be tested, and so on.*

FIGURE 3: SAMPLE KNOWLEDGE BASE

rule list is constructed by examining the premises of each rule. If a rule's premise references a fact, then that rule is placed on the fact's rule list.

The inference engine executes an iteratively recursive inferencing process. It performs rule testing and firing, and fact value updating.

When a fact value is updated all of the rules on its fact list are iteratively tested. Rule testing is the process of determining if the premise is true. It involves obtaining fact values, and resolving the premise expressions. If the premise is true then the rule is fired

Rule firing is the process of executing the rule's conclusion. If the process of executing a conclusion statement updates another fact value, then the process is interrupted. The inference engine recurses and starts the process over with the new fact, hence the designation "iterative recursion."

Code Generator

The knowledge base code generator encodes the interpreted rules and facts into procedural code. The resultant code allows the interpreted aspects of the inference process to be replaced by direct compiled code and access. This speeds up the inferencing process by several orders of magnitude. Each rule is turned into a procedure and the rule fact links are turned into an efficient lookup table.

Knowledge Driven Programming

The assertion of a new fact value automatically triggers the appropriate rules to be invoked. This is termed knowledge driven programming. Rule execution generates new facts, which cause more rules to be executed. This iterative process is forward chaining. See Figure 4.

In knowledge driven systems the assertion of facts automatically initiates the appropriate processes. External sensors and conditions can automatically focus and control system actions by their collective states. This provides an extremely efficient processing paradigm.

Knowledge driven programming is efficient because the logic in the knowledge base can be interpreted by the user and the computer. Changes in system states are noted by new values in the data base. Key states automatically trigger the appropriate code to be executed.

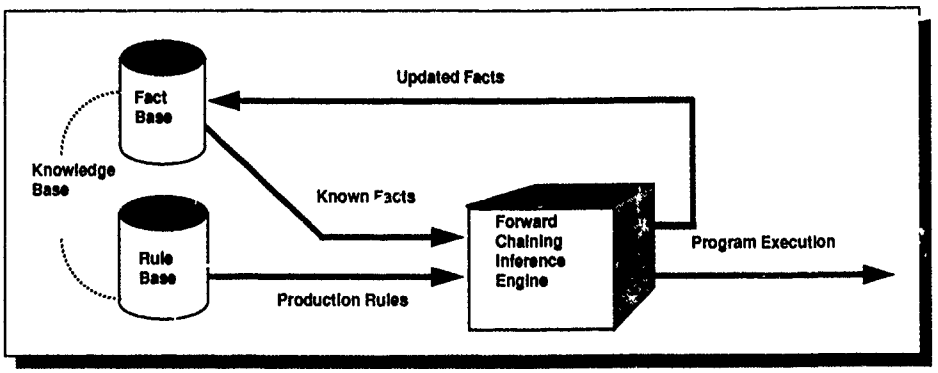
System Functions and Procedures

Standard programming functions and procedures make up the bulk of the programming code. These are modular and discrete, and are invoked and controlled by the rules. Their interactions are defined in the knowledge base. This provides a dynamic yet simple control structure that requires little control code to be developed. Very little time is spent developing complex control code, as the control is maintained in the knowledge base. These procedures perform a variety of tasks, but perform little reasoning and decision making.

Debugging Facilities

Smart embedded avionics systems are usually highly automated and do need little operator interface. However, debugging, testing, validation, and development requires an interactive environment. The interactive environment allows fact and rule inspection and logic tracing.

The operator is able to inspect or change fact values which allows detailed analysis of system states and provides user controlled state transitions. It is essential that on-line logic traces are available. Logic tracing is a simple process because the inference engine is



- *Fact Base contains state information of the system.*
- *Rule Base contains knowledge (or production rules) stored as* ***If-Then*** *statements defining fact relationships.*
- *Forward Chaining is the iterative process of inferencing using production rules.*
- *Inference Engine combines known facts and relationships to create new fact values and control program execution*

FIGURE 4: KNOWLEDGE BASE INFERENCE METHODOLOGY

the focus of all logic related activity. All facts are updated, and all rules are tested and fired by it. Tracing rule accesses and fact value updates are two essential areas of logic tracing.

Real-Time Performance

Real-time performance can be achieved by taking advantage of the constraints and restrictions inherent in embedded avionics systems. These include: limited operator interface, well defined system interfaces, simple data structures, well established requirements, and rigorously defined algorithmic software procedures. They provide bounds for the AI problem by limiting the scope of the run-time problem. It eliminates the need for symbolic processing, which requires costly run-time interpretation, examination, and decomposition. All variables or facts, which are not numeric, can be statically enumerated.

Other keys to real-time performance are: reducing overhead, providing good procedural attachment facilities, and increasing processor throughput.

The iteratively recursive paradigm is well suited for real- time performance. It allows the development of an inference engine that is very compact, and consumes little processing overhead, since it contains only several procedures and small data structures for each fact and rule. It is only invoked when facts are updated that may trigger rules to be fired.

There is no run-time overhead in determining which rules are candidates for firing or the order of firing. They are fired sequentially from the fact's rule list. Also, since the problem is decomposable by facts and rules it can be broken down and solved concurrently.

Cooperating Expert Systems

Concurrent distributed processing can be accomplished by breaking the knowledge driven system up into multiple cooperating expert systems. Each expert system operates on its own set of facts and rules. Cooperation and communication are accomplished via shared facts.

Defining expert systems interfaces is a very straight forward process. Each expert system has its own knowledge base, shared facts are declared in the fact definitions. When the value of a shared fact is changed by an expert system, a message is sent to the other expert systems with the new value.

When another expert system changes the value of a shared fact, a message is received by the expert stating the fact name and the new value. This message is stored in a queue and is not processed until the system finishes the current recursive processing. The messages are then processed in turn. The messages change fact values and may cause inferencing.

Messages from other experts are treated exactly like inputs from sensors, systems, or the operator. There is very little processing or size overhead in this method of distributed processing.

Ada REAL-TIME INFERENCE ENGINE (ARTIE)

To develop an mission manager to run on a parallel embedded environment and provide all of the required capabilities it was necessary to develop a real-time inference engine. Several generations have been developed, in various programming languages, using a variety of knowledge based techniques and features. The current inference engine includes all of the features outlined above. It is written in Ada and is called the Ada Real-Time Inference Engine or ARTIE. ARTIE is very fast, compact, and runs interactively or embedded, alone or with multiple cooperating expert systems

ARTIE consists of approximately 13,000 lines of Ada source code, and performs the following key functions. knowledge base parsing, rule-fact-code linking, on-line trace and debug facilities, on-line rule and fact modification, and runtime inferencing. ARTIE fires over 1400 rules per second on a typical work station in the interpretive mode, it is significantly faster using the compiled code genera-

tor. ARTIE has been successfully executed without modification on: Apollo 3000, 4000, and 590, Sun 2/60,3/60 and 4/60, Vax, Microvax, and Silicon Graphics IRIS workstations. ARTIE also runs on a parallel processor which consists of 10 embedded 68030 processors on a VME backplane.

ARTIE can function as a stand-alone system controlled by the operator through an interactive interface, or autonomously embedded in an Ada program controlled by external Ada code that is knowledge driven.

DEVELOPING INTELLIGENT SYSTEMS

Developing intelligent systems requires the use of specialized software development techniques. Knowledge based software development techniques can reduce the time and costs required to develop smart avionics software. In addition, it facilitates the creation of comprehensible, maintainable, and verifiable systems. Knowledge based software development is a methodology that uses Artificial Intelligence knowledge based techniques to provide: 1) less complex software; and 2) rapid development, coding, test, and verification of intelligent and autonomous software.

Software complexity is reduced by extracting embedded decision and control software from the complex code and putting it in a knowledge base. Decision and control software is the reasoning software that represents the logic that drives the system by linking together the algorithms that perform system tasks. This software is not represented efficiently in conventional high order languages. It will be placed in the knowledge base.

The knowledge base contains high level, user understandable, modular descriptions of the decision and control logic. These modular logic descriptions are the production rules. Removing the logic from the complex code leaves behind simple, discrete algorithms and procedures. The knowledge base is interpreted by the inference engine which creates the actual logic links.

Reduced software complexity makes it possible to create software that is comprehensible as well as verifiable and maintainable. The knowledge base provides a common language and interface between the customer, systems engineers, coders, testers, and maintainers. An interpreted inference engine with an automatic code generator allows efficient incremental or iterative software development.

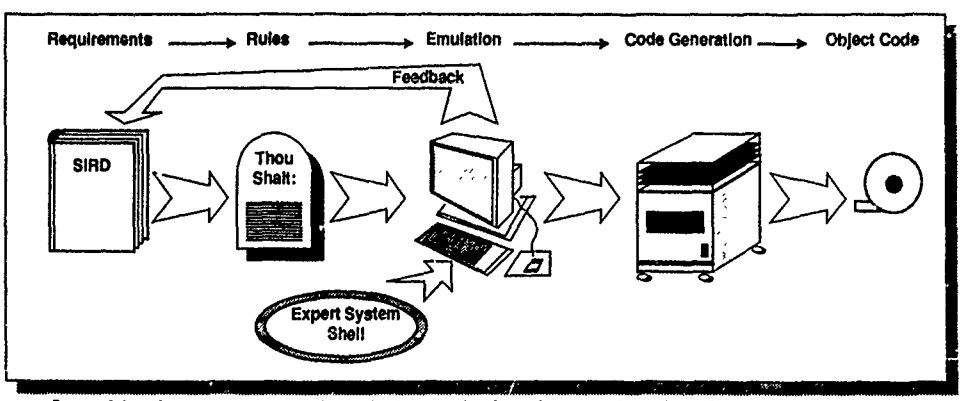
Rapid software development is accomplished by bringing the entire software team together and using the advanced features of the interpreted inference engine to develop, test, and verify the software. The automatic code generator is invoked to rapidly create the embedded high order language code.

The key to knowledge based software development is having a inference engine that: is very small, efficient, embeddable, performs real time inferencing, provides on- line reasoning and software execution trace and debug facilities, supports high level code generation for embedded operations, and is coded in a high order language using standard engineering practices and procedures.

With an understanding of the inference engine, experts can develop the rules to solve the problems. Computer programmers need only assist the experts and code the low level functions described by the experts as necessary to provide information or processing to be controlled by the rules.

The knowledge based software development process is streamlined to provide two major capabilities: rapid prototyping and efficient performance. To meet these to objectives the development process uses multiple levels of interpretation and symbolic processing. Software tools automate the transitions between these levels.

In the early stages of intelligent system development rapid prototyping and quick turn around are essential. This is true because development of intelligent systems requires an iterative approach. The systems are complex and logic errors will be found and must be eliminated. The rapid prototyping will be accomplished by using an interpreted inference engine and placing as much of the new code as possible in the knowledge bases. Modular knowledge bases allow incremental rule base development, similar to Ada's packages. The interpreted inference engine reduced the required number of recompilations and provides on- line debugging and modification. See figure 5.



- *Some of the advantages are: provides early viewing of code performance through emulation, semi-automatic object generation, and shorter development time.*

FIGURE 5: RULE BASED SOFTWARE DEVELOPMENT PROCESS

In later stages of development real-time performance and compact size become the priorities. At this stage, tested knowledge bases are compiled directly into procedural code. These rules are accessed directly like programming language procedures. Modification of these rules becomes more time consuming and complex but this is not a key factor at this stage of development.

Facilities are provided throughout the stages of software development to allow debug and logic tracing, even on the target processor. Facilities are provided to allow the use of multiple knowledge bases which may be at different levels of development, some may be interpreted, while others are already compiled as procedural code. The result is a very efficient process that allows rapid prototyping while preserving high performance.

MISSION MANAGER DEVELOPMENTS

Several Boeing research projects have contributed to our real-time mission management knowledge and experience. These include: (1) On Board Mission Management, a project that developed a prototype strategic inflight mission manager, (2) Advanced Tactical Cockpit, a project to develop and demonstrate a tactical mission manager in a domed simulator, (3) Knowledge Based Software Development, a project that developed a real-time Ada based inference engine, knowledge based development techniques, methods, practices, and tools, and (4) The Boeing Advanced Avionics Test Bed, a project that provides a Boeing aircraft for flight testing and evaluating advanced avionics systems.

On Board Mission Manager (IR&D BMA-031)

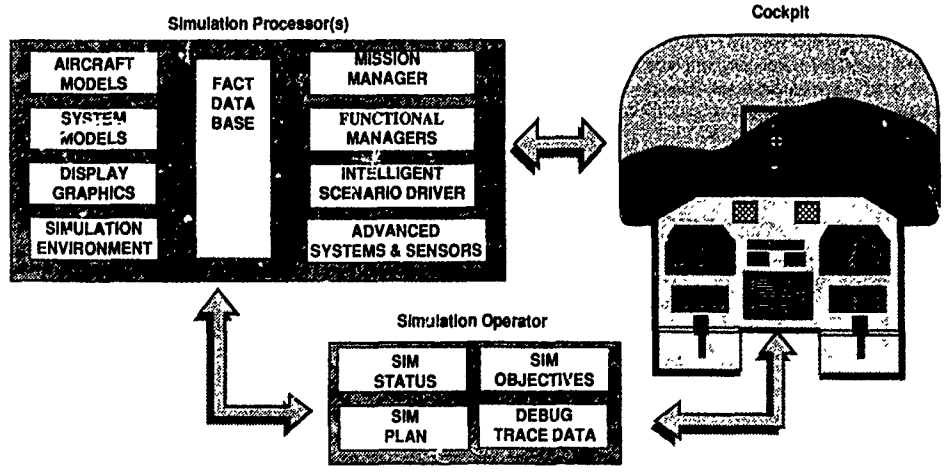
A strategic on board mission management system (OBMM) for bomber aircraft was developed at Boeing. The system provides inflight mission replanning, automated mission statusing and aircraft health monitoring, and evaluates multi-sortie nuclear deconflictions. It also models and assesses the threat environment and controls on board radio frequency sensors. Aircrew interface is via multiple color displays, a tracking handle, and programmable touch panels.

The OBMM system will aid the strategic aircrew in accomplishing its mission under the myriad of constraints and conditions that exist in the current and future strategic environment. It collects, synthesizes, analyzes, filters, and controls information gathered by modern aircraft sensors and avionics, and presents situations and status with alternative actions to the aircrew. OBMM provides adaptive sensor control, mission planning and replanning, aircrew interfaces, and aircraft status and monitoring as dictated by the mission.

The on board mission manager is controlled by distributed, parallel cooperating expert systems. OBMM experts and procedures communicate with each other through the blackboard, or global data bases. Global data bases contain the mission plan along with aircraft, weapon and mission status.

Tactical Mission Manager (IR&D BMA-897)

An advanced tactical fighter cockpit mission manager that manages air-to-ground tactical fighter and unmanned air vehicle missions was developed at Boeing. It performs inflight mission management, automated sensor control, threat recognition and countermeasure selection, inflight mission replanning, and advanced control and display processing. The system was developed in Ada and will on many platforms. The system was demonstrated in a distributed Silicon Graphics environment, linked into off-the-shelf simulation and graphics support tools. The system was flown man-in-the-loop and in a fully autonomous mode to represent an unmanned air vehicle. See figure 6.



- *This intelligent simulation architecture depicts major Tactical Mission Manager components and their interrelationships.*

FIGURE 6: TACTICAL MISSION MANAGER ARCHITECTURE

A key capability of the tactical mission manager is real- time threat avoidance. The system is designed to use knowledge of the low level mission environment to determine the best threat countermeasure. Threat countermeasures are not limited to electronic jamming or spoofing, but may include: aircraft maneuvers, altitude changes, expendable countermeasures, or mission replanning.

Mission planning and replanning includes weapon selection, mode control, and delivery tactics optimization. When the mission objectives, threat environment, or aircraft systems capabilities deviate from that required by the planned mission; the mission manager automatically replans the necessary mission segments. The replanning may require reconfigured weapons, a different delivery tactic, or an alternate flight profile. This capability provides real-time inflight tactics and mission management.

Advanced Avionics Testbed (IR&D BMA -032)

Key aspects of these prototype projects have been tested and flown on Boeing's Advanced Avionics Testbed Aircraft (AAT), a Boeing 720 aircraft that has been modified and instrumented to allow inflight evaluation and testing of advanced avionics systems. The AAT is dedicated to flight testing advanced offensive and defensive systems. Equipped with state-of-the-art sensors, instrumentation, and communications and navigation equipment the AAT has provided a facility for real world testing of inflight mission manager components. Real world conditions include: threat avoidance (via a Loral radar warning receiver); inflight mission replanning; threat recognition; simulated countermeasure selection; advanced controls and displays; sensor management (forward looking infrared radar, millimeter wave radar, laser line scanner, optical imagery); intelligent navigation (global positioning system, inertial navigation unit, digital map aided terrain following); advanced situation awareness (multiple color displays).

LESSONS LEARNED

Several lessons were learned in the process of developing real-time inflight mission managers and advanced avionics systems. Several key points follow.

Complex AI facilities are not required for state-of-the-art intelligent embedded systems. They add too much complexity, are too inefficient and never seem to quite fit. Facilities such as evidential reasoning, inheritance, and frame based a architectures were explored. These are useful features, but make the systems too complex and slow for most real-time systems.

Later versions of the real-time inference engine are as streamlined as possible A key requirement that evolved was to provide only the necessary AI facilities implemented with the least overhead possible.

Intelligent embedded systems can achieve real-time performance by using knowledge based AI techniques. They can perform the very intelligent and automated tasking required by modern combat systems. These systems can achieve very high performance levels using only basic knowledge based concepts. The inference engine must be extremely efficient.

By using knowledge based techniques the program logic is pulled out of the code and put in a knowledge base, where it is understandable, modular, and modifiable. This greatly reduces the complexity of logic intensive software and increases software quality

The inference engine provides an interactive user interface for debugging. Due to the complexity of the logic and the number of paths through the system, it is essential that the operator have complete visibility to system performance during all stages of system development. The inference engine also provides interactive debugging in the final embedded avionics environment.

REFERENCES

Expert Systems Aid On Board Mission Management, Defense Computing, January 1989 by G.F. Wilber.

Simulating Intelligent Missions, Defense Computing, November 1989, by G.F Wilber.

KNOWLEDGE ACQUISITION FOR EXPERT SYSTEMS USING STATISTICAL METHODS

Brenda L. Belkin* and Robert F. Stengel**

Princeton University
Department of Mechanical & Aerospace Engineering
Princeton, NJ 08544

ABSTRACT

A common problem in the design of expert systems is the definition of rules from data obtained in system operation or simulation. While it is relatively easy to collect data and to log the comments of human operators engaged in experiments, generalizing such information to a set of rules has not previously been a straightforward task. If an expert human operator is involved, he or she can be interviewed, in the hopes of identifying expenential knowledge concerning what does and does not work. However, this approach often is unsatisfactory, either because the expert has difficulty verbalizing techniques for performing the mission or because the mission is largely governed by quantitative issues not well analyzed by human operators. This paper presents a statistical method for generating rule bases from numerical data, motivated by an example based on aircraft navigation with multiple sensors. The specific objective is to design an expert system that selects a satisfactory suite of measurements from a dissimilar, redundant set, given an arbitrary navigation geometry and possible sensor failures.

This paper describes the systematic development of a Navigation Sensor Management (NSM) Expert System from Kalman Filter covariance data. The development method invokes two statistical techniques: *Analysis-of-Variance (ANOVA)* and the *ID3 algorithm*. The ANOVA technique indicates whether variations of problem parameters give *statistically* different covariance results, and the ID3 algorithm identifies the *relationships* between the problem parameters using probabilistic knowledge extracted from a simulation example set. ANOVA results show that statistically different position accuracies are obtained when different navigation aids are used, the number of navigation aids is changed, the trajectory is varied, or the performance history is altered. By indicating that these four factors significantly affect the decision metric, an appropriate parameter framework was designed, and a simulation example base was created. The example base contained over 900 training examples from nearly 300 simulations. The ID3 algorithm then was applied to the example base, yielding classification "rules" in the form of *decision trees*. The NSM expert system consists of 17 decision trees that predict the performance of a specified integrated navigation sensor configuration. The performance of these decision trees was assessed on two arbitrary trajectories, and the performance results are presented using a predictive metric. The test trajectories used to evaluate the system's performance show that the NSM Expert adapts to new situations and provides reasonable estimates of sensor configuration performance. The paper shows how the NSM Expert finds the best sensor navigation strategy from a pool of available sensors to provide the highest position accuracy.

NOMENCLATURE

e_i	Total number of examples with the i^{th} attribute value
e_T	Total number of examples in training set
$\hat{F}$	Computed F-ratio
H_A	Alternate hypothesis
H_0	Null hypothesis
(m)	m^{th} observation of the result
$p(c_j)$	Probability of occurrence of the j^{th} result value
p_{ij}	Probability of occurrence of j^{th} result with i^{th} attribute value
$Y_{i.}$	Sum of observed results using the i^{th} value
Y_{ij}	Observed result obtained when i^{th} and j^{th} values of Factors A and B are used
$\bar{Y}_{i.}$	Mean value of observed results using the i^{th} value of the factor under investigation
α_i	Effect of i^{th} value of Factor A on observed result Y_{ij}
β_j	Effect of j^{th} value of Factor B on observed result Y_{ij}
$(\alpha\beta)_{ij}$	Effect of interactions between i^{th} value of Factor A and j^{th} value of Factor B on the observed result Y_{ij}
ρ	Geodetic distance
θ	Bearing measurement
$\mu_{..}$	Mean of all observed results in data set being used in ANOVA experiment
ζ	Confidence level

* Formerly, Graduate Student, Department of Mechanical & Aerospace Engineering, Princeton University. Currently, Member of Technical Staff, AT&T Bell Laboratories, 480 Red Hill Road, Middletown, NJ, 07748
** Professor of Mechanical & Aerospace Engineering

INTRODUCTION

Knowledge acquisition is a major problem in the development of rule-based systems. The tools developed to date are not designed to extract information from data for which no generalizations are known *a priori*. Instead, these tools either rely on the expert to provide examples from which rules are generated or try to capture the expert's problem-solving methodology with interviewing techniques [1]. Unfortunately, it is often difficult for experts to describe their problem-solving methods or to detail the factors that come into play during the resolution of a problem. It is exactly this type of knowledge that is needed to design rule-based systems.

Since the early 1970's, adaptive navigation has been viewed as a highly desirable candidate for development in next-generation aircraft [2]. It is envisioned that future aircraft will have multi-sensor capability for navigation tasks requiring high reliability, optimal performance, and increased automation. With multi-sensor capability, the task of sensor configuration selection and management will become an additional pilot burden.

The performance of multi-sensor navigation systems (more commonly known as "integrated" or "hybrid" systems) has been explored since the late 1960's when results from modern control theory provided techniques for sensor mixing and optimal state estimation [3]. Hybrid systems refer to externally referenced navigation systems that "aid" an on-board inertial navigation system (INS) using an optimal state estimation mechanization. Hybrid systems combine the high- and low-frequency accuracy properties of INSs and external navigation aids (navaids) respectively. Many radio navigation and on-board systems aiding INSs have been modelled and their performance covariance results obtained [4-8]. When radio navigation systems are only partially operational, results show that navigation performance is superior to that of the pure INS [4]. Therefore it becomes advantageous to keep partially operational systems as candidates for integrated sensor mixing purposes.

With a large number of available navaids, choosing an optimal or near-optimal sensor set becomes a large combinatorial problem. Convergence towards an optimal sensor configuration requires an exhaustive computer search utilizing simulation results as the basis for selection. In contrast, a small number of available navaids reduces the decision space considerably. Hence, a dilemma occurs; increasing sensor capability (and thus reliability and performance) increases decision-making complexity.

The selection of an optimal configuration requires the application of some decision criteria. Most often, designers choose between navaids based on the relative accuracies of each system using a hierarchical approach [9]. This approach is "knowledge-based" in the sense that the nominal performance of each navaid is well-known and that this knowledge is built into the sensor hierarchy. The current hierarchical designs are not as "robust" with respect to sensor availability and performance changes as is necessary for future sensor management systems [10] Instead, these hierarchies represent "rules-of-thumb" that are useful in only the simplest cases. They do not resolve sensor configuration problems when more detailed information must be considered – for example when the number of each available navaid is specified, when partially operational systems remain viable candidates, and when trajectory effects degrade system performance. It becomes necessary to explore factors other than the performance of nominally operating navaids to determine how these factors affect the decision-making process, and to fully exploit the potential of hybrid systems.

The Analysis-of-Variance statistical technique (ANOVA) [11] was used to identify the factors that cause variation in navigation performance. Once the important factors were identified, the relationships between them were determined. The ID3 algorithm [12,13], an inductive inference technique based on the probabilistic occurrence of events, was used to find these attribute relationships.

The development of a navigation sensor management expert system using the ANOVA/ID3 technique [14,15] is described in this paper. The NSM system controls the selection of multi-sensor configurations. The methodology is applicable to any problem where the development of knowledge bases from multi-factor data studies is desired.

INTEGRATED NAVIGATION SYSTEMS

Optimal estimation techniques are used to combine inertial and radio navigational systems in order to provide stable continuous inertial navigation information [16]. The errors exhibited by these "hybrid" systems depend on the accuracy of the aiding system, and navaid accuracies are functions of many factors, such as navaid type, number of similar navaids, distance from the navaid, and whether the aircraft is approaching or receding from the station. The sensor selection criteria depend on the relative importance of these factors. Five external radio navigation and two on-board navaids were used to update a medium-accuracy (10 Nautical Mi./hr) INS. Hybrid system performance was simulated using the linearized inertial navigation error model and navaid measurement models as inputs into the optimal estimation filter. The hybrid errors were updated at a specified navaid fix rate. The systems simulated were (1) Global Positioning System (GPS), (2) Long-Range Navigation System (LORAN), (3) Tactical Navigation System (TACAN), (4) Distance Measuring Equipment (DME), (5) VHF Omnidirectional Range (VOR), (6) Doppler radar, and (7) Air Data sensor. The operational theory and the mathematical models used to simulate the navaids and the inertial navigation error model are discussed in detail elsewhere [14].

The numerically-stable discrete-time U-D implementation of the Kalman Filter equations [17] was used to mix the inertial system and navaid information optimally, providing covariance estimates of the navigation errors (e.g., north/east position) [14,18]. Each nonlinear measurement equation was linearized with respect to the inertial navigation states to obtain the observation matrix used in the U-D measurement update equations. Since sensor errors were taken into consideration in the measurement models, the inertial error state vector was augmented with the sensor shaping filter dynamics (e.g., random bias, first-order Markov model) to formulate the hybrid navigation model. Additionally, the measurement noise time history was simulated. As the aircraft moves along its trajectory relative to ground-based navaid stations, the measurement noise characteristics change. Therefore an equation for a distance- or time-varying measurement covariance matrix was found in order to realistically model ground-based radio navigation systems. According to Ref. 18, GPS measurement noise increases in a similar way; as the satellite descends near the aircraft's horizon, the noise increases. To simulate time-varying measurement noise for the ground- and satellite-based navigation systems, each noise variance was modelled as the sum of initial and range-dependent variances. The latter component increases linearly with the square of the distance from the station or satellite.

Position accuracy was selected for the rule-based system decision metric. Here, position accuracy is defined as the root sum of squares (RSS) of the north and east component errors. The RSS decision metric provides sufficiently consistent quantities to compare hybrid performances. For a detailed discussion of the RSS decision metric, the reader is directed to Ref. 14.

HYBRID NAVIGATION SIMULATION RESULTS

Using the RSS position error metric to measure hybrid system performance, the following U-D filter simulations were performed:

1. Single-type hybrids: GPS, LORAN, TACAN, DME, VOR, Doppler Radar, or Air Data sensor aiding an INS
2. Number of stations used in a single-type hybrid
3. Multi-type hybrids: Combinations of different navaid types aiding an INS
4. Aircraft trajectories simulated: High-performance, commercial, general aviation

Comparisons of Single-Type Hybrid Performance

Consider the four ground stations A, B, C, and D spatially oriented with respect to the high-performance, commercial, and general aviation trajectories in Fig. 1. The four ground stations are simulated as LORAN slaves, TACAN, DME, or VOR stations. Figure 2 shows the performance differences of ground-based, GPS, and on-board type hybrid systems. When the results from all ground station A types (LORAN, TACAN, DME, VOR) are compared on the high-performance trajectory, the relative performance from best to worst may be listed as follows: (1) LORAN, (2) TACAN, (3) DME, and (4) VOR. For example, a hybrid system utilizing LORAN Slave Station A provides better performance than a hybrid system utilizing TACAN A; a TACAN A hybrid in turn outperforms a DME A hybrid which in turn outperforms a VOR A hybrid. This pattern is repeated for stations B, C, and D [14]. The best hybrid performance was obtained from three GPS satellites aiding the INS. Figure 2 also shows how the performances of the Doppler radar hybrid and the air data sensor hybrid compare with the GPS and ground-based navaid hybrids.

Referring to the LORAN results in Fig. 3, there is a striking variation in the performance of the individual Stations A-D; this figure reveals that single stations of the same type aiding an INS give highly variable performance results. The same variability in performance of the remaining ground-based single-station navaids was found [14]. From Fig. 3, the variation in Station A-D's performances is attributed to the position of each ground station relative to the aircraft's trajectory. For example, LORAN Slave A gives the smallest position error of the four stations; referring to Fig. 1, the aircraft makes a close approach to Slave A on the trajectory's second leg. Hence the RSS error becomes very small. These errors begin to increase towards the end of the trajectory leg, due to the increasingly uncertain north component. In contrast, LORAN Slaves B, C, and D are farther from the aircraft's trajectory. The first trajectory leg results in good relative north information to B, C, and D, whereas the east component uncertainty grows due to the lack of relative east information. The variations in performance observed from Stations A-D are due to trajectory effects; using Station B instead of A to update the INS is equivalent to using A and changing the aircraft's trajectory.

Effect of Increasing the Number of Navaids in a Hybrid System

Next, the effect of the number of ground stations was studied by simulating all possible combinations of single-, double-, and triple-station hybrids formed from Stations A-D. There are six possible combinations of two stations and four combinations of three stations that may be integrated to aid the INS. These simulations were carried out for LORAN, TACAN, DME and VOR.

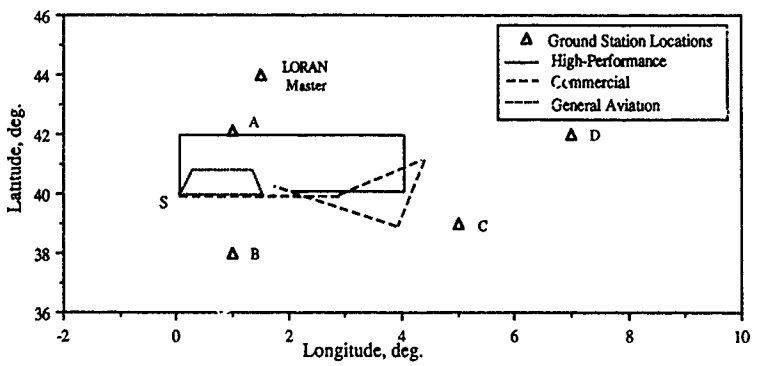


Figure 1. Aircraft Trajectories Used in Simulations

Referring to the LORAN results in Fig. 4, the performance variation among the double-station combinations and triple station combinations is less pronounced than the single-station variations. The magnitude of the RSS errors decreases dramatically when two stations are used instead of one station. The RSS errors decrease further when three stations are used, although the magnitude differences are not as great. The RSS magnitudes of the double- and triple-station combinations are much lower because the aircraft receives more complete navigation information as the number of stations increases. This also explains why there is much more variation in the results for the double station combinations than for the triple stations. Similar performance trends were observed for GPS, TACAN, DME, and VOR [14].

Effect of Trajectory on Hybrid Performance

It already has been shown that an aircraft's trajectory relative to a single ground station hybrid plays an important role in the estimator's performance. The RSS results in Fig. 5 illustrate the performance differences of the LORAN Slave A hybrid on the high-performance, commercial transport, and general aviation trajectories. Two parameters that contribute to these performance differences are distance to a station and heading with respect to a station. A third trajectory parameter that contributes to a hybrid

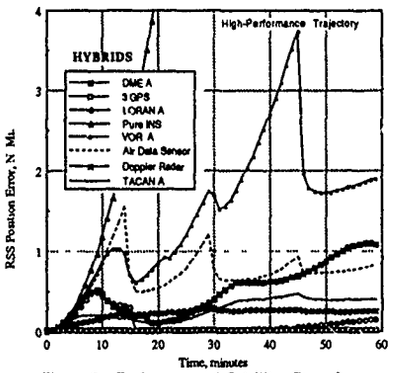


Figure 2. Performance of Satellite, Ground Station-Based and On-Board Hybrid Navigation Systems

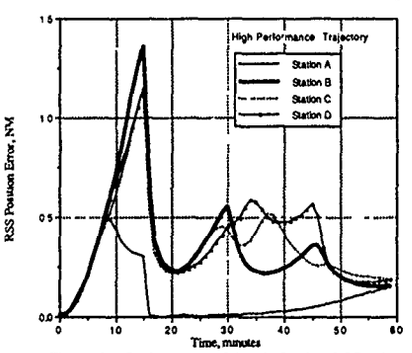


Figure 3. Performance of Single-Station LORAN Navaids Aiding an INS

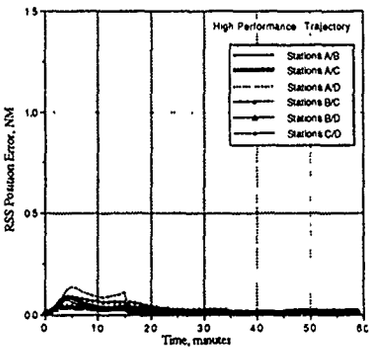


Figure 4. Performance of Double-Station LORAN Navaids Aiding an INS

system's performance is the number of heading changes along the trajectory. The effect of heading changes is discussed in more detail in [14]. Trajectory factors affect the INS dynamics, which in turn affects the error estimation performance. The trajectory factors also change the measurement dynamics since the measurements depend on the trajectory's geometric properties and aircraft states (such as velocity). The results in Fig. 5 clearly show that when the trajectory changes, the navaid selection decision most likely changes as well, since the relative accuracies of the navaids change.

Hybrid Performance of Mixed Navaids

Figure 6 shows various combinations of integrated navaids. The individual performances of LORAN Slave B, Doppler radar, and Air Data hybrids are shown in Fig. 6 along the high-performance trajectory. The LORAN/Doppler and LORAN/Air Data hybrids also are shown for comparison. Both combinations gave better results than their individual components operating alone. For example, the LORAN/Doppler combination outperformed the LORAN hybrid and the Doppler hybrid; similarly, the LORAN/Air Data combination performed better than either the LORAN or the Air Data sensor hybrids alone. The latter combination did slightly better than Doppler hybrid on this trajectory after the initial transient period. These results show that good navigation performance is still possible when a "failed" LORAN system (only one slave station operational) is integrated with an on-board navaid such as Doppler radar or a standard equipment Air Data sensor.

Effect of Navigation Sensor Reconfiguration on Hybrid Performance

This section shows how reconfiguration can be used to improve hybrid filter performance, resulting in an optimal navigation strategy. Here, the term "reconfiguration" is exemplified as follows: If navaids "A" and "B" give the best filter performances on trajectory legs "1" and "2" respectively, then the filter can be reconfigured (changed) to use navaid B rather than A on the second trajectory leg. The results in the previous sections demonstrated that the integration of several systems gives greatly improved performance over the use of each system by itself. However a dilemma arises if computational resources on board the aircraft prevent the filter from using more than one ground station for real-time measurement processing. This situation can occur if part of a distributed computer architecture or parallel processing system fails. A navigation expert system should have the capability of recommending a navigation strategy in this situation in addition to making recommendations when the navigation computational resources are fully operational.

Figure 7 shows the performance obtained when the best TACAN station information is used on each leg of the high-performance trajectory of Fig. 1. In the figure, TACAN station A provides more accurate position information than TACAN station C on the first two trajectory legs (from 0 to 35 minutes). On the last two legs, TACAN station C provides better information to the filter than TACAN station A. When TACAN station A is reconfigured to TACAN station C, the reconfigured filter provides better position estimates than does a TACAN C filter operating on the last two trajectory legs. This is because the good measurement information from TACAN A is propagated in time.

DEVELOPMENT OF A NAVIGATION SENSOR MANAGEMENT EXPERT SYSTEM

This section describes a novel methodology that uses established statistical techniques to develop the NSM expert from the simulation data. The primary function of this expert system is to select the external navaid sensors that provide the smallest possible RSS position error from a large set of available sensors. The Analysis of Variance (ANOVA) technique [11] is used to identify the

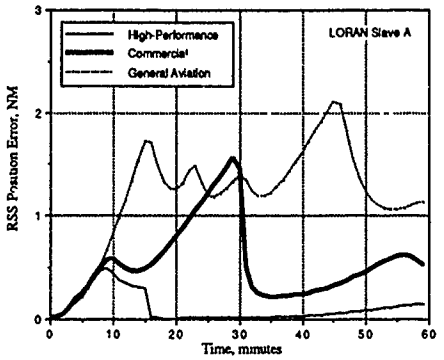


Figure 5. Comparison of RSS Results for LORAN Hybrids on Three Different Trajectories

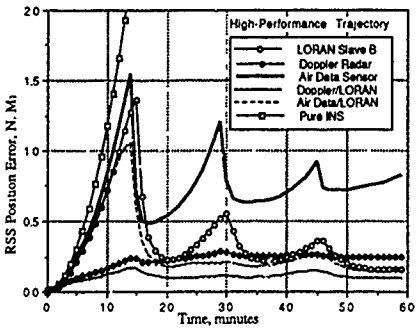


Figure 6. Comparison of RSS Results with LORAN, Doppler Radar, and Air Data Sensor Hybrid Combinations

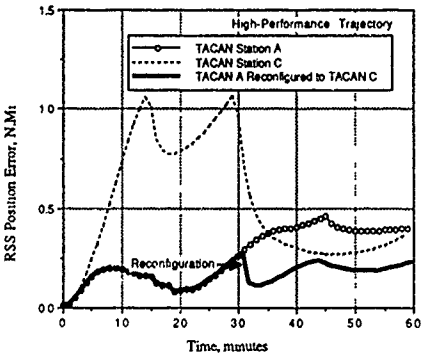


Figure 7. Performance of a Reconfigured TACAN Hybrid System

factors that make statistically significant contributions to the decision metric. Then, the ID3 algorithm determines the relationships between these factors [11,13]. The mathematical formulations of the ANOVA and ID3 techniques are briefly reviewed, followed by a discussion on how these techniques were applied to the covariance data.

Mathematical Formulation of ANOVA

Consider that two factors, A and B, are to be studied concurrently in an ANOVA "experiment". The ANOVA experiment identifies the two factors' effects on an observed result. The mathematical model describing the observed result obtained with different values of Factors A and B is given by:

$$Y_{ij(m)} = \mu_{..} + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \varepsilon_{ij(m)} \qquad (1)$$

$Y_{ij(m)}$ is the observed result corresponding to the i^{th} and j^{th} values of Factors A and B, respectively. On the right-hand side, μ is the mean of all observed results in the data set, α_i and β_j denote the individual A and B factor value effects, $(\alpha\beta)_{ij}$ represents the effect of the A-B interaction, and $\varepsilon_{ij(m)}$ is a Gaussian random error that represents an unaccountable contribution to Y_{ij}. The subscript (m) denotes the m^{th} observation of Y_{ij}.

An estimator for each term in Eq. (1) is found in terms of the observed results using the method of least squares. The random error $\varepsilon_{ij(m)}$ is minimized and the minimum variance estimators are computed from [11,14]:

$$\hat{\alpha}_i = \bar{Y}_{i.} - \mu_{..} \qquad \hat{\beta}_j = \bar{Y}_{.j} - \mu_{..} \qquad \widehat{\alpha\beta}_{ij} = \bar{Y}_{ij} - \bar{Y}_{i.} - \bar{Y}_{.j} + \mu \qquad \hat{\varepsilon}_{ij} = Y_{ij} - \bar{Y}_{ij} \qquad (2)$$

where $\hat{\alpha}_i$ is the i^{th} least squares estimator of α_i, $\bar{Y}_{i.}$ is the mean value of the observed results having the i^{th} value of Factor A, $\bar{Y}_{ij}$ is the mean value of the observed result (if "m" observations are available) having the i^{th} and j^{th} values of Factors A and B respectively. The dot notation in Eq. (2) is defined as follows:

$$Y_{i.} = \sum_{j=1}^{b} Y_{ij} \qquad \bar{Y}_{i.} = \frac{1}{b}\sum_{j=1}^{b} Y_{ij} \qquad \mu_{..} = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b} Y_{ij} \qquad (3)$$

In Eq. (3), Y_{ij} is summed over "a" values of Factor A and "b" values of Factor B. For example, in the navigation sensor selection problem:

Factor A, Navaid Type={TACAN, VOR, DME, LORAN}, a=4 (Model 1)
Factor B, Number of Ground Statiions={One, Two, Three}, b=3

Substituting the least squares estimator equations into the model in Eq. (1), the deviation of an observed result from the overall mean is given by:

$$\underbrace{[Y_{ij} - \mu_{..}]}_{\text{Deviation from overall mean}} = \underbrace{[\overline{Y}_{i.} - \mu_{..}]}_{\text{Factor A Contribution to total deviation}} + \underbrace{[\overline{Y}_{.j} - \mu_{..}]}_{\text{Factor B Contribution to total deviation}} + \underbrace{[\overline{Y}_{ij} - \overline{Y}_{i.} - \overline{Y}_{.j} + \mu_{..}]}_{\text{A-B Interaction Contribution to total deviation}} + \underbrace{[Y_{ij} - \overline{Y}_{ij}]}_{\text{Residual Contribution to total deviation}} \quad (4)$$

If the sums of squares of each component in Eq. (4) are computed, the total variation of all the observed results is expressed in terms of the individual factor, interaction, and residual variations:

$$SS_{TOTAL} = SS_A + SS_B + SS_{AB} + SS_E \quad (5)$$

where SS refers to the sum of squares. SS_{TOTAL} is the total variation in the observed result, SS_A and SS_B are the contributing variations in the observed results when values of Factors A and B are compared, and SS_{AB} is the contributing variation in the observed results due to A-B interactions The residual, SS_E, is the unexplained (random) variability in the result when observations with different factor values are made. In the NSM example in Model 1, SS_{TOTAL} is the total variation in the RSS position error based on 12 simulations averaged over the entire flight time. SS_A is the contributing variation in the RSS position error due to using different navaid types, SS_B is the contributing variation in the RSS position error due to using different numbers of ground stations, and SS_{AB} is the contributing variation in the RSS position error due to interaction effects. Finally, SS_E is the variation in the RSS position error that cannot be attributed to navaid type, the number of stations, or interactions between these two factors.

Equation 5 is the foundation of the ANOVA technique When each sum-of-squares component is computed, many conclusions concerning the factor effects can be made. For example, if $SS_A >> SS_B$ then Factor A has a greater effect on the observed result than does Factor B. The F-test determines whether Factor A's apparent effect on the observed result is real or can be attributed to the residual variation, SS_E. Defining the F-ratio for Factor A, $\hat{F}_\alpha$, assuming more than one observed result ($m > 1$):

$$\hat{F}_\alpha = \frac{SS_A/(a-1)}{SS_E/(ab(m-1))} \quad (6)$$

The sum of squares for each source of deviation is normalized by the *degrees of freedom* (DOF), which is simply one less than the number of factor values. Since Factor A in Model 1 has four levels ($a = 4$), then the DOF_A is 3, and the *Mean Squared Error* of A is:

$$MSA = \frac{SS_A}{3} \quad (7)$$

The F-test enables a choice between two hypotheses to made made Stated mathematically for the NSM problem, the hypotheses are:

H_o: The RSS errors are statistically equal for different values of the factor under consideration

H_A: The RSS errors are statistically unequal for different values of the factor under consideration.

H_o and H_A are called the null and alternate hypotheses respectively. The null hypothesis suggests that there are statistically insignificant differences in the observed results for the factor values under consideration. The alternate hypothesis suggests the opposite case is true. When the null hypothesis holds, the $\hat{F}$-ratio follows the F-distribution [11]; when the alternate hypothesis holds, the $\hat{F}$-ratio follows a complex non-central F-distribution. The computed $\hat{F}$-ratio is compared to tabulated values that indicate how the ratio is distributed, allowing a hypothesis to be selected. Small values of $\hat{F}$ support H_o whereas large values support H_A. Using Factor A as an example, the decision rules needed to choose the hypothesis for a specified confidence level, ζ, are:

$$\text{If } \hat{F}_\alpha \le F_{TABLE}(\zeta; DOF_A; DOF_E), \text{ then conclude } H_o \quad \text{(Model 2a)}$$

$$\text{If } \hat{F}_\alpha > F_{TABLE}(\zeta; DOF_A; DOF_E), \text{ then conclude } H_A \quad \text{(Model 2b)}$$

ζ identifies the "risk" the designer is willing to take that the F-test will reject the null hypothesis when in fact the null hypothesis is valid. If H_o holds, the effect of Factor A on the result is *statistically insignificant*, whereas when H_A holds, the effect of Factor A on the result is *statistically significant*. Table I summarizes the computed variables needed to analyze a two-factor ANOVA experiment.

When two observed results are compared, the statistical difference between the results is determined using the Scheffé method. For example, in a navaid ANOVA experiment, multiple comparisons of radio navaid hybrids lead to a ranking of these systems from best to worst. Two results are statistically unequal if their computed difference exceeds the *Scheffé critical difference*, given by:

$$D_{CRITICAL} = \left[MSE\left(\frac{1}{n_1} + \frac{1}{n_2}\right) \times DOF_{FACTOR} \times F_{TABLE}(\zeta; DOF_{FACTOR}; DOF_E)\right]^{1/2} \quad (8)$$

where n_1 and n_2 are the sample numbers of the two results being compared. Although the discussion above used a two-factor ANOVA experiment as an example, the theory is extendable to experiments for which any number of factors are to be investigated

Table I ANOVA Table for a Two-Factor Experiment

Source of Variation	*Sum of Squares*	*Degrees of Freedom*	*Mean Squared Value*	$\hat{F}$	F_{TABLE}	*Confidence Coefficient*
Factor A	SS_A	a-1	$SS_A/(a-1)$	MSA/MSE	$F_{TABLE}(\zeta;a-1;ab(m-1))$	$(1-\zeta)\times 100\%$
Factor B	SS_B	b-1	$SS_B/(b-1)$	MSB/MSE	$F_{TABLE}(\zeta;b-1;ab(m-1))$	$(1-\zeta)\times 100\%$
A-B Interaction	SS_{AB}	(a-1)(b-1)	$SS_{AB}/((a-1)(b-1))$	MSAB/MSE	$F_{TABLE}(\zeta;(a-1)(b-1);ab(m-1))$	$(1-\zeta)\times 100\%$
Error	SS_E	ab(m-1)	$SS_E/(ab(m-1))$			
Total	SS_{TOTAL}	abm-1				

Identifying Important Factors Using ANOVA

The RSS position error histories from over 200 covariance simulations were obtained, and the data were used in an ANOVA four-factor navaid experiment. The goal of the experiment was to identify which of the factors (navaid type, number of ground stations, trajectory effects, performance history) and their interactions had statistically significant impacts on the RSS position error. The factor values used in the ANOVA experiment were:

Navaids={VOR, DME, LORAN, TACAN, GPS} (Model 3)
Number of Ground Stations={One, Two, Three}
Trajectory Type={High-Performance, Commercial Transport, General Aviation, from Fig. 1}
Performance History = {Intervals: I, II, III, IV}

Since each trajectory consists of four, 15-minute legs, the Performance History (or "time interval") factor refers to the RSS performance obtained within each 15-minute time frame. Four single-station, six double-station, and four triple-station hybrids were simulated using combinations of Stations A-D in Fig. 1.

The ANOVA summary in Table II shows that three of the four factors, navaid type, number of aiding ground stations, and performance history, are "strongly significant" with 99% confidence. Scheffe multiple comparison tests were applied to the navaid

Table II Analysis of Variance Summary and F-test Results for Four-Four Navaid Experiment

SOURCE OF VARIATION	SUM OF SQUARES (Sq. N.Mi.)	DEGREES OF FREEDOM	MEAN SQUARE VALUE (Sq N.Mi.)	COMPUTED F-RATIO	TABULATED F-RATIO	CONFIDENCE COEFFICIENT
MAIN EFFECTS						
Between Navaids	4 815E+02	4	1 204E+02	9 909E+02	3.320E+00	99%
Between Number	2.423E+02	2	1.211E+02	9,973E+02	4.610E+00	99%
Between Trajectory	6.234E-01	2	3 117E-01	2.566E+00	2.300E+00	90%
Between Time Intervals	5.729E+01	3	1.910E+01	1.572E+02	3.780E+00	99%

and number-of-ground-station factors to identify the specific differences within each groups; for example, the RSS performance difference between GPS and TACAN, all other factors being equal, was statistically significant. Referring to Table III, GPS provides the best RSS position performance when compared to TACAN, LORAN, DME, and VOR; this is consistent with the results in Fig. 2. On the other hand, the RSS performance difference between LORAN and TACAN with all other factors being equal, was not statistically significant. This means that a LORAN hybrid could perform better or worse than a TACAN hybrid, depending on the values of the other factors (e.g., number of ground stations). As expected, the comparison test concluded that VOR and DME give higher position errors than either TACAN or LORAN, and that DME performs better than VOR. The multiple comparison test results in Table III yielded the same performance ranking depicted in the graphical results (e.g., Fig. 2), while utilizing the information content of a large number of independent simulations.

Further investigation into the ANOVA interaction effects revealed that the ranking should be cautiously applied to single-station hybrids, since these are highly-sensitive to trajectory effects. As seen from the magnitudes of the computed differences in Table III, double-station hybrids provide much smaller RSS position errors than single-station hybrids. The performance difference between

Table III Scheffé Multiple Comparisons of Simulation Parameters for a Four-Factor Navaid Experiment

NAVIGATION PARAMETER	MEAN (N. MI)	COMPUTED DIFFERENCE (N. MI)	CONCLUSIONS
NAVAIDS [Scheffé Critical Difference (99%) = 7.484E-02 N. MI.]			
Compare GPS with	1.483E-02		
TACAN††	2.070E-01	1.922E-01	TACAN worse than GPS
LORAN	2.408E-01	2.260E-01	LORAN worse than GPS
DME	5.483E-01	5.334E-01	DME worse than GPS
VOR	1.181E+00	1.166E+00	VOR worse than GPS
Compare TACAN with	2.070E-01		
LORAN	2.408E-01	3.380E-02	LORAN no better/worse than TACAN
DME	5.483E-01	3.412E-01	DME worse than TACAN
VOR	1.181E+00	9.742E-01	VOR worse than TACAN
Compare LORAN with	2.408E-01		
DME	5.483E-01	3.074E-01	DME worse than LORAN
VOR	1.181E+00	9.404E-01	VOR worse than LORAN
Compare DME with	5.483E-01		
VOR	1.181E+00	6.330E-01	VOR worse than DME
NUMBER [Scheffé Critical Difference (99%) = 4.830E-02 N. MI.]			
Compare "3" with	2.062E-01		
"2"	2.617E-01	5.554E-02	"2" worse than "3"
"1"	8.474E-01	6.412E-01	"1" worse than "3"
Compare "2" with	2.617E-01		
"1"	8.474E-01	5.856E-01	"1" worse than "2"
TRAJECTORY [Scheffé Critical Difference (90%) = 3.412E-02 N. MI.]			
Compare "GA" with	4.186E-01		
"Jet Trans."	4.429E-01	2.431E-02	"Jet Trans." no better/worse than "GA"
"High P"	4.538E-01	3.519E-02	"High-P" worse than "GA"
Compare "Jet Trans." with	4.429E-01		
"High-P"	4.538E-01	1.088E-02	"High-P" no better/worse than "Jet Trans."
TIME INTERVAL [Scheffé Critical Difference (99%) = 6.186E-02 N. MI.]			
Compare "I" with	2.091E-01		
"IV"	4.405E-01	2.314E-01	"IV" worse than "I"
"II"	5.284E-01	3.193E-01	"II" worse than "I"
"III"	5.758E-01	3.667E-01	"III" worse than "I"
Compare "IV" with	4.405E-01		
"II"	5.284E-01	8.791E-02	"II" worse than "IV"
"III"	5.758E-01	1.353E-01	"III" worse than "IV"
Compare "II" with	5.284E-01		
"III"	5.758E-01	4.737E-02	"III" no better/worse than "II"

†† Interpretation. A TACAN hybrid gives higher (worse) position errors than a GPS hybrid

two and three stations is not as dramatic but is nonetheless statistically significant. One way to further investigate the sensitivity of single-station hybrids to trajectory is to use multiple Scheffé comparisons, presented in Table IV. The LORAN and DME results clearly show that aircraft trajectory significantly affects RSS performance when single-station INS hybrids are used; however, the results show that RSS performance using double- and triple-station INS hybrids is less sensitive to the aircraft's trajectory. This result verifies the RSS performance graphically observed in Fig. 5. Table IV also provides interesting insight into the behavior of the TACAN and VOR systems. TACAN is less sensitive to trajectory effects than LORAN, DME, and VOR for single-station hybrids because TACAN provides distance (ρ) and bearing (θ) measurements (i.e., ρ–θ navigation) from the aircraft to the station, enabling a position fix to be made. In contrast, the single-station LORAN, DME, and VOR hybrids each use only one measurement to recalibrate the INS so that a position fix is not made. However, when two or three stations of LORAN or DME are used, a position fix is provided (ρ-ρ navigation), and the RSS performance becomes less sensitive to trajectory effects. This explains the "damped" response in navigation error shown in Fig. 4.

The ANOVA summary in Table II shows that the factor "trajectory" is weakly significant (90% confidence). Indeed the term "trajectory" is extremely vague. The significant differences in single-station hybrid performance shown in Table IV and Fig. 3 suggest that trajectory should be decomposed into attributes that describe, in better detail, what these effects really are. The differences between the high-performance, commercial, and general aviation trajectories (Fig. 1) are: distance from a station, airspeed, and heading with respect to the ground stations. These factors play an important role in affecting navigation performance. In summary, the ANOVA and Scheffé methods systematically identified trends in the simulation data without recourse to tedious graphical analysis.

Extracting Rules Using Induction: The ID3 Algorithm

The ID3 Algorithm uses inductive inference to extract rules [13] from an example set. The problem space is described in terms of attributes, where each attribute is characterized by a set of values that define the possible "states." For example, in Model 3, the navaid type and number of ground stations were shown to be attributes affecting RSS position error. The attribute values for the factor "navaid type" were {GPS, LORAN, TACAN, DME, VOR}, and the attribute values for the factor "number of stations" were {One, Two, Three}. The ID3 algorithm defines the shape of the decision tree as it identifies the most important attribute at each decision node. The algorithm uses an Information-Theoretic Measure (ITM) that minimizes the number of tests (attribute nodes) necessary to define the decision tree. The information contained in an attribute is quantified using the attribute's *entropy content*, found in the example set. This measure is the information gained by testing the attribute at a given decision node; it is defined as the difference between the information content within the complete example set and the information content within the attribute:

$$ITM_{attribute} = -\sum_{j=1}^{M} p(c_j) \ln p(c_j) + \sum_{i=1}^{N} \frac{c_i}{c_T} \sum_{j=1}^{M} p_{ij} \ln p_{ij} \qquad (9)$$

where $p(c_j)$ is the probability of occurrence of the j^{th} result in the total example set; p_{ij} is the probability of the i^{th} attribute

Table IV Scheffé Comparisons of Trajectory Effects on Single-, Double- and Triple-Station Hybrid INSs

KEY
(+) Difference is Statistically Significant
(-) Difference is Statistically Insignificant

Scheffé Critical Difference (95%) = 0.22 N Mi				
Mean Difference N. Mi. \ Navaids	LORAN	TACAN	DME	VOR
Single-Station Hybrids				
Between Jet Transport and GA	8 57 (+)	0 08 (-)	0 91 (+)	1 0 (+)
Between Jet Transport and High-P	0 22 (+)	0 07 (-)	0 22 (+)	0 1 (-)
Between GA and High-P	0 78 (+)	0 15 (+)	0 70 (+)	1.1 (+)
Double-Station Hybrids				
Between Jet Transport and GA	0 026 (-)	0 003 (-)	0 04 (-)	0 56 (+)
Between Jet Transport and High-P	0 008 (-)	0 000 (-)	0 02 (-)	0 00 (-)
Between GA and High-P	0 34 (-)	0 003 (-)	0 02 (-)	0 56 (+)
Triple-Station Hybrids				
Between Jet Transport and GA	0 009 (-)	0 001 (-)	0 01 (-)	0 36 (+)
Between Jet Transport and High-P	0 000 (-)	0 003 (-)	0 00 (-)	0 02 (-)
Between GA and High-P	0 009 (-)	0 002 (-)	0 01 (-)	0 38 (+)

value occuring with the j^{th} result, c_j; e_i is the total number of examples having the i^{th} attribute value, and e_T is the total number of examples in the training set, the limits M and N denote the number of result and attribute values respectively.

The ID3 algorithm uses the ITM in a splitting strategy [12] to decide which attribute provides the most information from the example set. When several attributes are tested for their information content, the one giving the largest ITM value (hence, information) is the attribute chosen The ID3 algorithm constructs a decision tree as follows:

1. The root node is chosen using the ITM values. The ITM having the largest value is the root node, and the examples associated with the root node are examined
2. If the examples associated with the current node all have the same class value, then an *endnode* is reached. The next attribute is chosen by recalculating the ITM values of the remaining attributes. The largest value determines the next attribute.
3. The current node has at least two branches. The examples are associated with their branches according to their values.
4. For each branch node, repeat step 2, until all endnodes have been found.

A simple example using the navigation sensor management problem will illustrate how decision trees are formulated using the ID3 method. Consider that a decision tree to predict RSS position errors for different navaid types and number of ground stations is to be extracted from a set of covariance simulations. The navaid types and numbers studied were:

Navaid Type = {LORAN, TACAN, VOR} (Model 4a)
Number of Ground Stations = {Two, Three} (Model 4b)

The RSS position error is determined from five simulations, where the RSS performance for each simulation has been classified into four categories:

RSS performance classes = {c-1, c-2, c-14, c-15} (Model 4c)

where the values c-1, c-2, c-14, and c-15 represent user-defined intervals of RSS position error. The five simulations and corresponding classified RSS errors are shown in Table V:

Table V Training Example Set for Decision Tree Induction

Navaid Type	Number of Stations	RSS Position Error Class
LORAN	2	c-2
LORAN	3	c-1
TACAN	3	c-2
VOR	2	c-15
VOR	3	c-14

First, the ITMs for the navaid type and number of stations attributes are computed. In this example, $N_{NAVAID} = 3$, $N_{STATIONS} = 2$, and $M = 4$. From Table V, the probabilities that each value of each factor occurs with each of the RSS position error classes is:

Table VI Value-Result Occurrence Probabilities for Training Example Set

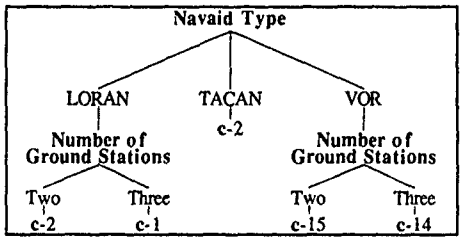
	c-1	c-2	c-14	c-15
LORAN	p_{11}=0.5	p_{12}=0.5	p_{13}=0.0	p_{14}=0.0
TACAN	p_{21}=0.0	p_{22}=1.0	p_{23}=0.0	p_{24}=0.0
VOR	p_{31}=0.0	p_{32}=0.0	p_{33}=0.5	p_{34}=0.5

	c-1	c-2	c-14	c-15
Two	p_{11}=0.0	p_{12}=0.5	p_{13}=0.0	p_{14}=0.5
Three	p_{21}=.33	p_{22}=.33	p_{23}=.33	p_{24}=0.0

Using Eq. (9) and the probability tables above, the ITSs for the two attributes are:

$$ITM_{NAVAIDS} = 0.776 \text{ bits}, \qquad ITM_{STATIONS} = 0.394 \text{ bits}$$

From step one of the ID3 algorithm, the navaid type attribute is chosen as the first node because its ITM value is larger than that of the number-of-stations attribute. Applying the remaining steps in the algorithm, the decision tree extracted from the example set above would look like the following:



Developing the ID3 Attribute Framework Using ANOVA Results

The Model 3 covariance simulations were used to extract decision trees for the NSM Expert system. Eleven attributes were defined for the ID3 framework:

- Navaid type,
- Trajectory leg,
- Aircraft groundspeed,
- Number of ground stations,
- Minimum geodetic distance from station,
- Maximum geodetic distance from station,
- (Max - Min) distance on trajectory leg,
- Maximum line-of-site angle from station,
- Minimum line-of-sight angle from the station,
- Direction of flight (approaching or receding) relative to station,
- RSS position error class on previous trajectory leg.

The distance from a ground station is an important attribute since the signal-to-noise ratio decreases as the distance to the station increases. The direction of flight with respect to the station influences position accuracy through its effect on the line-of-sight angle.

The trajectory parameters were computed for each of the high-performance, jet transport, and general aviation trajectories on each trajectory leg. The maximum and minimum distances to the aiding station were also determined on each trajectory leg, in addition to the difference between the maximum and minimum distances.

When more than one station was used the maximum and minimum distances were the closest and farthest distances computed to the stations. The distance difference is the algebraic difference between the farthest and closest distances determined on the trajectory leg. A similar definition was applied to the line-of-sight angle; from the angles computed to each station, the largest and smallest were selected. The ID3 algorithm's task was then to determine how these attributes were related to each other and to the final RSS position error.

The classification scheme chosen to represent the RSS position error endnode in the NSM decision trees is depicted in Table VII. Since an approximate prediction of the RSS position error was of interest, it was appropriate to represent the RSS performance in terms of an error range.

Table VII RSS Position Error Classification Scheme

[High-Accuracy]		[Medium-Accuracy]		[Low-Accuracy]	
Error Range, N. Mi.	Code	Error Range, N. Mi.	Code	Error Range, N. Mi.	Code
0.0-0.02	c-1	0.10-0.20	c-6	1.0-1.5	c-15
0.02-0.04	c-2	0.20-0.30	c-7	1.5-2.0	c-16
0.04-0.06	c-3	0.30-0.40	c-8	2.0-2.5	c-17
0.06-0.08	c-4	0.40-0.50	c-9	2.5-3.0	c-18
0.08-0.10	c-5	0.50-0.60	c-10	3.0-3.5	c-19
		0.60-0.70	c-11	3.5-4.0	c-20
		0.70-0.80	c-12	4.0-4.5	c-21
		0.80-0.90	c-13	4.5-5.0	c-22
		0.90-1.00	c-14	> 5.00	c-23

The velocity, distance, and line-of-sight angles were expressed in terms of ranges instead of individual values, so that the expert system weights trends more heavily than specific examples. This renders the expert system more adaptable to new conditions, because matches between the actual and knowledge-base cases could be obtained more frequently.

The example set was developed using the attribute framework described above. The RSS position errors for each simulation were classified on each trajectory leg using the scheme in Table VII. The ID3 example base was then created from each single-, double-, and triple-station simulation.

NSM Decision Trees

The NSM example set was divided into 17 smaller example sets. The GPS and on-board navaid examples were grouped into one expert, whereas the ground-based navaid examples were divided according to navaid type and time (15-minute intervals). The ID3 algorithm constructed decision trees for each of the 17 small expert systems that comprise the larger NSM Expert. The breakdown of the NSM Expert into smaller systems provides greater manageability of the training example base. The total number of examples used to develop the NSM Expert System was 932, based on 260 Kalman Filter covariance simulations. An additional 37 simulations were performed to obtain a decision tree to estimate RSS performance when different navaid types are combined. The NSM expert system prompts the user for a set of flight conditions commensurate with the attribute/value lists used in the example set, and the resulting RSS classification code is returned to the user from the decision tree.

A typical decision tree obtained for the ground-based navaids is exemplified by the TACAN results. Figure 8 presents the decision trees for single-, double-, and triple-station combinations on the first 15-minute trajectory leg. Here, the majority of the testing nodes are trajectory parameters (distance, LOS angle, direction of flight with respect to the stations). The top or root node in Fig. 8 is the aircraft's direction of flight. This is expected because the distance and LOS angle attributes are dependent on directional motion. Distance, LOS angle, and groundspeed are results of the aircraft's motion; hence, they represent more specific problem parameters, and it is expected that these parameters appear at a lower depth in the decision tree. Figure 8 also shows that distance, ground velocity, LOS angle, and hybrid performance history are significant factors that enable a prediction of the RSS error to be made. The RSS classification results verify that the closer the aircraft is to a station, the smaller is the RSS error; other results show that the larger is the LOS angle, the smaller is the RSS error [14].

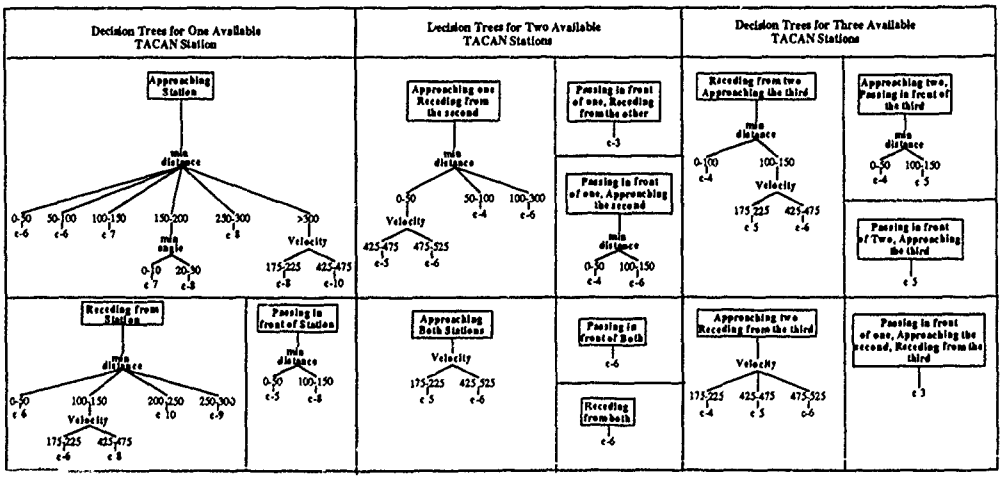


Figure 8. Decision Trees Predicting RSS Position Error Range for an INS Aided by TACAN During the First 15 Minutes of Flight

The expected performance of the GPS system on each trajectory leg is shown in Fig. 9. Note that the aircraft's groundspeed plays an important role in the GPS hybrid's performance. Velocity affects the measurement dynamics (history) and is therefore classified as a trajectory effect. From Fig. 9, the two-satellite hybrids are more sensitive to these velocity effects than are the three- and four-satellite hybrids.

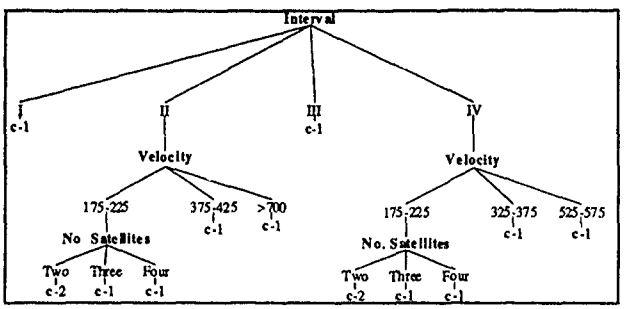


Figure 9 Decision Tree Predicting RSS Performance for an INS Aided by GPS

Finally, the decision tree showing what position error range is expected when different navaid types are integrated in a hybrid system is presented in Fig. 10. Note that the decision tree is not specified for a given trajectory leg. The RSS position errors for these simulations were averaged over the entire flight time for the high-performance trajectory. The tree is organized in terms of the navigation method used: (1) Distance-Velocity (ρ-V), (2) Bearing-Velocity (θ-V), (3) Distance-Bearing (ρ–θ), (4) Distance-Distance (ρ–ρ), (5) Bearing Bearing (θ–θ), and (6) Velocity-Velocity (V-V). These results show that LORAN is a better distance-measuring navaid than DME and that Doppler Radar provides better navigation accuracy than the Air Data Sensor when ρ-V navigation is used. The ρ–θ results show that it is possible to obtain good performance when LORAN and VOR are used. The LORAN/DME hybrid gives better results than two DME stations but worse performance than two LORAN stations. By far the worst results are obtained using two VOR stations. As discussed before, the VOR system is the least accurate measurement device of the seven systems studied, which greatly affects INS-VOR hybrid results.

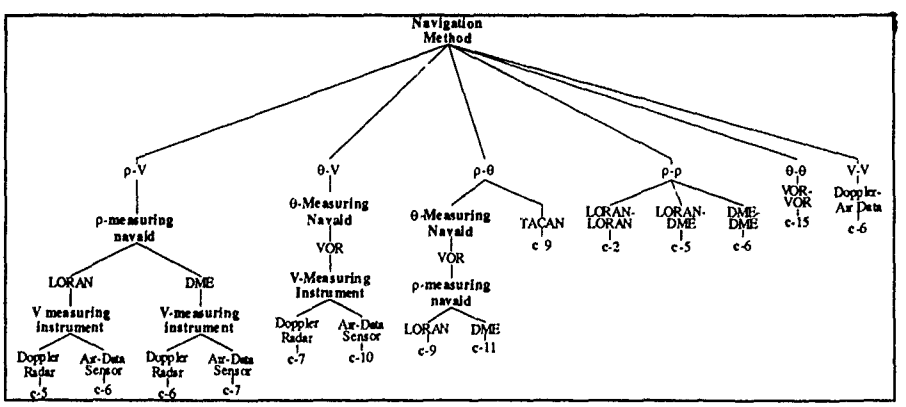


Figure 10 Decision Tree Predicting RSS Performance When Different Navaid Combinations are Used to Aid an INS

PERFORMANCE RESULTS OF NSM EXPERT SYSTEM

It is important to quantify the NSM Expert's performance for several test scenarios. It also is desirable to determine the factors that affect the system's performance, so that these factors can be exploited in future system development.

Two high-performance trajectories were used in the performance evaluation of the NSM Expert. The two trajectories each consist of four 15-minute legs. Trajectory #2's flight pattern was in a counter-clockwise direction, whereas clockwise flight patterns were used to develop the NSM Expert (Fig. 1). Additionally, the takeoff point on Trajectory #2 was five degrees farther north than the training trajectories' takeoff points. These trajectory differences change the measurement and INS dynamics, affecting the hybrid performance. Trajectory #2 was designed this way intentionally, so that the NSM Expert System's adaptability could be determined.

Single-, double-, and triple-station combination hybrids were simulated on each test trajectory for the DME, VOR, TACAN, and LORAN systems. The combinations were formed using four ground stations located as in Fig. 1 with respect to each other. Additionally, two-, three-, and four-satellite hybrids were simulated on the test trajectories, as were Doppler Radar and Air Data sensor hybrids. In total, 60 covariance simulations were performed for the two test trajectories.

Test Trajectory Data Preparation, Performance Metrics, and Results

The performance results for each of the 60 simulations were classified on each trajectory leg according to the scheme in Table VII. The total number of matches was counted on each leg of each test trajectory for the seven navaid types studied. A match was declared between the actual and predicted RSS classification if and only if the RSS classification codes differed by one or less. For example, if the NSM Expert predicted an RSS classification code of 6 whereas the covariance results determined a performance of Class 7, a match was declared. A match also would have been declared if the actual performance was Class 5. Since the NSM Expert is only expected to *estimate* a hybrid's performance, it is allowed some room for error.

The NSM Expert System was run 488 times in order to determine the number of matches for each system on the test trajectories. Figure 11 shows the NSM Expert's performance in predicting the RSS position error for each hybrid configuration. The *predictive performance metric* for each navaid is defined as the percentage of number of matches obtained from the total number of combinations tested for that navaid. The matches on all four trajectory legs are reflected in this figure.

The NSM Expert performed very well on the two test trajectories. Figure 11 shows that the NSM Expert correctly predicts the RSS position error better than 70% of the time on test Trajectory #1. The system required only the trajectory information and its knowledge of hybrid system performance to make these predictions. However, its predictive capability on test Trajectory #2 is slightly worse for the LORAN hybrids (69%), considerably worse for the VOR (45%) and Air Data sensor hybrids (53%), and identical for the remaining configurations. Hence, the results from Trajectory #2 suggest that additional investigation into trajectory effects on VOR's and Air Data Sensor's performance may be necessary.

The results in Fig. 11 are encouraging for expert system designers. We have shown that an expert system can be designed from data, and that good results are obtainable even from relatively small training sets. The total number of examples used to obtain the NSM decision trees was slightly less than one thousand.

Next, we investigated how the NSM Expert's performance could be improved. Two factors affecting the Expert's performance were examined: the number of problem attributes selected to describe the problem, and the number of examples in the ID3 training set. Figure 12 shows the NSM's predictive performance metric plotted as a function of the number of attributes used in the knowledge bases of four navaid expert systems The complete attribute set may be found in Ref. 14.

To study the effect of number of attributes on the NSM Expert's performance, the number of attributes used in each of the 17 expert modules was decreased one at a time and new decision trees were obtained. The "root node," representing the most important attribute on which a decision tree is be based, was eliminated each time in order to establish a systematic method of attribute elimination. The results in Fig. 12 were obtained by running the NSM Expert each time an attribute was deleted. The results reflect the cumulative number of matches for both test trajectories on all four trajectory legs; in total, 3,296 runs of the NSM Expert were

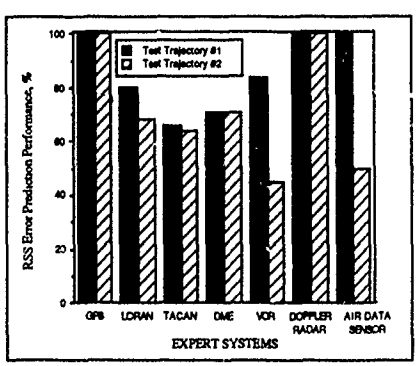


Figure 11. Performance of Navaid Experts on Test Trajectories

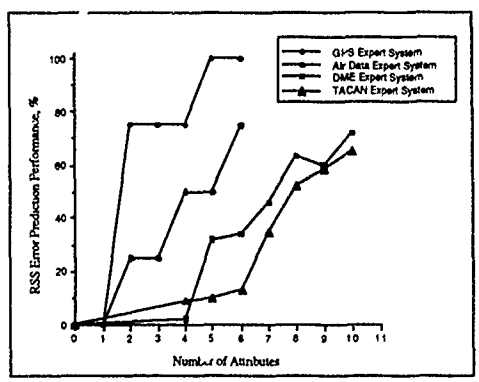


Figure 12. Performance of Selected Navaid Experts With Increasing Number of Attributes Describing the NSM Problem Space

required to produce the results in Fig. 12. As seen in the figure, the performance of each navaid expert increases as the number of attributes increases. There is a small dip in the DME expert's predictive RSS error capability when nine attributes are used; since a relatively small number of training examples was used to develop the NSM expert, small dips can be expected, and they are relatively insignificant when compared to the overall trend.

The effect of number of examples in the ID3 training set on the NSM Expert's performance is shown for the TACAN Expert in Fig. 13 for both test trajectories. This figure was obtained by deleting examples in the training set, reinducing the decision trees on each time interval, and counting the number of matches between covariance-determined and NSM-determined RSS classification.

As seen in the figure, there is an increase in expert system performance with increasing number of examples. An important trend in Fig. 13 is the reduced rate of performance increase with increasing numbers of examples. Although it is difficult to extrapolate directly from the curve, it is estimated that an 80% prediction performance could be attained with greater than 300 TACAN hybrid examples.

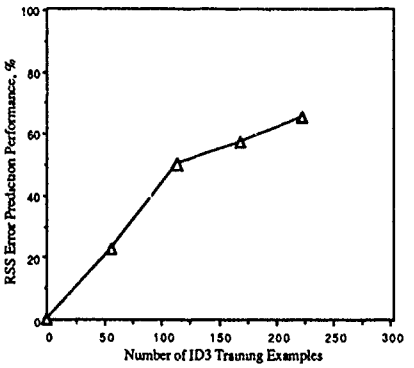
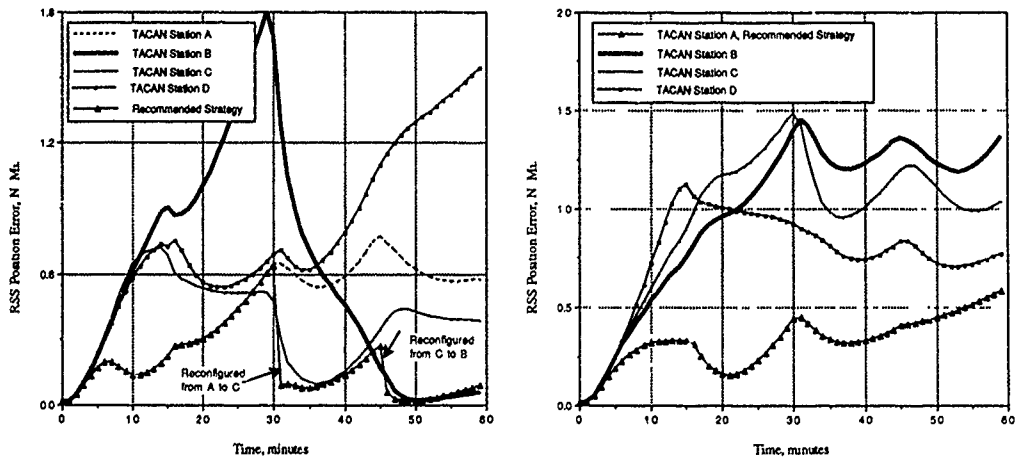


Figure 13. Performance of TACAN Expert With Increasing Number of Training Examples

Selecting Navigation Strategies Using the NSM Expert System: One Navaid Available

The NSM Expert was used to determine navigation strategies for several sensor availability scenarios on two high-performance test trajectories. The NSM-recommended strategies were then compared with strategies obtained from examining covariance data. In the first test, the NSM Expert was limited to choosing a strategy when only hybrids of one navaid type may be used or when only one measurement may be processed. In the second test, the NSM Expert chose strategies when two measurements of similar or dissimilar navaids could be processed.

In Figs. 14a-b, the NSM Expert found the best navigation strategies on both test trajectories when TACAN Stations A-D are available. For test trajectory #1 shown in Fig. 14a, the NSM Expert recommended Station A for the first 30 minutes, Station C from 30-45 minutes, and Station B from 45-60 minutes. For test trajectory #2 shown in Fig. 14b, the NSM Expert recommended Station A throughout flight. Similar strategies were determined when DME Stations A-D were available. This is because Stations A-D are fixed with respect to the two test trajectories. Therefore the relative RSS performance for the stations would be similar whatever the station type might be.



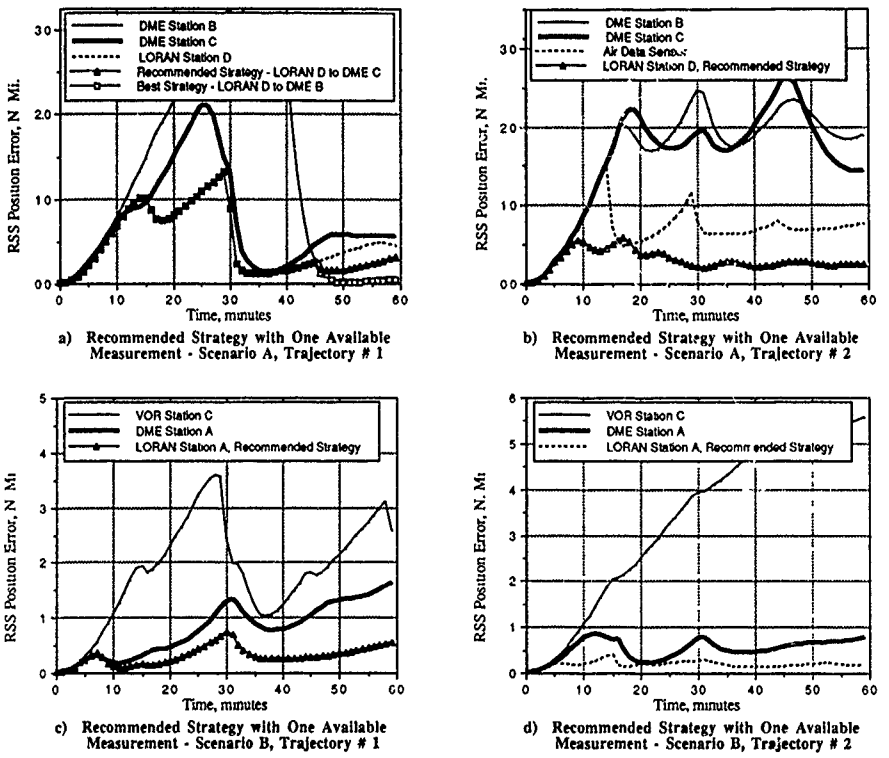
a) One TACAN Station Aiding an INS - Trajectory #1

b) One TACAN Station Aiding an INS - Trajectory #2

Figure 14. Comparison of NSM-Recommended and Covariance-Determined Nav Strategies when One Navaid Type Available to Aid an INS

Comparisons of single-measurement NSM-recommended and covariance-determined strategies are shown in Fig. 15 when navaids of different types are available. Two scenarios were investigated in these figures, as follows:

Scenario A - DME Stations B and C, LORAN Station D, and Air Data Sensor are available.
Scenario B - VOR Station C, DME Station A, and LORAN Station A are available.



a) Recommended Strategy with One Available Measurement - Scenario A, Trajectory # 1

b) Recommended Strategy with One Available Measurement - Scenario A, Trajectory # 2

c) Recommended Strategy with One Available Measurement - Scenario B, Trajectory # 1

d) Recommended Strategy with One Available Measurement - Scenario B, Trajectory # 2

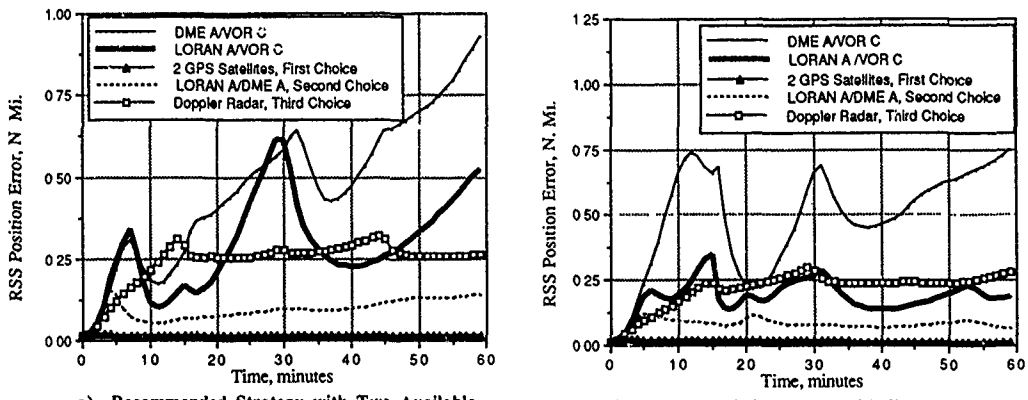
Figure 15. Comparison of NSM-Recommended and Covariance-Determined Nav Strategies with One Measurement from Different Navaids Available to Aid an INS

On test trajectory #1 when Scenario A navaids are available, the NSM Expert recommended that LORAN Station D be used until the fourth trajectory leg (Fig. 15a). Then, the NSM Expert recommended reconfiguring the hybrid filter from LORAN Station D to DME Station C; however, the computed covariance results show that the best reconfiguration would be from LORAN Station D to DME Station B. The resulting difference in performance is slight. From Fig. 15b, the NSM Expert chose LORAN D to aid the INS throughout the entire flight. For Scenario B, the NSM Expert chose LORAN A over DME A and VOR C to aid the INS on both trajectories, as shown in Figs. 15c and 15d. This suggests that the NSM Expert "learned" the colocation rule exhibited in Fig. 2. When two stations are colocated, the selection is based on the navaid hierarchy LORAN, TACAN, DME, and VOR

Selecting Navigation Strategies Using the NSM Expert System: Two Measurements Available

Using the navaids available in Scenarios A and B above, the NSM Expert was given the task to find the best two-measurement hybrid strategies. Since hybrids were to be formed from different navaid types, the navaid mixing decision tree in Fig. 10 was used in addition to the 17 individual navaid experts. Recall that Fig. 10 provides an approximation of the expected position error for the various navaid mixes and does not include trajectory and performance history effects

The results obtained by mixing the Scenario A navaids together are shown in Fig. 16. On both test trajectories, the NSM Expert recommended that two GPS satellites be used to aid the INS. The second and third choice configurations were the LORAN A/DME A and Doppler Radar hybrids, respectively. From Fig. 16b, the LORAN/VOR hybrid performed on a par with the Doppler Radar hybrid on test Trajectory #2, although this was not predicted by Fig. 10.



a) Recommended Strategy with Two Available Measurements - Scenario A, Trajectory # 1

b) Recommended Strategy with Two Available Measurements - Scenario B, Trajectory # 2

Figure 16. Comparison of NSM-Recommended and Covariance-Determined Nav Strategies with Two Measurements from Different Navaids Available to Aid an INS

CONCLUSIONS

The performances of seven navigation systems aiding a medium-accuracy Inertial Navigation System (INS) were investigated using Kalman Filter covariance analyses. Hybrid performance decisions were based on the RSS position error history metric. A Navigation Sensor Management Expert System was designed from covariance simulation data using a systematic method comprised of the two statistical techniques, the Analysis-of-Variance (ANOVA) method and the ID3 algorithm.

ANOVA results show that statistically different position accuracies are obtained when different navaids are used, the number of radio navigation ground stations or Global Positioning System satellites used to aid the INS is varied, the aircraft's trajectory is varied, and the performance history is varied. By indicating that these four factors significantly affect the decision metric, an appropriate parameter framework was designed, and a simulation example base was created.

The example base was composed of over 900 training examples from nearly 300 simulations. The example base was divided into 17 smaller groups to enhance manageability. The ID3 algorithm then was used to determine the NSM Expert's classification "rules" in the form of decision trees The performances of these decision trees were assessed on two arbitrary trajectories, by counting the number of times the rules correctly predicted the RSS position accuracy. These performance results then were presented using a predictive metric.

The ANOVA/ID3 method was very effective for the systematic development of the NSM Expert using simulation data. Results show that the NSM Expert can accurately predict the expected RSS position error for a specified navigation sensor suite operating on a specified aircraft trajectory between 45% and 100% of the time. The test trajectories used to evaluate the system's performance show that the NSM Expert adapts to new situations and provides reasonable estimates of the expected hybrid performance. The system's good performance with relatively few examples clearly shows how the ID3 algorithm maximizes the information content contained in the example base. The performance results strongly suggest that operational systems can be designed from simulation or experimental data using the ANOVA/ID3 method for knowledge acquisition The systematic nature of the method makes it a useful tool for expert system designers.

With slightly less than 300 navigation hybrid simulations used to develop its knowledge base, the NSM Expert exhibits a good capability for finding the best or second-best navigation strategies. The most important aspect of the system is its development using mathematical models and digital computer simulations, and not through knowledge-extraction using human expertise. Also important is the systematic development of this Expert System using the well-known ANOVA and ID3 statistical methods. The exercise also showed how potentially very large expert systems can be broken down into several small experts, thereby facilitating system maintenance and enhancing future development.

It was shown that the NSM Expert's ability to solve navigation sensor management problems increases with the number of examples used in its training set and with the number of problem descriptors. The NSM Expert's good performance, given its relatively small training set, is very encouraging. It demonstrates that large, carefully-planned simulation experiments can be used in a systematic manner to develop a fully-operational expert system with a designer-specified performance effectiveness.

Other aerospace applications that are good candidates for the ANOVA/ID3 method are air combat pilot strategies from simulation or flight test data and air traffic control solutions to multi-configuration problems. The expert system design methodology also is pertinent to problems such as nuclear reactor control strategies, chemical process control strategies, automated highway driving, and robotics applications. In each case simulation or operational experiments may be executed for the systematic development of an expert system advisor.

REFERENCES

[1] G. Kahn *et al*, "MORE: An Intelligent Knowledge Acquisition Tool", in *Proceedings of the Ninth International Conference on Artificial Intelligence*, Los Angeles CA, 1985.

[2] F.G. Unger and R.S. Sindling, "Systems Tasks for Advanced Aircraft Navigation Systems", in *Computers in the Guidance and Control of Aerospace Vehicles*, AGARD-AG-158, February 1972.

[3] A. Gelb and A. Sutherland, Jr., "Software Advances in Aided Inertial Navigation Systems", *Navigation*, The Institute of Navigation, Washington DC, Vol. 17, No.4, Winter 1970-71, pp 358-369.

[4] J. Richman and B. Friedland, "Design of Optimum Mixer-Filter for Aircraft Navigation Systems", in *Proceedings of the IEEE 1967 National Aerospace and Electronics Conference (NAECON) 1967*, pp.429-438

[5] W. Zimmerman, "Optimum Integration of Aircraft Navigation Systems", in *IEEE Transactions on Aerospace and Electronics Systems*, Vol. AES-5, No.5, September 1969, pp. 737-747.

[6] J. W. Burrows, "Combined Inertial-ILS Aircraft Navigation Systems", *Journal of Aircraft*, Vol. 8, No 6, June 1971, pp.439-443.

[7] D. B. Cox Jr., "Integration of GPS with Inertial Navigation Systems", in *Global Positioning System Vol. II* , papers published in *Navigation*, The Institute of Navigation, Washington DC, 1984, pp. 144-153.

[8] E. J. Rose *et al.*, *A Cost/Performance Analysis of Hybrid Inertial/Externally Referenced Positioning/Orientation Systems*, Report ETL-R-086, U.S. Army Engineer Topographic Laboratories, September 1985.

[9] A.D. Pisaro and H.L. Jones, "An Expert System Approach to Adaptive Tactical Navigation", in *Proceedings of the First Conference on Artificial Intelligence Applications*, IEEE Computer Society, 1982, pp. 460-464

[10] S. Berning, D P. Glasson and J. L. Pomarede, "Knowledge Engineering for the Adaptive Tactical Navigator", in *Proceedings of the IEEE National Aerospace and Electronics Conference (NAECON)*, Dayton, OH, May 1988, pp. 1266-1273.

[11] G. E. P. Box, W. G. Hunter and J. S. Hunter, *Statistics for Experiments: An Introduction to Design, Data Analysis and Model Building*, John Wiley & Sons, New York, 1978.

[12] *Expert Ease Manual*, Human Edge Software, Intelligent Terminals Ltd., Palo Alto, CA, 1983.

[13] J. R. Quinlan, "Discovering Rules by Induction From Large Collections of Examples", in *Expert Systems in the Micro Electronic Age*, D. Michie, Editor, Edinburgh University Press, 1979, pp. 169-201.

[14] B. L. Belkin, *Cooperative Rule-Based Systems for Aircraft Navigation and Control*, M.S.E. Thesis, Report 1856T, Department of Mechanical and Aerospace Engineering, Princeton University, June 1989

[15] B. L. Belkin and R. F. Stengel, *Quantitative Knowledge Acquisition for Expert Systems*, Presented at the Space Operations, Applications, and Research Symposium, Albuquerque, NM, June 1990

[16] P. S. Maybeck, *Stochastic Models, Estimation and Control*, Vol. 1, Academic Press, New York, 1979.

[17] R. F. Stengel, *Stochastic Optimal Control -- Theory and Application*, J Wiley & Sons,New York, 1986

[18] M. Kayton and W. R. Fried, *Avionics Navigation Systems*, John Wiley & Sons, New York, 1969.

ACKNOWLEDGMENTS

This project was sponsored by the U.S. Office of Naval Research and Army Research Office under Contract No. DAAG29-84-K-0048, and supported by NASA and the FAA under Grant No. NGL31-001-252. The first author is grateful for the support of the Natural Sciences and Engineering Research Council of Canada (NSERC).

Knowledge Extraction Methods for the Development of Expert Systems

Manuel Perez,
Directorate of Aerospace Studies,
Air Force Systems Command,
Kirtland AFB, New Mexico 87117-6008, USA

Leopoldo Gemoets and Robert G. McIntyre,
University of Texas at El Paso, El Paso, Texas 79968, USA

ABSTRACT

The development of expert systems require the use of engineering techniques which can be used to efficiently and correctly extract the domain knowledge resident within the human expert. To apply these techniques, certain conditions must be met. These conditions are that the candidate expert system domain must be suitable for implementation, that there be a knowledge engineer with a certain level of domain knowledge, and that the right human domain experts be selected in the expert system development effort. This paper presents a semi-sequential approach to development of techniques which can be used to extract the knowledge from the human expert. Presented are both direct and indirect methods which a knowledge engineer can use to extract this knowledge.

INTRODUCTION

In recent years, expert systems have been developed for a wide variety of subject domains such as medicine, engineering, and the sciences. With these developments, there has been a creation of techniques which serve to extract the domain knowledge from the human experts. The methods used to extract this information from the human domain experts is commonly referred to as knowledge engineering. There are two main approaches used in knowledge engineering. First, there is the object-attribute-value approach which uses forward-chaining from the inference engine and knowledge structure facts to reach a designated goal. Second, there is the inferential approach which uses if-than rules from the goal and backchains through the knowledge structure to deduce the facts. Knowledge engineering, still in its infancy as a field, consists of the knowledge acquisition or extraction process which is used to develop an expert system. Consequently, the knowledge extraction techniques in existence today are largely the result of the emerging technology which has risen out of the development efforts of current expert systems.

Before an expert system is developed, the knowledge engineer must understand that this effort is not a trivial task. He must also determine whether the candidate domain is suitable for implementation into an expert system. In making this determination, it is important that the following conditions be met:

(1) the domain must be stable enough and precisely defined so that it can be decomposed into the various subdomains which are resident with it.
(2) there must be a user's requirement for having such a system.
(3) there must be human domain experts from which the domain knowledge can be extracted.
(4) there must be a knowledge engineer with an indepth understanding of the subject domain.
(5) there must be some engineering tools or methods with which the knowledge engineer can extract the domain knowledge from the human expert.

The most difficult of these tasks is the extraction of the knowledge from the human domain expert. The knowledge engineer or knowledge extractor must design methods for revealing the complex structures and processes used by the expert in solving various types of problems. This engineer must understand that the expert has knowledge stored in various types of structures. Some of these types of structures include information stored in lists, tables, decision logic trees, and in hierarchies of relationships which consist of nested categories or clusters.

The knowledge engineer must also recognize that an expert possesses the skill to adapt old patterns which he/she sees applicable in a new problem through the application of known good rules used before. In this case, the knowledge engineer should obtain a set of representative problems which the expert is familiar with and have him/her articulate the steps and methods used in problem solving. The information which the knowledge engineer extracts then consists of rule bases which can then be implemented into an entity called the inference engine. This engine, a very important component of the expert system, is then modified as new rules are defined during the course of user-interaction with the prototype expert system.

This paper presents a semi-sequential approach to the development of techniques which can be used to extract the knowledge from the human expert. Presented is a description of the procedures and techniques in developing an expert system. The discussions include consideration of the domain suitability for an expert system, the knowledge engineer requirements for effective knowledge acquisition, the selection of the domain expert, the actual direct and indirect methods for knowledge acquisition, some discussions on system prototyping and user interfacing, and on testing and knowledge base maintenance of the developed expert system. This familiarization can be

acquired in the following ways: To accomplish this, the engineer should:

DOMAIN SUITABILITY FOR AN EXPERT SYSTEM

Before beginning the development effort for an expert system, the proponents for system development must determine whether the proposed domain is suitable for incorpoation into an expert system. This is done by obtaining surveys from well-known experts in the domain on what they believe are the areas where such systems are needed most. The experts should also be able to identify the intended users of such a system. For example, if a development proponent was looking at the feasibility for developing an expert system to aid physicians in diagnosing a certain disease, the propcnents would seek the advice from specialists in the domain area and have them identify areas where they believe that such a system would prove of some intrinsic value to the intended user. Following this, the knowledge engineer determines whether the domain is stable enough for development by getting additional opinions from well-known experts on the variations or conflicts which may exist in how processes and their outputs are viewed within the domain. If the domain is continuously changing or evolving, then development may have to be deferred until the domain stabilizes or there is a majority of agreement on how the domain is viewed by the experts. After establishing the stability of the domain, the knowledge engineer must then determine whether the candidate domain knowledge can be extracted given the current technology and knowledge engineering methods. This is important because a domain may be good for system development but there may not be any existing methods for extracting the knowledge from the expert or the knowledge engineer (or candidate) may not have sufficient domain knowledge to develop a plan for extracting the information.

KNOWLEDGE ENGINEER REQUIREMENTS

The knowledge engineer should have a working-level familiarity of the domain which is candidate for implementation in an expert syster. To accomplish this, the engineer should:

(1) attend tutorial sessions with some domain experts to obtain a general understanding on the domain concepts and technology.
(2) prepare a domain knowledge document which contains general domain knowledge. This document also serves the purpose of being used as a training tool for project members which later join the team and is valuable because it contains the common grammar used in the expert system.
(3) read as many domain reference materials to include any literature on past expert system development efforts which are related to the current effort. For example, if a knowledge engineer were trying to develop knowledge for development of an expert system to diagnose a certain disease, he/sne would make sure that he/she has read all pertinent literature on systems like MYCIN (Reference 1), EMYCIN (Reference 2), and INTERNIST-I (Reference 3) to name a few.
(4) conduct short knowledge acquisition sessions with the experts on a time-available basis.

A required output of the familiarization phase is that the knowledge engineer should prepare a paper (which contains the knowledge document identified in (2) above) which identifies the knowledge base in domain facts and rules which are clear, descriptive, concise, and preferably in a pseudo-English format. This allows for ease in the initial prototyping of the system, the ability to backtrack if necessary with complete information/knowledge available to determine problem areas, and make it easier for the expert to communicate in a format which is known to him/her.

SELECTION OF THE DOMAIN EXPERT

Since the human domain expert is the most important driver for development of the expert system, the knowledge engineer needs to make sure that the experts which he/she select for the knowledge extraction process are well-known, acknowledged experts within their domain community. The engineer also needs to determine whether the expertise lies with certain individuals or within the confines the expert community. He/she also needs to establish some criteria for the selection process. Other important factors are that the candidate experts must have the following:

(1) have current experience on working problems within the domain of their expertise.
(2) the support of their management so that they can devote considerable time to the development project.
(3) be cooperative, and willing to allow for the knowledge engineer to extract the resident knowledge through a variety of methods (some of which might seem boring, time-consuming, and perhaps a bit contrived).
(4) see the potential benefits that he/she can reap by having a companion advisor to aid him/her in doing some of the routine, detailed, or can then allow the expert to work on areas which interest him/her because of having more time for these efforts.
(5) complete confidence that such a system, once available, is not meant as a replacement for his/her expertise.

The knowledge engineer can accomplish this by selecting a panel of domain experts (not the ones from which the knowledge is to be extracted) to identify potential candidates for selection. These candidates are then given typical, non-contrived problems which are common within the domain and evaluated based on their observed strengths and weaknesses as experts. The evaluation would be based on the ultimate recommendation of the panel of experts but with final judgement made by the knowledge engineer. Once these potential candidates have been selected, the knowledge engineer, with the consultation of the expert panel, must evaluate the experts, and select one or several (in some cases) experts for the project.

KNOWLEDGE ACQUISITION

Before actually starting the knowledge acquisition phase, the knowledge engineer must make the assumption that this phase is the single, most important phase in the development of inference strategies because it is a driving variable in the development of the main part the expert system, the inference engine. The engineer must also understand that developing an expert system prototype can take several months to a year as the knowledge base is expanded and reformulated many times. Therefore, it is important that the knowledge engineer choose an approach which is flexible enough to allow for ease in prototyping a system as early as possible while at the same time having a definitive enough format which is easy to follow in backtracking procedures, should the need arise.

Initially, knowledge engineer should:

(1) organize knowledge acquisition meetings so as to maximize his/her access to the expert with only minimal interruptions.
(2) allow the meeting attendees access to the information gained and the prototype which is developed as soon as possible.

The first consideration is based on the recognition that the expert's time is very valuable and that this time is not entirely available to the development effort. For this reason, it is important to hold the meetings away from the expert's office so as to avoid interruptions. However, the knowledge engineer must also recognize that he/she might want to see the expert perform his tasks in his/her own habitat. So it is important that tradeoffs be established to avoid possible conflicts and maximize the amount of time which the knowledge engineer has with the expert. The second is to insure that expert system can be run as the prototype is being developed from the acquired knowledge to checkout the parts of the program. In this way, the outputs of the prototype are evaluated and used as drivers for the development for subsequent modifications to the prototype. The knowledge acquisition process follows the subprocesses during the knowledge acquisition phase:

(a) the experts is given or is told to provide typical domain problems which he/she solves in a step-wise approach.
(b) the knowledge engineer acquires the knowledge acquired through a variety of direct and indirect techniques and documents the observed rules and facts in the knowledge document using the descriptive, concise pseudo-English format. These techniques are discussed in the following sections.
(c) from the information/knowledge gained in the initial problem-solving, the knowledge engineer generates new problem sets (either by hand or through the use of a computer) and has the expert solve these problems. These problems should, over the course of several iterations, converge on the fringes or edges of where the existing rules and facts are approaching the limit applicability. In this manner, the knowledge engineer can be assured that he/she has covered the complete spectrum of problems feasible within the bounds of the expert domain.
(d) as the rules are acquired, the prototype build-up is started. The expert is then called upon tc provide critiques and analyses of the current state of the prototype by reviewing the applicability of the current rules given the generation of new problems which he/she must consider. Hopefully, the expert continues to generate new problem sets which the knowledge engineer can observe and start the knowledge acquisition process on.

Throughout the entire cycle, the knowledge document is continuously reviewed and updated as new rules/facts and their modifications are uncovered. The following subsections discuss the individual direct and indirect methods which can be used by the knowledge engineer to acquire the knowledge.

KNOWLEDGE EXTRACTION METHODS

DIRECT TECHNIQUES

The direct techniques for knowledge acquisition or extraction consist of interviews, questionnaires, simple observation procedures, verbalized or thinking-out-loud protocols, interruption analysis, drawing of closed curves, and inferential flow analysis. These methods are used to have the expert make known the domain knowledge through the description of the processes used in solving typical domain problems. Each one of those techniques is further described below.

INTERVIEWS

The initial objective of the knowledge engineer is to get the expert's full cooperation at all times so that the resident knowledge can be harnessed. This requires that the expert be willing to share his/her knowledge. Hence, the engineer should select only those experts which meet the criteria previously identified. In addition, the knowledge engineer must make the expert completely at ease so that he/she is willing to be interviewed.

The interviews can be structured or unstructured. However, interviews, by their very nature, are very time-consuming. Therefore, the knowledge engineer needs to maximize the knowledge gathering potential of these interviews so that there is no wasted time with the expert. This is done by having a very well organized plan for conducting interviews with the expert. The interviews can be used to identify the sets of objects and relationships used in the performance of his/her domain tasks. The engineer needs to ask a mixed style of questions which might focus on a particular aspect of a problem and then attempt to generalize it to other types of problems. This can be done by noting the expert's responses, rules, and objects which can be examined for generality in later sessions. The engineer also needs to find out if the expert thinks of the objects in special relationships, lists, tables, or physical spaces. In addition, the engineer must find out any relational/organizational factors, judgement processes, problem-solving methods, and solution designs. When interviewing multiple experts, in a case where the expertise perhaps lies within a community of experts, the knowledge engineer should also find out if there are any special design considerations which the community uses.

In an unstructured interview, the knowledge engineer can let the expert verbalize as he goes through a problem. This is hard to do because the knowledge engineer might interfere with the expert's thought processes by his/her mere presence.

The knowledge engineer should also not try to impose his/her opinions or level of domain understanding on the expert. In other words, the engineer should not try to prompt the expert's answers by influencing his/her remarks. This could result in getting the expert sidetracked away from his/her reasoning path and might mean that the knowledge which could have been obtained is lost. The engineer should let the expert talk even if the current subject area seems momentarily unrelated to the main purpose. If the expert remains for an extended amount of time on these areas, the knowledge engineer should look for the opportune moment (without interrupting) and gently steer the expert back on discussion of the problem at hand.

The knowledge engineer can therefore be content with what little knowledge the expert might verbalize and defer the verbalized or thinking-out-loud protocol (discussed later) until a future time. In any event, the engineer might want to make an audiotape or videotape of the interview so that any of the expert's speech pauses, nongrammatical segments, or unintelligible segments may be reviewed and reconciled with the copious notes the engineer takes during the interviews. These knowledge sources can reveal the inference making processes present (Reference 4). The interviews should be conducted in a small, quiet room with only the minimum essential project personnel present. In most cases, this includes only the knowledge engineer and the expert. However, in some cases, the engineer might want to have several experts (not more than say three) if it is believed the knowledge lies within the community of experts. (The subject of the interview of several experts is discussed in more detail below).

The structured interview, on the other hand, is a derivative of the unstructured type. In this case, the knowledge engineer uses information which was gained in previous interviews or sessions and dwells on the expert's familiar tasks. In order for this to work, the knowledge engineer must make an initial pass of all available knowledge (even if general in nature) which has been analyzed or reviewed. The expert then goes over the data one entry at a time and makes comments which the engineer records. This method forces the experts to systematically go over the knowledge. The expert's comments can be used to change the data base which exists as new information is made available. This information could include (1) the creation or deletion of entries, (2) the qualification or explanation of the current entries, (3) a reorganization of the hierarchical or categorical structures of the data base, or (4) the addition or deletion of categories. After such changes are made, the expert is again taken over a review of the data base until the knowledge engineer believes the knowledge has been extracted.

In some cases, it might be necessary for the knowledge engineer to interview multiple experts. This happens when the knowledge which is desired for extraction is found to be resident within the community of the experts as a whole. For this situation, the knowledge engineer might want to have a single residence expert act as a consultant. The consultant's duties are to make sure that the problems which are given to the experts are not contrived and present all the information required for solutions or elaboration. The consultant does not participant in the design process because of the danger of inserting bias information. However, he/she might prove invaluable in that subproblems are then given to each expert for general solution. This means that the experts do not have to solve the problem all the way through as long as the main solution processes are identifiable and representative of what is required in the problem specifications provided to them. The knowledge engineer then examines the individual solution processes to find out if the experts used the same strategy in terms of decomposition of the main problem into subproblems, subproblem solution methodology, and differences in specialization among the experts.

QUESTIONNAIRES

The knowledge engineer can also use questionnaires to derive the objects of a domain and their relationships and uncertainties. This is a very efficient way of acquiring knowledge and information. It offers the advantage of allowing the expert to complete the questionnaire in a relaxed manner. However, the questionnaire needs to be properly designed to insure that the maximum amount of knowledge/information is extracted from the expert. The questionnaire should be designed with a considerable amount of effort expended to structuring the questions such that they elicit a response on how the expert solves a typical domain problem. The two basic types of questionaires are variable and relationship elicitations. These types are discussed below and are not the same as those used in survey research. In variable elicitations, the expert is asked for the variables which he/she uses and a description of the following:

(1) the variable name.
(2) a description of the variable.
(3) the type of values that the variable can assume.
(4) the size of the values.
(5) the range of the values.
(6) whether there is any uncertainty in the values assumed.
(7) whether the variable is known at the start or uncovered as the reasoning progresses.

In relationship elicitations, the expert is asked to state his/her beliefs on what the relationships between factors are. For example, the expert might be asked about the stock market whether there are any relationships between the price of gold, the number of shares traded in a single day, and the prime lending interest rate. Through this process, the expert is asked to specify the possible relationships.

Once the methods described above are conducted, the knowledge engineer should ask the expert to scale his/her responses for any uncertainty in the particular inferences which might have been reported. The method of using questionnaires has justification and is appropriate because the normal verbal response obtained from the expert might not contain reliable information. The reason for this is that the expert may not be good at estimating the problem. Often, he/she might perceive the problem to be very difficult before he/she starts the problem-solving process, and will have thought that the problem is trivial when finding out how easy it was to solve. In some cases, the expert might show extreme conservatism in his/her responses and therefore cloud the true, underlying answers. In other cases, the expert might be overly optimistic that what he/she are conveying to the knowledge engineer is correct and factual.

The rating scales which are used to exact the information from the expert consist of the bar scaling and the five-point verbal (or Meister) scale. With the first, the expert is asked to approximate his/her level of uncertainty from a range of levels reflecting total uncertainty to total certainty. With the second, the expert is asked to apply a qualitative rating on his/her uncertainty through the use of the terms COMPLETELY, REASONABLY, BORDERLINE, MODERATELY, and EXTREMELY to the uncertainty.

OBSERVATION OF TASK PERFORMANCE

The knowledge engineer can also determine how the expert makes his/her judgements, diagnoses, or design decisions when working through a problem. In order to do this, the engineer must look at the expert solve some typical domain problems. Through observation the engineer can discover the objects, relationships, and inferences which the expert uses. The only problem is that the engineer must make sure that the information/knowledge is correctly recorded so that the domain knowledge is captured in its entirety. The two methods which can be used to do this are for the knowledge engineer to take copious notes and obtain audiotape recordings (video recordings also if time and the development budget permit) as the expert articulates the processes he/she used in problem solving. This method, however, is burdensome on the engineer in that the knowledge engineer might be pressured to try to capture every possible step which the expert demonstrates or articulates. The end-results could be that the engineer might miss significant portions of the processes which are described or might read too much into what the expert is really conveying. The videotaping (and to a certain extent the audiotaping) processes offer the flexibility of allowing the knowledge engineer to review the tapes later and obtain answers to what he/she missed taking notes on. However, the disadvantage is that the knowledge engineer might rely too heavily on the tapes and will not take down notes on important facts which the expert conveys while going through the problem solving procedures. Also, this method should be applied expeditiously because in some cases it relies too much on assuming that the expert may be stating his/her true line of reasoning in the process. This might not be the case.

PROTOCOL ANALYSIS

Another technique which is closely related to observation of task performance is protocol analysis. This technique is not applicable to all kinds of tasks (Reference 5). The difference between observation techniques and this technique is that the expert performs the domain task by going through an applicable problem and thinking the processes used out loud as he/she does so. Normally these types of sessions are videotaped and the knowledge engineer annotates the displayed behavior after the session by

reconciling the behavior with the notes taken in real-time. Typical entries noted on these annotations are statements about what the expert's goals are at particular steps in the problem-solving process, the problem-solving methods used by the expert, and statements about the expert's cognition of the processes he/she is using.

This method offers the advantage of letting the knowledge engineer see the inferences made by the expert about objects, their relationships, and other information from the session transcripts or notes. It also has the advantage over task performance observation in that there is relatively no delay between the act of thinking and the recording of the act. In this manner, the knowledge engineer details the kinds of tasks the expert performs by noting (1) those tasks where verbalization is a natural part of the thought process or that verbal information is produced while someone makes inference on them, or (2) in identifying the features of the objects in the situation. However, this method is not applicable to all types of tasks. Tasks in which idiosyncratic language processes are used, are not well-defined enough to enable knowledge extraction. An example of an area where such would be the case would be in the arts (such as music). In addition, some tasks cannot be described by natural language descriptions. Domains which involve perception or motor tasks, do not lend themselves to these types for tasks. The knowledge engineer must also consider that in some type of tasks, the technique of verbalizing might distract the expert into subtasks which are normally used only as a side-consideration by the expert. In this case, the expert might be channeled down a path which deviates from the correct problem-solving reasoning path. Once obtained, the protocol analyses must be examined to determine if there are any changes in the attention focus used by the expert. Also, they are examined for identification of the kinds of objects which the expert sees in the process and the relationships and inferences that he/she sees between protocols.

INTERRUPTION ANALYSIS

When the knowledge engineer no longer understands the expert's thought processes, the engineer can use a technique called interruption analysis. The knowledge engineer then asks the expert to furnish details on what he/she did and why. By doing this, the engineer is trying to capture the specific momentary focus and inferences which the expert is using. This method is very instructive about the processes being observed at the moment. However, the drawback is that once the expert is interrupted, there is very little chance of getting the expert to continue at precisely the place he/she was prior to the interruption. Consequently, there is a propensity for the knowledge engineer to lose information/knowledge with this technique. Therefore, this technique should be used sparingly and only as a last resort. It is most applicable and offers the most promise if it is applied to the expert system prototype when comparing performance with the human expert.

DRAWING OF CLOSED CURVES

A specialized technique which can be used to indicate the relationship among objects which are in some physical space configuration is called the drawing of closed curves. With this technique, the expert is asked to indicate which collection of a set of objects go together, and to draw related objects in a closed curve. This technique is applicable to any spatial representation such as the mapping of the possible position locations on a game board. In effect, the knowledge engineer is looking for the aspect of the responses, the position currently displayed, and the order which matches a closed curve.

INFERENTIAL FLOW ANALYSIS

A technique which is a variant of the interview, is known as inferential flow analysis. This technique is ad hoc in nature, simple to use, and displays to the expert the aspects of expertise which have been uncovered at a certain time. It can also be used to stimulate the expert's mind on the subjects to be covered on future interviews. Typically, the expert is asked questions about the causal networks amplifying the objects of concepts in the domain of expertise. The knowledge engineer looks for a list of key objects of expertise and then asks the expert to cite any relationships which he/she sees between two seemingly related objects. The answers could reveal linkages among items which are linked in an inverse relationship and which may have another intervening variable. The answers should also uncover some consistency in the relationship between the intervening variables. Each time a specific item is mentioned in the expert's answer, it is linked with the other items in the answer with a relative link labeled either positive or negative. The linked items are then joined in an all-inclusive network of relations. Normally, the link is started with a standard weight and strengthened with each successive mention during the development of the network.

INDIRECT METHODS

The indirect methods for knowledge acquisition do not rely on the expert's ability to articulate the knowledge/information which he/she uses in the solution of problems in the domain. These methods exploit other types of behaviors within the domain. The bases for these methods lie on the recalled or scaled responses from which the knowledge engineer can make inferences about what the expert must have known in order to respond in the way he/she did. In some instances, the expert is completely unaware of the mental or knowledge processes which he/she uses to obtain some very complex answers. With these methods, the expert is not asked to express the knowledge directly but is given a variety of tasks. From the results, the knowledge engineer infers the underlying

structure among the objects rated. This technique makes different assumptions about the form of the understructure which may consist of physical spaces, lists, networks, or tables. The knowledge engineer must use only those methods from which the assumptions fit the expert's best guesses. This information is obtained through the initial inferences through careful questioning and noting of objects, names, and/or notations which the expert makes. The methods discussed below consist of (1) multi-dimensional scaling, (2) hierarchical clustering, (3) general weighted networks, (4) ordered trees from recall, and (5) repertory grid analysis. All these techniques have been validated in experimental studies and have shown psychological validity.

MULTI-DIMENSIONAL SCALING

A method which allows for similarity judgements to be made on all pairs of objects and concepts in the domain is referred to as multi-demensional scaling. This technique is easy to use but the interpretation of its results are not. The technique produces a layout of the items in space. From this, a similarity judgement matrix is derived where the objects are paired off and compared to each other to determine their relative similarity with each other. The matrix is then used as an input to an analysis program which searches for the best placement of the object in the user-defined dimensional space. Each dimension solution has an association factor measurement which is a total distance differential from the perfect fit association. The knowledge engineer then looks for solutions which have distance factors which are small and aggregates dimensions for the smaller factors. In doing this, less dimensions are plotted. The plots are then examined to judge the best placement of the axes and the appropriate labeling for them. These plots are then mapped as solution-to-solution-to-similarity matrix types. The advantage of this technique is that it can derive interesting clusters of objects, the distance factors for the closer lying objects, and the outliers. The difficulty with this technique is that it is a tedious process which requires multiple collections of pair-wise similarity judgements. It is also difficult to see the dimensionality with the smallest distance factor and then perceive the best placement of the axes and their names. The judgements are, for the most part, assumed to be symmetrical and to have a value from zero to one. This method is also good for producing a diagram which the expert can inspect and describe in more detail as the knowledge engineer asks more questions.

HIERARCHICAL CLUSTERING

A simplified, straight forward algorithm which consists of half-matrix similarity judgements is known as hierarchical clustering. The underlying assumptions for this algorithm are in direct contradiction to those for multi-dimensional scaling (multi-dimensional scaling assumes a symmetrical distribution of graded properties whereas hierarchical clustering assumes that the item is not a member of the cluster). The judgements are made as a function of the nested clusters which two items have in common or the height at which two items become members of the same category. To start the algorithm, the knowledge engineer initializes the half-matrix set by entering the distances between items. This results in a hierarchical representation of the items where the pairs of items closest in the matrix are joined and assumed to be members of the same cluster. Then a new matrix is drawn up with one cluster serving as a new item. The new matrix is then reexamined to find items which are closer together. Each time a new matrix is drawn, the inter-item distances between unclustered items are calculated to determine the distance factors (minimum distance of all cluster items to each item, and also the maximum and average distance factors). These values are then copied from the original matrix onto the new matrix. The advantage of using this technique is that it can be done through hand-calculations which are easy to follow. However, the procedure is tedious because you have to collect the items as in multi-dimensional scaling and without some valid justification for employing a particular joining algorithm, the enjoining process is somewhat arbitrary. The result is that you could derive different algorithms for the different hierarchies and the analysis is very subjective.

GENERAL WEIGHTED NETWORKS

Like with multi-dimensional scaling and clustering techniques, the general weighted network technique gives the knowledge engineer a way of extracting the expert's symmetrical distance judgements on all possible object pair combinations. The distance is assumed to arise from the expert traversing a network of associations where the network is one in which the single primary path between every two items and from some difference enclosed for the secondary path. From the distance matrix, the knowledge engineer derives the minimally connected network where each connection must involve closely connected or linked items. Then additional links are added to the tree and the structure is referred to as the minimally elaborated network. From that point, a link is added if and only if it is shorter than links currently serving the network between those two nodes. The minimally connected and minimally elaborated networks are then examined for any dominating concepts with large numbers of connections to many nodes and for items which are linked into circles. This technique can then be used to reveal significant aspects of expertise which should be encoded into the expert system.

ORDERED TREES FROM RECALL

A method which is used to investigate the memory organization of the expert is commonly referred to as ordered trees from recall. This method has been investigated to show the relative differences on memory organization between experts and novices in

a domain. The ordered trees begin with recall trials. This method assumes that objects either belong to a cluster or do not like in the hiarchical clustering. However, a difference is that it is built on a model of how the data is produced by a subject. This method also assumes that people recall all items which are from a stored cluster before recalling items from another cluster. The result is that the technique builds on data from recalled items from known or learned organizations. The similar patterns which are found to exist over a set of recall orders are then assumed to be indications of organized memory. The analysis of these patterns can be done manually but is extremely tedious and subject to perceptual error. Hence, computer analysis is best. This technique is used most in studies where expert and novice domain practitioner differences are being investigated. Experts usually show much more similarity in their organization than do novices. They also show more of a trend toward convergence in their methods. Once the ordered sequences or trees are obtained, the knowledge engineer must then examine them and determine what and how the expert perceives a situation resident within his/her domain. Reitman and Rueter (Reference 6) show good examples of how to apply this technique.

REPERTORY GRID ANALYSIS

A technique which is based on clinical psychology construct theory is known as repertory grid analysis. This technique has been used to develop several hundred prototype systems and is thought by some expert system developers to be the most complete technique. The technique is applicable to classification type problems where new problem observations and the objects are sorted into one known set of categories. An initial dialogue is established with the expert through an unstructured interview. The expert is then asked to name objects in the domain. Once this is done, the expert is asked to distinguish the traits between objects and to name the dimensions of the objects in his/her own words. The expert is then told to indicate the high and low values for the objects. The knowledge engineer then records the dimensions and scales provided by the expert for three items taken at a time (triples) until the major dimensions of the similar and dissimilar objects are uncovered. This forms a grid of objects and the associated dimensions. The knowledge engineer then asks the expert to fill in all the missing values. The initial dialog is followed by a rating session in which the expert makes inferences about relationships among the objects and the relatedness of the dimensions. Then the objects and dimensions are analyzed to look for object clustering and for the dimensions used to rate the items. This is done based on examination of the similarity of the dimensions. For the clustered objects, a matrix showing distances between objects is required. For the pairs of objects, the absolute differences between the object values are determined. For the dimensions, however, the determination of distance factors is not so straightforward. Some cases might show correlation but show opposite values. A system developed by Boose (Reference 7) allows for examination of clustered dimensions to determine correlation. Following this analysis, a second interview is conducted with the expert to determine the rules obtained from traits and dimensions found at this phase. These rules are then prepared for implementation into the prototype system. One advantage of this technique is that the procedures for developing similarity matrices are less tedious than by directing rating similarity in all pairs. Another advantage is that it allows for the combination of expertise from two experts with different specialization within the same field. Individual repertory grids can also be used as the basis for discussing and negotiating differences among the experts.

SYSTEM PROTOTYPING

It is important that the system be prototyped at the earliest time possible so that the rules obtained to date can be tested and modified, if necessary. The reason for this is that problems in the expe tem are not known until actual implementation because the system does not ha rete specifications at the start of the development effort. It should, however, ealized that major rework might be required after finding these problems. The prototype development efforts should concentrate on a single representative problem at a time and should involve the intended user at the earliest time possible. The advantages of doing this are that (1) the user can immediately find any major problems early in the program and before a considerable amount of work has been done, and (2) this also gives the user an ability to submit recommendations on how the system design can be made user-friendly. The prototyping phase is cyclical in that the development effort pace varies between the active subphases where new packages, tools, and interfaces are being incorporated and those times where the knowledge engineer is going back to the expert for more knowledge or information. When initially starting the prototype phase the knowledge engineer should (along with support of his/her management and the other members of the development team) determine whether the implementation technology for the prototype will be based using an off-the-shelf system shell or whether a special purpose system will have to be built. The first type offers the advantage of having the shell immediately ready for the incorporation of rules as they are developed. However, extreme care should be taken by the knowledge engineer to insure that the shell selected is oriented on rule programming. The second type offers the advantage of having a tailor-made system which is perhaps more conducive to producing the exact desired outputs. However, the disadvantages are that the cost will be considerably higher for both development and maintenance, it will take more time to develop and the shell might not be transportable to other applications and expert system development efforts. Additional factors which must be considered are whether the tailor-made shell development cost include support for programming and consultation by the developer. When using commerically available shells, the knowledge engineer needs to ask questions about the types of expert systems

that have been built before using that shell, and the types of support which were required and available from the vendor.

USER-INTERFACE DEVELOPMENT

Once the system has been prototyped, the first real test of its value is that it be user-friendly. Consequently, the knowledge engineer must factor this into the system design and guide the development effort toward that end. This process takes considerable time and should therefore be a front-end development item. It is important that the system be simple enough to use such that the user does not have to learn the whole system. The user should be required to know only the necessary inputs which he/she must provide to get the expected outputs. These factors make the system more acceptable and of more value to the user. By the same token, the system should readily allow for the knowledge engineer to modify or create new prototype packages with relative ease. Some helpful aids which might be incorporated into the prototype are goal browsers which lay out the design process as a network of different goals and displays the status of each goal when prompted. The browser allows for more detailed querying of the goal structure through menu interaction on the screen. It also allows the user to edit, undo, advise, and re-execute goals. This feature will allow the company managers to know the exact stage or phase at which the effort is on the milestone chart. This display is also updated by the system itself as the exploration of the design space continues. In addition, a separate interface design document resembling the knowledge document should be prepared and updated so that it includes any major design decisions and analyses conducted on the interface.

TESTING AND REDEFINITION

Immediately after the first version of the prototype is developed, the system should undergo intense testing. If these developments consist of submodules, the knowledge engineer, the expert, and any other intended users should start testing the system by using test cases to verify the rules contained. This process will inherently generate new problems which might give the expert system development effort new direction and indicate areas where more work (i.e. more and/or better knowledge) is required. The end result is that a second version prototype will usually be built as new knowledge structure and inference procedures are uncovered. In addition, the solving of real problems causes a reimplementation of continued knowledge programming or knowledge base maintenance.

KNOWLEDGE BASE MAINTENANCE

After the knowledge engineer and the prototype developing team have tested the system, the maintainers of the system must continue the process of updating the system as new rules/facts or changes are identified. This should be done at all user stations to help calibrate the user interface and extend the knowledge base. When this has been accomplished at all stations, it is then easier to evaluate the cost of the resources required as opposed to the value of solving new problems.

SUMMARY

This paper has presented some methodologies for acquiring or extracting the human expert knowledge via both direct and indirect techniques. Also included are some considerations for how to determine whether a domain is suitable for an expert system, what the knowledge engineer requirements are, and procedures for selecting the right expert knowledge via both direct and indirect techniques. Also included are some considerations for how to determine whether a domain is suitable for an expert system, what the knowledge engineer requirements are, and procedures for selecting the right expert or group of experts for development of an expert system.

REFERENCES

1. Shortliffe, E.H., Computer-Based Medical Consulations: MYCIN, (Elsevier, New York, 1976).
2. Van Melle, W., "A Domain-Independent System That Aids in Construction Knowledge-Based Consultation Programs", PhD. Dissertation, Report No. STAN-CS-820, Computer Science Department, Stanford University, 1980.
3. Miller, R., Pople, H., and Myers, J., INTERNIST-I. An experimental computer-based diagnostic consultant, New England J. Medicine, 307, 1982, 468-476.
4. McNeill, D. and Levy, E., Conceptual Representations in Language Activity and Gesture. In Speech, Place, and Action, eds. R. Jarvella and W. Klein, Wiley, New-York, 271-295.
5. Ericcson, K.A. and Simon, H.A., Protocol analysis: Verbal reports as data, MIT Press, Cambridge, Massachusets, 1984.
6. Olson, J.R. and Rueter, H.H., "Extracting expertise from expert: Methods for knowledge acquisition", Expert Systems, August 1987, 152-168.
7. Boose, J.H., "A Knowledge acquisition program for experts based on personal construct psycholcgy", International Journal of Man Machine Studies, 23, 1985, 495-525.

A Methodology for Producing Validated Real-Time Expert Systems

S.A.Cross
FS9, Royal Aerospace Establishment,
Farnborough, UK, GU14 6TD.

M.Grisoni
Logica Cambridge Ltd, 104, Hills Rd,
Cambridge, UK, CB2 1LQ.

Summary

VORTEX is an experimental methodology for building validated expert systems. It considers validation to be an exercise in building confidence in a system in the users, procurers, experts and developers. It identifies the components of validation concerning each participant and techniques for achieving validation and embeds them in a life-cycle suitable for a novel technology. This paper presents the essential points of the methodology and some experiences from the airborne ASW application developed in parallel with it.

1. Introduction

The move towards more technologically complex mission systems, for coping with increasing mission sophistication, coupled with increasing limits on available manpower resources is leading to increased crew workload and more lengthy training of operators for many existing and future aircraft. RAE have identified that expert systems technology may relieve this problem by enabling the development of decision support aids to reduce workload and improve operator performance. However, before this technology can be used operationally MOD need to have confidence in its procurement, use and maintenance. Validation is the basis on which confidence in a system is built.

The validation of expert systems, though, remains a complex issue. Through VORTEX (Validation Of Real Time EXpert systems) RAE has developed a methodology that consists of a description of the development process from initial idea through to operational use, and a design approach, with tool support, that facilitates this task. The development process has some common aspects with the life-cycle for any new technology. For example, it emphasises phased development and prototyping as a way of establishing both requirements and confidence. However, expert systems have their own unique aspects which cause problems for validation, notably the sometimes individualistic and heuristic knowledge contained in them and the inherent complexity of the tasks they address, and are recognised to require expert ability to solve. This means that the correctness of a generated solution is necessarily based on expert opinion and that an initial specification, against which to validate, is extremely difficult to produce. These expert system-specific problems are not insurmountable but do require commitment from an expert to the validation task.

It is not just the expert who is concerned with validation. Issues such as user-friendliness and timeliness of response are important concerns of the user, while the developer assigns priority to maintenance and reusability and the procurer is concerned with operational impact. All these aspects are encompassed by validation and each should be considered in a fully validated system.

However, there may be some confusion about the meanings of these terms. For instance, timeliness could be considered an element of user-friendliness. The ESPRIT project "VALID, Quality and Metrics" [1] itemises the criteria for quality software from the developers' point of view. It includes items such as clarity, coherence and generality, without asserting their independence. The authors also describe components which are the quality factors from the users' point of view, including the items mentioned above such as correctness and performance. The users' factors are built up from the developers' criteria forming a two-tier model, indicating that a number of criteria are related in that they contribute to a common user factor.

VORTEX draws loosely on this model by building another tier on top of the "user factors", where the units of the new tier are the various participants in an expert system life-cycle, and the links to the (extended) set of "user factors" indicate their different perspectives on the validation task as a whole. The other main influences on the VORTEX methodology are listed in the bibliography.

In this way, there is a fusion of life-cycle, participants, validation components (such as expert understandable knowledge) and techniques (such as expert understandable knowledge) attempting to answer the

fundamental questions of when, who, what and how to validate for an expert system.

2. Application Details

The expert system methodology resulting from VORTEX was based on actual experience of building an expert system demonstrator. This expert system, called the Sensor Adviser, needed to be both a plausible application for improving mission performance and sufficiently constrained for the purposes of investigating and developing an expert system methodology. It was originally intended to consider Anti-Submarine Warfare (ASW) data fusion but after applying the techniques described in the initiation phase the choice moved to a decision aid for Merlin Observers for sensor selection and deployment in ASW.

Sensor selection chooses the best sensors from the Merlin sensor suite to deploy in a particular tactical situation, whilst sensor deployment chooses the best way to use the sensor. Merlin is the name given to EH101 helicopters planned as the aircraft for Invincible-class ships. It has a long range and good endurance, and is equipped with sonobuoys, active-dipping sonar (ADS), radar and magnetic anomaly detector (MAD). Operating autonomously in its ASW role, the crew consists of a pilot in the forward cockpit, and an Observer and an acoustics system operator in the rear cockpit.

Observers are likely to have spent two years in the Navy either airborne or with some command responsibility on-board a ship, followed by 18 months training in ASW tactics, followed by three to four years of practical experience. They are responsible for the mission, but need to negotiate with the pilot over tactics and risks, particularly where these represent a threat to the aircraft. The Demonstrator aims to increase the effectiveness of Merlin Observers on sorties. It demonstrates how its operational equivalent could substantially assist Observers in achieving their mission objectives, by improving performance and smoothing workload.

The Demonstrator improves performance by helping an Observer to:

- consider all relevant factors in the selection and deployment of available sensors
- avoid narrow-thinking, and so reduce the risk of making poor decisions
- consider overlooked options at times when it is not clear what subsequent actions to take, such as losing contact with a submarine that was previously being tracked

An Observer's workload can vary considerably during a sortie. For example, workload is often low when flying to a barrier and excessive when prosecuting a target. The Demonstrator helps to even out these wide variations. When workload is low the Observer can use the Demonstrator to explore alternative tactics and prepare for the mission. When workload is high the Observer can use the Demonstrator for advice on sensor selection and deployment.

3. The Sensor Adviser

The VORTEX Demonstrator is written in Common Lisp with Flavours to run on a Symbolics 3650 machine. It consists of a mouse driven graphical interface, a set of validation tools, an inference engine, a Knowledge Base Manager (KBM) and the KB itself. These are described below in sufficient detail to understand illustrative examples which will be raised during presentation of the methodology. Figure 1 gives a block diagram of the relationship of the various parts.

3.1. Graphical Interface

In order to aid both user and expert understanding of the proposed system, the interface has been designed to simulate that which will be available to Merlin Observers. One window simulates Merlin's Common Control Unit (CCU) using mouse sensitive buttons. The Observer interacts with the system through the CCU, navigating short menu paths which may require input from the CCU keypad. The alert key allows the Observer to interrupt the Adviser and provide unsolicited information which can change the current line of reasoning. The Sensor Adviser outputs its advice on a simulation of the Common Display Unit (CDU), together with a concise explanation of the rationale for this advice. It is intended that this output would be imposed on Merlin's tactical plot.

Additional windows are used for demonstration and development purposes. One window provides a detailed trace of the reasoning being undertaken by the Sensor Adviser's inference engine. Another controls the scenario and provides access to both the internal state of the Sensor Adviser and to a suite of validation tools. It enables system level interrupts to allow user input of information which would normally be available from Merlin's mission system, such as sonobuoy detections. These windows would not be present on board Merlin but might exist in an operational support station. Their purpose is to build the confidence experts and

developers have in the Sensor Adviser and, therefore, they form part of the validation tool support.

3.2. Knowledge Base

It is important that the knowledge incorporated in an expert system be easily understood by experts if they are to fulfil their validation responsibilities. Although individual production rules may be understood, their interaction is much harder to grasp. Moreover, implementational constraints may compound this problem

The Sensor Adviser knowledge base consists of two kinds of knowledge:

- domain knowledge such as threat, sensor and weapon data
- problem-solving knowledge, ASW tactics and their application

Domain knowledge is organised in the familiar object hierarchy with inheritance, so that a MAD is a type of sensor, which is a type of object, and so on. Problem solving knowledge is organised in a procedural form which reflects the way experts decide on the best tactics for a given scenario. Tasks are therefore linked with key decisions made by experts. For example the task *go-passive* is linked to the decision variable *active-versus-passive*. Tasks may depend on sub-tasks which are necessary in order to perform the parent task. For example, *go-passive* is a sub-task of *decide-active-versus-passive*. The latter will reason about such things as water conditions while the former will actually update the decision variable. Tasks are also represented as objects in the object hierarchy.

The internal reasoning within a procedural task is actually implemented using declarative production rules. The above problem with understanding the interactions of rules is removed since they are all constrained to work on one task, and it is the interaction of the tasks which trace the flow of reasoning, and this is much more natural for the expert. In this way knowledge base modularity is achieved and the natural expressiveness of production rules is retained without the associated disadvantage of complexity of interaction. Of course, from a wider perspective, different tasks could have different problem solving methods within them, including algorithmic techniques, but validation requirements have prompted the use of expert understandable production rules.

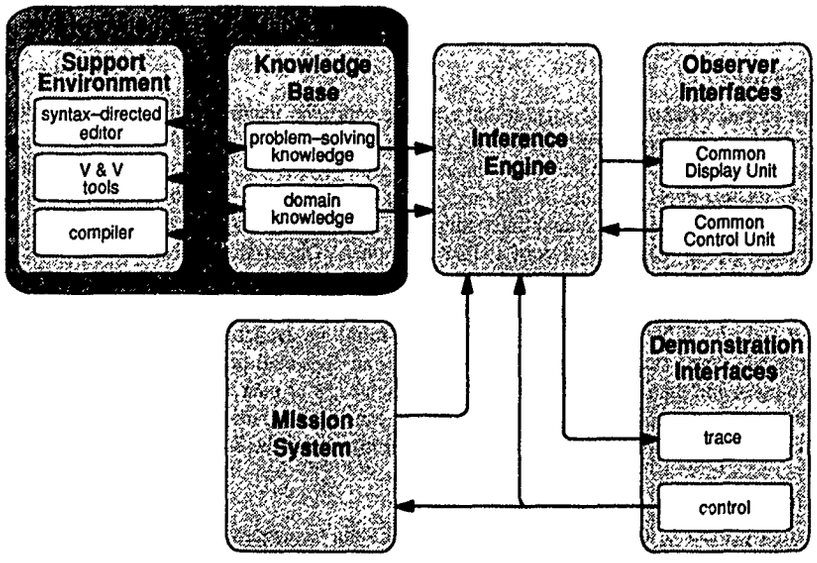


Figure 1: Sensor Adviser Architecture

3.3. Inference Engine

The inference engine is the piece of software which systematically applies the knowledge in the Sensor Adviser. It is therefore conventional software and is suitable for verification and validation using conventional techniques. VORTEX concentrates on that part of the validation process which is special to expert systems and so does not consider validation of inference engine code.

The operation of the inference engine is dependent on the knowledge representation so a brief description of its operation is given here for completeness and to aid understanding of the knowledge representation.

The inference engine maintains a schedule of tasks for processing in order. These tasks may schedule sub-tasks, which are inserted in the schedule before the task following the parent task. Completed tasks are not removed from the schedule, but remain as a record of the reasoning process. This record is necessary not only for validation purposes, but during execution when interrupts may arrive from the user, in the form of unsolicited information or from the (simulated) mission system. These interrupts may invalidate previous reasoning, eg a change of expected threat type. This will lead to the status of some decision variable moving from "decided" to "undecided". Since tasks are connected to decision variables, reasoning will need to be backtracked to the last point before the modified decision variable was used, by truncating the schedule at the appropriate task. This then becomes the current task and reasoning resumes.

3.4. Knowledge Base Manager

Any changes to the knowledge base required during reasoning are performed through the Knowledge Base Mangaer (KBM). It performs limited verification by the use of "strong typing". For example, a variable such as *sea-state* is a numerical variable with cardinality one and range 0-9. If this is updated during a mission, the KBM will prevent anything other than a single number in the specified range becoming the new value. Strong typing also applies to all objects and their attributes in the KB. Note that the constraints do not have to be language primitives, e.g. the variable recording expected threat must have a value which is actually some submarine from the domain knowledge.

The KBM also performs the appropriate housekeeping functions when backtracking of reasoning occurs. The inference engine only records the schedule of tasks and schedules new tasks. It is the KBM which determines the effects of external interrupts, deciding which decision variables are affected and how far back to undo reasoning. The KBM operates both at run-time and compile-time, so it also provides verification support when developing and maintaining the KB. Strong typing and tracing tasks which depend upon a modified object detects a very large number of problems which would otherwise be extremely expensive to track down.

3.5. Tool Support

Experience and common sense suggest that effective validation requires tool support. The VORTEX project has, therefore, developed a number of tools dealing with the validation and verification issues specific to expert systems. The KBM described above includes verification tools. The trace and explanation facilities mentioned in the graphical interface section are validation tools. A number of other tools bridge the gap between assessing specific advice and inspecting the incorporated knowledge. These are the Task-Task Identifier, Task-Object Identifier, Decision-Task Identifier, Object-Task Identifier and Task-Circularity Identifier. The first four produce graphical displays indicating a specific item and the items connected with it. For example, the Task-Object Identifier displays all objects read from and written to by a particular task. More details direct from the KB (and thus in expert readable form) are available by mouse selection of any of the objects displayed. The Task-Circularity Identifier locates any potential problems caused by a task that could schedule sub-tasks and then itself be scheduled by the sub-tasks or their descendants. The circularities might not occur at run-time due to the run-time conditions not causing that particular reasoning path to be followed. However, it is necessary to know where they *might* occur to validate the KB.

4. Validation and Verification Techniques

There are a number of different validation and verification techniques which address the special issues of expert systems:

Application assessment - we have identified a number of criteria for assessing the suitability of expert-system technology

the existence of human experts
the demand for expertise is too great for the available supply
there is a need to improve the consistency of operational decision making
there is current planning for future unavailability of expertise

the decision making quality is key to a task
the area of the task requiring expertise is well defined and can be packaged practically
the user can share in the decision making process
the amount of user input and output is not onerous
there is an existing or potential mechanism for maintaining 'best practice'
the users' trust can in principle be established
the envisaged system supports rather than replaces the user's decision making process
the reasoning experts employ is explicit rather then perceptual
there is access to experts who either do the job or have done it recently
uncertainties in the problem-solving process are handled in a well-defined and meaningful way
the necessary 'input' and 'output' to the problem-solving process is well understood
the "radio-link" test works [this test requires the expert to enquire about situations as if over a radio; the development team then suggest responses for criticism and are thus able to establish an understanding of the decisions made and the factors required to make them]
the vocabulary and form of the knowledge statements that comprise the knowledge base can be made familiar to the expert and to the users
the development team is experienced in expert-system development
early review and prototyping points are established, and feedback from experts and users is encouraged
that real-time and spatial considerations in the problem are handled in a way which is understood

Test cases - These are by far the most useful validation techniques that we have used in VORTEX. Different test cases can be used for different validation elements. However, a single test case gives only partial confidence in the incorporated expertise. Typically expert systems have an extremely large number of equivalence classes, and so exhaustive testing is inappropriate.

Panel of taste - This is a mechanism favoured by those participating in our validation experiments for establishing the "right" answers to questions uncovered with test cases is to establish an authorized "panel of taste" to arbitrate on what knowledge is correct.

Subjective judgement - many of the aspects of developing expert systems discussed elsewhere in this report conspire to make subjective judgement much more important than is apparently the case in more conventional technologies.

Asking the right questions at the right time - the weaknesses of a subjective judgement approach can be significantly ameliorated if the right questions are asked at the right time. This gives the maximum chance of detecting problems early, when they are much cheaper to fix. Our results-oriented life-cycle and validation approach are designed to achieve this end.

Tool Support - Developing and employing V & V tools is an important technique: tools fall into a number of different categories:

- Type-checking verification tools, which can operate at both compile time and run time. These catch many "clerical" mistakes, which would otherwise be very expensive to track down.
- Structured editors, which ensure that only syntactically acceptable knowledge statements can be entered into the system. Again, by catching such errors at the syntactic level, much effort can be avoided.
- Completeness tools, which verify that no referenced items of knowledge are duplicated or omitted.
- Explanation tools, which build confidence in users that the answers will meet the required level of competence.
- Trace tools, which build confidence in experts that the knowledge is correct, in that correct answers are being generated by appropriate lines of reasoning.
- Inspection tools, which build confidence that the knowledge is both correct and complete, by allowing the expert to inspect the knowledge base flexibly, and follow relationships between knowledge fragments.
- Consistency tools, which assist validation, for example by ensuring that all dependant elements of problem-solving knowledge are re-examined when a statement of domain knowledge is modified.

In the VORTEX project, versions of these tools have been incorporated in the KBM.

5. Life-Cycle

This section describes the VORTEX methodology in detail. It presents the Initiation, Feasibility, Development, Delivery and Operation phases, identifying the outputs and the most important validation components of each phase. For each component, the appropriate participants and their perspectives are identified and an indication of the techniques employed to validate the components is given.

5.1. Initiation Phase

The first stage is the idea that some aspect of a requirement may be met by an expert system, either through apparent relevance of the technology or through belief that conventional systems cannot do the job. Neither is a guarantee of applicability or success. The assessment indicators described above may be used for this purpose.

An initial specification of all aspects of the expert system should be produced. It provides the first opportunity for validation but it must be accepted that for expert systems the specification will evolve during the life cycle because the technology is new, so procurers cannot understand it well, and because the nature of expertise is such that it is difficult to specify and hard to manipulate by analogy. For expert systems in general, testing the system is a fundamental part of elicitation, and hence development, and so leads to modifications of the KB, which has an effect on the specification. As a case in point, the Sensor Adviser was originally only intended to make the decision to go active or stay passive. After a number of elicitation sessions it became apparent that both sensor selection and deployment advice should be provided.

The procurer is interested in the impact of the system, ie the observable benefit. This requires the production of a statement of impact which should address the following questions:

- What externally observable measure of performance will be improved through deploying the expert system? For the Sensor Adviser, examples include reducing time to weapon in the water or reducing the number of failed prosecutions.
- Who will achieve the improved measure of performance by using the expert system? E.g. novice, average, expert and fatigued crews.
- When will the improved measure of performance be achieved? E.g. against modern, unco-operative, multiple and expert threats.

A key insight from the VORTEX validation experiments is that expert system decision support tools may often yield benefit from catching mistakes and preventing errors rather than from systematically outperforming crews. This should influence the way that the statement of impact expresses the observable benefits.

The validation components associated with this phase are described below, together with those concerned with them. The output of this phase is paper based and so the only technique that can be used is eliciting subjective opinion on the proposals made.

Impact

The statement of impact is usually elicited by developers from procurers, perhaps with the assistance of experts. Experienced developers should seek to reduce over-ambitious requirements, but all concerned should realise this will probably only be a target statement.

Application assessment

The developers apply the indicators of section four to initial discussions with experts. For the Sensor Adviser the radio link test was found to be particularly useful. For instance, two initially suggested applications were improving tactical display quality to show only "interesting and relevant" information, and automatic alert of mission critical situations which require an Observer's immediate attention. Both were eliminated since they are highly perceptual tasks that are difficult to analyse in the manner required for the radio-link test. Expert systems cannot model perceptual tasks. On the other hand, the spatial information required for the Sensor Adviser which might be considered perceptual in nature can be verbalised in terms of courses of threats and standard patterns of buoy development.

Knowledge Availability

This can be a problem when the procurer views expert system techniques as a panacea. Developers must have access to knowledge to build the KB. Domain knowledge such as ship characteristics will often be available in reports and manuals. Problem solving knowledge may be available from tactics manuals but expert level knowledge will rarely be documented. Commitment from experts is therefore vital for success.

Functionality

The user needs to be satisfied that the system will provide all functions required. Functionality is considered to include capabilities, explanations, changes of mind, division of labour and what-if questions. During this phase, the developer seeks confirmation of valuable and workable functionality. For the

Sensor Adviser, experts were consulted since Merlin is a future system and there are no users currently.

Integration

The plausibility of integrating an expert system with the anticipated platform requires the involvement of expert system developers and those who maintain the platform.

5.2. Feasibility Phase

The output of this phase is a feasibility prototype and a feasibility report. Since tangible software is being produced, the subjective speculations of the initiation phase become mostly-subjective evaluations of a real artifact. This is a major improvement. It allows use of the techniques of inspection and test cases and is important for developing the knowledge representation, which has been identified as a key element in validation. The original proposal for VORTEX omitted the feasibility prototype on the grounds of cost but one was built since it was discovered to be so important.

However, it is important to remember that the feasibility prototype is just a prototype. As such it is a tool for conducting experiments that will lead to a final system. It is not the final system itself, even though elements of it may be retained. Prototyping is an experimental activity and, like all experiments, should be well planned before commencing, with an expected use for the results. In the expert system context, the results are typically used to improve knowledge representation and inference, with the goal of obtaining better solutions for the current test cases. Planning prevents prototyping becoming hacking.

The validation components associated with this phase are described below.

Confidence

Experts want to know their expertise can be captured, procurers that the statement of impact can be met, and developers that the technical approach is appropriate. Confidence is a subjective matter. The feasibility prototype provides procurers with a demonstration of the feasibility of expert system technology applied to the application, experts with advice from test cases and an expression of their expertise, and developers with feedback from the participants involved in the development process. There is, therefore, tangible evidence to aid confidence building.

Cost

Developers will be able to revise cost estimates from building the prototype. This leads to a decision point for the procurers about whether the anticipated benefits warrant the cost.

Competence

At this stage it is a matter of expert opinion as to whether the competence displayed by the limited feasibility prototype is likely to support the anticipated operational impact. This opinion should be explicitly sought and recorded.

Human Computer Interaction

If the prototype environment is sufficiently realistic, the users should evaluate the proposed HCI. Otherwise, the judgement is made on the basis of general HCI considerations, such as Schneidermann's work on direct manipulation, and on expert opinion about the operational task. With the Sensor Adviser, the CCU and CDU proposed for Merlin were emulated as described above. This meant that users could evaluate the HCI (although for Merlin the experts are in effect the only users). Initially, the system was designed to ask for any missing information when advice was requested. This proved unpopular so the format was changed to provide advice using defaults, while making observations that information was missing and allowing the user to provide it as unsolicited information.

Timeliness

The user is concerned that advice is generated at the right time. As well as speed of response, appropriateness of advice is important. By this stage, the analysis of the application should have identified any areas of time criticality. In tactical decision aids, these are likely to be relatively well contained, as was the case with the Sensor Adviser, but they should be addressed in the prototype. Selectiveness of advice is a problem for knowledge elicitation. In fact, VORTEX considers solutions to response time problems to be a knowledge elicitation task also. It makes sense to determine how the expert responds to time pressure and to emulate these methods, rather than to attempt to deal with the problem in some automatic manner. The task structure described above allows different problem solving methods to be used as

required.

Scope

The user is concerned with the breadth of applicability of the expert system. This will be expressed in terms of scenarios that can be addressed. The scenarios will form the basis of the test cases and are therefore selected by the expert to cover the different categories that must be addressed and the differing complexities. Scenarios which do not need to be considered may also be specified. The user then needs to know the range of scenarios, how competence varies over the range and how performance degrades at the boundaries. The feasibility prototype only deals with some of the test cases but the results of these should be used to provide answers to the users' requirements. In terms of ASW, scenarios will be specified by a particular threat in a particular theatre of operations, such as the North Atlantic, with details of weather and threat behaviour and so on. The scope at this stage also needs to be validated by both experts and users against the anticipated impact.

Knowledge

At this stage the expert is concerned with whether the knowledge required to achieve the anticipated impact can be incorporated, rather than trying to validate the limited knowledge in the prototype. The developers are concerned that the representation and inference facilities are adequate for the final system. The former decision comes from the test case results and the latter from feedback from the expert on these results. For the Sensor Adviser, the inclusion of decision variables (see above) aided expert understanding and allowed dependancy-directed backtracking of reasoning to be conveniently implemented. These were missing in the prototype, which highlighted the problem.

Integration

Having derived an integration strategy, the developers must validate its feasibility within the context of the known requirements for the expert system module for the target operational platform. The Sensor Adviser is written in Common Lisp. This would be easier to translate to Ada for military operational use than if it had been developed using a toolkit such as KEE or ART.

5.3. Development Phase

Development of the expert system can be divided into a conventional part, the infra-structure which includes the inference engine, and an expert-specific part, the knowledge representation and the knowledge base. The infra-structure is highly dependant on the knowledge representation chosen and configuration management of the two is a practical issue of some importance. However, it is a conventional project management problem and so is not covered here. The Sensor Adviser infra-structure provides the interface, KBM, Inference Engine and validation tools and is described in section three.

Development of the KB continues throughout the life-cycle. During delivery, (which covers extensive user trials) the close relationship between testing and acquisition will inevitably spawn new knowledge to be included as part of the maintenance phase. As tactics change with improvements in sensor capability or threat characteristics and behaviour, so must the knowledge base be maintained. Disposal of the system occurs when the benefit it delivers outweighs the cost of its maintenance. A common mistake during initial development is to perform too much domain modelling with no clear idea about how the problem-solving knowledge will utilise it. Usually it is better to be driven by the problem-solving knowledge and acquire domain knowledge as required. This is based on the assumption that for validatable decision support systems it is better to adopt the procedural approach to knowledge representation discussed in section three.

In acquiring knowledge, the following techniques have been found to be most useful:

- The radio-link test - good at getting experts to articulate their problem-solving approach and determining appropriate interaction with system.
- Knowledge animation - illustrating the consequences of knowledge statements aids the expert in spotting shortcomings and suggesting extensions and qualifications.
- Constructed test cases - varying test cases to identify boundary conditions between different solutions and methods.

Inspection of the knowledge base is a further potential means of knowledge acquisition, but is time intensive on the expert and so has not been extensively used by the VORTEX project.

The output of this phase is the final system for delivery. The components of validation associated with the development phase are described below.

Impact

By the end of the development phase, the validation of the target statement of impact will be to a much greater degree of confidence because, while still dependant on the judgement of the expert, the procurers will be able to see the results for a full range of example scenarios.

Confidence

All the critical expert system issues will be resolved so confidence of procurers and experts will be high. The remaining areas of uncertainty concern integration and the accuracy of projected operational conditions compared to the real world.

Cost

The technology related cost uncertainties will be largely resolved and so validation of cost will be reliable. The procurers carry out this validation with information from developers.

Competence

The expert system now embodies the competence statement. The advice generated for all specified scenarios defines the competence achieved, which the user validates (drawing on expert opinion and panel of taste) against the target competence.

HCI

The HCI is fully defined and should be validated by users against the operational environment in which it will operate.

Timeliness

The full range of scenarios are available for selection of time critical test cases by the expert for validation by the user, using developers' projections of relative performance in the operational environment.

Functionality

Now fully defined to be validated by the user against the specification.

Scope

All test scenarios can be validated against the specification by users. However, the users and expert must also consider the flexibility to deal with future developments in the problem domain.

Knowledge

By the end of development, validation by the experts of the incorporated knowledge will depend upon a number of factors:

- verification, primarily through the use of tools, of typing (compile and run time), syntax and completeness of the knowledge base
- inspection of KB listings, both by experts involved in the development and by other experts
- assessment of the advice generated in scenarios selected by experts to probe completeness and correctness of the knowledge
- browsing, using tools, of knowledge related to issues arising from and suggested by the test cases.

The developers will now have full confidence that appropriate facilities exist for the initial KB release. They must also validate the appropriateness of the facilities against an assessment of future needs as external requirements evolve. Validation of the software constituting the knowledge representation and inference facilities is addressed by the developers in a conventional manner.

Integration

At this stage a detailed integration plan will exist and the developers will validate this against the requirement.

In practice, military systems progress through the development stage using increasingly realistic world models which include paper studies, laboratory demonstators, simulators and experimental aircraft. This progression increases the complexity and scope of the validation process, but to compensate for this there are fewer artificial barriers distracting the the expert from the knowledge and its validation. The specification and

confidence building also develop in parallel with this progression.

VORTEX is currently at the laboratory demonstator stage. A number of validation experiments have been carried out using four experts other than the main expert used for most of the project, each with slightly different orientations on the ASW domain, e.g. training as opposed to operation. The experts followed the same set of scenarios, and were asked to say what they would do at each stage, just before the Sensor Adviser gave its advice. If there were discrepancies the experts were asked to make their subsequent decisions assuming that they had followed the Sensor Adviser's advice, to ensure consistency over the experiments. The discrepancies and the reasoning behind them were recorded. This meant that knowledge was elicited, indicating the close relationship between testing and elicitation.

These experiments contributed to confidence building, competence assessment, scope and HCI component validation, by identifying areas for further development. The knowledge representation, timeliness and functionality component reqirements were demonstrated to be satisfied for the current scope. Future work is intended to address the knowledge elicited during the experiments. This will expand the scope to deal with more realistic scenarios. The HCI will also be developed to produce a graphical display of buoy deployment as all experts found it difficult to keep track of the textual advice currently provided. The work is intended to remain as a laboratory demonstrator, but once the new scope has been validated and there is sufficient competence across it, there should be enough confidence to move to an integation task with a simulator.

5.4. Delivery and Operation Phases

The major activity of delivery is integration with the platform hardware and software, a conventional engineering exercise, and validation of the system using the real environment. This latter activity will progress into the Operation phase. The applicable validation components are confidence, cost, HCI, timeliness, knowledge and integration. At delivery, these are all nearly complete, but only after significant operational use can full validation be achieved due to actual working practice evolving with exposure to the system, the changing nature of threats and requirements and the actual operational impact.

VORTEX obviously has no direct experience of these activities. However, the progressively realistic development environment from laboratory to simulator to experimental aircraft will provide "mini" - delivery and operations phases within the overall development phase. The mini delivery phase for the first phase of the Sensor Adviser development is described above. The Sensor Adviser has been widely demonstrated since then and this has reinforced the current adequacy of the knowledge representation and timeliness, as well as the need for further development of the scope and the HCI. This could be viewed as a mini-operation phase.

6. Conclusions

VORTEX is a project to develop a methodology for building validated real-time decision aiding expert systems for military operational use. It is still under development but, even if completed, it could not hope to provide a foolproof step-by-step guide to building complex systems. Instead it seeks to provide a framework for thinking about and discussing the validation issues and provides checks and guidance for those interested in the development of expert system modules.

It does this by identifying the participants in the validation activity, the components of validation, some useful techniques for performing validation and a suitable life-cycle. It attempts to fuse these together by indicating the various participants' perspectives on the components at each point within the life-cycle and the appropriate techniques to apply. This is intended to answer the questions:

- what should be validated?
- who should do the validation?
- when should validation take place?
- how should validation be performed?

It is accepted that the answers to the last question are incomplete, although the framework presented here may help others to discover appropriate techniques for their particular applications. However, the methods that we consider to be generally applicable and of great importance are

- use an application specific knowledge representation that incorporates vocabulary and structure familiar to application experts
- build a feasibility demonstrator to execute elicited knowledge as early in the development life-cycle as possible
- elicit and use test cases which test the boundaries of the expert system

- develop tools that support validation and verification.

Knowledge base inspection by the expert is also expected to prove useful, although little practical experience of this has been gained.

This methodology has been developed in conjunction with the development of an ASW decision aid for sensor selection and deployment. Future work seeks to concentrate on validating the KB of this application, which will itself be increased in complexity to develop the methodology for a more realistic expert system.

BIBLIOGRAPHY

[1] ESPRIT II, 2148, VALID, State of the Art, 1989.

[2] Ackerman A, Buchwald L, Lewski F, Software Inspections: An effective verification process, IEEE Software, May 1989, 31-36.

[3] Boehm B W, Characteristics of software quality, North-Holland, 1978.

[4] Boehm B W, A spiral model of software development and enhancement, Computer, May 1988.

[5] Cupello J, Mishelevich D, Managing prototype knowledge/expert system projects, Comms of ACM, 31, 5, 1988, 534-541.

[6] Garg-Janardan C, Salvendy G, A structured knowledge elicitation methodology for building expert systems, Intl Journal Man-Machine Studies, 29, 1988, 377-406.

[7] Ginsberg A, Knowledge base reduction: a new approach to checking knowledge bases for inconsistency and redundancy, Proceedings AAAI 88, Vol 2, 585-589.

[8] Muir B, Trust between humans and machines and the design of decision aids, Intl Journal Man-Machine Studies, 27, 1987, 527-539.

[9] Nazareth D, Issues in the verification of knowledge in rule-based systems, Intl Journal Man-Machine Studies, 30, 1989, 255-271.

[10] Parnas D, Clements P, A rational design process: how and why to fake it, IEEE Trans SE-12, 2, 1986, 251-257.

[11] Reason J, Cognitive aids in process environments: prostheses or tools?, Intl Journal Man-Machine Studies, 27, 1987, 463-470.

[12] Wallace D, Fujii R, Verification and validation: techniques to assure reliability, IEEE Software, May 1989, 10-17.

[13] Weitzel J, Kerschberg L, Developing knowledge based systems: reorganising the system development life-cycle, Comms of ACM, 32, 4, 1989, 482-488.

A REVIEW OF SOME ASPECTS ON DESIGNING FUZZY CONTROLLERS

E. Trillas*, M. Delgado**, J.L. Verdegay**, M.A. Vila**

* Instituto Nacional de Tecnica Aeroespacial
Ministerio de Defensa
Torrejon de Ardoz
Madrid (España)

** Departamento de Ciencias de la Computacion
e Inteligencia Artificial
Facultad de Ciencias
18071 Granada (España)

SUMMARY

The paper focus both theoretical and practical aspects of the fuzzy control systems according to the following scope. First, foundations of the fuzzy controllers, and the different ways for implementing them, are described. Second, we concentrate ourselves in the management of the information, that is, the way in which the inferences are made from the expert's knowledge. Usually this is carried out by means of the Generalized Modus Ponens for which, the so called implication function is the main tool to handle it. Hence, as depend on the selected type of implication one has a different version of inference, finally, the possible implications functions and the consequences from its use are analyzed and discussed.

INTRODUCTION

The automatic control of complex industrial processes is very difficult. The difficulty arises from non-linearities, the time-dependent and/or ill-defined behavior of systems and the low quality of available performance measures. In most real cases the automatic control is carried out through those (subsidiary) variables that accept measurement and control, whichever them may be. In such situations only partial goals may be forecasted by the control model and the achievement of the global objectives are left to the human-expert-operators.

Thus in a lot of industrial processes the human operators work better than the automatic control systems. Their high performance lies on the experience. Like in other fields, the only solution to design and construct more performing automatic control systems, it seems to be the use of knowledge engineering to elucidate and formalize the control strategies, which the human-expert-operators often know and justify from an empirical point of view.

Even for well defined systems, usually the experts explain their control strategies in a linguistic way, giving a set of heuristic decision rules on the time. Obviously is a very difficult (even impossible) task to translate such linguistic laws into a set of numerical ones without loss of important pieces of knowledge.

Obviously the problem increases when the nature or the behavior of the system is ill-defined or ill-known. Thus, studying direct implementations of linguistic control rules is very interesting from both a theoretical and practical point of view.

The Theory of Fuzzy Sets (introduced by L.A. Zadeh [29]), and more particularly the developments on Fuzzy Reasoning, provides tools to represent and handle linguistic and/or vague statement (for instance rules) in an automatic manageable formulation. The use of that tools in the analysis of systems, starts with the Zadeh's works about fuzzy algorithms and linguistic analysis [30] and [32].

When expert's heuristic control rules are translated to a fuzzy formulation, that set constitutes a fuzzy algorithm (linguistic rather than numeric [32]) which, in this case, may be defined as (logical) fuzzy controller. Fuzzy control research was started from Mamdani's pioneer work [8]. Since then, this new approach for analyzing and controlling complex systems has received a growing attention.

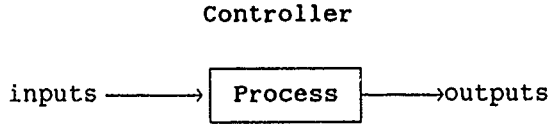
Two different (but complementary) trends are traditionally distinguished in Fuzzy Control,

1.- **Analysis:** To describe and model the (fuzzy) behavior of a system under specific (fuzzy) constraints or (fuzzy) control conditions.

2.- **Design:** To obtain a fuzzy algorithm being able to control the (fuzzy) behavior of an specified (fuzzy) system.

Historically, the first works about analysis deal with constructing suitable theoretic models, as well as defining appropriated performance measures. As the researchers on fuzzy control arrived at from classical Control Theory, the earliest approaches extend to the new context the concepts of state, controllability and stability, giving fittest versions to the Optimal Control Problem.

Further works attempt to generalize some aspects of the classical control models. Obviously controlled fuzzy systems may be outlined, like the non-fuzzy one, according to the diagram in Fig. 1,



(Fig. 1)

Usually the system's outputs produce non-fuzzy inputs for the controller, these signals being obtained by means of appropriated sensors. In its turn, the controller responses need to be changed into non fuzzy signals, and then they are inputs to the system.

Two areas of research may be distinguished according to the place of the interfaces between the process and its controller. When the interfaces are inside the controller, then this is just a kind of discrete look-up table. The cases in which the interfaces are into the process correspond to truly fuzzy controllers, and present more

interesting properties than the former ones (see [20]).

Although works about analysis were largely motivated by practical problems, they are quite theoretic themselves. The design trend however has a more practical character. It began with Mamdani's work [8], and its main aim is to develop a methodology to specify and model fuzzy control rules, in order to construct and implement effective fuzzy controllers.

The remain of the paper is mainly devoted to different aspects of the design, paying an special attention to the models of Approximate Reasoning which underlies any fuzzy algorithm.

DESIGN AND IMPLEMENTATION OF A FUZZY CONTROLLER

Five components may be distinguished in any fuzzy controller (Fig.2):

1) A **Rule-Base** containing the control rules or statements.
2) A **Database** which contains the operational definitions of the fuzzy sets appearing in the fuzzy rules.
3) A **Condition interface** receiving non-fuzzy signals from the process and transforming them into a fuzzy set form.
4) An **Action interface** which translates the controller's (fuzzy) response into a non fuzzy action, and
5) A **Computation unit** inside which the fuzzy action is computed from the fuzzy input signals provided by the condition interface.

It is clear the most difficult tasks are to construct the Rule Base the Computation Unit and the Database.

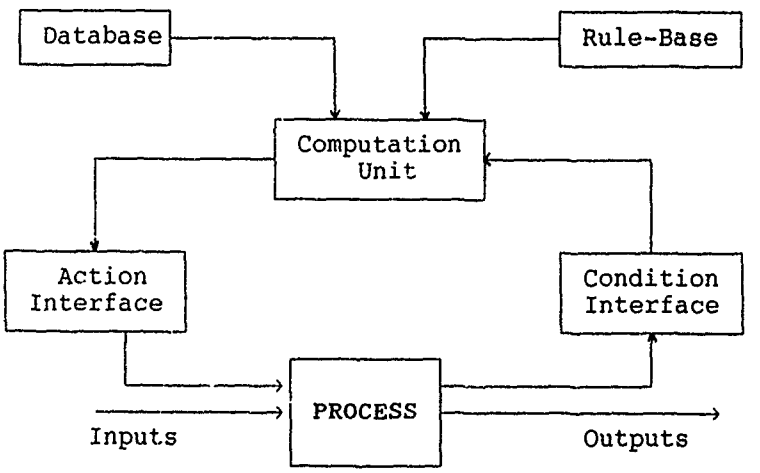


Fig.2

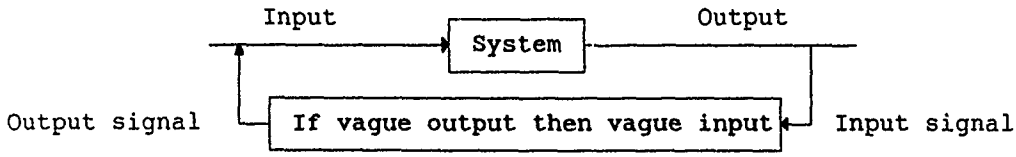
There is no systematic knowledge engineering procedure to obtain the control rules to construct a "complete Rule Base", that is, one being able to give the control action against any signal from the current process. Most part of applications uses a trial-and-error

approach ([2], [4]). Some attempt to develop automatic procedures for this task may be found in the literature ([5], [10]). Anyway, three modes of derivation of fuzzy control rules may be distinguished ([16], [14]):

1. Based on operator's experience and/or the control engineer's knowledge.
2. Based on the fuzzy model of the process controlled, and
3. Based on operator's control actions.

The first method is the most used because it is well-fitted with knowledge engineering approaches. Any case the rules usually take the form of "if-then" statements, with fuzzy data in the antecedents and consequences of every implication.

Another part of the fuzzy controller which has received an special attention, is the computation unit. Its functioning may be outlined in the following feedback loop,



But, usually the controller contains several rules and thus (denoting the relation if.. then.. by an arrow) the feedback loop may be represented as in Fig. 3.

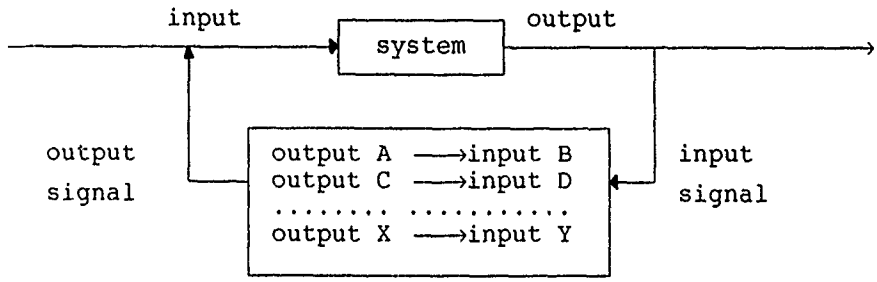


Fig. 3

As the inputs and the outputs signals in the computation unit are fuzzy or vague statements or properties, its functioning may be just seen as an instance of Approximate Reasoning. Thus, the computation unit may be seen as a device which makes inferences to obtain the outputs signals (input for the system) from input signals (output of the system) and rules. In the literature two models of computation unit may be found associated with the two existing approaches of fuzzy reasoning:

a) Based on compositional rules of inference (generalized "modus ponens") [32, 33, 21, 22].

b) Based on fuzzy logic, for instance fuzzified Lukasiewicz logic [22, 23, 1].

Most part of existing fuzzy controllers use generalized "modus ponens" (implemented by means of compositional rules of inference) for the sake of simplicity in computation [15].

Another crucial task in the design of a fuzzy controller is to establish the Data- base, which involves determining the universe of discourse of fuzzy variables in the input and the output, as well as their semantics, that is the term set of linguistic assessments, and the membership functions of the fuzzy sets representing that labels. With respect to this task two approaches may be found in the literature.

Since the Mamdani's pioneer works [8, 9], the fuzzy controllers based on generalized "modus ponens" usually assess the fuzzy variables on a term set with seven labels:

NB: Negative Big, NM: Negative Medium, NS: Negative Small

ZO: About ZerO, PS: Positive Small, PM: Positive Medium

and PB: Positive Big

whose semantics is different according to the characteristics of the universe (discrete or continuous [15]).

On their turn, the authors using fuzzy logic assess variables on a three labels term set:

N: Negative, ZO: ZerO, and P: Positive

with monotonic membership functions (see [15]).

In following sections we will specially turn our attention to the fuzzy controllers which use generalized "modus ponens".

The final step in the construction of a fuzzy controller is the effective implementation of its design. Like the classical control model implementations, there exist two different ways for implementing a fuzzy controller: 1) by means of software systems, and 2) through hardware devices.

In the first case, the central task is to write computer programs (in a suitable programming language) being able to make the aforementioned inferences. Some kind of suitable interfaces between computer and system are obviously needed. Most part of works in fuzzy control are in this way (see references about applications).

In the second approach the central task is to construct a hardware device with the specific purpose of making the control inferences. Suitable interfaces with the system are needed again, but their nature is quite different to the ones in the software approach.

One of the most active researchers in the field of hardware implementations of fuzzy controllers is T. Yamakawa, [24, 25, 26] which has constructed the so called "Rules Chip", a chip that implements control inferences using Mamdani's compositional rule of inference (see next section) to model the generalized "modus ponens". We must point out Yamakawa's research goals are beyond fuzzy control, as this author seeks for designing and constructing a "Fuzzy Computer", that is, an electronic device having the inference as intrinsic operation, like digital computer are designed to make arithmetic [25, 27, 28].

Hardware implementations of fuzzy controllers, and more generally of fuzzy inferences are receiving a growing attention [see for instance 17, 18, 19, 6].

GENERALIZED "MODUS PONENS"

Systems using internal representations of knowledge need to have the ability of making inferences, that is, they must be able to obtain new facts from already known ones.

The "modus ponens" (the basic deduction rule from Predicate Calculus) is the best known inference method, and it has been extensively used in Artificial Intelligence developments. It may be established as follow.

Assumed that the implication if P then Q is true, and given that P happens (that is, the fact or proposition P is true), then one must conclude the fact (proposition) Q is true. Usually this rule of inference is denoted as

$$\begin{array}{l} P \longrightarrow Q \\ \underline{P \qquad\qquad} \\ Q \end{array}$$

In most cases P and Q capture knowledge about variables. The simplest instance consider P and Q are statements about the variables x and y respectively. That is, P and Q stand for "x is A" and "y is B" respectively, where A and B represent properties of x and y, i.e. subsets of the corresponding universes of discourse. Whith such formulation, the "modus ponens is written as

$$\begin{array}{l} x \text{ is } A \longrightarrow y \text{ is } B \\ \underline{x \text{ is } A \qquad\qquad\qquad\qquad} \\ y \text{ is } B \end{array}$$

The deduction rule in that cases, where P and/or Q involve several variables, is straightforward obtained.

Usually expert's rules or statement about facts are vague, imprecise or fuzzy, rather than exact or precise. Inference methods for that cases are obviously needed if one is looking for modeling human reasoning to implement it in an automatic system. In the setting of Approximate Reasoning, Zadeh [32], introduced an extension of the classical "modus ponens", that is known as "generalized modus ponens", to deal with fuzzy propositions or fuzzy statements about variables, i.e. fuzzy or linguistic variables.

Let us consider a rule if x is A then y is B, where A and B are fuzzy properties (constraints) about the variables x and y respectively, i.e. fuzzy subsets of the corresponding universes of discourse. Moreover, let us suppose an observation about x that allows to state x is A', where A' is a fuzzy statement about x being related, but in general different, to A. By generalizing the classical scheme, the conclusion must be in the form y is B'

$$\begin{array}{l} x \text{ is } A \longrightarrow y \text{ is } B \\ \underline{x \text{ is } A' \qquad\qquad\qquad\qquad} \\ y \text{ is } B' \end{array}$$

where B' is to be a fuzzy set on the universe of y, related, but in general different, to B. There the key problem is to effectively determine B'.

The most usual approach to obtain B' is the so called "Compositional Rule of Inference" developed by Zadeh [30, 31, 32]. The very idea is the rule "x is A $\longrightarrow$ y is B" induces a fuzzy relation R between x and y, that is, a fuzzy subset in the cartesian product of the corresponding universes of discourse whose membership function depend upon the membership functions of A and B:

$$m_R(x,y) = F[m_A(x), m_B(y)]$$

and then B' is the transformation of B through R, i.e.

$$B' = R \otimes B$$

where $\otimes$ denotes some composition operation.

Let us observe F(a,b) is just an implication function in the classical sense of the logic.

According to Zadeh's Extension Principle we may write

$$m_{B'}(y) = \underset{x}{\text{Max}} \; (m_{A'}(x) * m_R(x,y))$$

* being an associative and monotonous application from [0,1]x[0,1] to [0,1], i.e. a t-norm, and in this way the problem of determining B' has been changed into establishing F and *.

Several options are possible and each conveys to a different instance of "generalized modus ponens" i.e. to a different approximate reasoning model, which may be use to model human ways of reasoning. Next section deals with a comparative study of several choices of F and *.

Roughly speaking we can say the control unit of a fuzzy controller may be designed just as a device that performs some model of "generalized modus ponens". To illustrate its operation, let us consider a system for which the control rules are

$$\text{If } x_1 \text{ is } A_{11} \text{ and } x_2 \text{ is } A_{12} \text{ then y is } B_1$$

$$\text{If } x_1 \text{ is } A_{21} \text{ and } x_2 \text{ is } A_{22} \text{ then y is } B_2$$

A_{ij} and B_i, being labels belonging to the term sets where the variables x_j and y are assessed, i,j = 1,2 (usually as we have described in last section). Moreover, let us consider that Mamdani's version of the compositional rule of inference is used [8, 9], that is

$$F(a,b) = \min \{ a,b \}$$

$$* = \text{Min}$$

If the (non-fuzzy) signals from the system (obtained by means of a suitable sensor) are x_1°, x_2°, belonging to the universe of discourse of x_1 and x_2 respectively, then the compatibility of these observation with the premises of each control rule is given by

$$w_1 = A_{11}(x_1) \wedge A_{12}(x_2)$$
$$w_2 = A_{21}(x_1) \wedge A_{22}(x_2)$$

and thus the conclusion B' must be characterized by

$$B'(y) = \text{Max } \{w_1 \wedge B_1(y),\ w_2 \wedge B_2(y)\}$$

Let us observe the output from the computation unit is a fuzzy subset in the universe of discourse of y. For control, a crisp response being applicable to the systems is needed, and then B' ought be defuzzified before its going out of the fuzzy controller. Synthesizing B' in its gravity center is quite usual.

USING DIFFERENT IMPLICATION FUNCTIONS IN THE GENERALIZED "MODUS PONENS"

In [32] Zadeh proposes to use the Min operator as *, and F the Lukasiewicz's implication function, i.e. F(a,b) = min { 1,1-a+b}

In [11, 12, 13] the behavior of the "generalized modus ponens" under other different implication functions (keeping * = Min) are dealt with. These studies have a theoretical character rather than a practical one.

At present, our own research is mainly concerned with the possibility of designing hardware devices, able to make inferences by means of "generalized modus ponens". Thus we are looking for implication functions and/or t-norms with two key features:

a) they must produce an inference method representing an average human reasoning,

b) they must be simple enough to be implemented through electronic circuits.

Moreover, for a formal coherence principle, it seems reasonable to want obtain output with the same characteristics than the input, in the sense that if the input is, for instance, a trapezoidal fuzzy number (tfn), the corresponding output must also be another tfn.

With these goals in mind, we are carrying out studies rather different to the aforementioned ones, paying an special attention to the behavior of "modus ponens" in relation with points a) and b) below. In the following, we summarize a simulation experiment which attempt to discover the possibility of B' preserving with respect to B, the relative position of A' with respect to A, under different implications functions.

We have assumed, on a hand, X and Y, the universes of discourse of the variables x and y respectively, are closed and bounded intervals of the real line. Thus, both X and Y may be taken equal to [a,b] or more particularly X = Y = [0,1]. On the other hand every membership function ($m:[0,1] \longrightarrow [0,1]$) is sampled on 30 equally spaced points, to obtain an internal computer representation. Each sampled grade of membership is, of course, ranging from 0 to 1. Thus, any fuzzy set is represented by a vector of 30 elements, that is a point in $[0,1]^{30}$, and we can write:

$$A \cong (a_1 \ldots\ldots a_{30}) \text{ with } a_i = m_A(x_i)$$

$$A' \cong (a'_1 \ldots\ldots a'_{30}) \text{ with } a'_i = m_{A'}(x_i)$$

$$B \cong (b_1 \ldots\ldots b_{30}) \text{ with } b_j = m_B(y_j)$$

Obviously the relation R appears as a 30x30-matrix defined by $r_{ij} = F(a_i, b_j)$ and thus

$$B' \cong (b'_1 \ldots\ldots b'_{30}) \quad \text{with} \quad b'_j = \max_i (a'_i \wedge r_{ij})$$

In our experiment, A, A' and B' are supposed tfn, because they are simple to handle, and good enough to represent any kind of vagueness or uncertainty of data. With respect to the relative position of A and A', three cases have been considered: 1) A' is included in A, 2) A' is included in A, but A'and A have a non empty intersection and 3) A is included in A'

Fifteen different implication functions have been considered to define R. They are the most usual ones [13], and are listed in the first column of the table 1.

Table 1 summarizes the results of our experiment. Each row presents the assessment of the behavior of each implication function, against the aforementioned three cases of relative position for A and A', in such a way that any pigeonhole contains three labels L1, L2, L3, with the following characteristics:

A) L1 measures the goodness of F in preserving, on B' and B, the relative position of A' with respect to A. The possible linguistic values we have used are vb = very bad, b = bad, m = medium, g = good and vg = very good

B) L2 assess the increment of fuzziness of B' with respect to B, measured through the value $I = \min\{b'_j : j = 1,2,\ldots,30\}$. The used linguistic values are E = non increment (I = 0), F = fair increment (I < 0.5), UF = unfair increment (I > = 0.5), T = total fuzziness (I = 1)

C) L3 is a boolean parameter that indicates whether B' is (Y) or not (N) a tfn.

References

[1] J.F. Baldwin, A new approach to approximate reasoning using a fuzzy logic, Fuzzy Sets and Systems, 2, 309-325, (1979).

[2] M. Braae, D.A. Rutherford, Selection of parameters for a fuzzy logic controller, Automatica, 15 (5), 553-557, (1979)

[3] M. Braae, D.A. Rutherford, Theoretical and linguistic aspects of the fuzzy logic controller, Automatica, 15, 535-577, (1979)

[4] E. Czogala, W Pedrycz, Some problems concerning the construction of algorithms of decision making in fuzzy systems, Int. Journal Man-Machine Studies, 15 (2), 210-211, (1981)

[5] E.Czogala, W. Pedrycz, Fuzzy rule generation for fuzzy control. Fuzzy Sets and Systems, 13, (3), 275-294, (1982)

[6] K. Hirota, K. Ozawa, Fuzzy Flip-Flop as a basis of fuzzy memory modules, in Fuzzy Computing, Theory, Hardware and Applications (M.M. Gupta, T. Yamakawa, eds.) 173-183, North Holland, (1988)

[7] P.E. Kloeden, Fuzzy dynamical Systems, Fuzzy Sets and Systems, 7, 275-296, (1982)

[8] E.H. Mamdani, Applications of fuzzy algorithms for control of simple dynamic plant. Proc. IEEE, 121 (12), 1585-1588, (1974).

[9] E.H. Mamdani and S. Assilian, An experiment in linguistic synthesis with a fuzzy logic controller. Int. Journal Man-Machine Stud. 7 (1), 1-13, (1975).

[10] E.H. Mamdani, J.J. Ostergaard, E. Lembessis, Use of fuzzy logic for implementing rule-based control of industrial processes, in Fuzzy Sets and Decision Analysis, (H.-J. Zimmermann, L.A. Zadeh, B.R. Gaines eds.) North Holland, 429-445, (1984)

[11] M. Mizumoto, Extended Fuzzy Reasoning, in Approximate Reasoning in Expert Systems, (M.M. Gupta, A. Kandel, W. Bandler, J.B. Kiszka, eds.), North Holland, 71-85, 1985.

[12] M. Mizumoto, Fuzzy controls under various fuzzy reasoning methods, Information Sciences, 45, 129-151 (1988)

[13] M. Mizumoto, H.-J. Zimmermann, Comparison of fuzzy reasoning methods, Fuzzy Sets and Systems, 8, 253-283 (1982)

[14] W. Pedrycz, Design of fuzzy control algorithms with the aid of fuzzy models. In Industrial Applications of Fuzzy Control (M. Sugeno ed.), North Holland, 153-173, (1985)

[15] M. Sugeno, An introductory survey of Fuzzy Control, Information Sciences, 36, 59-83, (1985).

[16] T. Takagi, M. Sugeno, Derivation of fuzzy control rules from human operator's control actions, Proc. Symp. Fuzzy Information, Knowledge Representation and Decision Analysis, Marseille, 55-60, (1983)

[17] M. Togai, H. Watanabe, A VLSI Implementation of Fuzzy Inference engine toward an expert system on a chip. Proc. 2th Int. Conf. on AI and Applications (IEEE), 192-197, (1985)

[18] M. Togai, H. Watanabe, A fuzzy inference engine on a VLSI chip: design and implementation. Proc. 2th Fuzzy Systems Symposium, 106-114, Japan, (1986)

[19] M. Togai ,S. Chiu, A fuzzy chip and a fuzzy inference accelerator for real time approximate reasoning. Proc. 17th IEEE Int. Symp. Multiple valued Logics, 25-29, (1987)

[20] R.M. Tong, An annotated bibliography of fuzzy control. In Industrial Applications of Fuzzy Control, (M. Sugeno, ed.), 249-269, Elsevier Science Publishers, B.V., (1985).

[21] E. Trillas and L. Valverde, On Mode and Implication in Approximate Reasoning. In M.M. Gupta, A. Kandel, W. Bandler and J.B. Kiszka (Eds): Approximate Reasoning in Expert Systems. North-Holland (1985).

[22] **E. Trillas and L. Valverde:** On Implication and Indistinguishability in the Setting of Fuzzy Logic. In J. Kacprzyk and R.R. Yager (Eds): Management Decision Support Systems using Fuzzy Sets and Possibility Theory. Verlag, TUV (1985), 198-212.

[23] **Y. Tsukamoto,** An approach to fuzzy reasoning method. In Advances in Fuzzy Set Theory and Applications (M.M. Gupta, R.K. Ragade and R.R. Yager, eds.), North Holland, 137-149, (1979).

[24] **T. Yamakawa,** High-speed fuzzy controller hardware system, Proc. 2nd Fuzzy Systems Symp., Japan, 16-18, (1986)

[25] **T. Yamakawa,** A simple fuzzy controller hardware system employing MIN & MAX operations-A challenge to 6th generation computer. Preprints of 2nd IFSA Congress, Tokyo, 827-830, (1987).

[26] **T. Yamakawa,** High-Speed Fuzzy Controller Hardware System: the Mega FIPS Machine, Information Sciences, 45, 113-128 (1988)

[27] **T. Yamakawa,** Intrinsic fuzzy electronics circuits for sixth generation computer. In Fuzzy Computing ,Theory, Hardware and Applications. (M.M. Gupta, T. Yamakawa, eds.), 157-171, (1988)

[28] **T. Yamakawa,** A programmable fuzzifier integrated circuit: Synthesis, Design and Fabrication, Information Sciences, 45, 75-112, (1988)

[29] **L.A. Zadeh,** Fuzzy Sets. Information and Control, 8, 338-358, (1965)

[30] **L.A. Zadeh,** The Concept of a Linguistic Variable and its Application to Approximate Reasoning. Parts 1 and 2, Information Sciences 8 (1975), 199-249, 301-357. Part 3, Information Sciences 9 (1976), 43-80.

[31] **L.A. Zadeh,** Calculus of fuzzy restrictions. In Fuzzy Sets and their Applications to Cognitive and Decision Processes, (L.A. Zadeh, K.S. Fu, K. Tanaka and M. Shimura, eds.), Academic Press, 1-40, (1979).

[32] **L.A. Zadeh,** A theory of Approximate reasoning. In Machine Intelligence 9 (J.E. Hayes, D. Michie, L.I. Mikulich, eds.) Elsevier, 149-194, (1979)

Table 1

F(a,b) =	A' ⊆ A	A' ∉ A	A ⊆A'
1 if a<>b or b=1 0 otherwise	g, UF,Y	vb,T,Y	vb,UF,Y
1 if a<=b 0 otherwise	vg,E,Y	vb,T,Y	vg,F,Y
1 if a<=b b otherwise	m,E,Y	vb,T,Y	vg,F,Y
1 if a<=b min(1,b/a) ot.	b,E,Y	vb,T,Y	vg,F,Y
1 if a=0 or (1-b)=0 min(1,b/a, ,(1-a)/(1-b)) ot.	g,E,Y	vb,T,Y	vg,F,Y
min(1,(1-a)+b)	b,F,Y	vb,T,Y	vg,F,Y
(1-a)+a*b	b,F,Y	vb,T,Y	vg,F,Y
max((1-a),b)	m,F,Y	vb,T,Y	m,F,Y
max((1-a),min(a,b))	m,F,Y	vb,T,Y	m,F,Y
min(a,b)	m,F,Y	m,F,Y	m,F,Y
1 if a=b (1-b) if a<b 0 if a>b	vg,F,N	g,UF,N	vg,F,N
1 if a=b (1-b) if a<b b if a>b	m,E,Y	g,F,N	vg,F,N
1 if a=b 0 if a<b b if a>b	m,E,Y	g,UF,N	g,F,N
1 if a=b 0 otherwise	g,E,N	m,UF,N	b,F,N
z1=max(1-a,b) ; z2=max(1-a,a) z3=max(1-b,b) ; min(z1,z2,z3)	m,F,Y	m.UF,N	m,F,N

A Neural Network for the Analysis of Aircraft Test Data*

J. B. Golden
Vanderbilt University
Nashville, Tennessee
and
B. A. Whitehead
The University of Tennessee Space Institute
Tullahoma, Tennessee

ABSTRACT

With the advent of the USAF's Advanced Tactical Fighter and NASA's National Aerospace Plane, demands for concise test data reduction and interpretation will increase beyond the capabilities of current methodologies As mission complexity increases it becomes apparent that real time data analysis for flight safety, mission control and test conduct becomes a necessary tool.

A neural network is a biologically inspired mathematical model, which can be represented by a directed graph, that has the ability to learn through training. Neural networks have many advantages over current aviation computing systems including the ability to learn and generalize from their environment. Neural networks are excellent for parameter estimation and recognizing patterns in signal data This research discusses a prototype system designed and implemented at the University of Tennessee Space Institute to discover patterns in test data from an engine test cell in order to determine if any part of the system is in failure. The results of this research show that a neural network can be used for fault diagnosis in an engine test cell when the problem of fault monitoring and diagnosis is seen as one of pattern recognition A two layer semilinear feed-forward neural net is able to separate simulated sensor data into normal and abnormal classes and the addition of a hidden layer makes the network more resistant to noise and improves the ability of the network to classify the type of fault that is occuring.

Neural networks offer test personnel a new tool for the analysis of critical test data. A general working network model can be applied to any system where sensor data is evaluated and fault diagnosis is critical Significant benefit to the aerospace community is gained by improved safety of test engineers by rapid detection of faulty conditions. This will be of extreme importance as testing begins on more advanced engine systems such as the National Aerospace Plane and the Advanced Tactical Fighter

(1.0) INTRODUCTION

(1.1) Problem Statement

Simulated flight tests conducted in an engine test cell produce large quantities of sensor data at rates that overwhelm flight test personnel. The task of monitoring signals from test cell sensors and watching for the onset of possible fault situations requires significant experience and a number of test cell personnel Because of the variety and quantity of test cell data, extensive knowledge and experience is required for accurate differentiation of data types. A neural network architecture that could accurately diagnose the onset of fault situations and alert test cell engineers would result in a significant savings in personnel, downtime because of damaged equipment, and improve test cell safety.

(1.2) Approach

The main approach of this research is to design and analyze a network architecture to monitor sensor data from an engine test cell to detect data that falls outside learned parameters, thus signifying a fault. In this research, diagnostic problems in general are conceptualized as a mapping of an input pattern (representing sensor data) to an output pattern which is interpretable as a diagnosis.

 This research was supported by Sverdrup Technology under contract number R02-400489 All references to the Aeropropulsion Systems Test Facility (ASTF) are from public domain documents (see Polce's MS Thesis). This paper was cleared for public release by USAF/DOCS and USAF/PA on 20 Feb 1990.

A neural network is a biologically inspired mathematical model, which can be represented by a directed graph, that has the ability to learn through training. The action of a neural net may be viewed principally as a mapping through which points in the input space are transformed into corresponding points in the output space on the basis of designated attribute values (of which class membership might be one) (Pao,1989). It is the goal of this research to design a neural network architecture that can classify test data as part of a nominal engine test or as data from a system in failure. The words neural network, network, net, and network model will be used interchangeably throughout this thesis.

(2.0) PROBLEM ANALYSIS

(2.1) The ASTF Domain

The Aeropropulsion Systems Test Facility (ASTF) is designed for testing airbreathing engine propulsion systems over a wide range of altitudes and Mach numbers. Simulated flight tests conducted in the facility will determine the operational and performance characteristics of aeropropulsion systems; thus, the time required for flight tests is shortened, and the risks and expense of flight tests are minimized (Polce, 1982).

The ASTF was designed to permit development and surveillance tests of large airbreathing, jet propulsion engines throughout their operational envelopes by providing the capability for true simulation of flight altitude and flight Mach number conditions. The facility is capable of performing direct-connect testing of turbojet and high-bypass (8:1) turbofan engines up to 75,000 pound-rated, sea level, static thrust throughout the appropriate engine operating envelopes (Polce, 1982). The facility is designed to allow testing in both direct-connect and freejet modes.

In Polce's (1982) thesis he describes different types of tests that will be conducted in the ASTF. These test types require the capability of transient testing, e.g., engine power transients, flight profile transients, or combinations of the two. The facility operating conditions and the corresponding engine operating conditions for testing in the ASTF are:

(1) Steady Environment — Steady Engine Power Setting.
(2) Steady Environment — Transient Engine Power Setting.
(3) Transient Environment — Steady Engine Power Setting.
(4) Transient Environment — Transient Engine Power Setting (Mission Profile Simulation) (Polce, 1982).

Looking at the testing requirements that the ASTF must perform, it becomes obvious that systems within the test cell must be able to alter both pressures and temperatures, along with a number of valve openings, to regulate test cell conditions. As the system increases in complexity, it becomes an overwhelming task to supervise all the sensors that monitor the facility. Instrumentation and data handling equipment must be provided to measure, transmit, acquire, display, and process those measurements which are required to determine the performance of the test article. The measurement capability for each test cell includes 970 pressure measurements, 820 temperature measurements, and 152 miscellaneous measurements, such as forces, flows, speeds, geometries, vibrations, accelerations, and strains (Polce, 1982).

Three major subsystems make up the ASTF instrumentation and control system They are the Plant Instrumentation and Control System (PICS), The Test Instrumentation System (TIS) and the Automatic Test Control System (ATCS).

The ATCS performs a number of control functions inside the test cell. The ATCS is used to:

(1) Control test conditions.
(2) Control engine operating conditions.
(3) Coordinate control of test and engine operating conditions with plant configuration changes
(4) Detect abnormal plant and engine operating conditions and their response thereto
(5) Communicate with facility operators (Polce, 1982)

Because of the importance of the ATCS and because it is used for detection of abnormal conditions inside the test cell, this research concentrates on applying neural network technology to enhancing the ATCS

The ATCS controls the parameters which establish a desired test condition by positioning forty-six of the facility configuration and control valves. The ATCS is used to control various engine controls and the interface to the test cell engineers is effected at a panel in the test building control room. Pertinent test information is communicated to test cell personnel through four CRT terminals. It is the end goal of this research to establish a better means of monitoring test data from the ATCS.

(2.2) Fault Monitoring and Diagnosis

In order to apply neural networks to fault monitoring and diagnosis, the task of fault detection must be seen as one of pattern recognition. In approaching the ASTF we will say that fault detection is observing data that fall outside expected parameters previously regarded as "normal" operation. Sensor data that fall outside normal boundaries do not necessarily indicate that the system is beginning to fail, but it does indicate that an abnormal occurrence is taking place inside the system that is being studied and warrants further attention. As stated previously, in this research diagnostic problems are viewed as a mapping of an input pattern representing sensor data to an output pattern which is interpretable as a diagnosis.

Analysis of a complex system should use an opportunistic strategy — there is no computationally feasible "legal move generator" that defines the space of solutions in which pruning and steering take place as in classical AI systems (Nii and Feigenbaum, 1978). Bits and pieces of information must be used as the opportunity arises to (relatively) slowly build a coherent picture of the world (system). This is one way in which neural networks are better than current AI methods at analyzing signal data from complex systems.

(3.0) THEORETICAL DEVELOPMENT

(3.1) Neural Networks

As stated previously, we will define a neural network as a biologically inspired mathematical model, which can be represented by a directed graph, that has the ability to learn through training. A neural network model typically has eight major aspects. They are:

(1) A set of processing units
(2) A state of activation
(3) An output function for each unit
(4) A pattern of connectivity among the units
(5) A propagation rule for propagating patterns of activities through the network of connectivities.
(6) An activation rule for combining inputs to produce a new activation level for a unit.
(7) A learning rule whereby patterns of connectivity are modified by experience.
(8) An environment within which the system must operate (Rumelhart and McClelland, 1988)

These will each be described in detail below.

Each processing unit in a neural network model represents some feature-like entity such as an engine sensor. It is the pattern as a whole coming from the system to be studied that is the meaningful level of analysis. The purpose of a unit is to receive input from its neighbors and as a function of the inputs it receives to output a value to its neighbors. There are three types of processing units in a neural network model: the input units which receive data from sources external to the model, one or more layers of hidden units which are internal to the model only, and a series of one or more output units which sends signals out of the model to effect motoric systems or influence other external systems. Figure 1 shows a three layer feed-forward neural network.

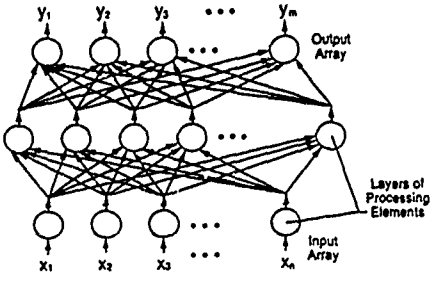


Figure 1 A Feed-Forward Neural Network Architecture

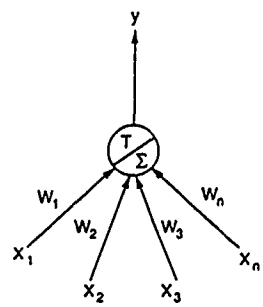
A state of activation is the representation of the state of the system at some time t representing the pattern of activity over the set of processing units. It is the pattern of activation over the set of units that captures what the system is representing at any time.

Associated with each unit y_i there is an output function $y = w \cdot x$ which maps the current state of activation to some output y_i; $y_i = \sum_j w_{ij} x_j$. There is often some threshold function so that a unit has no effect on its neighboring unit unless the output function exceeds some certain value (Rumelhart and McClelland, 1988).

The pattern of connectivity constitutes what the system knows and determines how it will respond to some arbitrary input. Neural networks are often called connectionist networks because the computations in the net are largely performed by the connection strengths. The signal coming into a processing unit is the inputs from all incoming units multiplied by a weight and summed to get the overall input to that unit, as in Figure 2

The total pattern of connectivity can be represented by specifying the weights for each of the connections in the system. Positive and negative weights can be seen as excitory or inhibitory input respectively. The absolute value of the weight gives the strength of the connection. How to determine appropriate weights is dependent on the learning algorithm

Propagation in the net is the sum of the net excitory inputs (the weighted sum of the excitory inputs to the unit) and the net inhibitory inputs (the weighted sum of the inhibitory inputs to the unit). (Rumelhart and McClelland, 1988). The activity of a unit is regarded as the degree of confidence that a preferred feature is present. To reach a new state of activation a function f (activation rule) will be used to take x and produce some new state of activation. f should be differentiable in order to use certain learning algorithms



$$\bar{X} = (x_1, x_2, x_3, \ldots, x_n)$$
$$\bar{W} = (w_1, w_2, w_3, \ldots, w_n)$$
$$y = \sum_{i=1}^{M} x_i w_i > T$$

Figure 2. A Single Neural Network Processing Element

What makes a network useful in the analysis of complex systems is the ability to modify its connectivity through experience. There are three types of modifications to a network: the development of new connections, the loss of existing connections and modifications of the strengths of connections that already exist in the net The derivation of these modification (learning) rules will be covered under the section on learning algorithms

The last aspect of the model is the environment in which it will operate. Formally, we will define this as a time varying stochastic function over the space of the input patterns. More simply; at any point in time, there is some probability that any of the possible set of input patterns is impinging on the input units The environment for this research is the input vectors composed of sensor data coming from the ATCS in the Aeropropulsion Systems Test Facility.

A description of how a neural network processes information follows: An input signal (stimulus) is received by the nodes in the input layer and produces some activation level in a given input unit, which conveys a signal of proportional strength based on the weighting matrix to its [synaptic] connections, or nodes in the next layer The global effect is that a pattern of activation across the set of input units produces a distinct pattern of activations across the set of hidden units (Churchland and Churchland, 1990). The process is repeated again as many times as there are hidden layers. As before, an activation pattern across the hidden units produces a distinct activation pattern across the next layer of nodes In this way the network becomes a device for transforming some input vector representing sensor data into a uniquely corresponding output vector. It is a device for computing a specific function. Exactly what function it computes is fixed by the global configuration of its synaptic weights (Churchland and Churchland, 1990).

There are a number of procedures for training a network and adjusting the weights Many learning algorithms have been derived over the last two decades. In choosing a learning algorithm for a system to monitor the ASTF we must realize the function that we want to specify is initially unknown to us because of the myriad possible fault situations We can impose an unspecifyable function on the network as long as we can supply a set of examples of the desired input/output pairs. The network is then trained until it performs the input to output transformation desired.

(3.2) Learning Algorithms

When feasible, learning is a very attractive alternative to explicit programming. This is particularly true when problems do not lend themselves to systematic programming, such as pattern recognition. All neural network learning algorithms have their basis in the neural learning rule discovered by Hebb in 1949 His rule for "Hebbian learning" in a network is as follows:

> When an axion of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased

Neural Networks can be classified on the basis of the patterns stored and their method of recall Two labels for networks used as associative memories are autoassociative networks and heteroassociative networks Autoassociative networks associate a data item with itself. Heteroassociative networks associate two different data items — one is used to recall the other Heteroassociative type memories must be trained with a data pair representing the input and the desired output pattern associated with it. To store the pattern in a training set each input is presented to the system along with the desired output and some learning rule is used to adjust the weights usually based on a difference equation and error between actual and desired outputs To recall stored information an unfamiliar pattern from a different set of patterns is presented to the network and the system outputs the second member of the training pair, the associated pattern of output.

A heteroassociative network will be used for fault monitoring of the ASTF Interpretation of the network's outputs and weights should give insight into the meaning of signals in a complex system If a network is able to respond to a set of "interesting" patterns then this will form the basis for a feature detector which we will use for detecting faults in our complex system.

The learning rule used for this system to discover normal and abnormal patterns of data from the ASTF is backpropagation of error which was discovered simultaneously by a number of different research groups in the early 1980s. Backpropagation is a heteroassociative, function estimating, neural network that stores arbitrary analog spatial pattern pairs, using multi-layer error-correction encoding between a target output vector and actual output values from the net

The learning procedure consists of the net being assigned a random set of weights and receiving a training set pair as input and evaluating the output in a feedforward manner The initial difference error between target and actual outputs will be quite large. Using the backpropagation procedure, the algorithm calculates a change in weight Δw_{ij} for all w_{ij} in the net for that particular pattern This procedure is repeated for all patterns in the training set to yield a resulting Δw_{ij}.

$$\Delta w_{ij} = \eta \delta_{ij}(q) y_j(q)$$

where $\delta_{ij}(q) = -\sum_q z_i^{error}(q) \cdot z_i^{out}(q)[1 - z_i^{out}(q)]$

$z_i^{desired}$ is the desired output for node z_i and z_i^{out} is the actual output value of node z_i.

Corrections are made to the weights and the outputs again are evaluated in a feedforward manner based on the number of iterations specified by the system. Discrepancies between actual and target output values again result in evaluation of weight changes (Pao, 1989). If the training is successful, the system error will decrease with the number of iterations the system performs and the procedure will converge to a stable set of connection weights. For a complete derivation of the backpropagation learning rule see Appendix B: Derivation of the Backpropagation Learning Rule.

Backpropagation is not guaranteed to find the global error minimum, only the local error minimum which should be good in enough in determining a fault situation in the ASTF facility. Backpropagation was chosen for this research because of its ability to store many patterns, its ability to generalize and its ability to perform arbitrary nonlinear mappings (Simpson, 1990).

(3.3) The Network's Ability to Generalize from Example

When considering neural networks for fault monitoring and diagnosis it is important to consider how well the network can generalize from the set of training examples It is impossible (and impractical) to present the network with all possible examples of a fault We are expecting the network to learn from a set of examples and be able to generalize for all possible faults.

In diagnosing faults in the ASTF we have an environment, which is a set of sensor values, and call this set X. In this environment we have a concept defined as a function $g : x \rightarrow \{0,1\}$ which is the presence or absence of a fault. The goal of learning is to produce a hypothesis $h : x \rightarrow \{0,1\}$ that approximates the concept g (Abu-Mostafa, 1989). In our case h is a pattern recognition system that recognizes the presence of faulty data. To do this we have a number of examples of the concept $(x_1, g(x_1))...(x_n, g(x_n))$, in this case a set of input vectors from the ASTF simulator and the desired outputs. A learning algorithm is one that takes the examples and produces the hypothesis. Performance of the system can be measured by the number of examples needed to produce a good hypothesis.

When backpropagation is used in a feed forward manner, we are guessing that there is a good hypothesis to be gained by setting the weights and thresholds of the network. The set of hypothesis H would then be the set of all functions h that are obtained by setting the weights and thresholds of the net. The learning algorithm picks a hypothesis $h \epsilon H$ that mostly agrees with g on the examples in the training set. In thinking about the generalizability of the network it is necessary to ask: Does this choice, which is based on the behavior of the examples, hold in general (Abu-Mostafa, 1989).

Generalization can be expected if the number of examples is large enough. Without sufficient number of examples the algorithm cannot produce a good hypothesis With too many examples the computational task of digesting the examples into a hypothesis proves intractable.

(3.4) Adaptive Pattern Recognition

We will view pattern recognition as the ability to map a series of input patterns into groups of output patterns (such as classes of patterns) or as Pao (1989) defines "a mapping from representation space to interpretation space". The goal of this research with respect to pattern recognition is to determine if a neural network can be used as the operator to carry out this mapping for use in fault diagnosis. In the past, operators were installed in specific pattern recognition systems. We will try to develop a network that can learn the optimal operators through training. The word adaptive means that the system will be able to deal with patterns that are distorted with noise, be able to modify classification rules in accordance with changing circumstances, and be able to self-organize the connectivity of the network in accordance with the structure of pattern data (Pao, 1989).

In order to achieve an optimal operator we must understand the use of discriminants in pattern recognition. A discriminant is a function or an operator that, when applied to a pattern, yields an output that is either an estimate of the class membership of that pattern or an estimate of the values of one or more of the attributes of the pattern (Pao, 1989). Simply put, a discriminant is a type of operator that produces a mapping from pattern space into attribute space. The discriminant we are after is one that processes a complete pattern at one time rather than one feature at a time The output of the net can be plotted as a map of decision regions created in the multidimensional space spanned by the input variables. The outputs from a single layer network form half-plane decision regions A two layer net can form many unbounded regions in the space spanned by the input. The output of multi-layer networks form complex convex regions based upon the intersection of the half-planes formed by each node in the first layer of the network. (Lippman, 1987). These regions, bounded by hypersurfaces, become our discriminants and patterns are classified in accordance whether they are on one side or another of a hypersurface or a set of hyperplanes (Pao, 1989) Using these hyperplanes divides the pattern space into volumes of unique class membership. In this case the classes are simple; normal or faulty operation of the system.

(4.0) RESULTS

(4.1) A Two Layer Semilinear Feed–Forward Neural Network: A Generalized Perceptron Approach

The Automatic Test and Control System (ATCS) in the ASTF uses 33 sensor values for diagnosis of the health of the facility. Eleven of those are commands values from the test cell engineers for changing the test cell environment. The other 22 are sensors that monitor the conditions of the engine and the test cell. In order to catch all possible faults that occur in the ASTF it is necessary to monitor all 22 data sensors. The network was trained using a training set of 60 scaled vectors composed of 20 normal followed by 40 abnormal vectors made up of 22 sensor values and their desired outputs. The network was allowed to iterate 1000 times over the training set to achieve the lowest possible total system error while still achieving 1000 iterations. Finalized connection weights for the input nodes (sensors) to the output node ranged between -1.61 and 3.78 The sensors with the strongest connection weights (highest absolute value) were a pressure sensor (P0) with a weight of 3.78, pressure sensor PL3H with a weight of 3 57 and a valve opening sensor (LVC1B) with a connection weight value of 2.02. The final threshold for the input nodes was -0 46. The hyperplane formed by the two layer network is shown in Figure 3.

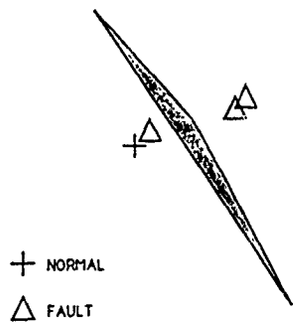


Figure 3 Hyperplane Formed by a Two Layer Neural Network in Three Dimensions (Sensor PO, Sensor PL3H and Sensor LVC1B)

In order to interpret exactly what the network found, other than by testing it on new data sets, we can look at how the final weights converged and try to decide if the weight values are telling us anything about the system being analyzed Graphing the final weights in Figure 4 we can see the pattern of connection weights with peaks at sensors P0, PL3H and LVC1B As far as this two layer network is concerned, the majority of relevant information as to the health of the system can be gained from these sensors. In other words, these sensor values give the best indications as to the status of the system

The two layer neural network was tested on seven data sets composed of ten vectors made up of 22 sensor values. Two of those sets were from the training set in order to judge whether or not the system was working correctly The other five were composed of arbitrary input vectors from the ASTF simulator representing engine test runs new to the network. As expected, the network computed a low output node value for the normal vectors extracted from the training set (0 013) and a high output node value for the vectors representing faulty data taken from the training set (0 999) This is not surprising since we would expect the network to correctly identify input vectors it has seen before The true test of the system, of course, is its ability to make an accurate

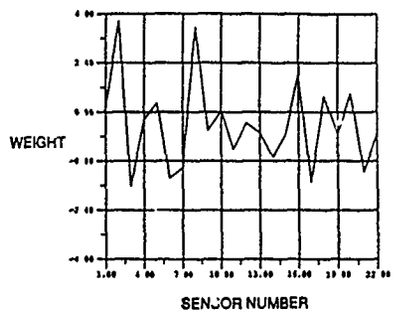


Figure 4. Final Connection Weights for a Two Layer Neural Network

diagnosis of the health of the system based on some unknown input. When a set of normal ASTF sensor vectors was presented to the network the value of the output node value was 0.014 a correct diagnosis Four new data sets composed of faulty sensor vectors was presented to the network and the output node achieved values of 0.990, 0.998, 0.936 and 0.990 respectively. These output values correctly indicate that the system is in failure This shows that the network architecture is making accurate diagnosis and generalizing over the space spanned by the input variables. A summary of the output values for the network tested on various data sets is given in Figure 7

To confirm that the network was indeed accurately classifying ASTF sensor data from any sensor in the ATCS, the network was trained using a new training set composed of vectors different from those in the original training set. The network was tested against test vectors extracted from the training set As expected the system achieved high diagnostic accuracy with a mean output value of 0.997 for vectors representing faults and a mean output value of 0.0004 for vectors representing a normal test run To test the network for generalizability over all sensors from the ASTF two unknown (not part of the training set) faults were presented to the network from unspecified locations in the test cell. The first vector was an unknown representing a turbine trip at an undisclosed location in the ASTF and the second vector represented a compressor failure in an unspecified leg of the test cell. The network achieved an output value of 0.998 for the first vector and 0 992 for the second vector.

(4.2) A Three Layer Semilinear Feed-Forward Neural Network: A Multilayer Perceptron Approach

A network was constructed using a hidden layer composed of three and five processing elements. Tests were run for each type of network, both on the vectors extracted from the training set and on the same unknown test vectors as for the two layer network. Since a two layer network gave accurate responses, the three layer network should be thoroughly scrutinized to see if it enhances the amount of information gained from the network Since adding a layer adds computational complexity and adds to the time necessary to train and test the network, a hidden layer should be able to detect hidden features in the patterns of input data not seen by the two layer network, and be able to generalize over many different test sets in order to make it economically worthwhile

The first three layer neural network used three processing elements in the hidden layer. Connection weights from the input nodes to the first hidden node ranged from-1 60 to 0 76 with highest (absolute value) weights from sensors P0 (-1 55), PL3H (-1 60) Only these two sensors had connection weight strenghts greater than 1.0. Connection weights from the input nodes to the second hidden node ranged from -1.59 to 0.71 with greatest weight strengths from sensors P0 (-1 30) and PL3H (-1.59). Again, only these two sensors had weights whose absolute value was greater than 1.0. Connection weights to the last hidden node ranged from-1.27 to 2 50 with the strongest connections from sensors P0 (2 28), PL3H (2.50) and LVC2A (-1.27). By adding a hidden layer new information was gained on diagnosing the health of the test cell, namely the significance of data from sensor LVC2A. In the two layer neural net the connection strength between LVC2A and the output node was high (1.48), but its relative importance in diagnosing the status of the system was best seen in studying the connection weights between the input layer and the third hidden node The hyperplanes formed by the three hidden nodes can be seen in Figure 5. The addition of the hidden layer produced enough hyperplanes to form a more complex region to better classify the data

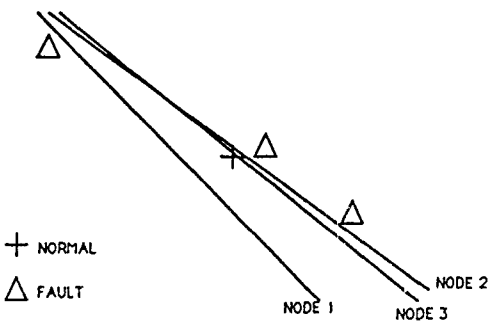


Figure 5 Hyperplanes Formed by Three Hidden Nodes Graphed in Three Dimensions (Sensor P0, Sensor PL3H and Sensor TL1H)

A three layer neural network with five hidden processing elements was also explored. For all five hidden nodes the strongest connections between the input layer and the hidden layer were from nodes representing sensors P0, PL3H, LVC1B and LVC2A Based on the finalized weights of the multilayer network, four sensors give the strongest indication as to the health of the ASTF. A summary of connection weights can be seen in Figure 6.

The three layer neural network with three and five hidden processing elements (PEs) was tested on the same data sets as the two layer network in order to compare diagnostic accuracy A summary of the results is given in Figure 7. Computing the output for the network with three nodes in the middle layer using data sets composed of normal and faulty vectors extracted from the training set, resulted in an output node value of 0 010 and 0.996 respectively. For the test set composed of new normal sensor data the value on the output node was 0 011. For the four faulty sensor data sets the network achieved the following values 0.991, 0.994, 0.973 and 0.991. By comparing these values with the outputs from the two layer network it can be seen that diagnostic accuracy was increased somewhat in four out of five test cases by the addition of the hidden layer.

Testing the three layer system with five hidden PEs, the network computed output values of 0 011 for the normal test case extracted from the training set and 0 997 for the abnormal test case similarly derived For the new unknown test cases the network achieved values on the output node of 0.013 for the normal vectors and 0.992, 0.996, 0.944 and 0 992 for the fault cases respectively. This architecture did better than the two layer system in the same four out of five test cases as the system with three hidden PEs, but achieved a higher diagnostic accuracy then the other three layer network in only three out of five cases.

SENSORS	HIDDEN NODES								
	0	3			5				
	0	1	2	3	1	2	3	4	5
PT2	0.88	-0.79	-0.60	0.70	-0.35	-0.50	-0.40	0.18	0.69
PO	3.78	-1.55	-1.30	2.28	-0.05	-0.97	-1.21	0.98	2.75
TT2	-1.61	0.03	0.16	-0.97	-0.54	0.08	0.10	-0.13	-0.33
PLA	0.56	0.49	0.36	-0.02	0.44	0.38	-0.09	-0.01	0.07
PHC1V	1.10	-0.27	-0.07	0.40	0.00	0.00	-0.46	0.01	0.84
PL1H	-1.35	0.76	0.69	-0.91	-0.02	0.54	0.56	-0.81	-1.01
PL2H	-1.02	0.22	0.16	-0.21	-0.33	0.06	0.72	-0.53	-0.64
PL3H	3.57	-1.60	-1.59	2.50	-0.15	-1.27	-0.87	1.45	1.90
P0EC1	0.20	0.32	-0.20	-0.08	0.3	-0.14	-0.05	-0.03	0.09
TL1H	0.84	-0.87	-0.83	0.92	-0.43	-0.75	-0.04	0.32	0.45
TL2H	-0.42	-0.42	0.29	0.49	-0.42	0.29	0.49	-0.23	0.03
TL3H	0.45	-0.26	-0.01	0.04	-0.08	0.05	-0.40	0.58	-0.14
THC1V	0.13	0.30	-0.10	0.28	0.22	-0.09	0.38	0.21	0.07
TC12	-0.68	-0.44	-0.38	-0.34	-0.40	-0.26	-0.41	-0.40	0.25
LVC1A	0.07	0.34	0.14	0.36	0.37	0.25	0.31	0.32	-0.40
LVC1B	2.02	-0.28	-0.54	0.98	0.46	-0.34	-0.64	0.95	1.13
LVC2A	-1.48	0.73	0.71	-1.27	-0.14	0.52	0.35	-0.29	-1.40
LVC2B	1.33	-0.42	0.00	0.86	0.00	0.07	-0.10	0.21	1.18
LVC3A	0.08	0.24	0.09	-0.41	0.13	0.09	-0.30	-0.14	0.21
LVC3B	1.41	-0.15	-0.83	0.07	0.25	-0.75	-0.81	0.47	0.82
LVTC1	-1.14	0.41	0.61	-0.47	-0.24	0.41	0.74	-0.48	-0.63
LV100	0.09	0.09	0.45	0.05	0.06	0.49	0.04	0.41	0.09

Figure 6. Final Connection Weights from Input Nodes to Nodes of Hidden Layer Based on Number of Hidden Nodes

Number Of Nodes In Middle Layer	Normal (train)	Fault (train)	Normal (test)	Fault 1 (test)	Fault 2 (test)	Fault 3 (test)	Fault 4 (test)
0	0.013	0.999	0.014	0.990	0.998	0.936	0.990
3	0.010	0.996	0.011	0.991	0.994	0.973	0.991
5	0.011	0.997	0.013	0.992	0.996	0.944	0.992

Figure 7. Final Output Values for Test Data Based on the Number of Nodes in the Hidden Layer

The results of these tests prove that the network is achieving accurate diagnosis of the health of the ASTF over arbitrary inputs and that the network is able to generalize over all possible faults in this instance Based on the comparisons in Figure 7, it can be seen that a three layer neural network with three hidden nodes achieves a slightly more accurate diagnosis than a two layer neural network The addition of two more nodes in the hidden layer only results in a small increase in performance, but significantly increases the time needed to train the network. Tests conducted using the two layer network seem to indicate that data coming from the ASTF sensors is linearly separable. Based on the increased run time and greater computational complexity it appears that a two layer network is able to diagnose the status of the engine test cell with high enough diagnostic accuracy not to warrant the addition of a middle layer, specifically the addition of five hidden nodes. However, in this case a multilayer neural net proved useful in discovering sensor information that give important insight into monitoring the health of the test cell. Also Figure 3 clearly shows that a two layer neural network is unable to seperate all classes of data, especially in the presence of noise.

The strongest connection weights from the first hidden node to the output node were from sensors PL3H (-1.60), PO (-1.55), and TL1H (-0.87). Using these weights to calculate the intercepts of the hyperplane on a three dimensional graph clearly shows that a multi-layer neural network with three hidden processing elements seperates data that was non-linearly separable. The addition of a middle layer adds diagnostic accuracy to the network.

(5.0) THE NETWORKS ABILITY TO HANDLE NOISE

A great deal of criticism is given to diagnostic systems designed purely around simulated data. Many systems, neural networks among them, do not perform as well on real world data as they did on data from a simulator. One reason for this is the noise that is created when sampling real time sensor data. If a neural network is to be used for diagnosing the status of an engine test cell it must be robust and resistant to noise.

Because of the unavailability of actual engine test data it was necessary to add noise to the data sets created by the ASTF simulator. The original backpropagation program was modified to allow the user to add a certain percentage of noise to both the training set data and to the test set data. Random noise was added to the vectors during every pass through the simulated data in order to better resemble the noise that might be encountered in on-line training and testing of the network Every time any piece of data was used by the network in either training or testing, the random number generator was re-seeded in order to give completely random noise. Tests of the system were conducted with 5%, 10%, 25% and 35% noisy data. The algorithm used for adding noise to previously used data sets was:

$$\text{Noisy Data Value} = (\text{data value}) + (\text{uniform random number}[-1,1]) * (\text{percent noise}) \qquad (5\,0\,0)$$

A comparison of the noise levels for all seven tests sets for a two layer backpropagation network is given in Figure 8 A two layer backpropagation neural network was trained using a training set modified to include 5% noise. Seven test data sets composed of ten vectors (two from the training set and five previously unknowns) were used to test the network. For the unknown normal vector set the network computed an average value of 0.051. For the four unknown fault vector sets the average computed values were 0.972, 0 993, 0.874 and 0.967. All these output values diagnose the condition of the noisy simulated data with a high degree of accuracy.

The network was trained again to include 10% noise. The average output value computed by the net for the normal test vector set was 0 051. For the four fault vector sets the respective average output values were 0 973, 0 993, 0 874 and 0.967. The performance of the network degrades slightly as the level of noise increases For tests conducted with 25% noise the average output value of the network was 0 051 for the normal vectors and 0 973, 0.992, 0.873 and 0.967 for the fault vectors. 35% noise in the training and testing sets resulted in average output values of 0.051 for the normal vector set and 0.973, 0.992, 0.872 and 0.966 for the fault sets respectively.

NOISE	NORMAL TRAINING SET	FAULT TRAINING SET	NORMAL TEST	FAULT TEST 1	FAULT TEST 2	FAULT TEST 3	FAULT TEST 4
5%	0 046	0 998	0.051	0.972	0.993	0 874	0.967
10%	0 047	0.998	0 051	0 973	0.993	0 874	0.967
25%	0 048	0.998	0 051	0.973	0.992	0 873	0.967
35%	0 049	0 998	0 051	0.973	0 992	0 872	0 966

Figure 8. A Two Layer Neural Network Trained and Tested with Noisy Data Over 100 Iterations

Because a multilayer neural network spreads its knowledge in the form of connection weights over more nodes, we would expect that it is less affected by noisy data and would diagnose the status of the ASTF with a higher degree of accuracy. This was indeed the case. The tests for noise in a two layer network were repeated with a three layer neural net with three processing elements in the hidden layer, with results as shown in Figure 9.

For a network trained and tested on vectors modified to include 5% noise the average computed output values were 0.044 for the normal test vectors and 0.970, 0 976, 0.927 and 0.968 for the fault vector sets. For a network modified to include 10% noise in training and testing the average output values did not change from the network trained and tested with 5% noise. The outputs also remained unchanged for a network that included 25% noise. Based on this test we can see that a multilayer neural network is robust and resistant to noise in some instances. A slight degradation in diagnostic confidence appears in training and testing a network to include 35% noise. The average output value for the network was 0.045 for the normal vector set and 0 969, 0.976, 0.927 and 0.968 for the fault vector sets.

By comparing the results in Figure 8 and Figure 9 it can be seen that the three layer network does better on the unknown normal test vector set and fault vector set four, and much better for the third fault test. A two layer network trained and tested with 35% noise for fault test set three computes an average output value of 0.872 whereas a three layer network computes an average output value of 0.927. It can be seen from these results that a multilayer neural network is more robust and resistant to noise than a two layer generalized perception. If training is to be conducted on-line in the ASTF a multilayer neural network should be used

NOISE	NORMAL TRAINING SET	FAULT TRAINING SET	NORMAL TEST	FAULT TEST 1	FAULT TEST 2	FAULT TEST 3	FAULT TEST 4
5%	0 043	0.986	0 044	0 970	0.976	0.927	0 968
10%	0 043	0 986	0.044	0 970	0.976	0 927	0.968
25%	0 044	0.986	0.044	0.970	0.976	0.927	0.968
35%	0 044	0.986	0 045	0 969	0 976	0 927	0.968

Figure 9. A Three Layer Neural Network with Three Hidden Nodes Trained and Tested with Noisy Data Over 100 Iterations

(6.0) THE NETWORKS ABILITY TO CLASSIFY TYPES OF FAULTS

To make this system more useful to test cell engineers it would be helpful for the network to tell us what type of fault is occurring.

The ASTF simulator was used to generate data representative of four types of faults: the ASTF loses one air side compressor during a ramp in test conditions; the ASTF experiences a turbine trip of the only turbine in use; the ASTF loses air side compressors while it is idle; and some unknown fault that has not been diagnosed by the test cell engineers These faults along with vectors representing normal test data are used to compose a training set in the following way: 30 normal vectors, 20 vectors representing the compressor failure during ramp, 20 vectors representing a turbine trip, 20 vectors representing loss of air side compressors while at idle, and 10 vectors representing an undiagnosed type of fault. This combinations was picked by availability of the data and the results discussed in 3.0.

Both a two layer and three layer architecture were tested. In both cases the network was composed of 22 input nodes representing the ASTF sensors and five output nodes representing normal (node 0), compressor failure (node 1), turbine trip (node 2), compressor failure at idle (node 3) and undiagnosed (node 4). Desired outputs were added to the training set with a high value (1) for the node representing the type of fault and a low value (0) for all other output nodes. To make the data more similar to real time test data 10% noise was added. The networks were trained for 100 iterations with a learning rate of 0.2.

After the network was trained it was tested on two vector sets from the training set and five unknowns. Figure 10 shows the results for the two layer network. For the test set composed of normal vectors the value of node 0 was high (0.371) with all other nodes being relatively low. For the first fault test set the value of node 1 representing a compressor failure was high (0.601) with all other output nodes being relatively low. Likewise for fault test two the value for node 1 was high (0.914) while all others remained low. Fault test three contained data taken from the ASTF during a turbine trip and the value for node 2 was high (0.479). Fault test four (compressor snap at idle) had a high value for node 3 (0.826) and a reasonably high value for node 0 (0.312).

A three layer network with three hidden processing units was tested for its ability to classify types of faults After the same training configu ation as the two layer network the net was tested against the same five test cases. The three layer network did poorly at diagnosing normal test vectors for both the unknown normal and normal data taken from the training set. In both cases the network diagnosed the data as being representative of the undiagnosed fault. However, the multilayer network did very well in diagnosing some types of faults. For fault test one the value of node 1 was high (0.535) For fault test two (another compressor failure) the value of node 1 was 0.876. For the turbine failure fault test three the value of node 2 was 0.848 and for fault test four the value of node three was high (0.864) with all other output nodes being relatively low. These results are summarized in Figure 11

Based on these results it would seem that the multilayer neural network can diagnose certain types of faults with high degrees of accuracy. However this network architecture is not able to differentiate between normal data and unknowns. While a two layer network does not give output values with as high a confidence as the multilayer network, it is able to diagnose the status of the system with accuracy and with no error during these tests

OUTPUT NODE	NORMAL TRAINING SET	FAULT TRAINING SET	NORMAL TEST	FAULT TEST 1	FAULT TEST 2	FAULT TEST 3	FAULT TEST 4
0	0 473	0.081	0.371	0 092	0 086	0 072	0.312
1	0.033	0 772	0.081	0 601	0 914	0 002	0 019
2	0 001	0.006	0 001	0 003	0 001	0 479	0 000
3	0 069	0 008	0 019	0 024	0 016	0.135	0 826
4	0 076	0 000	0 079	0 001	0 000	0 000	0 000

Figure 10. A Two Layer Neural Network Classification of Types of Faults with 10% Noise

OUTPUT NODE	NORMAL TRAINING SET	FAULT TRAINING SET	NORMAL TEST	FAULT TEST 1	FAULT TEST 2	FAULT TEST 3	FAULT TEST 4
0	0 279	0 000	0.090	0 001	0 000	0 006	0 097
1	0 012	0 866	0.069	0 535	0 876	0 033	0 003
2	0 000	0 084	0 000	0 043	0 050	0 848	0 048
3	0 057	0 003	0.038	0 015	0 003	0 064	0 864
4	0 514	0 035	0 777	0 062	0 059	0 000	0 067

Figure 11. A Three Layer Neural Network with Three Hidden Nodes Classification of Types of Faults with 10% Noise

(7.0) CONCLUSIONS AND RECOMMENDATIONS

This research has shown that a neural network can be used to differentiate between patterns of data representing a normal test run of the Aeropropulsion Systems Test Facility and sensor data coming from a test cell that is in failure A network was also built to classify the type of fault that is occuring For simulated ASTF data a near optimal diagnosis can be achieved using a two layer neural network (generalized perceptron). However, for simulated data with noise such as would occur in actual engine test cell data, a multilayer neural network with three hidden processing elements yields a more accurate diagnosis The neural network architecture described in this report diagnoses the health of the ASTF with a high degree of accuracy after being exposed to an input vector representing sensor data from only one time slice (0.2 sec) and an even higher accuracy after averaging ten time slices, or two seconds, worth of data.

In addition, by monitoring the converging weights of the input processing elements it was seen that the most relevant information used by the 22 input node network comes from four sensors, P0, PL3H, LVC1B and LVC2A based on connection strengths. Graphing the node connection weights and the total system error for various network parameters shows whether the weights oscillate while approaching error minima.

This research has demonstrated the usefulness of neural networks for fault monitoring and diagnosis. Further research and implementation of neural network based systems will give test personnel a new tool in the anlysis of critical test data. This system will help take neural network technology out of the laboratory and help create commercial systems for the analysis of complex systems. The most significant potential benefit of the system which has been developed is the improved safety of test cell personnel by rapid detection of faulty conditions. This will be of extreme importance as testing begins on more advanced systems such as the National Aerospace Plane and the Advanced Tactical Fighter.

BIBLIOGRAPHY

The following is a list of reference articles and texts containing information relevant to the research discussed in this paper:

Abu-Mostafa, Y.S.: "The Vapnik-Chervonenkis Dimension: Information versus Complexity in Learning" *Neural Computation* (1989).

Bedworth, M.D. and Bridle, J.S.: "Experiments with the Back-Propagation Algorithm· A Systematic Look at a Small Problem". *Royal Signals and Radar Establishment* (1987).

Bruck, J · "A Study on Neural Networks". *International Journal of Intelligent Systems*, Vol. 3, pp. 59-75 (1988).

Carpenter, G A. and Grossberg, S : "ART2: Self-Organization of Stable Category Recognition Codes for Analog Input Patterns". *Applied Optics* (1987).

Casasent, D P : "Neural Network Signal Processor (NSP)" *SPIE Vol 880 High Speed Computing* (1988).

Chow, M.Y. and Yee, S.: "Application of Neural Networks to Incipient Fault Detection in Induction Motors" Department of Electrical and Computer Engineering, N.C. State. Unpublished (1989)

Churchland, P.M. and Churchland, P.S.: "Could A Machine Think?". *Scientific American* (January 1990)

Cox, K S.. "An Analysis of Noise Reduction Using Back-Propogation Neural Networks". Master's Thesis, AFIT (1988).

"DARPA Neural Network Study". Final Report MIT (October 1987-February 1988).

Dietz, W E., Kiech, E.L , and Ali,M.. "Neural Network Models Applied To Real-Time Fault Diagnosis". *Journal of Neural Network Computing*, Vol. 1 (1989).

Hinton, G.E . "Connectionist Learning Procedures". *Artificial Intelligence 40* (1989).

Johnson, C. And Sehert, P.: "The Automated Flight Test Data System". *AGARD Conf. Porc. on Flight Test Techniques*, Neuilly-Sur-Seine, France (1976).

Lippman, R.P.: "An Introduction to Computing with Neural Nets". IEEE ASSP Magazine, Vol. 4 (1987).

Mitchell, J G · "New Test Capability for Propulsion System Testing" *AIAA Paper 73-1283*, AIAA/SAE 9th Propulsion Conf., Las Vegas, Nevada (November 1973)

NeuralWorks Professional II Instruction Manual. NeuralWare Inc. (1988).

Nii, H.P. and Feigenbaum, E.A : "Rule Based Understanding of Signals" *Pattern-Directed Inference Systems* Academic Press (1978).

Pao, Y.H.· *Adaptive Pattern Recognition and Neural Networks* Addison-Wesley Publishing Co. (1989).

Polce, R L.. "A Systems Description of the Aeropropulsion Systems Test Facility (ASTF)" Master's Thesis, UT Knoxville (1982).

Pomerleau, D.A.: "The Meta-Generalized Delta Rule: A New Learning Algorithm for Learning in Connectionist Networks". *Carnegie-Mellon University Report* (1987)

Rumelhart, D.E. and McClelland, J L · *Parallel Distributed Processing Explorations in the Microstructure of Cognition*, Volume 1. Foundations The MIT Press (1988).

Simpson, P K.· *Artificial Neural Systems. Foundations, Paradigms, Applications and Implementations*, Pergamon Press (1990).

Touretzky, D.S. and Hinton, G.E : "A Distributed Connectionist Production System" *Carnegie-Mellon University Report* (1986).

Werbos, P J · "Generalization of Backpropagation with Application to a Recurrent Gas Market Model". *Neural Networks*, Vol. 1, pp. 339-356 (1988).

AN ADA FRAMEWORK FOR THE INTEGRATION OF KBS AND CONTROL SYSTEM SIMULATIONS

M. J. Corbin
G. F. Butler

Royal Aerospace Establishment,
Farnborough, Hants, GU14 6TD, U.K.

SUMMARY

The application of Ada and an object-oriented approach to the design and construction of advanced defence systems are both attracting increasing attention. The Ada language contains some support for object-oriented programming but has some notable deficiencies. This paper shows how to overcome these deficiences by providing a library for object-oriented development in Ada (OODA) which contains facilities to create and manipulate objects and provides support for more general relationships between objects

One of the initial applications of this library has been to design a framework for integrating KBS with control system simulations comprising mixed continuous and discrete time elements. Using this framework it is possible to study the interactions between a Knowledge Based controller and the other more conventional elements of the closed loop to any level of detail required.

1. INTRODUCTION

The object-oriented approach to the development of complex systems is attracting increasing support for its promise of improved maintainability and reusability through modelling the software as a collection of interacting objects. There are a number of object-oriented languages (Simula, Smalltalk, C++ etc.) which all embody different versions of the four basic concepts of object-oriented programming, namely:

Encapsulation: the property that a software object is a completely self-contained entity, possessing all the data and code needed to perform a set of standard operations which form its only interface to other objects.

Data abstraction: the ability to treat the definition of an single object as an abstract entity which can then be used to define many further objects of the same type, all with independent data.

Inheritance: the ability to define new types of object which have some of the properties of old types, and some new properties of their own. These new types can be regarded as specialised variants of the old. **Multiple Inheritance** allows the properties of more than one parent to be combined.

Dynamic binding: a facility of many object-oriented languages which allows flexibility in the specification of object operations. A message can be sent to an object requesting it to perform some action, without having to specify the type of this object until run time.

It is important to emphasise, however, that the use of object-oriented techniques is not dependent on any particular programming language. Rather it is an approach to organising and planning computer programs which can be applied to a greater or lesser extent in all software development. The standard language for defence systems, Ada, contains support for some of the basic concepts of object-oriented programming. Encapsulation can be achieved by the use of packages, and data abstraction by defining abstract data types, instances of which can be replicated (Booch (1)). Inheritance and dynamic binding are not readily emulated, however, and other features, which aid the construction of a world of interacting objects, are entirely absent.

The OODA library described here remedies these deficiencies by providing support for a range of features usually regarded as essential for object-oriented programming, while staying entirely within the Ada language standard. No translators or other preprocessors are needed, in contrast with some other approaches to this problem (Simonian and Crone (2); Kovarik and Nies (3)). The library also supports the definition of more advanced structures and object relationships and has been designed to be as efficient as possible in execution to make it suitable for use in real-time systems. In this paper, we describe the OODA library and initial applications of it to the construction of knowledge based systems and to a framework for mixed continuous/discrete simulations, capable of interacting with these knowledge bases.

2. THE ADA OBJECT-ORIENTED LIBRARY

Within the library, the basic mechanism used to encapsulate the operations and data for an object is the Ada package structure. In practice, provision has to be made to replicate objects of one type any number of times. To achieve this, each package defines a class of objects, and the individual instances of this class are created as needed, and destroyed when needed no longer. The attributes, or data, for these independent objects are all stored within the class package as an array of reusable records. The

maximum size of the set of instances which can exist at any one time is determined at the start of a run, when this class set is initialised. While not being as flexible as a system which allows objects to be replicated *ad libitum*, this approach gives fast and efficient access to the attributes of either an individual object or the set of such objects. It should be noted that the strong typing capability of Ada is used to ensure that all references to objects of a particular class are made via a distinct type. Such a structure allows several categories of error in the use of classes to be identified at compile-time.

2.1 Accessing Objects

An object is uniquely identified by its name and the class to which it belongs. An object can be referred to by name, but, in order to improve efficiency, the principal method for accessing an object is via a privately-typed reference. The reference is assigned when the object is created, and includes both a unique identity for the object, and the location of the object's attributes within its class. The type of the reference is determined by the class of the object. The following fragment of Ada code from a typical defence application shows the creation of an object named "Harrier1", of class Fighter, accessed by a reference H1:

```
H1 : Fighter_type;                  -- define the type of the reference
Create( H1, name => "Harrier1" );   -- create the object
Refuel( H1 );                       -- access to it by reference
```

The alternative method of access is by giving the name of the object and using a specially defined operation, named after the class concerned, to return the appropriate reference, e.g.:

```
Refuel( Fighter( "Harrier1" ));     -- access by name
```

2.2 Inheritance and Multiple Inheritance

Inheritance is the method used in object-oriented languages to allow new classes of object to be constructed by reusing previously defined classes. A new object class can **inherit** the properties of an old one, adding new properties of its own and thus becoming a specialised variant of the old class. This can be thought of as constituting an "is-a" relationship between the two classes, e.g. a Fighter "is-a" Moving-object. The method used to support inheritance in this library is a form of **construction**, as proposed by Taenzer et al. (4) to allow more effective class reuse in Objective-C. When a new (child) object wishes to inherit from an old (parent) class, it creates a new instance of the parent class with the same name as itself, e g..

```
MovObj : Moving_Object_type;
                                    -- define the reference for the parent.
Create( MovObj, kin_of => Self );
                                    -- create the inherited parent object,
                                    -- "kin_of" denotes inheritance, "Self" the child
```

The operations of the parent are then available for use within the child object by reference, or external to it by name or relationship. Multiple inheritance is readily achieved since parent instances can be constructed from any number of classes. Where the parents are themselves derived from a common ancestor class (i.e. repeated inheritance, see Meyer (5)), the library will create only one instance of this class.

2.3 Component Hierarchies

Another way in which complex objects are created is to build them up out of simpler component objects or parts. The component objects are best thought of as being contained within the complex object, though in practice they have a separate existence within their own class, and can be accessed individually when required, e.g. to initialise them. These components can themselves be made up of others, resulting in a hierarchy of component objects.

Componency is more flexible than inheritance as a means of constructing complex objects since a class can be inherited only once, but it is perfectly possible to have more than one component from the same class. Each component is distinguished by a part-name, e.g. "Left_Wheel", "Right_Wheel" and the name given to a object component is made up of the name of its owner appended to its part name, e.g. "Harrier1.Left_Wheel". The components are accessed within the owner object by reference and outside it by name or relationship. For example, within the owner, we can create components and access them thus:

```
W1, W2 : Wheel_type;
    --define references to components
Create( W1, name => "Left_Wheel",  part_of => Self );
Create( W2, name => "Right_Wheel", part_of => Self );
    --create components, "part_of" denotes componency.
```

Outside the owner, name or relationship are used:

```
Inflate( Wheel( "Harrier1.Right_Wheel" ), Pressure );
    --access by name.
Inflate( Wheel( "Right_Wheel", part_of => H1 ), Pressure );
    --equivalent (but faster) access by part name.
```

2.4 Sub Sets and Set Operations

Inheritance and componency are both permanent relationships, holding true for the lifetime of an object. There are, however, many problems in which it is desirable to form temporary associations between groups of similar objects. The OODA library provides this facility by allowing the formation of **Sub-Sets** which can be populated by any selection of the objects in that class. Objects can be added or removed individually, or a new sub-set can be made up of a logical combination of old sub-sets, or from a general object specification.

Sub-sets are particularly useful in the construction of plans for groups of objects, the members of which can be altered as the plan matures. A complete sub-set of objects from one class can be inherited by a single object from another class, allowing one-to-many mappings to be created between objects of different classes. For example, an object describing a meeting could inherit a sub-set of people who are booked to attend it (a Meeting "is-a" Set-of-people). A single object can be included in several sub-sets at the same time, allowing many-to-many mappings to be constructed. In this example, one person could be booked to attend many meetings. Using the library, it is possible to find out who is booked for any two meetings by forming the intersection between their sub-sets.

A further enhancement to aid the manipulation of sets and sub-sets of objects is the ability to define **Set Operations**. Rather than operating on one instance at a time, a set operation can perform the same operation on the members of the whole set of objects in a class, or any of its sub-sets. This can considerably clarify the code, and improve run-time performance by reducing the number of procedure calls needed to perform a given task. Set operations are particularly useful when constructing general facilities, such as inference engines, for which access to the world of objects independently of any specific application software is desirable.

2.5 Additional Features

1. A template, containing a number of pre-defined operations, is provided to simplify the creation of class packages. The specifications and bodies of class packages are compiled separately to minimise recompilation after changes.
2. Different classes can use the same names for operations. The type of the object reference is used to select the right one at compile time.
3. Dynamic binding can be simulated for specific operations by including additional Ada procedures to select the correct operation at run time.
4. An interactive tracing and diagnostic package is included, allowing operations on individual objects to be logged, and their attributes displayed.
5. Objects have extended visibility - they can be accessed by name from any part of a large Ada task. It is possible, however, to prevent critical object operations from being accessed from unauthorised parts of a task by use of the optional access protection facility.

3. APPLICATION TO KBS

This illustrates the use of the OODA library to construct a set of interacting object classes which together constitute an inference engine (see Figure 1). Three classes are defined: Facts, Rules and Rulebases.

The first class defines objects which are **Facts**. Each fact has a name (as do all objects) and an attribute describing the current state of knowledge about that fact, which can be True, False or Unknown. Facts can be manipulated both by inference engines, and by other objects within the overall application program, allowing this class to act as a blackboard system.

The second class defines objects which are **Rules**. These have attributes describing the antecedent facts, and how to combine them using <and, or, not> functions to evaluate the rule. In addition, the attributes contain a list of consequent actions to perform if the rule fires and an explanation of the significance of the rule. The operations for rule class include Evaluate, which inspects the antecedents to see whether or not the rule is capable of firing, and Fire which fires it and performs the consequent actions, as well as several operations needed to define the rules.

The final class defines objects which are **Rulebases**. These are responsible for the overall control of the inferencing procedure for a group of rules. Each rulebase object inherits a sub-set of rules (a Rulebase "is-a" set-of-Rules), has attributes describing the type of inferencing to use with these rules and a reference to a goal fact and a constraint rulebase. Its principal operation is Invoke, used to start the appropriate inference procedure for a rulebase. While inferencing on one rulebase, another one may be invoked, either as a consequence of one of the rules firing, or as a consequence of the operation of the inference engine itself. Another possible consequence of a rule firing is to Queue a discrete event in the simulation framework described in the next section.

Rather than create a new language in which to define the rules, it was felt to be preferable to keep the definition within the syntax of Ada, since it is a well defined standard, supported by a range of software engineering tools. The basic form of a rule appears thus:

```
Start_Rule( <rule reference>, <rulename> );
      IF Established( <antecedents> ) THEN
            <consequents>;
      END IF;
End_of( <rule reference> );
```

The antecedents consist of a list of factnames (in quotes) separated by <and, or, not> operators as appropriate. The consequents comprise a list of operations to be called to redefine facts, Explain the rule, or Invoke another rulebase. The rulebase itself is defined similarly:

```
Create( <rulebase reference>, <rulebase name>,
                              <inference method> ),
      ...
      Rule definitions
      ..
End_of( <rulebase reference> );
```

These Ada definitions are compiled with the class packages for Rule and Rulebase, which overload the standard Ada operations AND, OR and NOT called within the definitions. The definitions are then executed once to initialise the objects before inferencing starts.

4. APPLICATION TO SIMULATION

The other main application of the OODA library so far has been to simulation. Below we describe a mixed continuous/discrete simulation system, constructed using OODA, which can run multiple continuous models under the control of a discrete event scheduler, and allows the models to interact with knowledge bases. Three groups of object classes are involved in this:

a) A class of **states**, which are the results of an integration.

b) Continuous or discrete **models**, which belong to various classes, some supplied by the user, others provided in a support library.

c) **Integration control** objects, inherited by continuous models, responsible for selecting and applying the integration method for that model.

These three types of object interact to provide the simulation facilities.

States are relatively straightforward objects, see Figure 2. They have operations for setting the differential of an individual state, and reading the back the value of that state; these operations are used within the models. The operations which can integrate the differentials to produce the next values of the states are implemented as **Set Operations**, which act on the whole set of states, or any sub-set of it. These operations are invoked under control of the event scheduler, transparently to the models which use each state.

Models contain the equations of motion being simulated, and any calls to the event scheduler to explicitly schedule discrete events. They are generally compound objects, built up of components which are sub-models provided by the user, or filters and transfer functions extracted from a library. Both a model and its sub-models would generally contain states as components.

Integration Controllers are inherited by models and contain the information needed to integrate the model, including the integration algorithm to be used (rectangular and Runge-Kutta 2nd and 4th order methods are currently provided) and the maximum time step. They in their turn inherit a clock object to record the progress of the integration, and a Sub-set of States containing all the states for the model. The inheritance structure is summarised in Figure 3. This use of sub-sets of states allows different models to be integrated independently of each other, with different algorithms and update rates.

As well as integrating a model to produce time responses, it is possible to perform a small perturbation analysis of it to extract its differential equations in linear form. Various forms of s-plane linear analysis techniques can then be applied to predict, for example, the stability and frequency responses of a closed loop control system. So far, stability analysis has been implemented.

Sampled data systems, involving difference equations rather than differential equations, can also be accommodated within the same framework. This allows systems consisting of a sampled-data digital control system controlling a continuous plant to be simulated. Linearisation can also be performed on these sampled data systems, opening the way to z-plane analysis of the overall control loop.

These various types of model can interact with a Knowledge Based System by executing discrete events which can Invoke the inferencing operation of a Rulebase. The Rulebase in its turn can fire rules whose consequents are to Queue the operation of further discrete events. Facts may be altered and inspected by the KBS and by the simulation. This allows the construction of control loops incorporating knowledge based systems as one of their elements. The monitoring facilities in the simulation framework and the diagnostic tracing capability of the OODA library itself can both be used to examine the detailed interaction between the various elements of the control system.

Neural Network simulations have also been constructed using the OODA library, and work is currently under way to investigate the use of neural networks to detect faults in control systems. The networks are trained using the stored responses of the control system model to various inputs and failure cases. The discrete event controller finds an additional use in supervising the progress of this training.

5. CONCLUDING REMARKS

The object-oriented view of software is rapidly gaining support as a flexible, maintainable and intuitive method for structuring complex systems. This paper has described a new approach to the problem of development and maintenance of such software: the OODA library retains all the software engineering properties built into the Ada language and its support environments, while providing a rich set of facilities for use with the object-oriented paradigm. These include support for temporary associations between groups of similar objects (Sub-sets) and the definition of operations which act on all the members of a sub-set (Set operations).

Although intended primarily for use in knowledge-based systems and simulation, the library is not specialised and can be used to support any application for which Ada is appropriate. Initial applications have included an framework for continuous and discrete simulations, inference engines for knowledge based systems and a method by which the two can interact when modelling control systems which incorporate a KBS. Neural network simulation has also been successfully undertaken.

REFERENCES

1. Booch G, Software Components with Ada,
Benjamin/Cummings, Menlo Park, California, 1987.

2. Simonian R, Crone M, InnovAda: True Object-Oriented Programming in Ada, *Journal of Object Oriented Programming, Vol. 1, No. 4, 1988, pp 14-21.*

3. Kovarik V J Jr., Nies S, Supporting Object-Oriented Programming Within Ada: Extending the Paradigm, *AIDA-88, Fairfax, VA, November 1988, paper 13.*

4. Taenzer D, Ganti M, Podar S, Problems in Object-Oriented Software Reuse, *ECOOP 89 Proceedings, Cambridge University Press, July 1989, pp 25-38.*

5. Meyer B,1988, Object Oriented Software Construction, *Prentice-Hall, New York, 1988.*

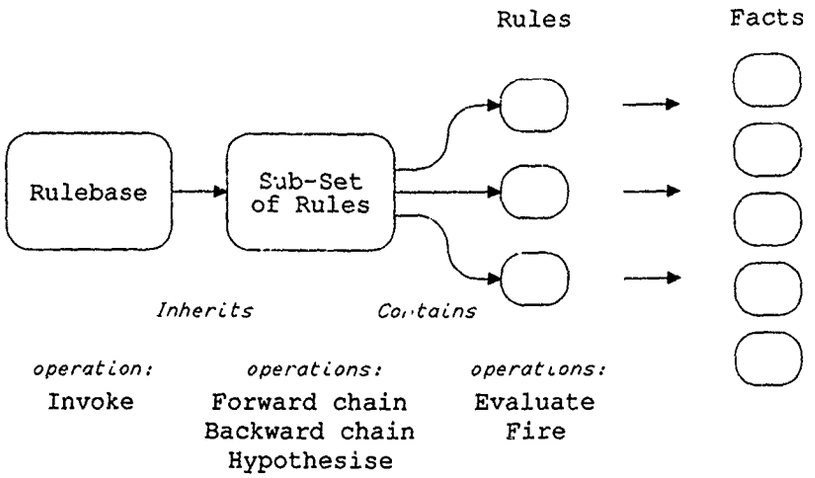


Figure 1. Object Class Relationships within the KBS System

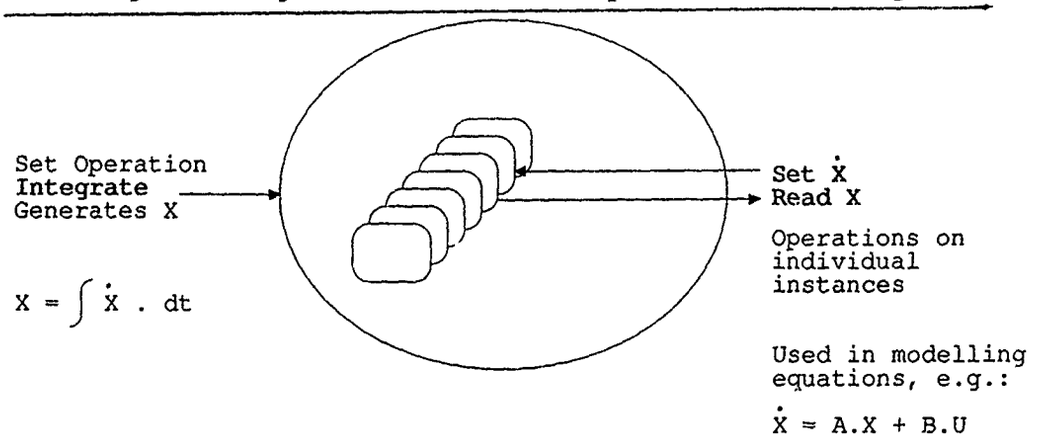


Figure 2 Object and Set Operations for the class of States

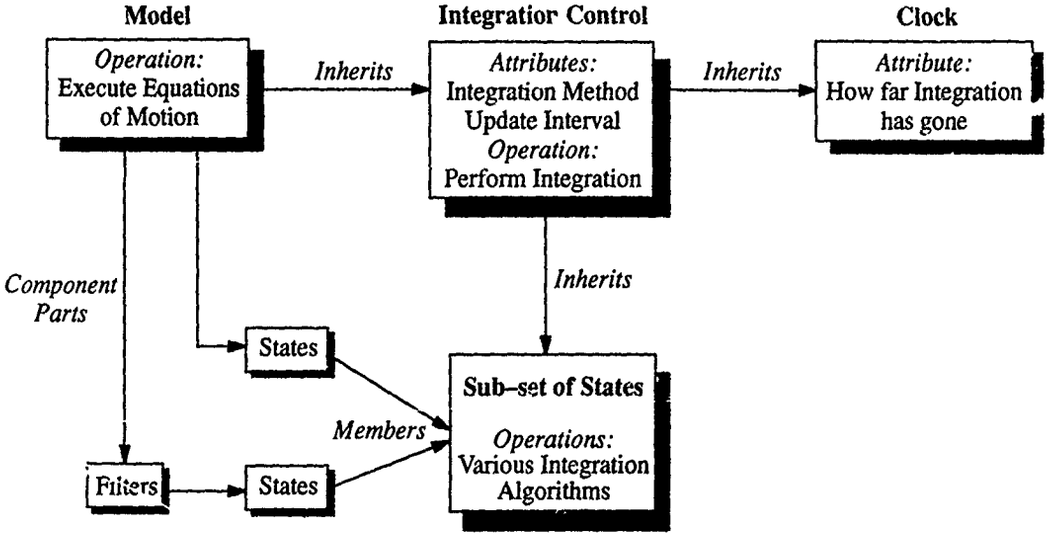


Figure 3. Inheritance Relationships for Continuous Simulation.

LA PROBLEMATIQUE MULTI-AGENTS DANS LE COPILOTE ELECTRONIQUE
(APPLICATION DES SYSTEMES A BASE DE CONNAISSANCES AU GUIDAGE – PILOTAGE)

par

Antoine Gilles
Dassault-Aviation
92210 Saint Clout
France

RÉSUMÉ

La conduite d'un avion de combat est une tâche difficile du type "contrôle de processus complexe" qui nécessite tout particulièrement une maîtrise de la complexité temporelle de la mission. Dassault-Aviation sous l'égide de la DRET[1] (contrat 86-34407) s'oriente actuellement vers la réalisation d'un système interactif d'aide au pilote appelé "Copilote Electronique". Ce projet complexe met en œuvre plusieurs entités expertes. L'existence de telles entités pose des problèmes de gestion de ressources de communication et de coopération. Un superviseur entre les différentes entités expertes du système embarqué, et une architecture logicielle adaptée à l'arbitrage et à la communication de ces entités est donc nécessaire. Cet exposé situe les travaux que nous avons effectués dans ce domaine et les résultats obtenus. Il indique les perspectives que nous envisageons en fonction des difficultés rencontrées.

LA PROBLÉMATIQUE MULTI-AGENTS DANS LE COPILOTE ELECTRONIQUE

Supervision d'entités expertes

La conduite d'un système embarqué lors de missions complexes a toujours posé des problèmes de charge de travail pour un pilote seul. Ces problèmes ont été généralement résolus par l'application de règles strictes de conduite de la mission.

A l'avenir les capteurs et les moyens de communication fourniront des informations de plus en plus nombreuses, donnant au pilote la possibilité d'une véritable gestion de la mission. Cependant, même si le pilote se trouve déchargé des tâches courantes et répétitives, la gestion d'une mission complexe risque de constituer une tâche trop lourde. L'Intelligence Artificielle pouvant proposer des réponses à ces problèmes, Dassault-Aviation travaille au développement d'un système d'aide au Pilote (projet "Copilote Electronique") et au lancement des recherches nécessaires.

---

(1) Direction des Recherches, Etudes et Techniques

Afin de pouvoir décomposer le problème complexe de l'assistance au pilote d'un système embarqué en sous problèmes plus simples et mieux identifiés, une approche résolument "multi-experts" a été adoptée pour le projet. Cette approche permet d'élaborer des entités expertes spécialisées, capables de raisonner sur un domaine d'expertise limité et mieux défini.

La validité des décisions proposées par un tel système doit toujours être envisagée dans le contexte global de la mission. De ce fait une assistance au pilote dans son ensemble ne peut être vue comme une collection de systèmes experts indépendants mais plutôt comme un ensemble d'agents intelligents collaborant à l'élaboration d'une décision. Une telle collaboration pose de nombreux problèmes de gestion temporelle des ressources, de communication d'information entre agents, et d'arbitrage entre décisions.

Le thème des univers multi-agents prend une importance grandissante en IA, les agents pouvant être aussi bien des entités physiques (robots), que logiciels (systèmes experts) ; ils doivent coopérer pour solutionner un certain problème. Ce thème pose des problèmes nouveaux de représentation des connaissances, de raisonnement, d'architecture des systèmes. Il s'inscrit dans le courant général de l'IA distribuée, concept appelé à un développement économique majeur.

Une approche multi-agents, consistant à faire coopérer un ensemble de systèmes de traitement d'informations et de bases de connaissances pour parvenir à résoudre un problème complexe, s'avère fructueuse à plusieurs titres :

- du fait de la nature distribuée de certaines applications : plusieurs acteurs peuvent intervenir; plusieurs sous-tâches peuvent être isolées; les données provenant de divers endroits géographiques et de divers capteurs doivent être fusionnées; etc.
- du fait de la simplification qu'il en résulte dans la résolution d'un problème (technique IA de la réduction de problèmes).

Il existe plusieurs façons d'implanter un système multi-agents, notamment les langages d'acteurs et le modèle de blackboard, ce dernier assurant une plus grande variété de structures de contrôle. Les langages "orientés objets" sont très intéressants pour réaliser pratiquement un tel modèle.

Quelles que soient les implantations retenues, l'existence et le contrôle des diverses entités expertes nécessitent la présence d'un superviseur qui puisse gérer dans le temps les traitements des entités informatiques. Nous avons élaboré un premier module de superviseur dans la maquette actuelle. Nous le décrivons ci-après.

Maquettage

La coopération entre différentes sources de connaissance est l'un des problèmes majeurs auxquels l'Intelligence Artificielle est confrontée dès lors qu'elle s'attaque à des problèmes de grande ampleur.

Pour évaluer la faisabilité d'un Copilote Electronique, basé sur des techniques d'intelligence artificielle, une maquette s'avérait indispensable. Une maquette a donc été réalisée. Elle constitue un outil de **développement** et d'**évaluation** des aides à la décision expertes proposées par le Copilote Electronique.

Cette maquette s'appuie sur une simulation simplifiée mais réaliste de l'environnement opérationnel permettant de "jouer" divers scénarios de mission.

1. Une partie du logiciel réalise les fonctions d'environnement expert, destinées à définir et intégrer de nouvelles logiques de raisonnements, à vérifier leur cohérence, et à les valider. Cela nécessite des possibilités de saisie, de visualisations, et de contrôle de l'expertise.

2. Une autre facette de la maquette comprend les fonctions d'environnement opérationnel, permettant de tester, d'évaluer et de valider les résultats des raisonnement experts, en terme de qualité opérationnelle comme en terme de temps de réaction. Cette partie nécessite une simulation simplifiée mais réaliste du SNA et du théâtre tactique.

3. On assure des fonctions d'environnement informatique, permettant outre le lancement de l'application, l'archivage des sessions en vue du dépouillement des résultats, le monitoring des sessions ainsi que la gestion de plusieurs versions des expertises codées dans la maquette.

4. Enfin on dispose de fonctions de communication permettant de dialoguer et d'échanger des informations entre simulation opérationnelle, simulation du SNA, Pilote et Copilote Electronique. Ces logiciels étant réalisés sur machines différentes, le dialogue établi consiste à un échange "en ligne" des informations en provenance de machines hétérogènes.

Nous ne décrivons ici que les aspects d'environnement expert, et plus particulièrement la supervision et le contrôle des entités expertes.

FONCTIONS D'ENVIRONNEMENT EXPERT

Le développement d'un logiciel expert comme le Copilote Electronique est caractérisé par la continuité de l'enrichissement du logiciel entre la phase d'élaboration des logiques de raisonnement et la phase d'évaluation et de validation de ces logiques. L'ensemble de ces fonctions représente la partie "embarquable" du concept de Copilote Electronique, à l'exclusion des traitements déjà présents dans le SNA. Ces fonctions s'exécutent en parallèle des simulations de manière à pouvoir évaluer leur applicabilité au problème temps réel.

Dans le cadre de la maquette, nous avons retenu un ensemble cohérent de fonctionnalités (parties grisées), respectant l'architecture fonctionnelle globale présentée ci-dessous. Cet ensemble permet de présenter des aides à la prise de décision tactique. Il illustre par ailleurs le principe d'échange et de coopérations entre les entités expertes retenues. L'ensemble des fonctionnalités retenues est le suivant :

- Niveau Réflexion :
  - Evaluation de la situation tactique
  - Evaluation des plans de mission
- Niveau Décision :
  - Gestion tactique

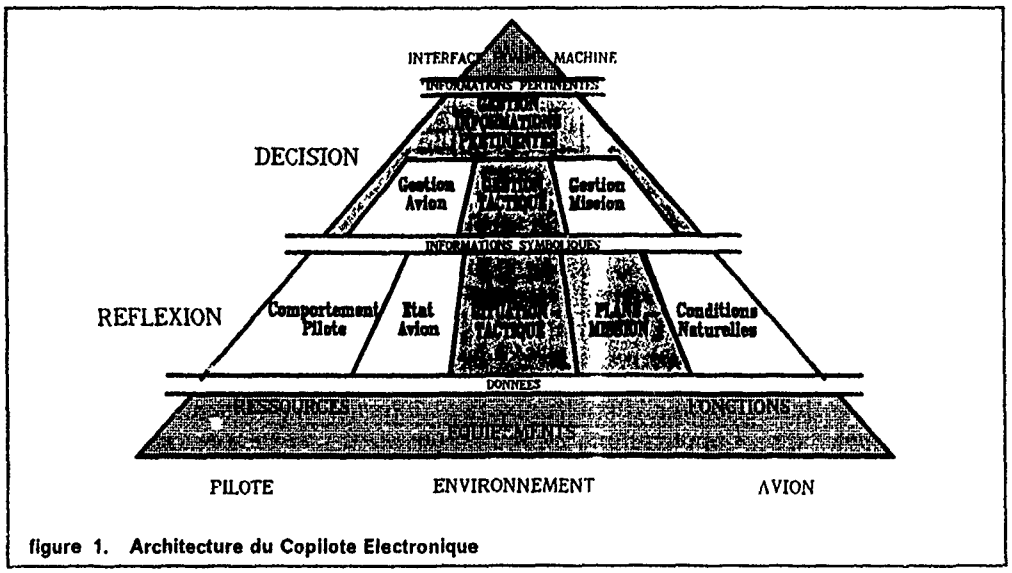


figure 1. Architecture du Copilote Electronique

- Gestion des informations pertinentes

Description fonctionnelle

Niveau Réflexion

Ce niveau regroupe les raisonnements permettant de manipuler l'ensemble des données disponibles sur l'environnement réel du décideur. Il correspond à l'activité d'*information*, qui consiste à traduire sous forme symbolique les phénomènes observés, ainsi qu'à l'activité de *prévision* qui consiste à construire des schémas du futur probables. On élabore ainsi une compréhension de l'environnement opérationnel, ainsi que des prédictions d'évolution de la mission. La finalité de ces raisonnements est d'élaborer des éléments de décision riches et efficaces à l'usage du niveau de décision.

1. **Evaluation de la situation tactique**

L'évaluation de la situation tactique revient à une corrélation "intelligente" continuellement rafraîchie de toutes les informations disponibles sur le cadre opérationnel (objectif, dispositif ennemi, dispositif ami). Cela suppose :

a. la synthèse des informations disponibles,
b. l'analyse des éléments de la situation tactique,

2. **Evaluation des plans de mission**

Cette tâche maintient en permanence un ensemble de plans de mission envisageables pour la poursuite de la mission. Pour cela, elle modélise les divers plans issus de la préparation de mission ou d'une replanification en vol. Elle évalue les plans de mission suivant deux critères qui sont le risque d'une part et l'efficacité d'autre part. Le risque correspond à la possibilité de ne pas mener la mission à son terme. L'efficacité correspond au résultat attendu de l'attaque pour un plan de mission donné en fonction de l'objectif

fixé en préparation de mission ou en cours de vol. Les fonctions assurées sont les suivantes :

a. Mise à jour des plans de mission
b. Mise à jour des m rges
c. Mise à jour des risques
d. Information du reste du Copilote Electronique

Niveau Décision

Ce niveau regroupe d'une part les raisonnements permettant de gérer l'ensemble des informations élaborées par le niveau de réflexion suivant les types de décisions à prendre. L'activité d'*élaboration* consiste à traduire les intentions en actes envisageables ou propositions d'actions, l'activité d'*exécution* consiste ensuite à interagir avec le pilote et avec le système de manière à produire des résultats efficaces, et agir sur le monde réel.

On génère des propositions d'actions adaptées à l'environnement opérationnel et permettant de réaliser efficacement la mission. La finalité de ces raisonnements est de fournir des informations pertinentes (propositions d'actions ou critères de décision) en fonction de la situation et des besoins du décideur. A ce niveau sont élaborées et choisies les solutions aux divers problèmes rencontrés pendant le déroulement de la mission, c'est le niveau des modules de gestion.

On génère des propositions d'actions adaptées à l'environnement opérationnel et permettant de réaliser efficacement la mission. La finalité de ces raisonnements est de fournir des informations pertinentes (propositions d'actions ou critères de décision) en fonction de la situation et des besoins du décideur.

- Les modules gestion avion, gestion tactique et gestion mission traitent en entrée les informations symboliques et fournissent en sortie d'autres informations symboliques de type proposition d'action ou explication de décision.
- Le module de gestion des informations pertinentes traite en entrée les informations symboliques et fournit en sortie des informations pertinentes de décision ou de synthèse, ainsi que des informations symboliques sur le dialogue en cours. Ces informations sont dites pertinentes car elles ne sont dirigées vers le pilote via l'interface qu'à bon escient, c'est à dire au bon moment et par le bon support.

A ce niveau sont élaborées et choisies les solutions aux divers problèmes rencontrés pendant le déroulement de la mission, c'est le niveau des modules de gestion.

1. **Gestion tactique**

Ce module détermine les actions adaptées aux tactiques défensives et offensives face à des menaces ou des objectifs en fonction du cadre opérationnel rencontré au cours de la mission (mise en œuvre d'une arme, procédure d'attaque, affectation de cible, manœuvre en duel aérien, évasive face à un missile, brouillage et leurrage...). Le traitement à effectuer et l'ampleur des logiques mises en œuvre dépendent de la pression temporelle.

a. Détermination de la pression temporelle

b. Sélection de la menace à traiter
c. Choix de parade
d. Détermination des parades compatibles
e. Détermination de l'efficacité des parades sélectionnées
f. Détermination de parades hors marge si besoin
g. Envoi des proposition de parades

2. **Gestion des informations pertinentes**

Il s'agit de gérer les échanges Copilote - SNA, et en particulier, les échanges Copilote - Pilote, via l'IHM. Ce module détermine les actions du Copilote Electronique et gère le dialogue et les passages de commandes qui en découlent. Pour cela, il gère les informations symboliques pour fournir au pilote les informations considérées comme les plus importantes à tout moment. Il gère les éventuels conflits de décisions en prenant en compte les facteurs humains dans l'assistance au pilote. Il définit le type de dialogue le plus adéquat en fonction de l'activité du pilote, de l'urgence de la situation et de l'état de l'Interface Homme-Machine. Enfin, il permet de varier le niveau d'aide à apporter au pilote, en définissant dynamiquement les actions possibles en automatique.

a. Filtrage des informations symboliques,
b. Dialogue pilote / Copilote Electronique.

Supervision des traitements

Niveau Réflexion

Les modules du niveau de réflexion doivent apporter une vision globale et cohérente du monde, et maintenir cette vision incrémentalement. Ceci amène naturellement à penser à une architecture à base de blackboard, permettant la coopération de diverses entités expertes. Les problèmes de définition d'une architecture de blackboard pour le niveau de réflexion du Copilote Electronique sont essentiellement les suivants :

- problèmes liés à une architecture parallèle et non pas séquentielle, comme la majorité des systèmes à base de blackboard actuels,
- problèmes liés à la coopération entre une architecture à base de blackboard pour le niveau de réflexion, et une architecture à base d'interruptions pour le niveau de décision (voir plus loin),
- roblèmes liés à l'intégration dans un univers évolutif et à des raisonements non monotones.

oints d'étude font l'objet d'un contrat DRET[1] avec l'Onera[2] (89-34347). A l'heure actuelle, une solution simple de contrôle par niveau de priorité a été retenue. Cette solution sera bien entendu améliorée en fonction des résultats de l'étude proposée.

Niveau Décision

A la différence des modules du niveau de réflexion, les modules du niveau de décision sont beaucoup plus indépendants les uns des autres, ils n'ont pas

---

(2) Office National d'Etudes et de Recherches Aérospatiales

besoin de communiquer beaucoup entre eux pour mener à bien leur tâche respective ("loosely coupled"), ce qui amène à concevoir une architecture de contrôle différente.

Pour chaque module de décision, il est nécessaire d'évaluer le temps de décision disponible, donc de déterminer quelle est la *pression temporelle* pour une situation donnée. En effet il faut avoir un ordre de grandeur du temps dont on dispose pour élaborer une décision. De plus, il faut pouvoir interrompre un raisonnement lorsque le temps imparti est écoulé. Actuellement, on distingue trois valeurs pour la pression temporelle :

L'urgence Elle caractérise toute situation nécessitant un raisonnement rapide conduisant à des actions réflexes consécutives à des risques majeurs ayant fait l'objet d'une analyse préalable. Cette analyse a pu avoir lieu antérieurement au cours de l'exécution de la mission (prévision) ou être mémorisée (conditionnement) dés la conception.

La pression temporelle moyenne Elle intervient dans les cas où, bien que la situation soit assez critique, il est possible d'entreprendre des raisonnements plus fouillés conduisant à des actions à court terme. Après avoir analysé les différentes possibilités, on procède rapidement à une sélection d'un des comportements possibles.

La pression temporelle faible conduit à une planification d'ordre stratégique sur la base d'informations plus raffinées et conduisant à des actions complexes. On peut s'y livrer à une analyse approfondie de la situation et en déduire des mises à jour sur le plan de mission tenant compte des diverses répercussions à moyen et long terme. C'est notamment en pression temporelle faible qu'il est possible de préparer les comportements réflexes à adopter dans les situations d'urgence en prenant en considération suffisamment de paramètres.

En fonction des événements ou des modifications induisant des répercussions importantes (diminution d'efficacité ou risque important), on détermine le module induisant la pression temporelle la plus importante. Le contrôle lui est alors transmis pour effectuer les raisonnements d'aide à la décision. La notion de pression temporelle sert ici une première fois pour la résolution de conflits *externe*, c'est à dire pour choisir le module devant apporter l'aide à la décision la plus urgente.

Le terme "*imprévu*" apparaissant dans ce document doit être pris au sens large comme définissant tout événement imprévu, devant faire l'objet d'une prise en charge par le Copilote dans l'un de ses modules de décision (Gestion Tactique, Gestion Mission, Gestion Avion et Gestion des Informations Pertinentes). La prise en charge de tels événements doit aboutir en fin de traitement à une proposition de décision à transmettre au pilote.

- **<u>Séquencement et priorités</u>**

  La décomposition fonctionnelle du niveau de décision fait apparaître des tâches d'aide à la décision pour des domaines d'expertise indépendants les uns des autres. Par ailleurs, chaque tâche élémentaire d'un module d'aide à la décision fait appel à des traitements informatiques indépendants. Il s'agit ainsi d'activer des traitements pour toutes les tâches connues à un instant donné. Les problèmes devant être pris en compte sont les suivants :

  - Détermination de la pression temporelle liée à chaque événement imprévu.

- Détermination de l'imprévu à traiter.
- Lancement du ou des traitements à appliquer.

Cependant le temps nécessaire à l'élaboration d'une proposition d'action est essentiellement variable. Il est nécessaire de tenir compte de la pression temporelle, donc du temps imparti, pour décider de lancer ou non des traitements élaborés. Lorsque le temps imparti est écoulé (et qu'il faut présenter à tout prix une proposition d'action), le module doit achever ses traitements en fournissant un résultat, fût-il de moins bonne qualité : une parade très efficace mais proposée trop tard n'est en fait d'aucune utilité.

Pour permettre un comportement adaptatif et opportuniste, dont les temps de réponse soient maîtrisés, il faut pouvoir disposer des fonctionnalités suivantes :

- Interruptions des raisonnements et des traitements selon les changements d'environnement et de pression temporelle associée.
- Interruptions des raisonnements et des traitements en cas de dépassement du délai imparti.
- Masquage des interruptions dans certaines phases critiques.

- **Cas nominal**

Reprenons et analysons chacun de ces points dans le cas d'un fonctionnement nominal, c'est à dire sans perturbation des traitements :

1. Détermination de la pression temporelle de chaque événement imprévu

   Ce niveau de pression temporelle, prenant trois valeurs (faible, moyen, fort) est déterminé par le niveau de risque induit par l'événement à traiter, la position de l'avion, l'urgence de la décision.

   Cette tâche peut être déclenchée à toute arrivée de nouvel imprévu susceptible d'avoir des répercussions importantes, et d'augmenter la pression temporelle ; elle peut être considérée comme une tâche privilégiée, pouvant se déclencher à tout moment et pouvant modifier les traitements. A la fin de cette tâche, la main est rendue au traitement précédemment en cours.

2. Détermination de l'événement imprévu à traiter

   Il s'agit d'ordonner l'ensemble des imprévus à traiter, en les indexant par leurs niveau de pression temporelle, afin de choisir l'imprévu à traiter, qui devient "l'imprévu courant". Il faut éventuellement décider de suspendre le traitement de l'imprévu en cours (imprévu 1) pour traiter un imprévu plus prioritaire (imprévu 2).

   - Dans ce cas il faut donner la main à ce nouvel imprévu (2). Lorsque cet imprévu aura été traité, la tâche "Choix de l'imprévu" sera réactivée pour décider de l'imprévu à traiter.
   - Dans le cas contraire on revient au point du traitement où l'on s'était arrêté pour l'imprévu 1.

3. Traitement de l'imprévu

   Au cours de cette tâche sont lancés successivement des traitements qui ont pour but de proposer une décision. Ces différents traitements

correspondent à des étapes fonctionnelles du raisonnement ; ils sont implantés de façon telle qu'ils affinent progressivement la proposition de décision. De cette façon ils peuvent être interrompus à tout moment en cas de temps imparti écoulé, et fournir un résultat.

- **Cas d'exceptions**

Il est nécessaire de définir un système d'interruptions de raisonnements qui doivent se déclencher dans les cas suivants :

- *lorsque la pression temporelle induite par l'événement imprévu augmente en dynamique*.

  C'est le cas par exemple lorsqu'une menace tire son missile et donc provoque une augmentation du risque, ou lorsque le segment courant devient celui de la menace, induisant une pression temporelle forte. Dans ce cas il peut être nécessaire d'abréger les traitements en fournissant une proposition de parade sans délai.

- *lorsque l'imprévu à traiter change*. (Un autre imprévu présente une pression temporelle supérieure à l'imprévu courant).

  C'est le cas lorsqu'un événement grave ou une menace survient pendant un traitement de moindre importance. La pression temporelle de l'imprévu dont le traitement est en cours n'est plus la pression temporelle la plus élevée. Il faut donc stopper le raisonnement en cours (quitte à le reprendre plus tard), pour élaborer une proposition de parade sur le nouvel imprévu.

- *lorsque le temps imparti est écoulé*.

  Le système doit impérativement proposer un résultat dans un délai déterminé à l'avance. A l'échéance fixée le traitement doit s'interrompre (tout traitement ultérieur ne serait d'aucune utilité et donc serait du temps machine perdu).

Toutefois, il est nécessaire de prévoir la possibilité de masquer ses interruptions du raisonnement. Sans cela le module risque de s'interrompre continuellement dans certains cas, et donc de ne jamais pouvoir fournir le moindre résultat.

- Dans les premier et troisième cas d'interruptions, le raisonnement est en fait stoppé ou abrégé pour fournir une proposition de résultat. Le module rend ensuite la main pour d'autres traitements. L'interruption ne pose pas de problèmes de masquage.

- Dans le second cas, le masquage d'interruptions répété est garanti par le nombre fini de valeurs de la pression temporelle. Si le module était en train de traiter un imprévu sous pression temporelle faible, il risque au pire d'interrompre ses traitements deux fois : une première fois pour un imprévu sous pression temporelle moyenne et une seconde fois pour un imprévu sous pression temporelle forte.

  Ensuite le module masque toute nouvelle interruption : l'idée sous-jacente est qu'en cas d'existence de plusieurs imprévus induisant toutes une forte pression temporelle, il vaut mieux continuer le raisonnement sur celui pour qui on est le plus avancé, donc sur l'imprévu en cours de

traitement, car c'est pour lui qu'on a le plus de chances de trouver rapidement une proposition d'action acceptable. De plus, cette notion de poursuite du raisonnement est conforme à l'idée d'une ligne de raisonnement unique, pour le pilote comme pour le copilote électronique, pendant les phases critiques.

PERSPECTIVES

- **Un système "Human-like"**

Les systèmes d'aides se différencient des automates de par la présence de l'opérateur, véritable acteur temps qui doit faire face à la complexité des systèmes et agit directement sur la dimension temporelle du processus. Il est donc important de tenir compte de cette dimension dans l'interaction du pilote et du "Copilote Electronique" car il semble qu'une grande partie des erreurs humaines soit due à une mauvaise appréhension du temps.

Nous travaillons dans ce sens avec le CERMA[3] pour étudier un modèle de comportement du pilote, reprenant des heuristiques humaines de gestion du temps.

- **Spécifications temps réel**

Les mécanismes d'interruptions sont inspirés de ceux utilisés dans les systèmes temps réel. Cependant l'analogie s'arrête là, nous n'envisagerons ici que les contraintes de synchronisation sur une horloge (qui dans notre cas n'est pas temps réel). La conception de modules experts capables d'interruptions et de synchronisation sur une horloge donnée permettra par la suite d'envisager des capacités temps réel au sens habituel du terme (c'est à dire dont les temps de réponse sont non seulement maîtrisés, mais aussi et surtout très courts).

La description de tels mécanismes n'est pas une spécification *logicielle*. On n'a pas décrit comment devaient être codés les mécanismes présentés, mais seulement les propriétés que devaient vérifier les mécanismes d'aide à la décision du Copilote Electronique. Le fonctionnement du contrôle des activités d'aides à la décision est guidé par la notion de pression temporelle. Il s'appuie sur des mécanismes d'interruptions. Dans ce cadre, la gestion des tâches en concurrence informatique peut s'apparenter à une gestion de système d'exploitation.

- **Optimisation d'architecture**

Enfin, nous menons avec l'Onera[2] une étude visant à améliorer l'architecture multi-expert actuellement retenue pour le Copilote Electronique, au moyen d'un simulateur. Il permettra de choisir les granularités des traitements, d'analyser les flux et les interactions entre les entités expertes et de maîtriser la complexité de la supervision

---

[3] Centre d'Etudes et de Recherches de Médecine Aérospatiale

EVALUATION OF THE OPTIMAL HOMING POINT FOR MISSILE GUIDANCE

by

B. Midollini, P.L. Torelli, G. Balzarotti
FIAR S.p.A.
Via G.B. Grassi 93 - 20157 Milano - Italia

1. INTRODUCTION

One of the main problems arising in the field of missile guidance is the automatic search and detection of targets, in order to gather the necessary information for the correct homing of the missile.

This problem is commonly approached by mounting a seeker (usually an IR seeker) with an image and data processor on board of the missile.

The ground scene, as recorded by the seeker, will present isolated targets and target formations, together with a high level of clutter which produces a number of false alarms. Thus, it is necessary to provide the image and data processor of the missile with algorithms which can automatically eliminate the clutter and the false alarms and detect the true targets.

Furthermore, in order to have an effective shot, only one formation among the various in the scene must be cued to the navigation and the weapon systems of the missile, the choice of it depending on the shape, the disposition and the distance between targets of the formation itself.

This paper presents an algorithm developed to evaluate the optimal point for releasing the ammunition. The algorithm is further illustrated by applying it to a practical case and showing the results of this simulation.

The paper is based on the work carried out by Italy within the NATO Research Study Group on Image Processing, RSG.9 - Project 4.

2. BACKGROUND AND GENERAL STRUCTURE OF THE ALGORITHM

The starting point for the algorithm is a file of data obtained from the preprocessing of a sequence of images.

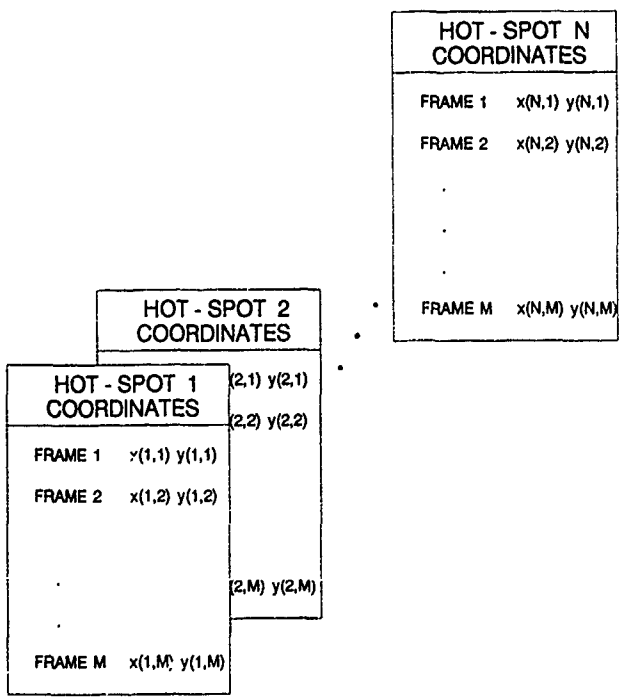


FIGURE 1 - INPUT FILE

The length of such a sequence depends on different aspects, such as the dynamics of the recorded scene (the higher the dynamics, the shorter the sequence), the action range of the missile and its flight characteristics (altitude, speed), the computing speed of the processor on board of the missile.

The preprocessing of the images, consisting of segmentation, filtering, thresholding, labelling, background registration and prediction of the position of those targets which are temporarily obscured (for example by smoke or vegetation) in some frames of the sequence, will supply the algorithm with the labelled coordinates of the detected hot-spots.

The analysis will be performed on this set of data, schematically shown in Figure 1.

The mathematical model used is based on the consideration that targets belonging to the same formation show some similar features, such as speed and mutual distance. For this reason the algorithm establishes, also by means of a comparison with predefined data and characteristics, which is the range of values for these figures that characterizes each group, thus being able to reject false alarms, to identify isolated targets and to group the remaining targets in formations.

Another comparison between predefined data of the missile weapon system and some evaluated figures will finally lead to the location of the optimal formation and, within it, of the optimal point for munitioning.

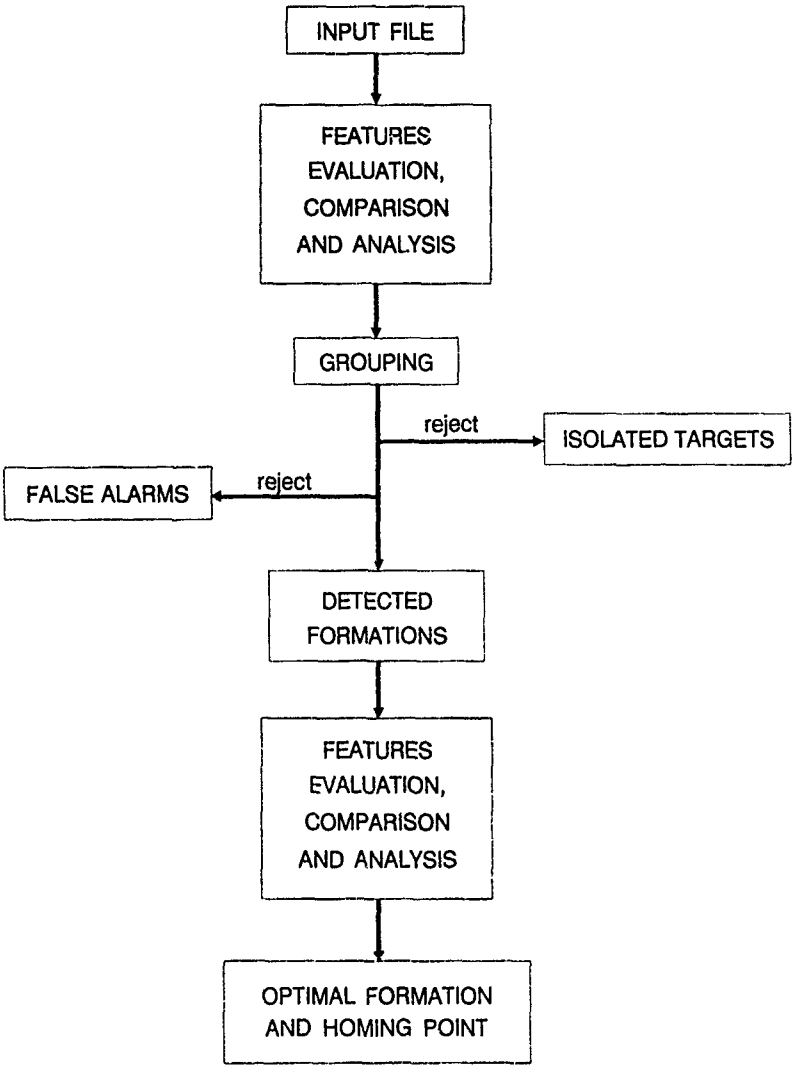


FIGURE 2 - ALGORITHM GENERAL STRUCTURE

Figure 2 shows the general structure of the algorithm. The calculation performed and the criteria used to implement the grouping process are discussed in details in the subsequent sections.

3. INTER-FRAMES ANALYSIS: GROUPING BY SPEED

During the first step of the algorithm, the hot-spots are grouped according to the similarity of their speed.

For each point i of the input file, the algorithm evaluates the relevant total displacement s(i) (i.e. the distance, in pixels, covered by the point throughout the sequence) as the sum of the frame by frame displacements.

The partial displacements are the instantaneous speed values (in pixels/frame) of the points: each of these figures is compared with a predefined maximum speed value, available to the data processor and evaluated as a function of the IR seeker altitude over the ground, of its Instantaneous Field of View, of the frame time and of the foreseen maximum target speed on the ground (km/h).

Those hot-spots having a speed value greater than the predefined one are rejected as false alarms.

The remaining points are then subdivided into groups by means of the method described in the following.

The mean value, $\mu(s)$, and the standard deviation, $\sigma(s)$, of the set of total displacements are evaluated:

$$\mu(s) = 1/N \sum_{i=1}^{N} s(i) ,$$

$$\sigma(s) = \sqrt{1/N \sum_{i=1}^{N} [s(i) - \mu(s)]^2} ,$$

where:

i = label of the hot-spot,
N = total number of hot-spots.

If the standard deviation is small:

$$\sigma(s) < \sigma^*(s),$$

where $\sigma^*(s)$ is a predefined threshold value, then the hot-spots show similar displacement values in the sequence, i.e. they have similar speed, and they are considered to belong to the same group.

If, on the contrary, the standard deviation value is greater than the threshold level, the points are subdivided into groups, according to the positive or negative sign of the difference:

$$[s(i) - \mu(s)] .$$

In this second case, the process is repeated once more on each set obtained, to confirm the validity of the grouping, or to perform a further subdivision.

Isolated targets obtained, in case, after the subdivision are rejected. A rejection can be performed also on groups of two targets.

Simulation trials carried out to test the algorithm have shown that a threshold value that can be used for the standard deviation is:

$$\sigma^*(s) \simeq 0.2 \cdot \mu(s).$$

4. FRAME BY FRAME ANALYSIS: GROUPING BY DISTANCE

In the second part of the algorithm the analysis proceeds frame by frame and separately on each group identified, in case, in the previous step.

In each frame of the sequence the algorithm evaluates the mutual distances of targets and, following the assumption that targets belonging to the same formation have homogeneous distances, subdivides the targets accordingly.

To accomplish this task, the distances between each target and all the others are evaluated and, for every target j, only the shortest, d(j), is retained. Proceeding this way, all the M targets are connected in a chain of (M-1) sides, in which every point is linked to its neighbour. Within this chain, targets can be classified as:

- End Point — if the point is linked only to one other point;
- Intermediate Point — if the point is linked to two other points;
- Branch Point — if the point is linked to more than two other points.

Figure 3 gives an example of a chain of targets: End Points are marked with EP and Branch Points with BP, the other points are Intermediate Points.

The analysis proceeds with the evaluation of the mean value, $\mu(d)$, and the standard deviation, $\sigma(d)$, of the set of minimum distances:

$$\mu(d) = 1/(M-1) \sum_{j=1}^{M-1} d(j) ,$$

$$\sigma(d) = \sqrt{1/(M-1) \sum_{j=1}^{M-1} [d(j) - \mu(d)]^2} .$$

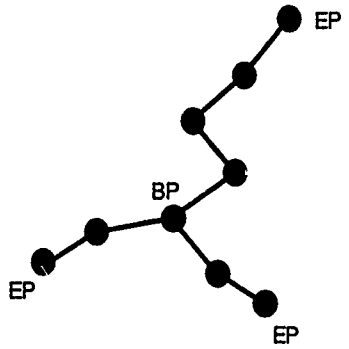


FIGURE 3 - CHAIN OF TARGETS (BP=Branch Point, EP=End Point)

If the standard deviation is small:

$$\sigma(d) < \sigma^*(d),$$

where $\sigma^*(d)$ is a predefined threshold value, then the targets show similar distances, i.e. they belong to the same formation, and the analysis proceeds on the next frame.

If, on the contrary, the standard deviation value is greater than the threshold level, then a counter K relevant to the group under examination is initialized and the following positive distances are evaluated:

(1) $$| d(j) - \mu(d) | \qquad j = 1,\ldots,M-1.$$

If, for a distance d(j), the value of the difference in (1) is less than $(1+Kc)\cdot\sigma(d)$, where $c > 0$ is a predefined value, then the corresponding link is retained. On the other hand, differences greater than $(1+Kc)\cdot\sigma(d)$ are analyzed and a cut is produced in the chain on the link relevant to the largest difference.

The whole process is repeated iteratively on each group obtained, incrementing of one unit the relevant counter at each step, until the current value of $\sigma(d)$ is smaller than $\sigma^*(d)$.

When a cut produces an isolated target, i.e. the cut is made on the link of an End Point, this target is rejected; once again it can be decided to reject also groups made up of only two targets.

The groups of targets obtained after the execution of the described process represent the formations of the current frame. The analysis proceeds analogously on each frame of the sequence.

Simulation trials carried out to test the algorithm have led to the following settings of the predefined constants:

$$\sigma^*(d) \simeq 0.2 \cdot \mu(d) ,$$

$$c \simeq 0.2 .$$

The counter K has been introduced in the calculation to reflect the decrease of probability of further sub-divisions in the current group, as the number of performed iterations increases.

5. ASSIGNMENTS OF THE TARGETS TO THE FORMATIONS

As it has been shown in the previous section, the execution of the algorithm, to this extent, leads to the determination of the formations present in each frame.

However, due to the movement of the targets, different sub-divisions may be operated by the processing in the various frames. For this reason, a track is kept of all the configurations obtained, so that, at the end of the sequence, the algorithm is able to decide to which formation each target must be assigned to.

This is accomplished by means of the compilation of a Frequency Table in which the algorithm records the number of presences of each target in the formations. Each presence is also weighted with a factor F that increases with the position of the current frame in the sequence: the closer the frame to the end of the sequence, the higher the factor F.

At the end of the sequence, those targets having the sum of their weighted presences lower than a certain predefined value P will be rejected, and each of the remaining targets will be assigned to the formation in which the sum of its weighted presences is higher.

In the simulations performed, the following values for P and for the weighting factor have been used:

$$P = 0.3\ L\ ,$$

$$F = 1 + 1/L\ ,$$

where:

l = current frame number,
L = total number of frames.

6. EVALUATION OF THE OPTIMAL HOMING POINT

The last part of the algorithm is devoted to the individualization of the most suitable formation to be shot and, within it, of the optimal point of homing.

This is accomplished by comparing the predefined data of the missile weapon system with the corresponding characteristics of each formation, and associating a "score" to every successful comparison.

One characteristic to compare is the distance of the targets: the predefined datum is, in this case, a range of values in which $\mu(d)$ should fall, evaluated (in pixels) as a function of the IR seeker altitude over the ground, of its Instantaneous Field of View and of the distance (in meters) of the points shot by the munitions. The distance between End Points can be considered too: this, together with the eventual presence of Branch Points, gives an idea on how much the formation is stretched out.

A check will be done also on the number of targets belonging to the formations, determining the optimal one with respect to this feature.

Finally, the shape of the formations will be recognized by evaluating the angles formed by the targets, taken three by three. If, for example, a wedge formation is preferred, a higher score will be associated to the group of targets that can be linked in such a way to present all the angle values close to 180°, except for one angle, corresponding to an intermediate target.

Once the optimum formation has been selected, the individualization of the homing point (the barycenter, the center of the smallest circumscribed circle, the medium, etc.) can be readily made with simple calculations on the coordinates of the targets.

7. A PRACTICAL CASE: EXPERIMENTAL RESULTS

In this section the algorithm previously described is illustrated by applying it to a sequence of preprocessed images of 256 by 256 pixels.

The algorithm has been implemented in FORTRAN on a DIGITAL VAX 780 computer.

For this example, a quite simple sequence has been chosen, in order to highlight the mathematical computation performed by the algorithm.

Moreover, in order not to be tied to a particular situation or scenario, the comparisons with predefined values of the IR seeker and of the weapon system of the missile, have not been accomplished.

Figure 4 and Figure 5 show the disposition of the hot-spots in the first and the last image of the sequence, respectively.

Thirteen hot-spots are present in the sequence, numbered with the labels set by the preprocessing.

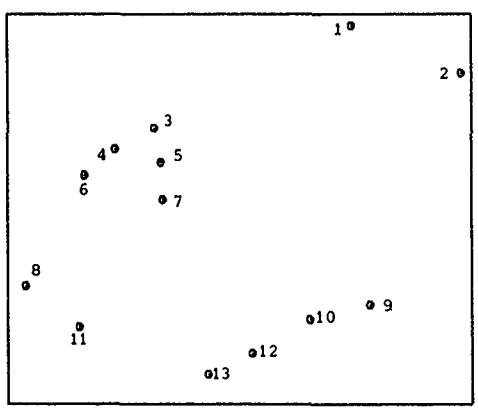


FIGURE 4 - LABELLED HOT-SPOTS IN THE FIRST IMAGE OF THE SEQUENCE

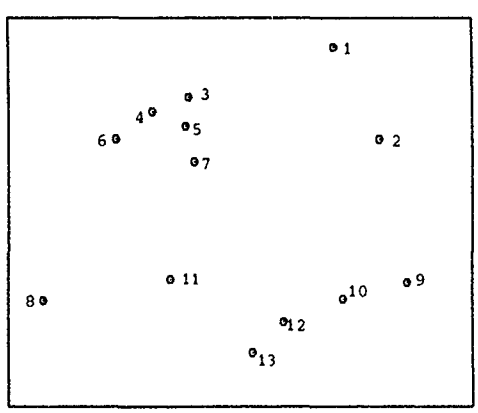


FIGURE 5 - LABELLED HOT-SPOTS IN THE LAST IMAGE OF THE SEQUENCE

The evaluation of the total displacements of the points lead to the values shown in Table 1, where SUM(i) is the name used by the computer program to indicate the distance, in pixels, covered by the i-th point throughout the sequence (i.e. the s(i) of section 3.).

SUM(1)= 16.621	SUM(2)= 61.877	SUM(3)= 20.105
SUM(4)= 22.190	SUM(5)= 15.318	SUM(6)= 19.028
SUM(7)= 19.385	SUM(8)= 13.207	SUM(9)= 19.868
SUM(10)= 18.000	SUM(11)= 60.908	SUM(12)= 25.042
SUM(13)= 23.023		

TABLE 1 - TOTAL DISPLACEMENTS OF THE HOT-SPOTS

The mean value and the standard deviation for the set of figures in Table 1 are:

$$\mu(s) = 25.74 ,$$

$$\sigma(s) = 15.50 ,$$

and the threshold value for $\sigma(s)$ is:

$$\sigma^*(s) = 0.2 \cdot \mu(s) = 5.15 .$$

Being the standard deviation greater than $\sigma^*(s)$, the set of points must be subdivided according to the sign of the differences: $[SUM(i) - \mu(s)]$.

Two groups are obtained: the first one composed by eleven points (number 1, 3, 4, 5, 6, 7, 8, 9, 10, 12 and 13) and the second one composed by the remaining two targets.

The values of $\mu(s)$, $\sigma(s)$ and $\sigma^*(s)$ for the first group are:

$$\mu(s) = 19.25 ,$$

$$\sigma(s) = 3.28 ,$$

$$\sigma^*(s) = 3.85 ,$$

and for the second group:

$$\mu(s) = 61.39 ,$$

$$\sigma(s) = 0.48 ,$$

$$\sigma^*(s) = 12.28 .$$

In both cases, no further subdivision is necessary; the second group (hot-spots number 2 and 11) is rejected and the remaining points, subject of the subsequent analysis, are re-labelled.

The algorithm considers now the first frame: it evaluates the distances from point number 1 to all the other points and retains the shortest, which lead to point number 2, as shown in Figure 6.

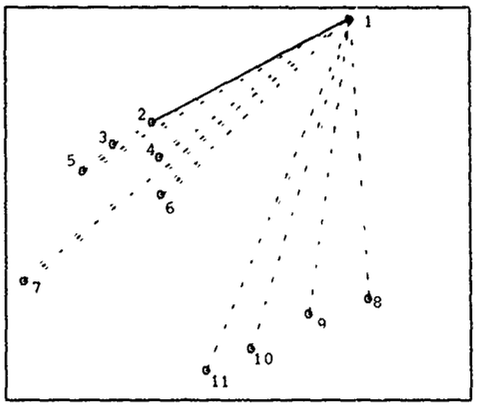


FIGURE 6 - MINIMUM DISTANCE EVALUATION

The composition of the chain proceeds following three rules:

- The algorithm is not allowed to close loops, nor to isolate couples of points: if point 2 is linked to point 4, point 4 must be either linked to another point not jet in the chain or be an End Point, but cannot be linked back to point 2.

- For every new point to be added in the chain, a check is done whether this point can be linked with a shorter edge: for example, the chain in Figure 6 goes from point 1 to point 2, then to point 4 up to point 6. This last should be linked to point 3, but the check allows to verify that point 3 is closer to point 2 (which already belongs to the chain), thus point 6 becomes an End Point and the link 2-3 is retained.

- A final check is done to control if some points have been left out (this might happen to points close to a Branch Point) and, in case, they are linked to the closer point of the chain.

Figure 7 shows the result of the chain construction in the first frame and Table 2 presents the values relevant to this chain, as they are output by the computer program: in each row, the first two values are the labels of the points linked and the third value is the relevant distance.

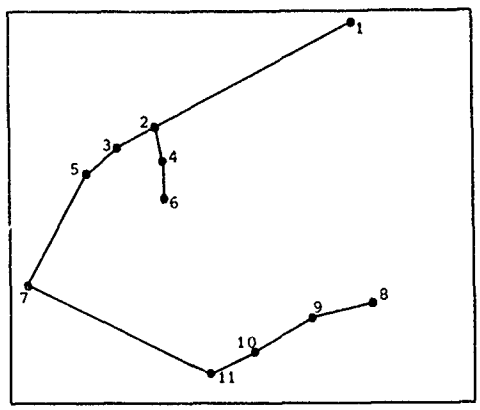


FIGURE 7 - CHAIN OF TARGETS IN THE FIRST IMAGE OF THE SEQUENCE

Counting the number of times that each point is repeated in the first two columns, the algorithm can classify the hot-spots: points 1, 6 and 8 are End Points (one presence); points 3, 4, 5, 7, 9, 10 and 11 are Intermediate Points (two presences); point 2 is a Branch Point (more than two presences).

DISTAM:	1	2	127.95
DISTAM:	2	4	23.35
DISTAM:	4	6	24.02
DISTAM:	2	3	25.24
DISTAM:	3	5	24.04
DISTAM:	5	7	80.11
DISTAM:	7	11	116.47
DISTAM:	11	10	27.78
DISTAM:	10	9	39.41
DISTAM:	9	8	34.21

TABLE 2 - LINKS AND DISTANCES OF THE CHAIN (first image)

Analyzing the values of the last column of Table 2, we have:

$$\mu(d) = 52.26 ,$$

$$\sigma(d) = 38.57 ,$$

$$\sigma^*(d) = 10.45 ;$$

being $\sigma(d) > \sigma^*(d)$, the counter K is initialized and the differences

(1) $$\left| d(j) - \mu(d) \right|$$

are compared with the value:

$$(1+Kc) \cdot \sigma(d) = (1+1 \cdot 0.2) \cdot 38.57 = 46.29 .$$

Two differences (1) are greater than this last value: that one relevant to the link of point 1 with point 2, and that one relevant to the link of points 7 and 11. The higher value corresponds to the first one, thus a cut is produced on the link 1-2 and point 1 is rejected, being an End Point.

The analysis proceeds on the group of the remaining distances, for which we have:

$$\mu(d) = 43.85 ,$$

$$\sigma(d) = 30.75 ,$$

$$\sigma^*(d) = 8.77 .$$

Again a cut must be produced, so the counter K is incremented and the differences (1) are evaluated and compared with:

$$(1+Kc) \cdot \sigma(d) = 43.05 .$$

The difference relevant to the link 7-11 is greater than this value: the chain is thus split, obtaining two groups to be analyzed.

For the first one, composed by the edges that link points number 2, 4, 6, 3, 5 and 7, we have:

$$\mu(d) = 35.35 ,$$

$$\sigma(d) = 22.39 ,$$

$$\sigma^*(d) = 7.07 ,$$

$$K = 3 ,$$

$$(1+Kc) \cdot \sigma(d) = 35.82 .$$

Evaluating the differences (1), it can be easily seen that the link 5-7 must be cut and point 7 eliminated as End Point. For the remaining set of edges (chain 5-3-2-4-6) we have:

$$\mu(d) = 24.16 ,$$

$$\sigma(d) = 0.68 ,$$

$$\sigma^*(d) = 4.83 ,$$

and the chain is not further split.

Analogously, the second group individualized in the previous step (chain 8-9-10-11) is not further subdivided, being:

$$\mu(d) = 33.80 ,$$

$$\sigma(d) = 4.76 ,$$

$$\sigma^*(d) = 6.76 .$$

The result of the analysis of the distances in the first frame leads, thus, to the location of a formation made up of five targets (number 2, 3, 4, 5 and 6) and of a formation made up of four targets (number 8, 9, 10 and 11).

The same analysis is then carried on considering all the subsequent frames of the sequence. Figure 8 shows the chain of targets obtained in the last image and Table 3, analogously to Table 2, shows the relevant links and distances.

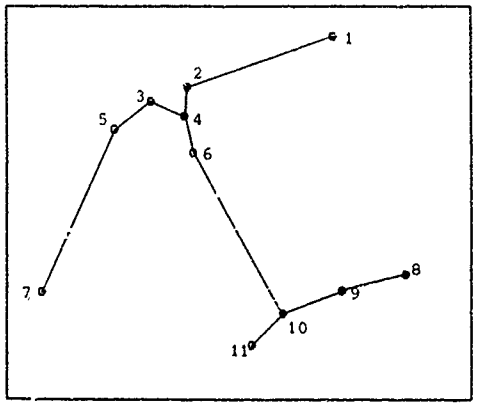


FIGURE 8 - CHAIN OF TARGETS IN THE LAST IMAGE OF THE SEQUENCE

If the algorithm is applied to this chain, firstly the link between targets 6 and 10 is cut, the group composed by targets number 8, 9, 10 and 11 is recognized as a formation and the analysis proceeds on the other group of targets. The second iteration cuts the link 5-7, and target 7 is rejected as End Point. Link 1-2 is cut in the third iteration, and target 1 is rejected. Finally, the group of targets 2, 3, 4, 5, and 6 is recognized as the second formation of the frame.

The Frequency Table, filled up during the analysis of the frames, confirms the existence of the two formations in the sequence.

DISTAM:	1	2	86.54
DISTAM:	2	4	19.10
DISTAM:	4	3	20.59
DISTAM:	3	5	26.91
DISTAM:	4	6	23.54
DISTAM:	6	10	115.87
DISTAM:	10	11	26.25
DISTAM:	10	9	36.25
DISTAM:	9	8	36.40
DISTAM:	5	7	113.30

TABLE 3 - LINKS AND DISTANCES OF THE CHAIN (last image)

8. CONCLUSIONS

The quantities used and evaluated and the comparisons performed during the analysis by the algorithm presented in this paper, are summarized in Table 4.

EVALUATED IMAGE FEATURES		PREDEFINED SYSTEM FEATURES	PREDEFINED CONSTANTS
.Partial Displacements (Instantaneous Speed)		.Maximum Speed Value	---
.Total Displacement .Total Displacement Mean Value .Total Displacement Standard Deviation		---	.Standard Deviation Threshold Value
.Differences of Total Displacements and Total Displacement Mean Value		---	---
.Minimum Distances .Minimum Distances Mean Value .Minimum Distances Standard Deviation	TO	---	.Standard Deviation Threshold Value
.Absolute Value of Differences of Minimum Distances and Minimum Distances Mean Value	BE COMPARED WITH	---	.Absolute Difference Threshold Level
.Sum of Weighted Presences		---	.Presence Threshold Level
.Mean Value of Distances in the Formation		.Range of Acceptable Distance Values	---
.Distances between End Points of the Formation .Number of Branch Points in the Formation		.Preferable Formation Lay-Out	---
.Number of Targets in the Formation		.Number of Munitions	---
.Angles formed by Targets of the Formation		.Preferable Formation Shape	---
.Homing Point		---	---

TABLE 4 - SUMMARY OF THE EVALUATED QUANTITIES AND PERFORMED COMPARISONS

The algorithm has been tested on various files, in order to tune the predefined parameters and to check the behaviour of the algorithm itself.

The results obtained are quite satisfying: thanks to the number of comparisons and controls performed, the algorithm can reject false alarms and isolated targets and can detect the true formations also in presence of a large number of points in the sequence.

In the frame by frame analysis, a wrong grouping could be achieved in case of very close targets, due to the quantities used to perform the comparisons (i.e. the mean value, the standard deviation and the differences (1)): in this case the standard deviation could be increased by the short distances and shortest links could be cut, since the differences (1) are taken with their absolute value.

Finally, the algorithm works very well in the evaluation of the optimal formation and of the point of homing, once the characteristics of the weapon system have been properly set.

REFERENCES

1. "Detection of Target Formations in IR Image Sequences"
North Atlantic Council - Research Study Group on Image Processing
AC/243(Panel 3/RSG.9)d/16 - 1st January 1990

2. "Time-Varying Image Processing and Moving Object Recognition"
Proceedings of the International Workshop - Florence, Italy, September 8-9, 1986
V. Cappellini Editor - North-Holland

3. "Real Time Image Processing: Concepts and Technolcgies"
SPIE Volume 860 - 17-18 November 1987, Cannes, France
Jean Besson Chair/Editor

DESIGN AND SIMULATION OF AN ADVANCED AIRBORNE EARLY WARNING SYSTEM

Chien Y. Huang, Senior Research Scientist, Corporate Research Center
Manikant D. Lodaya, Senior Engineer, Technology Development Section
Grumman Corporation, MS A08-35
Bethpage, NY 11714, USA

SUMMARY

The results of the design and simulation of an advanced airborne early warning (AEW) system are presented. The approach is based on modeling operator's reasoning and decision processes as well as battlefield strategies. The tasks are divided into threat assessment and tactical planning. The implementation of these subsystems is carried out in a generic expert-system shell developed specifically for this purpose. The AEW crew is provided with an advanced display that monitors all transactions. The functionalities of this prototype AEW system are demonstrated using an advanced simulation facility. Simulation results show that decision automation can be accomplished in real time and can prove to be a valuable tool in an airborne early warning environment.

1. INTRODUCTION

Airborne Early Warning (AEW) systems are responsible for acquiring, correlating, and tracking targets using radar, identification friend-or-foe (IFF) signals, infrared, and other passive sensors. They are used for battlefield management and often serve as command, control, and communication (C^3) centers. Increasing sophistication of vehicle technology and performance indicates that response time to threats will be shorter (due to, for example, stealth technology). At the same time, new battlefield concepts imply that the number of decisions and tactics will be higher (due to, for example, combat unit dispersion). Clearly, these requirements place demands beyond the capabilities of the current AEW systems, which must be upgraded to meet the challenges.

The tactical decisions made by the AEW crew are based on threat levels of the targets, availability of resources, and rules of engagement. They also are affected by geographic location, defense condition, and weather. All of these factors are subject to constant changes which, if not taken into account in devising strategies, can lead to incorrect decisions. Furthermore, the crew usually have to keep track of a large number (e.g., over 200) of targets. This situation translates to an overwhelming amount of information from which relevant data are extracted. The end result is a high-workload, error-prone environment where correct and timely decision making is severely handicapped.

Previous discussions indicate that the critical issues for the AEW systems are information management and decision making. It is said that most decision failures are due to the inability to extract and integrate relevant data in an effective time-critical manner [1]. These problems can be best addressed by decision automation. This approach would reduce the number of cases to be dealt with, thus allowing the AEW crew to concentrate on more critical cases, and thereby improving the quality of decisions.

Studies have been made in the past in applying artificial intelligence to battlefield management, specifically in the areas of threat warning [2], tactical indications [3], tactical situation assessment [4], tactical decision aids [5], and others. The US Air Force is funding the Pilot Associate project [6] to develop a cockpit information management and decision-aid system. To encompass force deployment and protection, a similar effort, but larger in scope, is being carried out by Grumman. This effort is concentrated in the Crew Associate project [7,8], which is being developed to assist the AEW crew in decision making. The objective of this paper is to describe the design and performance of a *real-time* automated decision-making system (ADEMS) to assist operators in an AEW environment.

ADEMS is an expert-system shell designed to extract information such as location, behavioral pattern, electronic surveillance status, and other relevant data to establish battlefield strategies. Decisions are made by incorporating resource management techniques and short- and long-term goals. Typical scenarios would include vector-intercept (by aircraft), patrol assignment, combat unit relief, etc. ADEMS has a restricted

application domain, which is decision making within the AEW environment, although it can be expanded to deal with other problems.

In this paper we describe parts of Grumman's Crew Associate (Fig. 1). It consists of a Threat Assessment (TA) module, which ascertains the threat level of a target; a Tactical Planning (TP) module, which makes tactical decisions based on overall battlefield strategies; a Monitor and Control System (MCS), which conducts on-line system monitoring of AEW aircraft and friendly targets; and a Crew Console Display (CCD), which provides an advanced console display for the AEW crew. Both TA and TP are implemented with ADEMS. Due to our emphasis, MCS will not be described here. The work described herein is a continuation of previous efforts reported in Refs. 9 and 10.

The organization of the paper is as follows: Section 2 describes the requirements and design of the TA system, while Section 3 details analysis and design of the TP system. Section 4 covers the internal structure of the ADEMS, with emphasis on approach. Section 5 describes the functionality of the CCD. Section 6 outlines an advanced simulation environment where the automated AEW functions are tested. Section 7 presents the simulation results using typical scenarios. Finally, Section 8 contains conclusions and plans for future work.

2. THREAT ASSESSMENT

The Threat Assessment (TA) system is a Crew Associate module whose function is to advise crews about the threat levels of detected targets. TA determines a target's threat level based on its position (distance, on/off corridor, etc.), dynamic behavior (heading, acceleration, altitude, etc.), actions (radar on, jamming, etc.), identification (F-14, F-16, enemy aircraft, friendly-known, enemy-unknown, etc.), and the number of aircraft in a raid.

Based on the available information, the TA system first assigns each target a classification, which can be confirmed-friend, assumed-friend, neutral, unknown, assumed-foe, or confirmed-foe. This information may come from a separate identification (ID) system or can be inferred, for example, from the target flying pattern. The threat level is then assessed from a combination of heuristic parameters such as actions, distance, ID, etc. A weighting scheme can be placed on each of the parameters to achieve desired emphasis. The threat level also is dependent on look-ahead depth, that is, how much TA anticipates enemy moves, thereby putting different "values" on incoming targets. The heuristic measures are added to arrive at a single, final number from which the threat level of no-threat, possible-threat, or high-threat is assigned. More specifically,

$$H_{total} = w_{tc} \cdot h_{tc} + w_d \cdot h_d + w_a \cdot h_a + w_{id} \cdot h_{id} + w_{la} \cdot h_{la} \quad (1)$$

where H_{total} is the final score, h_{tc} is the heuristic measure associated with target classification, h_d is the heuristic measure associated with distance, h_a is the heuristic measure associated with action, h_{id} the heuristic measure associated with ID, h_{la} the heuristic measure associated with look-ahead depth, and w are appropriate weighting factors. A realistic application of heuristic measures can be found in [11].

The Threat Assessment module is implemented using ADEMS. They implement most of the rules described in [12]; examples of some of them are shown in Fig. 2. The output of the TA system is fed to the TP module, which is discussed in the next section.

3. TACTICAL PLANNING

The Tactical Planning (TP) system is a Crew Associate module whose function is to recommend (to the AEW crew) or implement the best strategies necessary to engage threats and accomplish a desired mission. This is closely related to the desired combat posture, whether defensive or offensive. The tactics employed depend on the threat level of the hostile targets, availability of resources, the rules of engagement, defense conditions, and geographic locations. They can also take into account requirements, such as positional advantage; constraints, such as guaranteeing the safety of the carrier group; or goals, such as destroying a particular enemy target.

One of the primary functions of the TP system is to assign Combat Air Patrols (CAPs), tankers, and Deck Launch Interceptors (DLIs) to carry out a mission. This task is carried out so as to minimize the exposure of the carrier group to danger and maximize the favorable engagement outcome. TP may also mean look ahead on what targets may become a high threat and position interceptors in advantageous locations.

The other important role of the TP system is to manage resources such as CAPs, weapons, tankers, jammers, and DLIs. It keeps track of weapons and fuel used by CAPs and ensures their availability for a successful engagement. It also advises the AEW crew of unavailability of resources and recommends alternatives, such as retraction of the CAPs in self-defense.

Another possible usage of the TP system is to devise countermeasures by recognizing enemy plans and tactics. TP can also incorporate functions that anticipate engagement outcomes, thus revising the strategy or the overall approach (i.e., defensive, offensive, or neutral) accordingly. These notions are currently under study.

The functionality of the TP system reflects a number of generic battlefield management rules. Some of them are shown in Fig. 3. Note that although both TA and TP rules may appear to be simplistic, they become cumbersome when the number of targets is large. Furthermore, the combination and the interaction of rules are complex and are better addressed using decision automation. The Tactical Planning module is also implemented in ADEMS, which is described in the next section.

4. DESIGN OF AUTOMATED DECISION-MAKING SYSTEM

Earlier versions of the TA/TP systems were developed using the Automated Reasoning Tool (ART) [7] and the Procedure Reasoning System (PRS) [9]. However, the processing requirement was high and the system response was sluggish. It was felt that commercial expert-system shells were too general for our purpose and too slow to respond to the inputs. By utilizing these shells, we are effectively (and unnecessarily) paying the overhead for the tools that are superfluous. In an effort to achieve real-time performance, the functionalities and tasks of TA/TP systems were carefully examined and coded as LISP procedures. Although this approach was successful [10], the modularity and flexibility of the design were sacrificed, since the inference logic had to be "hard-wired." Therefore, it was decided to develop a generic expert-system shell (called Automated Decision Making System or ADEMS) to take advantage of the decision structure of the AEW scenario.

The desirable features of the ADEMS do not differ from a general purpose expert-system shell. They include a user-friendly front-end to declare and to modify the rules; a facility to display relevant and current rules; an explanation mechanism to describe the rules invoked, etc. In addition, because the data being given to the TA/TP systems are information rich and carry implicitly the preconditions that trigger the rules, they naturally lead us to rely heavily on data-driven methods.

The data-driven approach is used here not in its "classical" sense of forward-chaining, but rather it is a reflection of the decision process. It is accomplished by carefully organizing the logic and scheduling the execution of inference. Thus, specific actions are associated with the incoming data, and they are embedded within the rules. Each action consists of procedures, messages to other subsystems, or insertion of additional keywords to drive the inference engine.

A major step in setting the data-driven structure is to dissect the rule entries. ADEMS uses a pseudo-parsing system to understand a wide variety of rule specifications. It does so by extracting from the rule declaration a set of keywords from which the internal logic is reconstructed. Examples of keywords are distance, less-than, greater-than, equal-to, CAP, target, high-threat, low-threat, corridor, etc. Any input that cannot be deciphered in this fashion is redisplayed and the user is prompted for clarification. Extraction of keywords allows ADEMS to decide which rules should be invoked without use of any meta-rules (that is, rules about usage of rules). This is possible since ADEMS knows when the new information is available, and thus only those rules which are intersections of all rules will be executed. Lastly, any rule that cannot be parsed (such as computation of heuristic measures) is entered as a special case, and procedures can be attached to achieve the desirable effects.

After the rules are entered, they are stored on the property lists of the keywords. This is true whether the rules involve actions or references to other rules. This knowledge representation enables ADEMS's inference engine to quickly access the desired information and execute the required procedures. In fact, when new data become available, ADEMS performs a simple unification of all existing preconditions and runs any rule that satisfies the prerequisites. This procedure is illustrated with some examples below.

Rule Structure
IF subject + verb + description + modifier
THEN action

Example:
IF target's ID is enemy aircraft,
THEN the target is an assumed foe

Here the keywords are target_id, enemy_aircraft, and assumed_foe. If the incoming data satisfy the prerequisites, an "assumed_foe ' classification will be associated with target_id.

Example:

IF	target	is	a confirmed foe
AND	it	is	within high-threat zone,
THEN	the target is a high threat		

The keywords for this case are confirmed_foe, high_threat_zone, and high_threat. When the first two keywords are present in a target's property list, then high threat is declared and added onto the target's property list. This new information subsequently will trigger rules within the TP system.

Example:

IF	target's threat level	is	high threat
THEN	assign the closest CAP to engage the target		

All keywords are previously defined. The action consists of a procedure that figures out which available CAP is closest to the target within the assigned sector and then vectors it to intercept the target.

The approach described for the implementation of the TA/TP expert systems is somewhat narrow in scope. The intent is to achieve a good compromise between fast response and ease of use. The processing speed is gained by specializing the application, and the flexibility is reduced (but not lost) by concentrating on specific tasks. Although this methodology would not be applicable for general situations, it was deemed to be sufficient for the AEW problem.

5. CREW CONSOLE DISPLAY

The purpose of the CCD is to display the current combat scenario based on the radar and other sensor information to an AEW operator. It also communicates to the crew the findings of TA and strategic decisions made by TP. Thus, to logically display the current situation and to allow quick access to desired information are primary goals of the CCD. A very large part of the CCD design is human interface. This is necessary so as to alleviate the operator's workload in interpreting information and making queries and decisions.

CCD's design is graphic-intensive so as to simulate a conventional radar display, but with several additions such as pull-down menus and mouse-sensitive items. Each target has its own track information (which is accessible via mouse or keyboard) and can be commanded to move to other locations in a similar manner. Geographic maps are overlaid on the display as well as on (simulated) radar sweep. Zoom and multiple window facility are available. Color is judiciously used to provide information differentiation. The system provides automatic hand-over when the aircraft move out of the assigned sector of the AEW crew, thus ensuring proper information transfer.

Threat assessessment and subsequent tactical decisions are displayed graphically as well as textually. The AEW operator can override the recommendation made by TP (and TA). Optionally, the decisions by TP can be made to take effect within a specified time interval. Presently, all decisions are automatically transmitted to the Simulation Driver. The crew has additional flexibility in changing the current ID of the target and letting the TA/TP expert systems evaluate the consequences. This leads to better tradeoffs of strategies.

Although the CCD is an integral part of the AEW system, its detail design is not warranted at this stage. However, sufficient flexibility is built in to permit future improvements. In fact, much of the CCD is modeled after the Simulation Driver, which is to be described in the next section.

6. SIMULATION ENVIRONMENT

To test the decision-making process of the TA/TP modules, we have developed a generic Simulator Driver (Fig. 1) which allows us to set up and experiment with various combat scenarios quickly and repetitively. Moreover, the Simulation Driver would allow us the freedom to provide, emulate, and encode necessary data to other modules of the Crew Associate.

The Simulation Driver is designed to be user friendly and complete. Emphasis is placed on the ability to set up a new scenario or modify an existing one quickly and efficiently. The Simulation Driver currently supports any generic scenario with *arbitrary* user specifications. Targets are created by moving the mouse pointer over the desired location and clicking the appropriate mouse button. They are displayed according to the Naval Tactical Display Symbol (NTDS) set, which includes enemy air targets, friendly air tracks, unknown tracks, surface tracks, etc. Once positioned, a target can be deleted or dragged to a new location.

Targets can be made to move toward or away from fixed points; they can also be vectored or commanded to escort other targets. Target color can be changed for emphasis or bookkeeping.

Associated with each target is a set of target information such as speed, altitude, heading, target ID, etc. All data have defaults and can be displayed on command. Changes to data can be made individually (via mouse) or to all targets created. Both military and commercial corridors can be drawn at specified locations. All simulation scenarios and sessions can be recorded and saved as files to be played back at a later time with different parameters, if desired.

An impending engagement (between two targets) is highlighted (with a circle drawn around them). The user has the option of destroying either of the targets or selecting the "escort" mode, where the friendly target changes its course to parallel that of the enemy target. Decisions on the engagement result can be made automatically (based on a weighted preassigned probability) or entered manually by an operator. Engagement results are transmitted to the TA/TP system to be used in the generation of subsequent strategies.

The Simulation Driver is written in LISP in object-oriented format. It utilizes various advanced features offered by the Symbolics expanded Common-LISP environment, such as routines that keep track of the mouse and generate pull-down menus. The database for the targets created is kept using generic pointing structures, which allow direct reference by name and quick access to the data. Currently, the Simulation Driver resides on a Symbolics 3765 and is displayed on a Tektronix color monitor. The TA/TP subsystems reside on a Symbolics MacIvory, while the CCD uses a separate Symbolics 3765. All interactions among the TA/TP modules, the CCD, and the Simulation Driver are coordinated by a System Manager and transferred over Ethernet using proprietary network software.

7. SIMULATION RESULTS

Preliminary evaluation of the functionalities of the Threat Assessment and Tactical Planning modules are shown in the coordination of tactical maneuvers using a generic combat scenario (Fig. 4). The objective is the defense of the carrier group as a whole (which implies maintenance of proper defensive posture). Targets 0-2 are available CAPs and targets 3-5 are enemy (or hostile) aircraft targets. Targets 6-7 are commercial airlines flying in defined commercial corridors. Targets 8-12 are friendly and unknown surface tracks. Target 13 is the C^3 vehicle. The critical distance circles (the inner one is the high-threat region [with respect to the carrier] and the outer one is the minimum-threat region) are shown as drawn. The goal in this case is to protect the carrier force and associated ships. The sequence of events is as follows:

Event 1: Targets 3-4 move toward the carrier.
Action 1: High threat is not declared since none of the enemy targets are within the high-threat region.
Event 2: Target 3 is within the high-threat region and its radar is active.
Action 2: CAP-1 (or Target 1), which is the closest one within the assigned sector, is directed to intercept and destroy Target 3.
Event 3: CAP-1 destroys Target 3 during the engagement. Targets 4 and 5 are still minimum threats.
Action 3: CAP-1 becomes an available CAP.

The new scenario is shown in Fig. 5.

Event 4: Target 4 crosses the high-threat region, but its radar is inactive and it is flying at a lower speed; therefore, it is considered a minimum threat. At the same time Target 7 (an assumed commercial airliner) deviates from the corridor and heads toward the friendly surface ships (Target 8).
Action 4: CAP-1, again being closest to Target 4 within the sector, is assigned to escort Target 4. CAP-0 is repositioned closer to the surface ships as a precautionary measure.

The new scenario is shown in Fig. 6.

Event 5: Fuel on CAP-1 is running low and Target 7 moves within the high-threat region of the surface ships.
Action 5: CAP-2 is assigned to replace CAP-1, which returns to carrier for refueling, while CAP-0 is commanded to leave the patrol mode and intercept Target 7.
Event 6: Target 7 is verified to be an enemy airplane and is destroyed in the engagement. Target 4 moves away from the carrier force.
Action 6: Both CAP-0 and CAP-2 become available.

This final scenario is shown in Fig. 7.

Other combat scenarios have been tested. They show that all decisions can be made essentially in real time. This performance is mostly due to coding efficiency and partially due to scenarios used and strategies involved. It remains to be seen what performance will be achieved when more sophisticated tactics are used and operators become part of the overall system.

8. CONCLUSIONS

Design and simulation of an advanced airborne early warning system have been described. The work is carried out by applying automated decision-making techniques, which emulate the inference process of the AEW operators. The tasks are separated into threat assessement, which determines a target's threat level based on available information, and tactical planning, which attempts to use the available strategies to achieve the optimal response. Both TA/TP modules are implemented using a generic expert-system shell designed specifically for the AEW problems. The shell is based on data-driven concepts and provides necessary knowledge representation and inference engine to ease the design. Additionally, an advanced crew console display and a unique simulation facility have been developed to further evaluate the potential of the knowledge-based approach. The systems are being integrated and tested against several scenarios, and the results are encouraging.

Future plans call for refinements of current tactics and the expert-system shell. We plan to conduct experiments using real data from an E-2C Hawkeye. We also contemplate research on decision making based on partial data, which is due to, for example, radar jamming. In this situation, it would be necessary to rely on the outputs of secondary sensors and carry out data fusion in both the identification stage as well as in the inference process. Finally, we are looking at potential applications to programs such as Joint Surveillance and Tracking System (J-STARS), ground monitoring, and with some modifications, to the air-traffic control systems.

9. REFERENCES

1. Hopson J., Stark S., Detwiler, D., Zachary, W., and Fitzkee, S., "A Generalized Automated Decision Algorithm for the AEW Engagement/Intercept Planning Function," *Report No. NADC-80104-50*, Naval Air Systems Command, Washington D.C., July 1980.

2. Babcock, D. F., "An Architecture for the Application of AI Techniques to Threat Warning," *Army Conference on Application of Artificial Intelligence to Battlefield Information Management*, Navy Surface Weapons Center, White Oak, MD, April 1983, pp. 117-123.

3. Kiremidjian, G., Clarkson, A., and Lenet, D., "Expert System for Tactical Indications and Warning Analysis," *Army Conference on Application of Artificial Intelligence to Battlefield Information Management*, Navy Surface Weapons Center, White Oak, MD, April 1983, pp. 145-159.

4. Azarewicz, J., Fala, G., and Heithecker, C., "Template-Based Multi-Agent Plan Recognition for Tactical Situation Assessment," *IEEE Conference on Artificial Intelligence Applications*, March 1989.

5. Kurien, T., "A Tactical Decision Aid for the E-2C Hawkeye," *Report No. TR-473*, Office of Naval Research, April 1990.

6. "Phase 1 Interim Report of the Pilot's Associate System," Lockheed Aeronautical Systems Co., Marietta, Georgia, Aug. 5, 1988.

7. Kadar, I., and Baron-Vartian, E., "Process Modeling - A Situation Assessment Expert System," *AIAA Computers in Aerospace VI Conference*, Wakefield, MA, Oct. 1987.

8. Newman, T., and Tarpinian, R., "Crew Associate for Advanced Surveillance," *Avionics Technical Symposium*, Monterey, CA, Nov. 1989.

9. Huang, C., and Lodaya, M., "Automated Decision-Making in Advanced Airborne Early Warning Systems: Analysis, Design, and Simulation," *Proc. of SPIE Conf. on Applications of Artificial Intelligence*, Orlando, FL., April 1990.

10. Huang, C., "Real-Time Automated Decision-Making in Advanced Early Warning Systems," *Proc. of NAECON*, Dayton, OH, May 1990, pp. 434-439.

11. Huang, C., and Lane, S., "Solution to the Traveling Salesman Problem Using Heuristic Function Maximization," *J. of Guidance, Control, and Dynamics*, Vol. 11. No. 5, Sept.-Oct. 1988, pp. 430-435.

12. Lodaya, M., and Abrams, B., "Functional Specification for Threat Assessment Expert System," *Grumman Internal Memo*, Feb. 15, 1989.

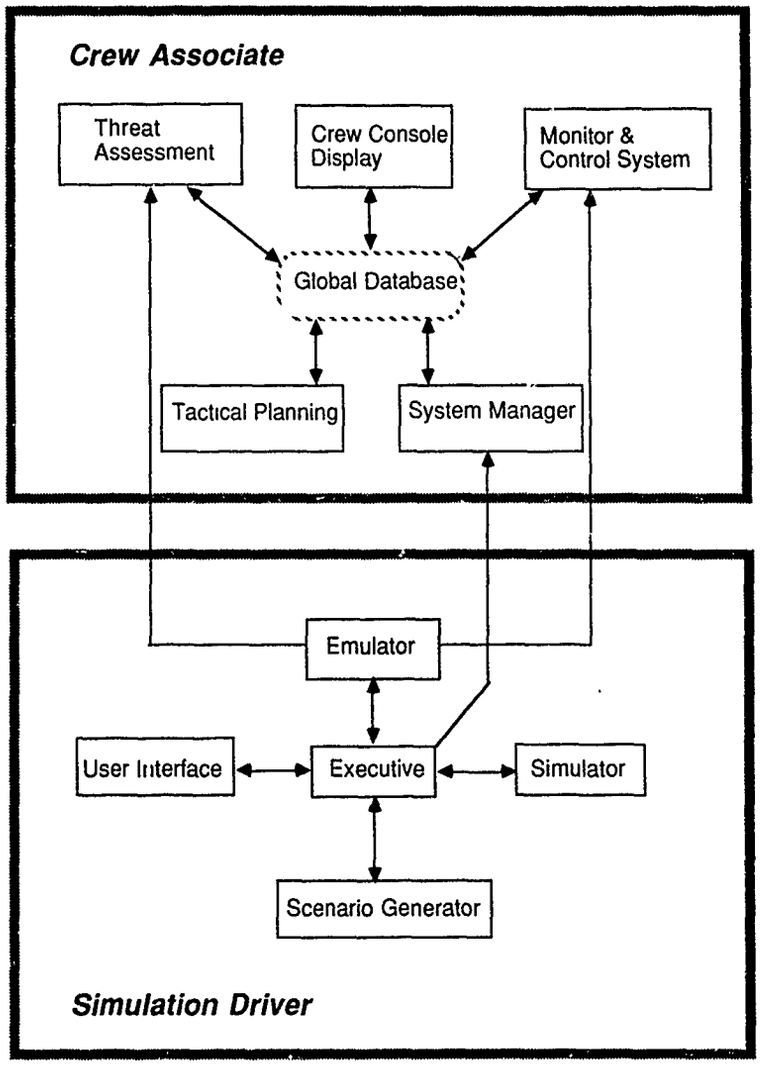


Fig. 1 Crew Associate System Block Diagram.

IF the target's ID is an enemy aircraft with probability greater than α,
THEN the target is an assumed-foe.

IF the target is flying in a commercial corridor and transmitting airline code,
THEN the target is no threat.

IF the target class is a confirmed-foe and it is within the high-threat zone,
THEN the target is a high threat with high confidence.

IF the target is an assumed friend at a specified distance and satisfies a threat criterion,
THEN the target is a possible threat.

Fig. 2 Some Threat Assessment Rules.

IF the threat level of a target is high threat,
THEN assign the closest available CAP to engage with the target.

IF a high-threat target has been destroyed,
THEN make assigned CAP available and update information.

IF there are no available CAPs in the area,
THEN dispatch a combat-ready CAP from the carrier.

IF the CAP available for assignment is low in fuel and a tanker is available,
THEN refuel the CAP in air.

Fig. 3 Some Tactical Planning Rules.

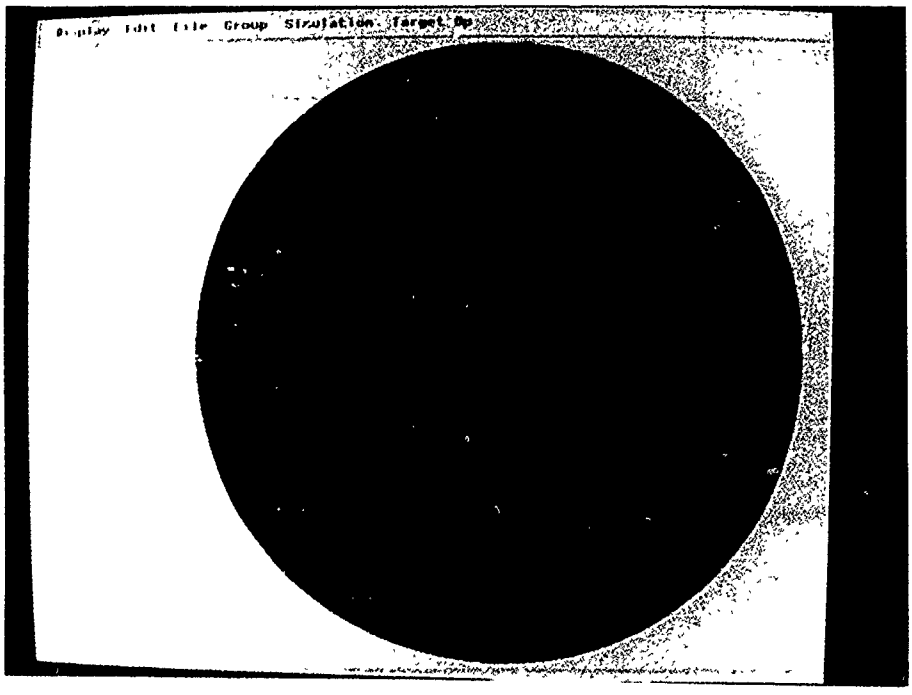


Fig. 4 Initial AEW Scenario.

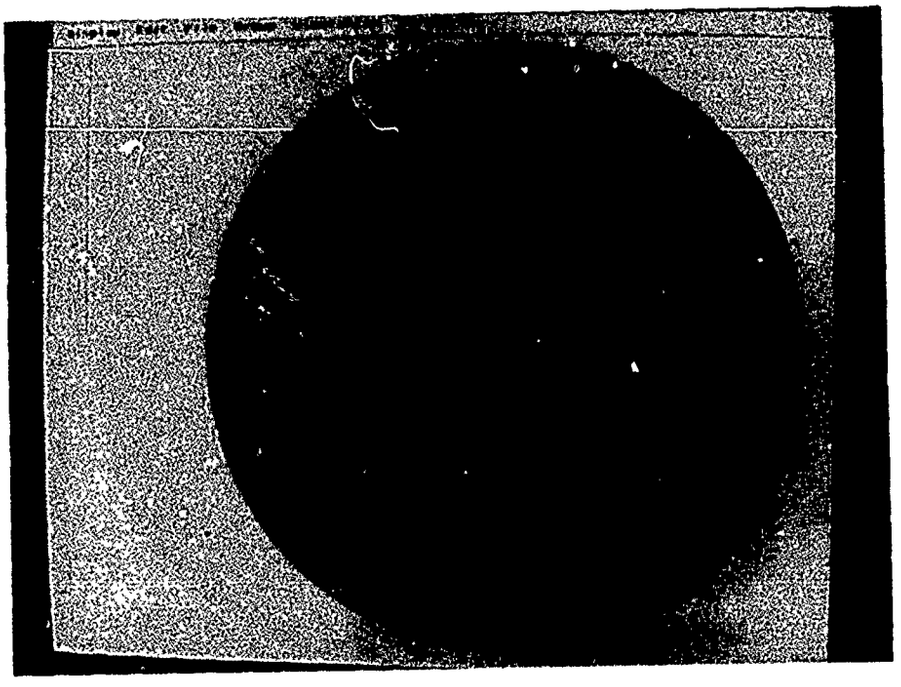


Fig. 5 AEW Scenario after Action 3.

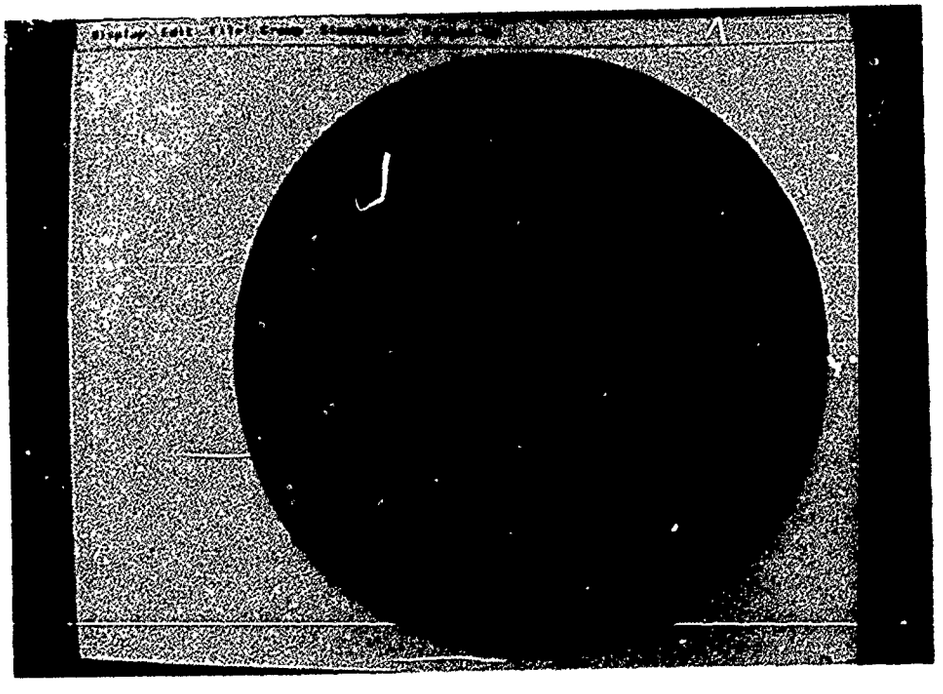


Fig. 6 AEW Scenario after Action 4.

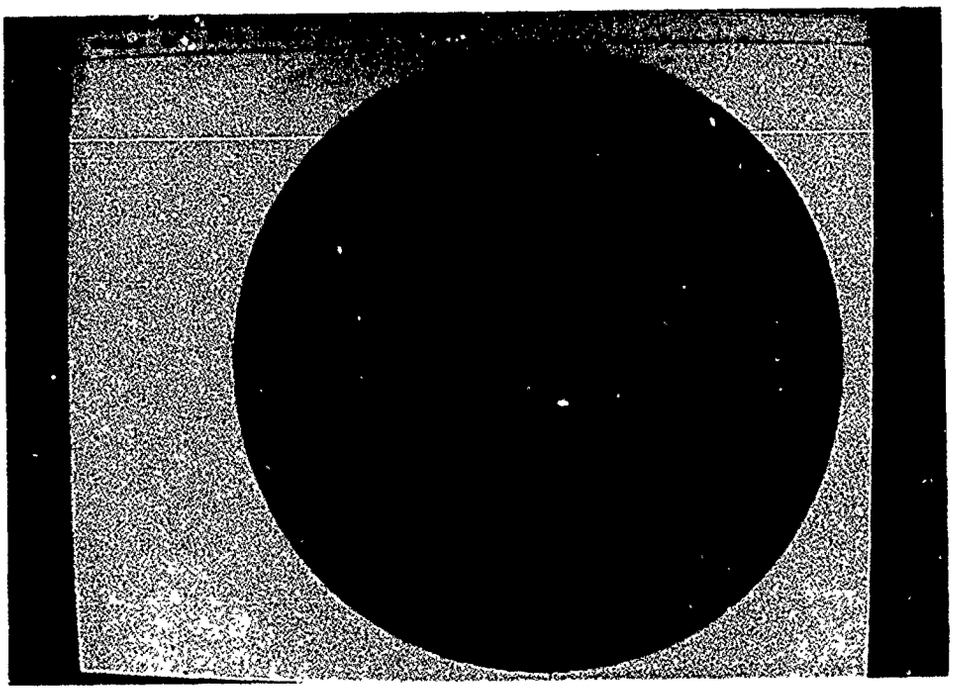


Fig. 7 AEW Scenario after Action 5.

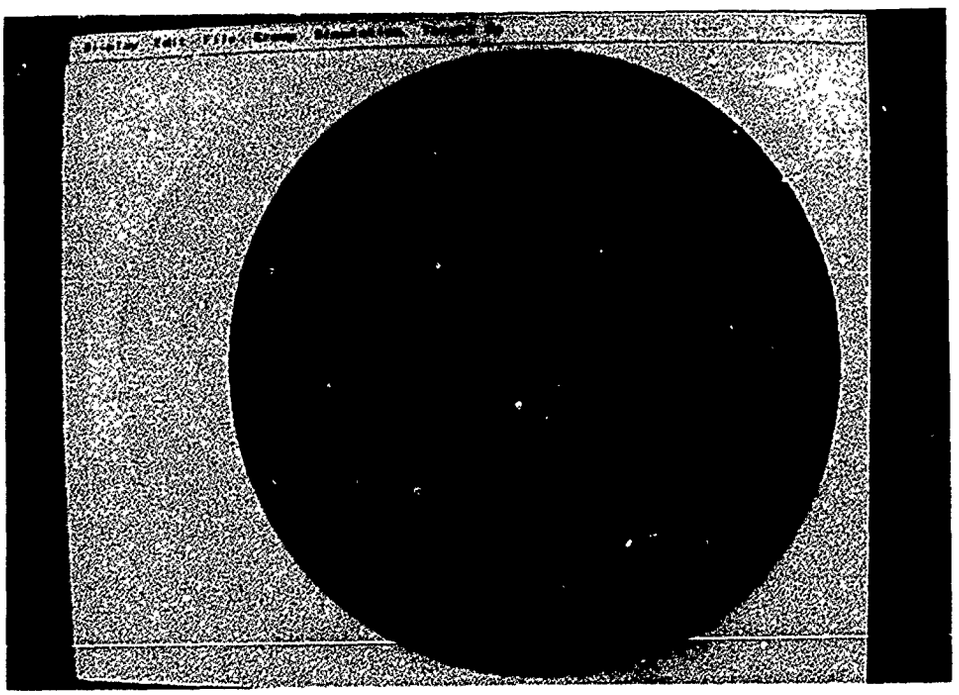


Fig. 8 Final AEW Scenario.

SEAN: UN SYSTEME EXPERT D'AIDE A LA NAVIGATION POUR AVION DE COMBAT

par

D.Morillon, T.Conter et M.de Cremiers
SAGEM. Unité de Recherche en Automatique
Centre d'Etudes de Pontoise
95301 Osny
France

Cette communication fait suite aux travaux réalisés dans le cadre d'un marché S.T.T.E

Résumé : La technologie des équipements d'avion de combat est en constante évolution L'augmentation de la précision des instruments de navigation, de la puissance des calculateurs, du nombre de moyens de recalage améliore la capacité de réussir les missions modernes. Cependant, la tâche du navigateur s'en trouve compliquée. SEAN (Système Expert d'Aide à la Navigation pour avion de combat) permet d'assister le navigateur en le conseillant dans ses choix, tout en surveillant le comportement du système de navigation. L'emploi d'une méthodologie cognitive performante (KOD) associée à un principe robuste de validation des connaissances a permis de développer de façon cohérente une maquette très complète du système expert et de son environnement embarqué opérationnel.

Mots clés : Système Expert, Navigation Inertielle, Diagnostic Technique.

Abstract : Technology of fighter aircrafts equipments is in constant evolution Increases in precision of navigation equipments, in number of available means of navigation updating and in onboard computers performance, has improved aircraft capability to satisfy modern missions requirements but also has complicated copilot's tasks. SEAN (Expert System for Navigation Aiding aboard fighters has been designed to assist copilot in its choices and supervise the inertial hybrid navigation system Use of KOD (Knowledge Oriented Design), a powerful cognitive methodology, associated to robust knowledge validation principles, has allowed development of a prototype of the expert system working with a realistic complete simulation of its onboard environment.

Key-words Expert System, Inertial Navigation, Technical Diagnosis

0 INTRODUCTION

Le système de navigation inertielle considéré dans le cadre de cette étude se compose de deux UNI (Unités de Navigation Inertielle) et dispose de quatre moyens de recalage de navigation associés. Chaque unité de navigation inertielle peut fournir deux types d'information de navigation :

- une navigation inertielle recalée classiquement,
- une navigation inertielle optimale i.e. recalée par filtrage de Kalman.

Le navigateur assure la mise en oeuvre du système et réalise des choix qui conditionnent l'obtention d'une bonne performance de navigation :

- sélection de la meilleure navigation (une parmi quatre),
- choix d'un moyen de recalage (un parmi quatre),
- validation ou non des recalages.

Ces choix sont assez critiques et le navigateur ne dispose pas toujours du temps et des moyens de réflexion suffisants pour assurer la prise de décision optimale. Un système expert a donc été envisagé afin de l'assister et de permettre la mise en oeuvre optimale du système de navigation. Il comporte deux volets :

- une surveillance constante des équipements pour avoir à chaque instant une idée qualitative de leur fonctionnement,
- une aide au copilote consistant en des prises de décision basées sur la connaissance précise de l'état du système.

Un avantage par rapport au navigateur est de pouvoir traiter l'ensemble des informations internes de la navigation et de pouvoir y appliquer une expertise de spécialistes (concepteurs et expérimentateurs du produit).

Plus précisément, l'apport d'un tel système est significatif à trois niveaux :

- aide à la prise de décision du navigateur, notamment dans les cas difficiles,
- amélioration des performances du système de navigation grâce à l'exploitation des redondances,
- surveillance constante des équipements permettant une maintenance préventive plus efficace.

Comme indiqué ci-dessus, ce travail a été réalisé dans l'hypothèse d'un équipage à deux comprenant un copilote chargé de la navigation. Cependant, la mise en oeuvre d'un tel système à bord d'un avion de combat monoplace deviendrait une nécessité

L'étude présentée dans cet article a donc consisté en la réalisation d'une maquette informatique mettant en oeuvre une simulation réaliste de l'environnement opérationnel d'un avion de combat.

1 METHODOLOGIE ET DEROULEMENT DE L'ETUDE

1.1 INTRODUCTION

L'étude a comporté deux phases principales :

- recueil intensif des connaissances,
- réalisation de la maquette de démonstration.

Toutefois, au delà de ce découpage primaire, c'est un long travail méthodique qui a permis l'aboutissement de cette maquette du système expert. Afin de bien maîtriser les aspects cognitifs de l'étude du système, deux méthodes ont été mises en place.

Tout d'abord nous avons retenu, parmi les méthodes de spécification et de conception de logiciel, une méthode propre au génie cognitif et non simplement du domaine du génie logiciel (méthodes type SADT, OOD, ...) Il s'agit donc véritablement d'une méthode d'acquisition et modélisation des connaissances

En complément, une méthode de gestion globale de l'étude a été élaborée visant à valider les connaissances, le plus tôt possible dans le cycle de vie de l'application. Ces deux aspects méthodologiques sont développés plus précisément dans les paragraphes qui suivent.

1.2 METHODE D'ACQUISITION ET DE MODELISATION DES CONNAISSANCES

Le recueil et la validation des connaissances ont été réalisés avec la méthode KOD tirée des travaux de Claude VOGEL [VOG 88]. Celle-ci s'appuie principalement sur deux idées :

- la décomposition structurale des objets manipulés par l'expert et des fonctions auxquelles ils sont assujettis,
- l'association de l'information aux conditions de l'énonciation ; on conserve ainsi une connaissance imprégnée de la vision de son énonciateur

Pour la présenter rapidement, disons qu'elle consiste à élaborer trois modèles successifs permettant de passer du discours de l'expert à la réalisation du système expert (voir figure 1.1) :

- **le modèle pratique** - Cette phase a pour but d'identifier et de spécifier l'expertise. On part d'une retranscription fidèle des propos de l'expert de laquelle on extrait les concepts maniés par ce dernier, les actions s'exerçant sur ces concepts et enfin la liste des inférences naturellement exprimées On obtient ainsi une base d'énoncés.
- **le modèle cognitif** - Cette étape essentielle conduit à l'élaboration d'un modèle abstrait intermédiaire permettant de reproduire les raisonnements profonds de l'expert et non pas seulement une mécanique de résolution dépendante des cas étudiés. Ce modèle, indépendant de toute représentation informatique, est élaboré par structuration de la base d'énoncés obtenue précédemment en classifications (taxinomies), actions génériques (actinomies) s'exerçant sur ces classifications et schémas d'inférence.
- **le modèle informatique** - C'est la transcription du modèle cognitif dans un langage informatique (incluant le plus souvent une représentation objet, un langage à base de règles et un langage procédural).

La volonté d'appliquer une méthode apparemment aussi contraignante, part d'un constat désormais classique : il est nécessaire d'établir au cours du processus d'acquisition de la connaissance un modèle abstrait intermédiaire afin de reproduire les raisonnements profonds de l'expert et non pas seulement une mécanique de résolution dépendante des cas étudiés.

En effet, lorsque cette étape est respectée, on obtient des systèmes à base de connaissance qui, bien que limités dans leurs capacités de raisonnement, conduisent toujours les expertises à l'image de celles que réaliserait l'expert.

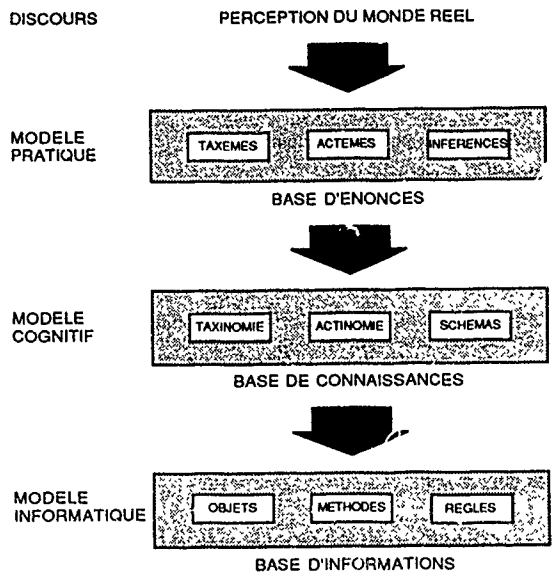


Figure 1.2 **METHODE KOD** (Knowledge Oriented Design)

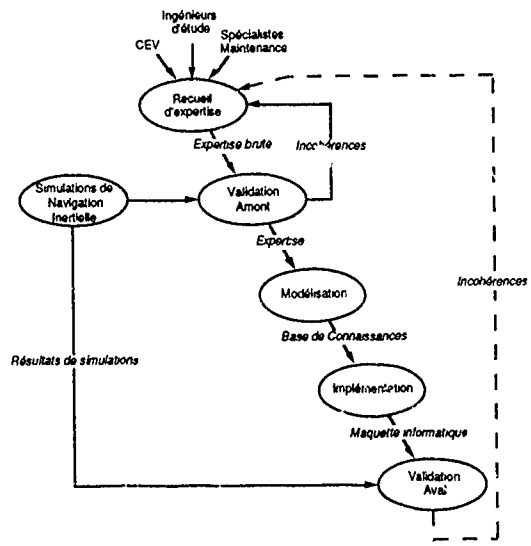


Figure 1.1 **CYCLE DE DEVELOPPEMENT**

1.3 APPROCHE GLOBALE : VALIDATION DES CONNAISSANCES

1.3.1 Introduction

Dans le développement d'un système expert, de même que dans tout projet informatique, il y a lieu de minimiser le travail de mise au point. Pour cela, une bonne spécification des fonctions attendues du système est nécessaire, mais de même que l'on effectue des test unitaires de parties de logiciel, il y a également lieu de valider la connaissance au fur et à mesure de son recueil.

De plus, nous disposons à la SAGEM de logiciels permettant de simuler très précisément le comportement d'une UNI. On peut ainsi reproduire l'évolution inertielle d'un avion en fixant la trajectoire et l'état des composants inertiels des UNIs.

Ce simulateur nous a permis d'envisager un processus de validation en deux temps (voir figure 1.2).

1.3.2 Validation amont

Dans notre application, nous avons confronté, dès leur recueil, les propos tenus par les différents experts aux phénomènes observables sur des résultats de simulations inertielles.

Nous ne commencions à élaborer le modèle pratique que lorsque tous les propos tenus par l'expert étaient cohérents avec les évolutions constatées sur les résultats de simulations.

Nous avons pu ainsi éliminer les expertises incertaines et favoriser l'expression de nouvelles connaissances.

1.3.3 Validation aval

Les mêmes simulations ont également été utilisées après chaque passe d'implémentation afin de vérifier la cohérence des connaissances introduites dans la maquette informatique.

Enfin, elles nous ont servi de base pour la validation finale de la maquette dans sa version stabilisée actuelle.

2 CONNAISSANCES ET ARCHITECTURE DU SYSTEME

2.1 INTRODUCTION

Afin de bénéficier d'une modélisation du domaine traité aussi complète que possible, nous avons fait appel à plusieurs experts :

- les ingénieurs d'étude en systèmes inertiels (ingénieurs "matériel" et "logiciel") qui ont acquis une expérience à travers les études de conception et la mise au point de systèmes inertiels hybrides, les simulations de performances et également l'analyse des résultats d'essais en vol. Ils ont donc fourni une connaissance que l'on peut qualifier de profonde.
- de même, les ingénieurs d'essais en vol ayant participé à la mise au point des systèmes qui ont donné leur vision plus expérimentale du comportement des UNIs. Ce sont eux qui ont recensé certaines anomalies ou dérives suspectes survenues au cours de vols d'essais.
- enfin, les pilotes / navigateurs d'essais des Centres d'Essais en Vol qui ont plutôt raisonné en terme de "boîte noire". C'est donc plus une connaissance de surface sur l'utilisation du système de navigation qui a été extraite.

2.2 DOMAINES DE CONNAISSANCES

Au cours des différents entretiens, nous avons mis en évidence 4 principaux domaines de connaissances :

- connaissances sur les systèmes inertiels : sources d'erreurs, leur ordre d'importance, leur influence sur les sorties de navigation, leur fréquence d'apparition,
- connaissances sur les moyens de recalage : précision a prion, influence sur les sorties de navigation, leur fréquence d'apparition,
- connaissances sur le filtrage de Kalman : sources d'erreurs, leur ordre d'importance, leur influence sur les sorties de navigation, leur fréquence d'apparition,
- expérience du navigateur

2.3 REPRESENTATION DES CONNAISSANCES

2.3.1 Introduction

Nous n'avons pas cherché à ce stade à innover au travers d'une architecture ou d'une technique développée spécifiquement pour notre problème. Notre but à travers cette étude n'était pas de concevoir un nouvel outil mais bel et bien d'appliquer des techniques rodées à l'aide d'un produit du marché.

Aussi, suite à l'élaboration du modèle cognitif, nous avons simplement dressé l'inventaire des modes de représentations nécessaires pour implémenter la base de connaissances :

- tout d'abord, une **représentation centrée objet** Celle-ci permet de poser la structure de la base de connaissance en donnant un canevas sur lequel s'appuie et se développe le raisonnement de l'expert .
- pour compléter cette représentation, des **actions** qui sont des morceaux de code procéduraux rattachés aux objets. Celles-ci peuvent être exécutées, soit à la demande (méthodes appelées en partie droite de règles), soit automatiquement lors de l'accès ou plus généralement la manipulation d'un attribut d'un objet (démons).
- les **règles** permettant de bâtir la logique du raisonnement
- enfin, pour permettre le raisonnement à l'intérieur de mondes clos ou pour générer des hypothèses, le **mécanisme de contextes**.

2.3.2 La structure centrée objet

Suite à l'analyse du discours des experts, quatre classifications principales ont émergé :

- Les **sources d'erreurs inertielles**. Celles-ci se décomposent en deux catégories : les sources d'erreurs entraînant des erreurs visibles à long terme et celles dont les erreurs sont observables à court terme. Au niveau le plus bas de cette taxinomie, on retrouve des décalibrations de facteurs d'échelles ou de biais de composants inertiels. Néanmoins, il ne s'agit pas d'une décomposition structurelle complète d'une UNI. Seules les sources d'erreurs significatives ont été conservées.
- Les **événements**. Il est apparu rapidement que parmi toutes les données inertielles possibles en entrée du système expert, certaines étaient plus intéressantes à observer que d'autres lorsque surviennent au cours du vol des événements particuliers qui excitent la dynamique de l'avion. Les virages, changements d'altitude et alignement constituent de tels événements.
- L'**historique**. Celui-ci est une collection de toutes les tendances qui ont pu être enregistrées au cours des vols précédents (erreurs de vitesse au retour parking, ... etc). Il constitue la mémoire du comportement à long terme de chaque UNI.
- La description fonctionnelle du **système de navigation** et de ses paramètres observables (longitude pure, longitude recalée, ...).

2.3.3 Les actions

On a distingué quelques types principaux d'actions à réaliser :

- tout d'abord, un ensemble de programmes qui permettent de détecter l'arrivée d'événements et de bâtir les structures "objet" nécessaires pour les représenter,
- ensuite des attachements procéduraux qui interviennent lors de la création des événements d'une part pour en renseigner tous les champs, d'autre part pour en extraire, le cas échéant, des indices intéressants pour alimenter la logique de résolution,
- enfin, des actions codées directement en parties droites de règles qui permettent de gérer les mécanismes d'hypothèses tant pour les causes d'erreurs inertielles que pour les recalages.

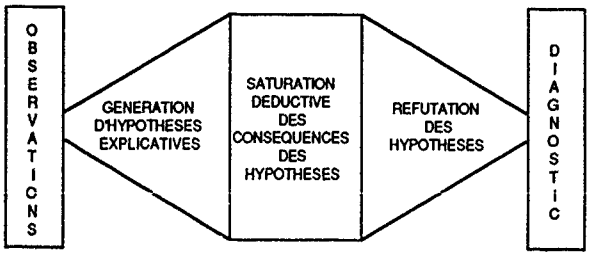
2.3.4 Les règles

On distingue deux ensembles principaux de règles assurant des parties distinctes du mécanisme de raisonnement.

- des règles chargées de traiter les recalages. Celles-ci se chargent de valider ou de ne pas valider les recalages, en lieu et place du navigateur.
- des règles chargées de prendre en compte les erreurs inertielles ainsi que leurs interactions avec les recalages. Celles-ci interviennent en association avec des mécanismes de raisonnement plus complexes que sont les contextes.

2.3.5 Les contextes

La forme de la connaissance, telle qu'elle a été exprimée par les experts, nous a conduit à envisager l'utilisation de contextes. En effet, elle laisse apparaître, entre autres, la nécessité de l'emploi de mécanismes de raisonnements hypothético-déductifs. Un tel mécanisme peut se décrire par le schéma suivant :



Pour implémenter de façon efficace ces mécanismes, une solution pertinente est l'emploi d'un contexte pour chaque hypothèse générée tant que celle-ci n'est pas réfutée.

Nous avons distingué pour l'instant deux niveaux principaux de raisonnement :

- la recherche de causes d'erreurs inertielles. Supposons que l'on constate une anomalie, celle-ci ayant deux causes d'erreurs possibles. On génère alors deux hypothèses et tous les événements détectés par le système sont interprétés en parallèle dans les deux contextes jusqu'à ce qu'un événement puisse prouver ou réfuter une de ces deux hypothèses.
- la détermination de la confiance que l'on peut accorder à un recalage (a posteriori). Dans les systèmes classiques la décision de validation d'un recalage est élaborée à partir d'une image instantanée donnée par le filtre de KALMAN. Dans notre système, nous utilisons toutes les données inertielles disponibles depuis le début du vol pour décider si le recalage est validable ou non. Cependant, lorsque nous avons un doute sur la qualité du recalage, nous générons deux hypothèses concurrentes (recalage correct / recalage mauvais) et nous attendons l'arrivée de nouveaux événements pour déterminer a posteriori la confiance que l'on peut accorder au recalage.

Ces deux niveaux de fonctionnement ne sont cependant pas décorrélés. En effet, la détection de certains phénomènes anormaux après un recalage peut déboucher sur deux interprétations :

- la présence d'une erreur inertielle sur un système bien recalé,
- un mauvais fonctionnement dû à un recalage de qualité moyenne.

Il est donc nécessaire d'avoir une vision différente des erreurs inertielles selon que le recalage est considéré comme bon ou mauvais

L'implémentation d'une telle structure à deux niveaux a été facilitée par l'existence du mécanisme de "View-points" multiniveaux sur l'outil utilisé (ART™).

2.4 ARCHITECTURE GLOBALE DU SYSTEME EXPERT

La figure 2.1 présente l'architecture globale très classique de la maquette du système expert. Elle fait apparaître plusieurs modules importants :

- les informations inertielles en provenances des deux UNI sont lues par un module d'**acquisition de données**. Celui-ci est également chargé de transformer les données brutes en faits acceptables par la base de faits qui est structurée en objets,
- la **base de connaissances** contient toutes les règles et les méthodes nécessaires au raisonnement du système,
- le **mécanisme d'inférence** sélectionne, en fonction des faits existants, des règles ou des méthodes. Il active les règles ou exécute les méthodes, le résultat étant renvoyé dans la base de faits. Il gère également plusieurs hypothèses en parallèle en maintenant la cohérence de la base de faits,
- enfin l'**interface utilisateur** permet d'observer les conclusions du système expert ou de lui fournir les informations supplémentaires (avis sur la validation du recalage par exemple)

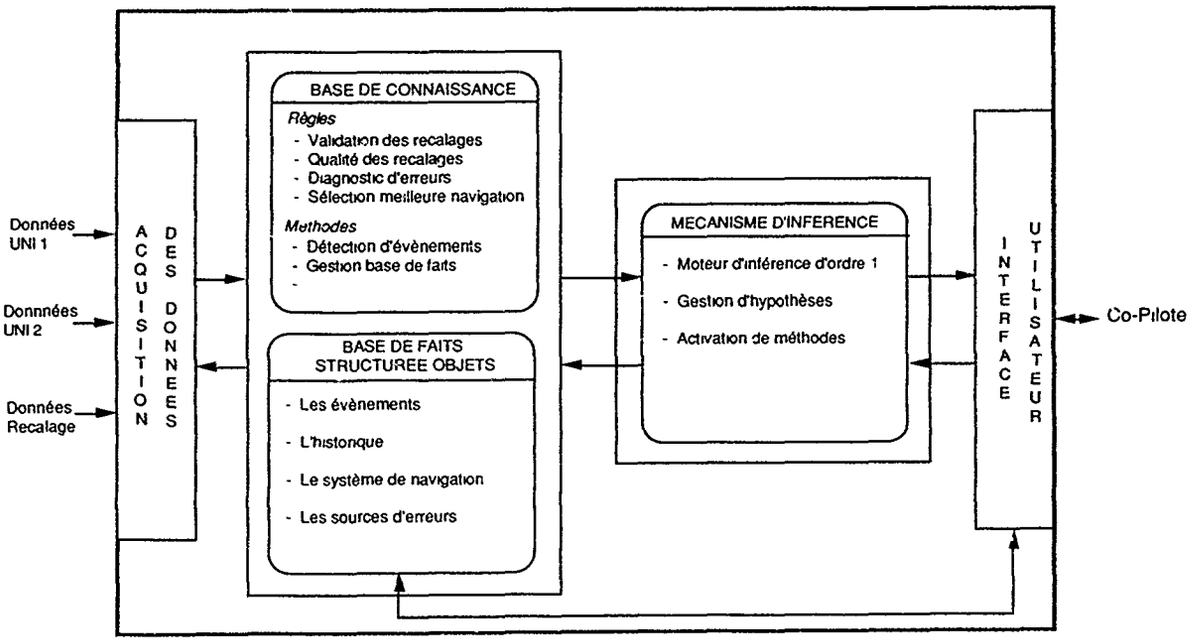


Figure 2.1 **ARCHITECTURE GLOBALE**

3 REALISATION INFORMATIQUE

La maquette de démonstration a été réalisée pour montrer les améliorations que peut apporter le Système Expert pour résoudre les difficultés rencontrées dans un cas d'utilisation opérationnelle. Pour cela, le Système Expert a été intégré dans un environnement simulant le fonctionnement des systèmes avec lesquels il sera appelé à travailler : les deux UNI et les instruments du poste de navigation.

Cette maquette réunit les fonctionnalités suivantes (voir figure 3.1):

- possibilité de pilotage de la trajectoire à l'aide d'une souris avec visualisation de type "Tête Haute" de l'image synthétique du MNT correspondant,
- simulation synchrone des deux systèmes inertiels hybrides, ainsi que des moyens de recalage,
- simulation du poste de commande et de navigation PCN et du poste de sélection de modes PSM afin de permettre la visualisation de divers paramètres et la prise en compte des recalages,
- aide à la navigation par l'intermédiaire du Système Expert,
- moyens de dépouillement.

Ces fonctionnalités ont été implémentées en **deux parties** :

- **simulateur pilotable de navigation inertielle**, réalisé sur mini-système HP1000 A-900,
- **Système Expert d'Aide à la Navigation et simulation des équipements PCN & PSM**, réalisés sur machine LISP SYMBOLICS avec le générateur de systèmes experts ART™ de INFERENCE Corp

La maquette est constituée de neuf processus asynchrones fonctionnant en pseudo temps réel. Le système expert et le simulateur des boîtiers PSM/PCN sont implémentés sous la forme de deux processus communiquant par une liaison de type "producteur/consommateur" : un premier processus fait l'acquisition des données, soit depuis un fichier, soit depuis une simulation inertielle et les envoie à un processus ART™.

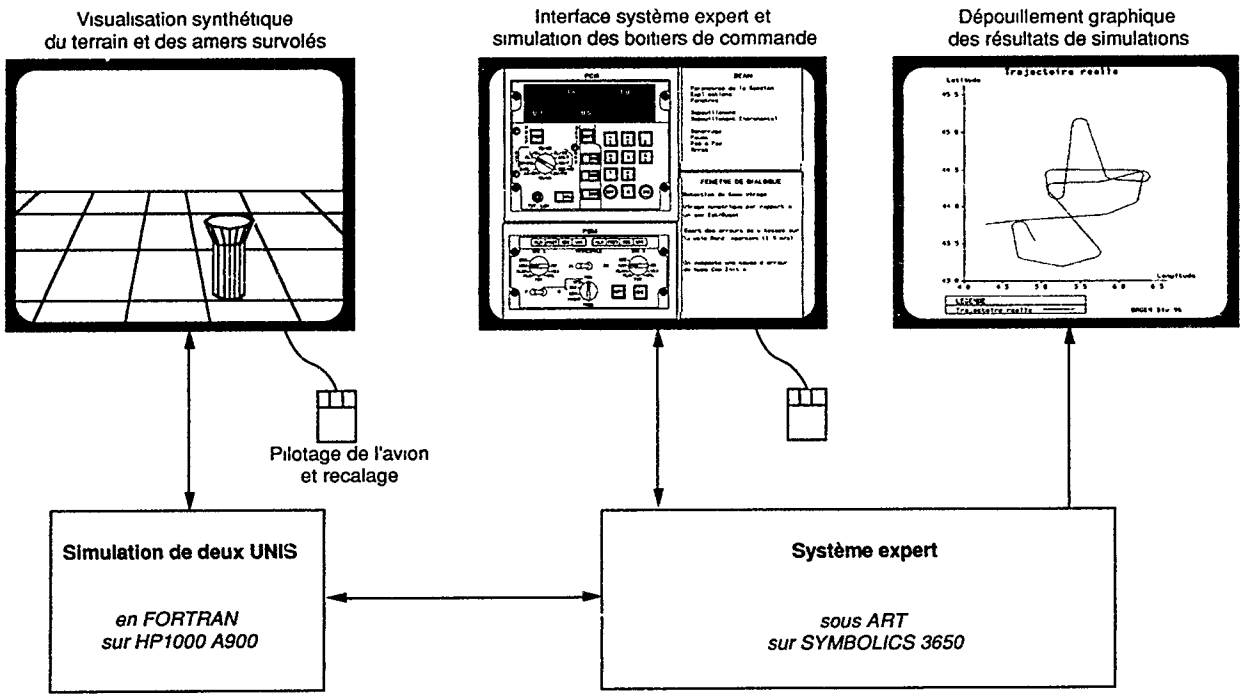


Figure 3 1 **MAQUETTE DE DEMONSTRATION**

4 RESULTATS OBTENUS

Afin de construire un ensemble cohérent dans le temps imparti pour l'étude, nous avons limité le nombre de causes d'erreurs prises en compte.

Nous avons traité complètement le diagnostic des six plus importantes causes d'erreurs ainsi que leurs corrélations avec la qualité des recalages validés.

Les bases de connaissances obtenues ont été testées sur une dizaine de simulations.

Plusieurs résultats ont été obtenus :

- d'une part, la prise de décision en ce qui concerne la validation ou non d'un recalage est au moins aussi bonne que celle d'un navigateur;
- d'autre part, pour le diagnostic des sources d'erreurs, nous avons atteint un taux de détection des anomalies de 95% et un taux de fausses détections voisin de 0%;
- enfin, en corollaire des deux points précédents, le système expert ayant effectué des choix plus judicieux tant pour la validation des recalages que pour la sélection de la meilleure navigation, les performances de navigation obtenues sont globalement meilleures.

Ce dernier résultat augmente l'intérêt du système expert en raison de l'amélioration de la sécurité du vol (localisation précise en phase de pénétration) et de l'efficacité de la mission (précision de la position et de la vitesse pour l'initialisation des armes).

Par ailleurs, cette première phase a permis de mettre en place un début d'approche méthodologique pour le projet, en utilisant une méthode déjà reconnue et en la complétant par un principe de validation spécifique au projet.

CONCLUSIONS ET PERSPECTIVES

Comme l'attestent les résultats exposés au paragraphe précédent, la maquette de démonstration réalisée dans le cadre de cette étude a réussi à montrer la validité, l'intérêt et la faisabilité d'un système expert pour la surveillance des systèmes inertiels et l'aide à leur mise en œuvre.

Cependant, avant d'aboutir à un produit opérationnel, plusieurs étapes semblent nécessaires :

- tout d'abord un complément de la base de connaissances afin de prendre en compte un nombre plus important de sources d'erreurs,
- ensuite la conception de l'ergonomie de l'interface d'un tel système avec le pilote,
- et enfin, l'étude de l'embarquabilité du logiciel ainsi écrit dans un environnement opérationnel.

Enfin, le système présenté ici s'intègrera tout naturellemment au projet de copilote électronique.

BIBLIOGRAPHIE

[BUC 84]
RULE-BASED EXPERT SYSTEMS
BUCHANAN & SHORTLIFFE
Addison Wesley 1984

[FAR 87]
ELEMENTS D'INTELLIGENCE ARTIFICIELLE
Henry Farreny Malik Ghallab
Hermes

[GAL 88]
TRANSFERT DE CONNAISSANCES
J.F. GALLOUN
EYROLLES

[GIL 85]
MILITARY APPLICATIONS OF EXPERT SYSTEMS
J.F GILMORE
Actes du Congrès "Les Systèmes Experts et leurs Applications" AVIGNON 1985 p 251-321

[HAR 88]
ACQUISITION DU SAVOIR POUR LES SYSTEMES EXPERTS
A.HART
MASSON Collection Sciences Cognitives

[HAY 83]
BUILDING EXPERT SYSTEMS
HAYES-ROTH, WATERMAN & LENAT
Addison Wesley 1983

[ROY 88]
AIDE A LA DECISION
B. ROY, D. BOUYSSOU
AFCET, "Interfaces" Mars 1988

[VOG 88]
GENIE COGNITIF
C. VOGEL
MASSON Collection Sciences Cognitives

INTEGRATED CONTROL AND AVIONICS FOR AIR SUPERIORITY: A KNOWLEDGE-BASED DECISION-AIDING SYSTEM

Donald J. Halski
Robert J. Landy
McDonnell Aircraft Co
PO Box 516
St Louis MO 63166-0516

James A. Kocher
Wright Research and Development Center (WRDC/FIGX)
Wright-Patterson AFB OH 45433-6553

ABSTRACT

This paper describes an ongoing program which is developing an effective, real-time, knowledge-based decision-aiding system for air combat. The Integrated Control and Avionics for Air Superiority (ICAAS) advanced development program is sponsored by the Wright Research and Development Center (WRDC) at Wright-Patterson Air Force Base, Ohio. A research and development contract was awarded to the McDonnell Aircraft Company, St Louis, Missouri, in September 1987. The program objective is to develop, integrate, and demonstrate critical technologies which will enable United States Air Force tactical fighter aircraft to kill and survive when outnumbered as much as four to one by enemy aircraft during air combat engagements. Primary emphasis is placed upon beyond-visual-range (BVR) multiple target attack capability with provisions for effective transition to close-in combat. Knowledge-based pilot decision-aiding techniques and expert system methods are used to achieve substantially enhanced offensive and defensive capabilities compared to current operational systems. Situation awareness information and recommended actions are computed to aid the pilot in selecting the most effective attack and defend engagement options. The system maximizes opportunities for missile launch against multiple enemy aircraft while maintaining options to defend when necessary. Sufficient integration and automation are provided for application to a single seat fighter An intraflight data link is used for enhanced mutual support between individual flight members to make them a more effective combat team.

SYSTEM INTRODUCTION

Specific functions of the ICAAS system include attack management, tactics, attack guidance, defensive assets manager, and aircraft performance monitor as illustrated in Figure 1. Attack management automatically controls onboard sensors and computes a correlated summary of track files from ownship sensors as well as information received over the data link from other flight members The track file summary is displayed to the pilot to maximize his situation awareness. The tactics function uses a rule-based approach which blends a number of knowledge-based values to assist the pilot in assessing the overall situation. Scenario attributes are evaluated in real-time against a tactics knowledge base, extracted from expert pilots, to recommend the most viable tactics. Specific coordinated assignments are provided to each flight member Attack guidance provides flight trajectory information for achieving missile launch solutions against multiple enemy aircraft while maximizing ownship survival. Faster than real-time missile fly-out models are used to perform engagement predictions and guide the aircraft to recommended launch points. The defensive assets manager is activated by the sensed or inferred launch of a threat missile against ownship. An extensive data base of early, midcourse, and endgame maneuvers is used to generate and validate available evasive options and to select the most promising for recommendation to the pilot. The aircraft performance monitor uses stored knowledge of aircraft performance parameters to aid the pilot in achieving the best real-time dynamic aircraft performance. The knowledge-based decision-aiding features of these specific ICAAS system functions are addressed in more detail in the following paragraphs.

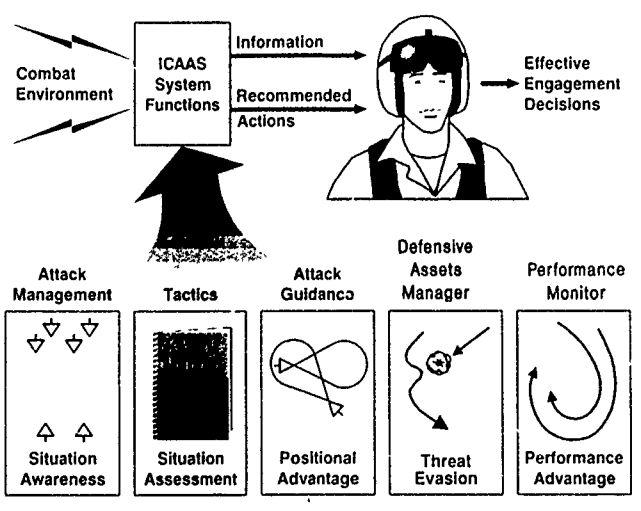


Figure 1. ICAAS System Functions

ATTACK MANAGEMENT

The ICAAS Attack Management (IAM) function provides a high degree of situation awareness to the pilot. It forms an interface between the pilot and his onboard sensors and weapons. IAM manages multiple sensors and correlates track files from ownship sensors as well as other flight members to form a single picture of current threat location and identification. Sensors are also coordinated to acquire sufficient data to launch and guide weapons. The IAM fire control function identifies current launch opportunities and provides aim-dot steering cues plus other launch parameters against the primary targets. The pilot maintains control of search volume, emission mode, and launch decisions. The IAM function is a special adaptation specifically

for ICAAS of the Air-to-Air Attack Management (A^3M) system developed by the WRDC Avionics Laboratory. Reference 1 gives a detailed description of A^3M. Individual features of IAM are presented below.

Sensor Control - Pilot workload associated with the task of individual sensor control can be very high. A modern radar has more than twenty different operating modes, and a pilot can easily dedicate too much of his attention to operating the radar. Since future fighters will likely have multiple sensors, the pilot of a single seat fighter can be overwhelmed if required to manually operate all sensors.

IAM sensor control logic commands each of the ownship sensors as required to update all track files according to their urgency as well as to search the volume defined by the pilot or the tactics algorithm. The knowledge-based sensor control logic accounts for the availability of each sensor, its particular operating modes, and the level of emissions allowed by the pilot. The emission modes provided are. active with full sensor emissions, low probability of intercept with limited emissions to periodically acquire range as required, and passive using only non-emitting sensors such as an Infrared Search and Track System (IRSTS). The pilot can assume total control of any individual sensor if he chooses.

Multi-Source Integration - In current fighter aircraft, information from individual sensors is presented to the pilot on separate cockpit displays. The pilot must mentally correlate the information into a common frame of reference. This limits his ability to establish and maintain a good awareness of the tactical situation. The ICAAS multi-source integration (MSI) feature correlates sensor track files from multiple ownship sensors into a single track file summary. MSI track files are derived from kinematic and covariance data reported from individual sensors. A nearest neighbor approach is used to examine sensor reports for correlation with existing MSI track files Track files are then created, updated, propagated, or deleted as appropriate.

Target identification is derived using a heuristic voting method. Each sensor provides, when available, information regarding target identification (friend, foe, neutral), class, and type. Information from each sensor is weighted and combined with other sources to produce a weighted sum. Comparison with specified thresholds is used to establish a confidence factor.

Internetting - An intraflight data link between ICAAS flight members permits rapid exchange of track files, targeting assignments, and selected tactics. This capability is referred to as internetting Additional information may be available from wide area data network sources such as Airborne Warning and Control System (AWACS) or ground control, but the intraflight information exchange is the focus for ICAAS application (see Figure 2)

The IAM internetting function compiles a correlated summary of track files which are received from all fighters This improves threat track file accuracy and increases identification confidence. The pilot of each aircraft within the intraflight network can view all track files, from his own geometric viewpoint, and can distinguish between internetted and ownship MSI track files IAM internetting dramatically increases situation awareness of the target environment with little or no voice communication required Sensor scan volumes can be coordinated to provide maximum total coverage and thus reduce the probability of undetected threats Two internetted fighters can triangulate with line-of-sight passive sensors, such as infrared, to accurately determine target range while both remain covert. Accurate relative position information can also reduce the need to maintain visual contact between flight members

Fire Control - Baseline weapons for ICAAS fire control system functions are the advanced medium range air-to-air missile (AMRAAM), AIM-9 sidewinder, and 20mm gun. Data from MSI and internetted track files plus embedded missile fly-out models are used to compute ownship missile launch envelopes and threat missile envelopes against the ownship

During internetted operation, a cooperative attack subfunction coordinates targets with each blue fighter within the flight according to the assignments by the blue pilot tactical lead or by the ICAAS Tactics Algorithm. Each pilot can alter his own attack sequence as recommended by the tactics algorithm. Aim-dot steering is provided for the top two priority targets, and the sensor manager ensures that ownship sensors are tracking these targets.

An internetted fire control capability called cooperative launch is being conceptually developed This allows one fighter to launch an AMRAAM missile which will be guided toward the target by another friendly fighter. The launching aircraft can approach the engagement as covertly as possible, execute the launch based on target track data supplied by the other flight members over the data link, then immediately disengage to avoid lethal enemy launch envelopes. From a greater standoff range, the guiding aircraft can provide midcourse guidance for the blue missile. Meanwhile, the launching fighter can re-enter the engagement in support

TACTICS ALGORITHM

As a pilot approaches a potential BVR engagement, the ICAAS tactics function assists with selection and implementation of an appropriate tactic to achieve a first missile launch opportunity. A "tactic" consists of target allocation and prioritization, intercept geometry, and a strategy for electronic emissions.

The tactics algorithm consists of four principal elements: menu generator, checkpoint generator, monitor, and internetting executive The menu generator makes tactical recommendations based on real-time threat information from IAM, blue aircraft status, and a tactics knowledge base Checkpoint generator defines the selected tactic in terms of relative geometry goals, airspeed, aircraft radar or infrared signature management, and electronic emissions strategy. The tactic monitor modifies checkpoints as the engagement unfolds and evaluates the success of the tactic in terms of accomplishing desired goals. The internetting executive coordinates tactics algorithms which are executing independently on internetted aircraft. The pilot can tailor the response of the tactics algorithm according to personal preferences or new threat tactics experienced in earlier engagements by altering preflight input parameters which affect tactics calculations. Examples include preferred tactic categories and desired launch ranges.

Tactics options are determined by assessing the enemy (red) formation and friendly (blue) formation capabilities. Candidate tactics are then generated and rank ordered within the context of the engagement scenario. This process is illustrated in Figure 3.

The tactics knowledge base designed for ICAAS is applicable to the following three scenarios for both the F-15 and a mid-90's fighter:

- AWACS defense against a high, fast threat
- Fighter escort of bombers
- Base defense

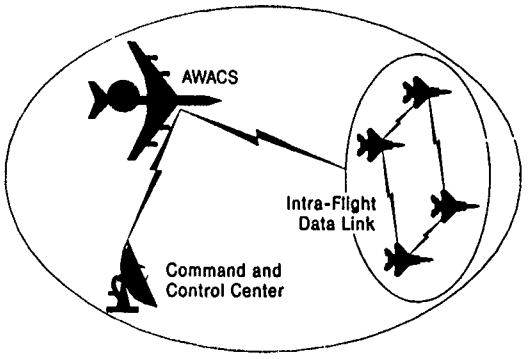


Figure 2. Data Link Network

Nine pilots were interviewed to determine current day tactical objectives for each of the ICAAS scenarios The pilots interviewed represented a combined total of over 17,000 flight hours in both Tactical Air Command (TAC) and Air National Guard (ANG) units. Over half of these hours were in the F-15. Interviews were structured to solicit information regarding geometry, speeds, altitudes, sensor management, target sorting and prioritization, missile selection and launch range, and salvo strategy.

Digital off-line analyses and rapid prototyping were conducted to refine the tactic specifications and tolerances. Scenarios were executed from early set-up to first launch position Shot opportunities and steering philosophies were then evaluated. The result is a set of 35 different tactics which are characterized by. mission type, intercept goals, altitude relative to the threats, speed of blue aircraft, relative position of the blue aircraft, and sensor strategy. Each tactic is defined for both the tactical lead and the wingman aircraft. The initial knowledge base contains current operational tactics which are tuned for the ICAAS weapons and avionics suites. Work is continuing in conjunction with Tactical Air Command pilots to expand the knowledge base with more innovative tactics designed to exploit the intraflight data link and signature features of an advanced aircraft.

Menu Generator - A tactics option menu is computed using the logic tree structure depicted in Figure 4. The most fundamental determinant in tactic selection is blue mission as the first node of the tree Branches emanating from this node represent the three mission scenarios which are implemented in the tactics algorithm. Secondary nodes of the tree assess the nature of the threat within geometric and kinematic constraints of the appropriate mission. Processing the relative geometries and velocities determines a level of threat along each red aircraft velocity vector. Branching occurs according to urgency in dealing with the threat. Subsequent nodes in the tree are activated by situational parameters such as altitude of the threats, number and type of blue defenders, and the rules of engagement

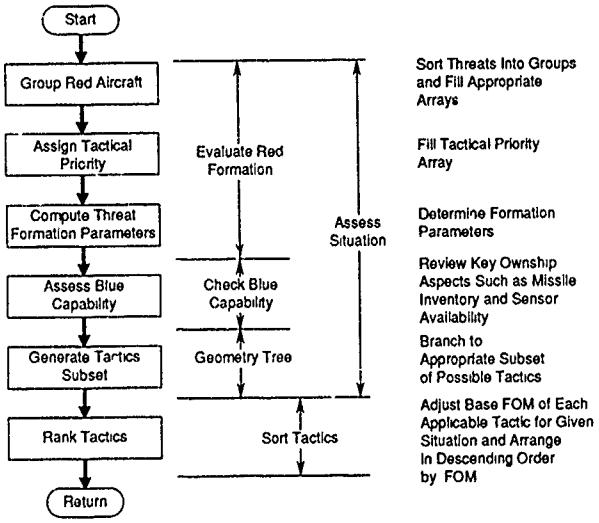


Figure 3. Tactic Options Menu Generation

At the end of any set of tree branches is the list of applicable tactics for that particular node and a stored base figure of merit (FOM) for each. The tactics included in each subset and corresponding FOMs are the result of pilot interviews and off-line analyses using digital air battle simulations The aircraft performance emphasis within these different tactics also varies, from a full afterburner direct approach to a minimum fuel consumption loiter.

Target assignments and launch sequences are computed for a particular tactic, subject to priority inputs from the pilot The solution is a function of the blue mission, threat positions relative to a high value asset, and target tactical priority Sorting between two blue aircraft considers threat separation in azimuth and elevation so a single blue is not responsible for widely separated threats. Launch sequencing then results from the intercept geometry of the tactic along the attack axis. The pilot's assignments and sequences are considered top priority for each blue aircraft. Targets assigned by the algorithm are adjusted if a blue aircraft has insufficient missiles to attack all assigned targets.

Relative ranking of tactics options is determined using the final subset of tactics defined by the logic tree. The base FOM stored with each tactic is incremented or decremented according to weighting factors assigned to the sixteen engagement attributes shown in Figure 5. Final values are normalized so the FOM assigned to each tactic has a value between 0 (very poor tactic) and 1.0 (very good tactic) Tactics

are ranked in descending order of FOM and made available for cockpit display. The lead pilot is free to select any tactic from the available menu.

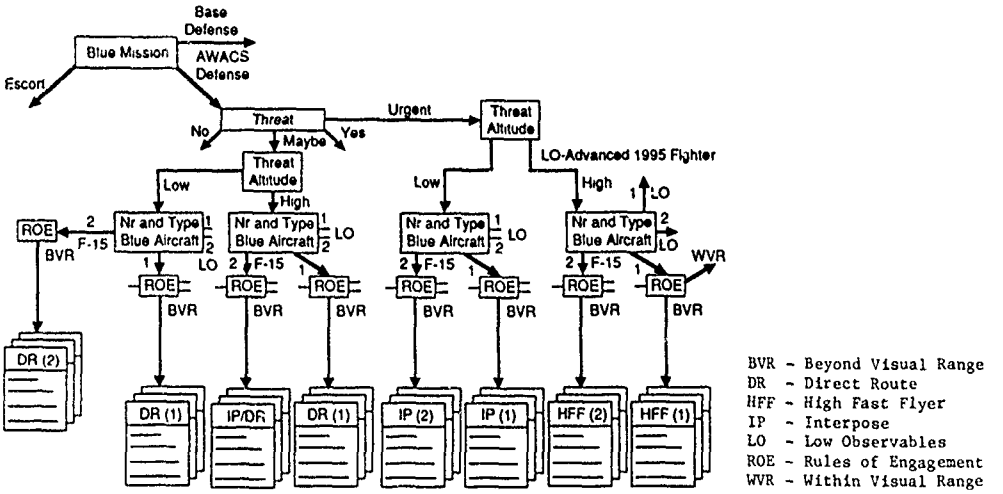


Figure 4. Partial View of Tactics Geometry Tree

Checkpoint Generator - Once the flight leader has selected a tactic, a series of checkpoints is computed. These checkpoints provide important information for effective, coordinated execution of the tactic including flight path constraints, goals for energy management, sensor emission levels and scan volumes, target priorities, countermeasure employment cues, and first weapon employment cues. Wingmen are automatically provided their respective checkpoints within the coordinated tactic. Thus, each flight member knows his own individual assignments and the flight can operate as an effective team. Figure 6 illustrates initial checkpoint generation for two aircraft executing a single-side offset tactic.

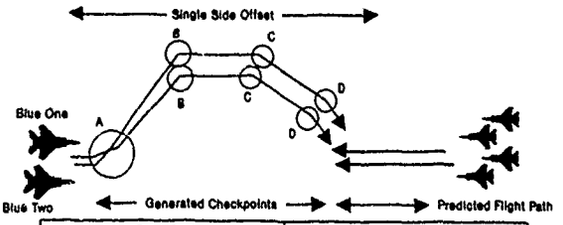
Associated with a set of checkpoints is a set of time or spatial tolerances. These are the blue flight path tolerances, the checkpoint tactical tolerances, and the red maneuver tolerances. The blue flight path tolerances are the most restrictive because the attack guidance function performs path predictions and optimization within these tolerances. Checkpoint tactical tolerances allow for variation in the implementation of a tactic by the pilot without readjustment of the checkpoint parameters and the ensuing shifts in the predicted path. The red maneuver tolerance defines a set of threat reactions within which a particular tactic automatically makes adjustments. Outside of this tolerance, the tactic FOM will be adjusted to reflect a situation which is not progressing as well as planned

Tactics Monitor - A tactics monitor subfunction continually monitors progress of a selected tactic to see if the desired results are being achieved. A general monitor observes those parameters of air-to-air engagements which are similar regardless of the specific tactic being implemented. Examples are a change in target assignments or threat formation, or the sudden detection of a new threat at or near a launch solution on blue The scenario events checked by the ICAAS general tactic monitor are listed in Figure 7 along with the monitor response. The general monitor also adjusts the FOM for the selected tactic based on changing conditions as the scenario unfolds.

Item	FOM Factor	Definition
1	Number of Red Aircraft	Total Number of Threat Aircraft
2	Number of Threat Groups	Result of Grouping Red Aircraft Processing
3	Number of Advanced Red Fighters	Number of Enemy and Unknown Aircraft
4	Number of Greatest Concern Fighters	
5	Formation Axis Angle	Angle Between Composite Threat Heading and Attack Angle
6	Formation Shape	Relative Sizes of Minimum and Maximum Moments of Inertia
7	Formation Size	Metric for Horizontal and Vertical Separation of Threats
8	Sensor, Electronic Countermeasure Availability	Point System to Establish Overall Blue Avionic System
9	Weapon Availability	Total Number of Blue Missiles Compared with Number Reds
10	Electronic Support Measures Status	Is Blue Detected By Threat
11	Desired Engagement Zone Achievability	Can Blue Achieve Desired Engagement Zone to Engage Threat
12	Emissions Control Preference	Compliance of Recommended Tactic with Pilot Preferred Emissions Level
13	Relative Altitude Preference	Compliance of Recommended Tactic with Pilot Preference
14	Attack Quarter Availability	Can Blue Aircraft Achieve Tactics Intended Attack Quarter Within Time Constraint
15	Pilot's Target Designation	Compliance of Recommended Target Assignment With Pilot Entries
16	Visual Mutual Support Preference	Compliance of Recommended Tactic With Preferred Lead/Wing Formation

Figure 5. Tactics Ranking Factors

A tactic-specific monitor observes those characteristics which are peculiar to the currently selected tactic, updates the parameters of the checkpoints, and determines the sequencing to the next step in the tactic plan. For example, the tactic-specific monitor recomputes the amount of time needed to obtain a desired offset and adjusts the guidance specification and tolerances for attack guidance accordingly. When the offset is achieved, the tactic-specific monitor triggers the next action, for example, recommended activation of a radar or ECM. The tactic-specific monitor also reacts to a degraded tactical situation. For example, if the blue pilot flies outside of the current checkpoint tolerance, alternative tactics are evaluated.



	State Goal		Equipment Cues	
	Blue One	Blue Two	Blue One	Blue Two
A	Initiate SSO	Initiate SSO, Enhance SA	Passive Sensors Only	Maintain Sensor Coverage
B	Achieve Offset/ Altitude Split	Achieve Offset/ Altitude Split	Passive Sensors Only	Maintain Sensor Coverage
C	Achieve Turn In Range	Achieve Turn In Range	Passive Sensors Only	ECM/MCM
D	Achieve Launch Range	Achieve Launch Range	Activate Sensors/ Launch	Launch

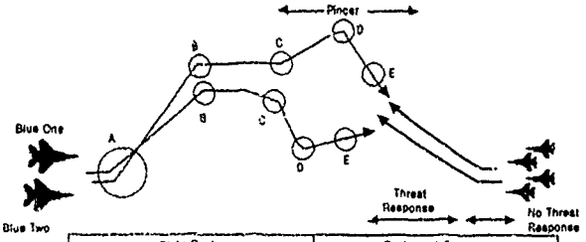
Figure 6. Tactics Initial Checkpoints

The tactic-specific monitor also contains a knowledge base for the expected set of threat responses for each tactic. Such parameters as spatial locations of checkpoints, offset angles, or target intercept angles are altered if the threat response falls within the threat maneuver tolerance. Figure 8 illustrates a modification to the initial single side offset tactic due to an altered, but anticipated, threat response. The FOM is significantly decremented if the threat package response falls outside the defined threat maneuver tolerance.

Scenario Event	General Monitor Response
Change in Target Assignments Sequence	Redefine Launch Sequence and Adjust Checkpoints According to Primary Target
Changes in Threat Formation Parameters	Assess Impact on Tactic Intent
Increase in Number of Threats	Insert Threat into Target Assignment Array
Pop-up Threat is Immediate Threat to Ownship	Make #1 Target and Shift to Attack Mode
Threat Turns Sufficiently Away From Blue	Shift to Situational Awareness
Red Threatens Blue Asset	Shift to Commit Tactic
Blue Near Desired Launch Range	Shift to Attack Mode

Figure 7. Tactics Algorithm General Monitor

The tactics monitor updates the FOM for the selected tactic as the engagement unfolds. It compares the current FOM to a predefined threshold input by the pilot. This minimum FOM is used to characterize a desperate situation when a tactical disengage should be recommended. If the FOM of the current tactic is decreasing, a search for a potentially better tactic is initiated. A recommendation to switch to a better tactic is made if a significant improvement is available. In all cases, the tactics algorithm provides only an advisory to the pilot. He still makes the decision to continue with the existing tactic or to select an alternative.



	State Goal		Equipment Cues	
	Blue One	Blue Two	Blue One	Blue Two
B	Achieve Offset/ Altitude Split	Achieve Offset/ Altitude Split	Passive Sensors Only	Maintain Sensor Coverage
New C	Initiate Pincer	Initiate Pincer	Passive Sensors Only	Maintain Sensor Coverage
New D	Bracket Threat	Bracket Threat	Passive Sensors Only	ECM/MCM
New E	Achieve Launch Range	Achieve Launch Range	Activate Sensors/ Launch	Launch

Figure 8. Tactics Checkpoint Adjustment

Internetting Executive - Tactics algorithms in each blue aircraft operate independently of each other. To coordinate the menu generation and tactics monitoring processes between the blue lead and his wingman, key tactics algorithm data and sensor information from each blue aircraft are exchanged

over the intraflight data link. The executive module (exec) then coordinates the interaction between the tactics algorithms of each aircraft. When two or more blue aircraft are present and the internetting data link is valid, exec checks the role of the ownship (lead or wing). In the lead aircraft, the exec computes the group FOM (average of lead and wing individual FOMs) which is displayed to both blue pilots. In the wingman aircraft, the exec ensures that the wingman is viewing the same tactic as the lead pilot. Thus, the wingman's display remains in unison with the flight leader. Also, the exec ensures that only the lead pilot can select a tactic for implementation by both blues.

When the internetting data link is not valid, the action of the tactics algorithm depends upon whether a tactic had been selected prior to the net going down. If no tactic was selected, the lead and wing pilots coordinate target assignment and tactic selection by voice call, each using his tactics algorithm with situational data provided by ownship sensors. If a tactic was being implemented when the data link went down, exec determines if the remaining ownship checkpoints of the tactic can be continued. This is a function of the continued availability of track files on the assigned targets and the steering philosophies associated with the remaining checkpoint legs. If the tactic can be continued, the tactic monitoring function in each blue aircraft continues to process its respective set of checkpoints. If the tactic cannot be continued, exec initiates a new menu generation based on non-internetted operation.

ATTACK GUIDANCE

Primary functions of attack guidance are to generate flight path trajectories which will satisfy criteria associated with the desired tactic and to compute recommended launch points against multiple assigned targets. It is designed to enhance pilot awareness of the engagement by providing information on predicted flight paths and missile impact zones. Specific kill and survival metrics associated with each assigned target indicate the likelihood of target kill as well as danger to the blue aircraft from all threat aircraft. Attack guidance overcomes a limitation in current fighter attack systems where only the highest priority target is designated for attack, and a fire control solution is computed without consideration of other threats to ownship. ICAAS attack guidance considers the overall situation and future ownship and threat position, thus greatly increasing pilot awareness of the future consequences of his near term actions.

The heart of attack guidance is an engagement predictor. This predictor uses embedded knowledge of ownship performance capabilities, energy management schedules, tactical goals, plus red and blue weapon envelopes to predict future relative positions and progress of the engagement. A flight plan is initially constructed based on inputs from the tactics algorithm and the pilot, such as target assignments, tactics checkpoints, and ownship aircraft speed and altitude goals A four-dimensional (time and space) trajectory is generated which is responsive to the goals and constraints from the flight plan Recommended launch points are determined for each individual target in conjunction with kill and survivability metrics. Predictions are continuously updated during all mission phases to account for both ownship and threat aircraft maneuvering. Outputs are used to drive cockpit displays and to generate steering commands for pilot manual control. An automatic flight control coupler is also available if activated by the pilot. Figure 9 illustrates the features of attack guidance versus two red aircraft.

The trajectory predictor algorithm integrates differential equations, which describe the aircraft motion, while controlling the aircraft through a set of autopilot-like guidance laws. The integration process takes place at a rate many times faster than real time. Variable integration step sizes are used so that near term predictions are computed more accurately than those further in the future, thus reducing the computer throughput requirements. The fast fly-out computations consist of five parts:

- Predict Engagement
  - Identify Missile Impact Zones
  - Recommend Launch Points
  - Compute Engagement Kill / Survivability Metrics
  - Generate 4D Trajectories
- Update Predictions
- Support Engagement Execution

Legend
- Tactical Engagement
- Attack
- Disengagement
- Threat
- Launch Point
- Missile Impact Zone Boundary

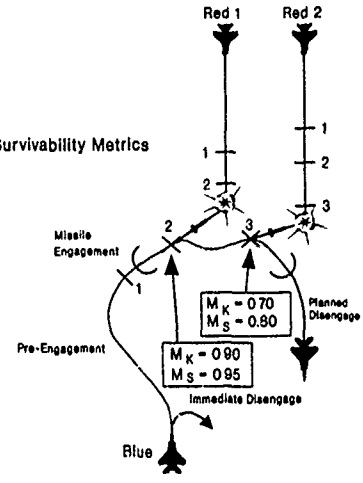


Figure 9. Attack Guidance

- Extrapolate enemy aircraft position based upon altitude and airspeed at the time prediction update begins;

- Computate error signals at every integration step to drive simulated blue aircraft into capturing the desired aircraft state goals contained in the flight plan;

- Integrate a set of differential equations describing aircraft motion;

- Determine control adjustments required to achieve the profile constraints specified in the flight plan; and

- Curve fit the initial portion of the predicted profile to generate input data for the Control Coupler.

Flight path predictions and associated flight guidance algorithms are tailored to support four mission phases: enroute, tactic, attack, and disengage.

- Enroute Phase: Enroute trajectory planning is provided for those portions of flight just before and after the threat engagement. Trajectory predictions are based on embedded performance schedules for optimal climb, cruise and descent speed, and altitude. These schedules account for aircraft weight, stores, and actual ambient conditions.

It is assumed that all navigation waypoints have an assigned altitude. Speed may be fixed or free. If speed is free, an optimal value is selected, either maximum range for waypoint steering or maximum endurance for CAP orbits. Direct path steering is used to navigate from waypoint to waypoint, flying over the waypoint on the inbound leg. The logic includes the ability to capture and hold a CAP orbit with a pilot-specified leg length and heading. CAP orbit is maintained until an exit command is given by the pilot.

- Tactic Phase: The checkpoints defined by the tactics algorithm are typically stated in terms of a desired geometry relative to the threat aircraft, and, as such, do not constitute an inertial flight path. Engagement predictions applies the guidance laws prescribed by tactics (such as line-of-sight offset) against an extrapolated threat flight path to evolve a four-dimensional inertial trajectory suitable for cockpit display.

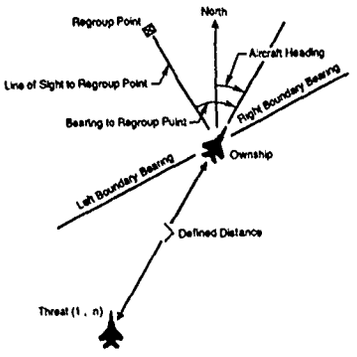


Figure 10. Danger Zone Clearance for Disengage

- Attack Phase: This processing computes the attack trajectory against each assigned target aircraft. Embedded in the generation of the trajectory are the kill and survival metrics and the recommended launch points against the assigned targets. The attack path starts when the tactic algorithm first launch objective has been achieved or when the pilot selects pure attack steering. The attack trajectory ends following the predicted (or actual) launch against the last assigned threat aircraft and completion of target illumination. Attack processing supports multiple launches against a single target as well as multiple missiles inflight against multiple targets.

The attack trajectory is constructed sequentially from target to target based on the defined launch sequence. Blue launch criterion is based on achieving a desired launch range within a defined level of survivability. Both the launch range goals and the minimum acceptable survivability are input by the pilot. The desired launch range is measured in percent of maximum missile launch range (RMAX). Survivability is a function of: the total number of threat aircraft, the danger each threat imposes based on maximum missile ranges, and the predicted success of blue missiles. Post launch missile illumination requirements are imposed on the trajectory.

- Disengage Phase: The disengage trajectory is a maneuvering flight path to escape danger from all threat aircraft, followed by a direct-to navigation path to a designated regroup point.

Determination of the danger posed by the combat situation during disengage processing is illustrated in Figure 10. Ownship is declared to be clear of the engagement if: all threat aircraft are behind a line through the ownship position perpendicular to the line-of-sight from ownship to the regroup point, all threats are farther away than a pre-defined distance, and ownship heading is within 90 degrees of the bearing to the regroup point.

DEFENSIVE ASSETS MANAGER

The Defensive Assets Manager (DAM) responds to detection of air-launched threat missiles and provides the fighter pilot with a recommended defensive response derived from a stored knowledge base. Options considered for response include infrared and radar countermeasures (expendable and non-expendable) applied in conjunction with early, midcourse, and endgame aircraft maneuvers. Early kinematic maneuvers to either avoid or penetrate and escape the maximum lethal intercept zone are highly effective when the threat missile has been launched from long range. If the threat missile was launched in a no escape condition, midcourse maneuvers with countermeasures are used to defeat the missile seeker, followed by endgame evasive maneuvers if required. DAM options are illustrated in Figure 11.

DAM is initiated when onboard sensors detect a missile launch or missile in flight. Missile launch can also be inferred from threat aircraft actions. Multiple missiles may be reported at any given time, but only the single highest priority threat is reported to the evasion algorithm for defensive response. A DAM scheduler eliminates missiles which are not a threat to ownship through range, closure, and guidance checks. Remaining missile reports are then prioritized according to predicted

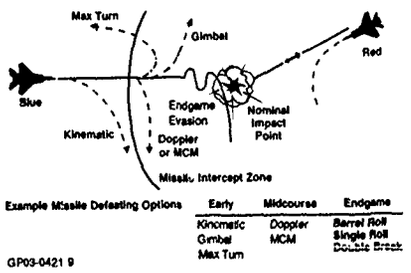


Figure 11. Defensive Assets Manager

time to closest approach. This assures that the missile selected as the greatest threat is within lethal range, is guiding on ownship, and has the shortest time-of-arrival.

An interface between DAM and the IAM sensor manager provides special management of missile track files. IAM initiates and maintains an MSI track following a launch alert from ownship radar or IRSTS sensors, just as it does for any new aircraft track acquisition. Missile tracks which are positively identified as a threat to ownship, or whose threat status is unknown, are given top track service priority over all other tracks. The predicted near-term missile flight path, computed by DAM, can be used to help re-acquire a missile should IAM fail to update the missile track DAM then issues special requests on the missile to IAM to ensure active MSI track files on known threat missiles are updated. It also monitors tracks of threat aircraft with a high probability of launch as determined by RWR detection

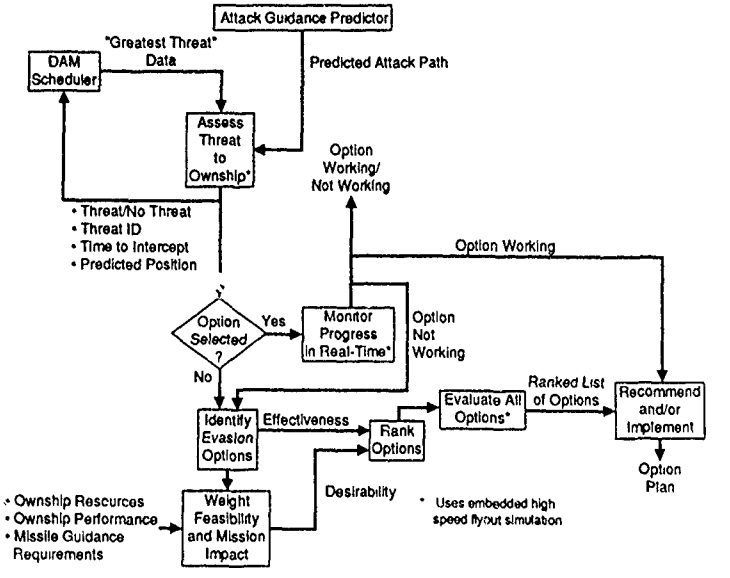


Figure 12. Defensive Asset Manager Missile Evasion Algorithm

DAM assembles the missile track reports from IAM track files for communication to the scheduler Due to the time criticality of information generated by the Missile Warning Sensor (MWS), MWS reports are not processed by IAM. This data is passed directly to the DAM scheduler for endgame response recommendation

The missile evasion algorithm is activated when the DAM scheduler determines the greatest threat to ownship. The evasion algorithm executes once per second as long as a missile poses a threat to ownship The major functional components are shown in Figure 12 and are described below.

Threat Assessment - To determine the level of danger to ownship, the threat missile flight path is projected forward in time to its terminal phase through the use of an embedded high speed flyout simulation. Ownship is assumed to fly along the predicted attack path. If the evasion algorithm determines the missile is not a threat to ownship, the DAM scheduler records this particular missile as no threat and sends the next candidate threat missile from its ordered list to the evasion algorithm.

Threat assessment must be capable of performing when incomplete or imperfect missile information is available. It can operate using sensor data reported on the missile during its trajectory or from the flyout model initiated from position and velocity · threat aircraft at launch. The source with the least uncertainty is used. When sensor data is used, flyout model is synchronized to help smooth data and extrapolate between infrequent or noisy data points The reported missile position and velocity are adjusted according to uncertainties to provide a "worst case" state of the missile. Uncertainty associated with sensor data will typically decrease with time. However, as data from the flycut model is continuously used, uncertainty will increase.

Threat assessment also attempts to establish positive identification of the missile type If identification cannot be established, available data will be used to characterize the missile as much as possible, such as radar guided or infrared guided, and long range versus short range. This information is needed by the evasion algorithm to evaluate response options.

Option Generation - With a missile determined as a potential threat, a list of candidate evasion options is generated. An embedded aircraft performance model is used to ensure that candidate maneuvers are achievable. Early and midcourse options are constructed to explore the effectiveness of different combinations of heading change and load factor. A stored knowledge data base is used to define endgame evasive maneuvers, reactions to be achieved with countermeasures, and conditions under which each option is most effective.

Early kinematic maneuvers are designed to defeat the threat missile by avoiding the intercept zone, escaping the intercept zone before the missile reaches the zone boundary, creating a missile sensor gimbal failure, or turning to generate a range rate failure. A constant altitude, maximum sustained turn by ownship is assumed. The evasion algorithm identifies options to provide the maximum time ownship can continue on its current flight path, the minimum amount of heading change required, and the time ownship must follow a radial from the launch site to escape the missile. Gimbal failures are created by taking advantage of the missile requirement to provide a lead angle for intercept to occur. Range rate failures are created by taking advantage of missile guidance system sensitivity to small closure rates.

Midcourse options combine countermeasures with maneuvers to defeat the threat missile seeker. Constant altitude turns are assumed at or below the maximum sustainable turn load factor. Countermeasures include flares and onboard and offboard jammers.

A knowledge base of candidate endgame options was identified through off-line analysis using aircraft and high fidelity missile model simulations. A barrel roll maneuver proved to be the most effective against various threat missiles because it is least sensitive to timing and produced consistent results. The effectiveness of the barrel roll is stored in tabular form.

Option Ranking - Desirability and effectiveness metrics are generated for use in ranking the candidate set of evasion options. The desirability metric is a function of the required heading change, fuel usage, time required to complete the option, and mission impact. Mission impact accounts for the inability to complete illumination for an AMRAAM already inflight. An option is considered even less desirable if it forces total abandonment of an offensive posture or mission objective. The effectiveness metric is a function of the ability to defeat the threat and the uncertainty associated with the missile track files. Options which cause hard failures of the missile system such as gimbal failures, closure rate limits, or time-of-flight limits are considered more effective than options which require precise timing and typically result in less miss distance, such as endgame maneuvers. The effectiveness metric accounts for the dependence on data accuracy. For example, the desirability of an option that is critically dependent on accurate missile velocity data is downgraded over one that requires less velocity accuracy. The preferences of the pilots are also reflected in the effectiveness metric. The products of the desirability and the effectiveness metrics are used to rank order the candidate options.

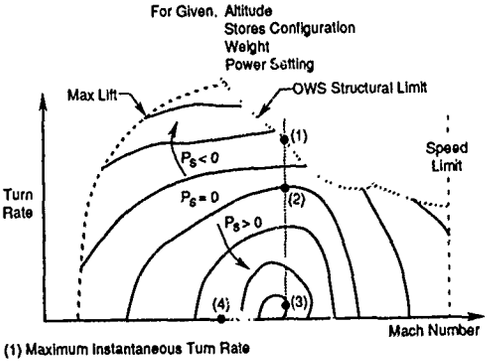
The top-ranked option is validated, before it is recommended to the pilot, using a high speed missile flyout model. The model is used to assess effects of the recommended flight path and countermeasure deployment throughout the predicted missile time-of-flight. An option is considered successful if the flyout model predicts an adequate miss distance.

AIRCRAFT PERFORMANCE MONITOR

Pilots of today's fighter aircraft do not get much assistance from onboard systems in optimizing the maneuver potential of their aircraft. A performance manual is available which a pilot reviews during ground training, but he is forced to work from memory when airborne. The ICAAS Aircraft Performance Monitor (APM) uses knowledge of ownship aircraft parameters to aid the pilot in achieving the best real-time dynamic performance of his aircraft in an effort to gain a performance edge over his adversary.

APM provides real-time corner speed and angle-of-attack (AOA) cues for cockpit display The angles-of-attack correspond to best acceleration, best sustainable turn rate, or maximum instantaneous turn rate, all at a given airspeed and altitude. The choice of the particular AOA parameters was based on discussions with pilots to determine cues that would help make optimal use of aircraft performance during an air-to-air engagement. Only one AOA cue is displayed at any given time based on pilot intent inferred from current stick commands.

The AOA cue for best acceleration was suggested due to a peculiar characteristic of some aircraft that best acceleration is achieved under a slightly loaded condition in some regions of the flight envelope. The pilots familiar with this characteristic reported using it to advantage during combat exercises. The cue for best sustained turn rate at current airspeed provides a zero specific excess power (Ps) indicator to help maintain energy relative to an opponent. The cue for maximum instantaneous turn rate was suggested to effectively fly near the structural limit without violating the structural limit boundary. Current overload warning systems advise the pilot with an aural tone when he is near the limit, but pilots still fly through and beyond the limit due to rapid onset rates. A visual cue related to AOA is believed to be more effective. Pilots typically do not wish to fly slower than corner speed in combat, but without a cue they use a constant value rule-of-thumb for all flight conditions. A cue better defines the corner speed for actual flight conditions and stores configuration. These performance parameters are pictorially represented on the maneuverability envelope, or "dog house" plot shown in Figure 13.



(1) Maximum Instantaneous Turn Rate

(2) Maximum Sustained Turn Rate ($T \cdot COS\ \alpha - D = 0$)

(3) Maximum Acceleration

(4) Corner Speed

Figure 13. Typical Maneuverability Envelope

SYSTEM IMPLEMENTATION

The Department of Defense standard Ada software language is being used for ICAAS implementation. Most of the software code has been developed, including 100,000 lines or 850,000 32-bit words of executable code plus 225,000 words of data base (Reference 2). Rapid update rates which are necessary to support real-time mission performance led to a computer throughput requirement of approximately 15 million instructions per second. No available flightworthy computer with this capability could be found, so a new computer was developed based on the MIPS Corporation R-3000 32-bit reduced instruction set computer (RISC) processor. This is one of two standard processors selected by the Joint Integrated Avionics Working Group. Each processor is conservatively rated at five million instructions per second. Three processors are included in a single computer chassis, resulting in a throughput capacity of 15 million instructions per second (Reference 3). A 100 percent reserve was achieved by using two of these computers to host the ICAAS software. One will host the Tactics, Attack Guidance, DAM, and APM functions and is designated the ICAAS

Integration Computer (IIC). The second computer, the ICAAS Support Computer (ISC), will host the attack management function. These computers are flightworthy and will be used in simulation evaluations as well as flight test.

Pilot/vehicle interface (PVI) is a major ICAAS system integration issue. No matter how much information can be developed within the system, effective combat performance can be achieved only if the pilot can easily understand and apply the information. Too much data and information can easily overwhelm the pilot. Efficient cockpit data management techniques are needed, including a proper balance between automation and manual task allocation.

An advanced cockpit instrument panel has been designed for ICAAS as shown in Figure 14. Three multifunction color displays will be used to present tactical situation data, weapon delivery information, threat warnings, recommended tactics, and flight trajectories. A large center display provides 9 1/2 inches by 9 1/2 inches of viewing area as the primary situation awareness display. The other two displays are 6 inches by 6 inches. A helmet mounted display (HMD) is provided for sensor cueing and threat warning. The HMD will also be used to cue the pilot on where to look for targets as he transitions from BVR to WVR, thus providing maximum visual acquisition range. A data transfer module will be used for preflight data entry to relieve the pilot from a lengthy initialization process. Inflight data entry will be managed through the Up Front Control (UFC) which is located just below the HUD Weapon selection, target designation, and display management functions are provided through standard hands-on throttle and stick switches, or by switches located around the perimeter of the three multifunction displays. An additional method is provided by a touch sensitive overlay for the large central display Pilot working group members have been involved in a series of static mockup, rapid prototyping, and part task simulation experiments as the specific cockpit layout and display formats have evolved It is essential to achieve a pilot-friendly implementation of the ICAAS technology

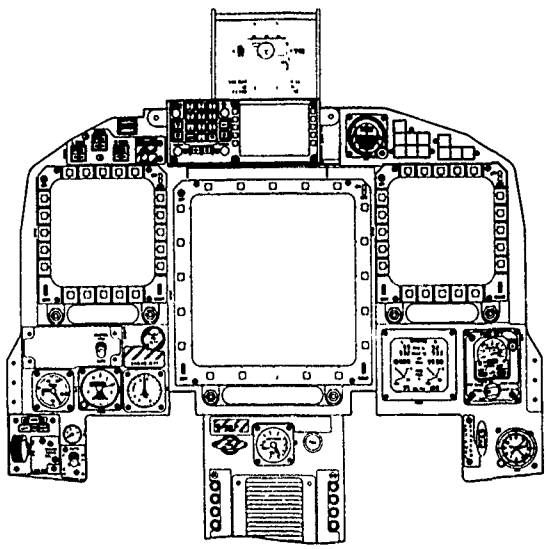


Figure 14. Cockpit Instrument Panel

TEST AND EVALUATION

Formal high-fidelity piloted simulations will be conducted starting in late 1990 to measure the combat performance of four internetted ICAAS aircraft engaging up to sixteen enemy aircraft An orderly buildup in complexity is expected, starting with two blue versus two red. Blue aircraft use cockpit domes which include fully configured cockpits with ICAAS controls, displays, information processing, and system functions. Red forces consist of up to four manned combat stations plus digital aircraft models Air Force tactical fighter pilots will serve as test subjects, with separate blue and red teams A mix of current and advanced threat aircraft are used to represent a projected 1995 air combat environment

Flight testing will demonstrate that ICAAS functions can be implemented on actual fighter aircraft using real sensors in real flight conditions. An F-15B testbed aircraft is being modified to incorporate the ICAAS advanced cockpit configuration and system functions, including IIC and ISC computers After initial flight testing in 1991, a second F-15 testbed will be modified in 1992. An intraflight data link will be installed on both aircraft to enable demonstration of the cooperative functions of ICAAS. Engagements will be initiated at BVR and proceed through simulated weapon employments, including transition to close-in combat

SUMMARY

For three years, the ICAAS program has been developing innovative design concepts to assist tactical fighter pilots in defeating larger forces of enemy aircraft. Avionics, guidance, and control functions are being improved and extensively integrated to enhance pilot awareness of the combat situation, provide decision aiding in evaluating offensive/defensive options, support the pilot with precise execution of his engagement decisions, and coordinate the actions of individual flight members into an effective battle team. Extensive use of knowledge based systems technology and emphasis on expert systems engineering techniques provide a basis for significant improvements in mission effectiveness compared to current operational systems. Efficient information management combined with advanced cockpit display concepts allow ICAAS to effectively support the pilot in achieving mission objectives.

A specific ICAAS system architecture has been defined to achieve real-time performance when embedded in a fighter aircraft. New flightworthy computers have been developed based on advanced 32-bit RISC processors The DoD high order Ada software language is being used to implement all ICAAS functions Integration of hardware and software is nearly complete in preparation for formal testing.

Extensive piloted simulations and flight test experiments will be performed to determine the mission effectiveness of ICAAS technology in realistic air combat engagement conditions. Test and operational fighter pilots will be used as test subjects to confirm that the system provides the intended level of support and meets performance goals. Final results are expected to be available in late 1992. Sufficient confidence will be established to facilitate technology transition to future fighters or upgrades of existing fighters

REFERENCES

1. Manske, Robert A., "Attack Management Decision Aids for Air-to-Air Combat", presented at the NATO AGARD Fall Avionics Panel Symposium, Copenhagen, Denmark, 3-6 Oct 1988.

2 Wagner, Steven M., and Rothstein, Steven W., "Integrated Control and Avionics for Air Superiority: Computational Aspects of Real-Time Flight Management", AIAA Guidance, Navigation, and Control Conference Proceedings, Part I, Boston MA, 14-16 Aug 1989.

3 Feinreich, Ben, and Wagner, Steven M., "Development of an Advanced 32-Bit Airborne Computer", prepared for NAECON, May 1990.

4. Kocher, James A., "Integrated Control and Avionics for Air Superiority", AGARD Conference Proceedings No. 425, pp 31-1 to 31-7, Oct 1987.

5. Rothstein, Steven W., et al, "Air Combat Flight Management· The System Integration Challenge", prepared for NAECON, May 1989

A KNOWLEDGE-BASED SYSTEM DESIGN/INFORMATION TOOL FOR AIRCRAFT FLIGHT CONTROL SYSTEMS

Dale A. Mackall
Aerospace Engineer
NASA Ames Research Center
Dryden Flight Research Facility
P.O. Box 273
Edwards, California 93523-0273
U.S.A.

James G. Allen
Staff Engineer
Charles Stark Draper Laboratories
555 Technology Square
Cambridge, Massachusetts 02139
U.S.A.

ABSTRACT

Research aircraft have become increasingly dependent on advanced electronic control systems to accomplish program goals. These aircraft are integrating multiple disciplines to improve performance and satisfy research objectives. This integration is being accomplished through electronic control systems. Because of the number of systems involved and the variety of engineering disciplines, systems design methods and information management have become essential to program success. The primary objective of the system design/information tool for aircraft flight control system is to help transfer flight control system design knowledge to the flight test community. By providing all of the design information and covering multiple disciplines in a structured, graphical manner, flight control systems can more easily be understood by the test engineers. This will provide the engineers with the information needed to thoroughly ground test the system and thereby reduce the likelihood of serious design errors surfacing in flight. The secondary objective is to apply structured design techniques to all of the design domains. By using the techniques in the top level system design down through the detailed hardware and software designs, it is hoped that fewer design anomalies will result. This paper will first review the flight test experiences of three highly complex, integrated aircraft programs: the X-29 forward-swept wing, the advanced fighter technology integration (AFTI) F-16, and the highly maneuverable aircraft technology (HiMAT) program. Significant operating anomalies, and the design errors which cause them, will be examined to help identify what functions a system design/information tool should provide to assist designers in avoiding errors.

1. NOMENCLATURE

AFTI	advanced fighter technology integration
AI	artificial intelligence
DFD	data flow diagram
FCC	flight control computer
FCS	flight control system
FMEA	failure modes and effects analysis
HARV	high-angle-of-attack research vehicle
HiMAT	highly maneuverable aircraft technology
H/W	hardware
ILS	instrument landing system
KB	knowledge base
KBS	knowledge-based system
KCS	knowledge capture system
KEE™	Knowlege Engineering Environment (trade-mark of Intellicorp, Mountain View, CA)
KR	knowledge representation
LE	leading edge
NWS	nosewheel steering
PLC	power lever control
S/W	software
TE	trailing edge
T/O	takeoff
V and V	verification and validation

2. BACKGROUND

System engineering has been recognized as an essential element in the development of complex systems. The top-down structured approaches to system engineering have been slow to catch on because of a lack of computerized tools. However, recent advances in personal computers and new software (S/W) tools have reestablished the use of these structured system engineering methods (ref. 1).

The system design aspects of the system design/ information tool expand on the current systems engineering methods by 1) automatically creating a knowledge base (KB) of the processes, data flows, and externals; and 2) including functions to verify consistency in design requirements unique to flight crucial control systems.

The information aspects of this tool address the need to provide design and implementation information throughout a flight control system's (FCS's) life cycle, and, specifically, to the test engineers. The verification and validation (V and V) effort for the digital FCS is of particular concern to the test engineer. Complete V and V is required to assure flight safety and requires the design information to establish, run, and analyze the V and V tests. Problems associated with V and V have caused major digital FCS developments to slip by as much as 18 months (ref. 2). The system design/information tool needs to include the flight control design knowledge and its hardware (H/W) and S/W implementation.

Figure 1 shows a typical life cycle for an FCS and how the system design/information tool would support all phases. Shown is a typical life cycle for an FCS and how the life cycle phases relate to the system/information tool's capabilities. Some of the current tools which would share information with the system design/information tool are shown in the lower half of the figure.

The following review of research aircraft and the unique design errors that were found shows how system complexity can hide design errors from even the most experienced engineers. These errors reflect, in part, the difficulty of adequately communicating the system design details to the test engineers in the multuple disciplines. These disciplines include flight control law development, H/W design and test; S/W specification, coding and test; system integration and test; and flight test operations.

2.1 X-29 Description and Airdata Single-Point Failure

The X-29A technology demonstrator aircraft is an experimental vehicle which integrates a number of advanced technologies. These technologies include a forward-swept wing, tailored composite wing structure, and full authority digital flight control. The aircraft is also highly unstable and is dependent on the triplex digital FCS for stability and handling qualities (ref. 3).

The FCS feedback gains are scheduled using airdata. Airdata errors can cause incorrect flight control gains and loss of the aircraft. To avoid incorrect gains, the X-29A has three sources of airdata. Redundancy management S/W takes the three airdata values, detects any failures, and selects a value to be used in the control law calculations.

After flying over 200 flights, a serious design error in the redundancy management logic was found during verification of a new release of flight S/W being tested in ground-based simulation. The error was attributed to the multidisciplinary nature of the system and had been in the flight S/W since the 38th flight. A lack of detailed understanding about the interactions between the airdata system, redundancy management S/W, and the flight control laws allowed for the design error to occur and is discussed below.

The fault detection level in the redundancy management S/W was set at a large value because of errors, such as position errors, possible between the probes at certain flight conditions (fig. 2). In the case of a probe failure, airdata errors as large as the fault detection level were allowed to pass through to the control laws. At the lower and slower end of the flight envelope, a fail-to-zero of the nose probe would not be detected. Simulations have shown that this single-point failure would change the gains to the point that the aircraft would become unstable and depart. For over 162 flights the aircraft was at risk of being lost because of a single airdata failure. The system requirement was that the aircraft be operational after two airdata failures. Until a subsequent software release corrected the problem, the aircraft was grounded.

2.2 AFTI F-16 Description and Flight 44 Anomaly

The advanced fighter technology integration F-16 program investigated the integration of emerging technologies into an advanced fighter aircraft. The AFTI's three major technologies investigated were (1) flight-crucial digital control, (2) decoupled aircraft flight control, and (3) integration of avionics flight control and pilots display (ref. 2).

The AFTI F-16 flight control system was a triplex, asynchronous digital system. The asynchronous architecture meant that input signals from sensors and controllers were read at different times into the three computers using a high-speed serial, digital data link (fig. 3). Concerns for S/W reliability were addressed with the inclusion of a triplex, analog-independent backup unit.

The following summarizes an in-flight anomaly which occurred on flight 44 (ref. 2). This anomaly was the result of the interaction of many design characteristics and a unique flight condition. The characteristics included asynchronous computer operation, forward integrators in the control laws, and output redundancy management S/W. These characteristics coupled with a unique flight condition and resulted in the divergence of the three computers' output commands to the control surfaces. The redundancy management S/W in each of the channels declared the other two channels as failed. The pilots indication of this apparent simultaneous failure of all three computers was a dual fail flight control light in the cockpit. The end result of this in-flight anomaly was that the aircraft safely landed on what was effectively a single-string flight control system, even though no actual H/W failure had occurred.

Like the X-29 example, the AFTI F-16 had a serious design error resulting from the lack of a detailed understanding of the interactions between the many different disciplinary areas. In this case the design error was not recognized until after an in-flight anomaly was experienced.

2.3 HiMAT Design and Gear Deployment Anomaly

The HiMAT demonstrator was a remotely piloted research vehicle which incorporated such advances as composite structures, aeroelastic tailoring, reduced static stability, and digital flight control (ref. 4). The aircraft was remotely piloted because the technologies represented too high a risk for a manned vehicle.

The HiMAT was flown remotely with the pilot in a ground-based cockpit and the control laws calculated in ground-based computers. Surface commands were telemetered to the aircraft as were aircraft sensor data which were telemetered to the ground (fig. 4). The onboard digital flight control computers were dual redundant and processed uplink and downlink data. In the case of a complete loss of the dual uplink commands, the onboard system acted as a backup FCS capable of orbiting the aircraft until control was reestablished.

An anomaly occurred during the flight test program which resulted in the aircraft landing with its landing skids retracted. However, the pilot performed an excellent landing and the aircraft was not seriously damaged. The anomaly was induced by a single failure in the redundant uplink H/W. The onboard redundancy management S/W identified the failure and allowed for continued control of the aircraft, except for the deployment of the landing skids.

The anomaly was caused by a timing change made in the ground-based system and the onboard S/W for uplinking the gear deployment command. This change coupled with the onboard failure of one uplink receiver to cause the anomaly. The timing change was thoroughly tested with the onboard flight S/W for unfailed conditions. However, the flight S/W operated differently when an uplink failure was present. This critical information about the S/W was not readily available to the flight test team.

3. REQUIREMENTS FOR A SYSTEM DESIGN/INFORMATION TOOL

These brief examples demonstrate that system complexity is overwhelming the individual's ability to understand the entire system and the interactions that can take place between the different functional areas. The X-29 example illustrates how important the airdata system and the redundancy management S/W design information to the flight control designers and test engineers. Using the experience gained from the above aeronautics projects, we have formulated the requirements for a system design/information tool. The requirements include:

1. A system design capability to ease the capture of design information. The system design capability will provide a graphical, structured method for designing complex systems. It will help the designer avoid errors and allow the capture of the design information as it is created. Later in the aircraft development this information, in the form of an intelligent documentation system, will provide information to the test engineers.

2. Online documentation of all the information describing an FCS and the relationships between different disciplinary information. This includes H/W, S/W, redundancy management, and flight control law disciplines. The test engineer can then easily and graphically see the design information needed to qualify the system, thus avoiding the in-flight consequences of design errors.

3. Expert system functions to help analyze the relationships between the disciplines and uncover where unwanted interactions can occur. These functions can be used by designers, as well as test engineers, to assess the system's operations and avoid serious design errors.

4. Ability to perform failure modes and effects analysis (FMEA) on the many design iterations. Currently, FMEA is only performed on the H/W, not on the system as a whole. Because of the time required to perform an FMEA, the FMEA is usually performed once and is done with an early design iteration. The inability to analyze the final design raises questions of the FMEA's value. Automated FMEA using the current online design is one example of a capability that would assist designers and test engineers in finding serious design errors in a timely manner.

5. Links from the system requirements to the S/W and H/W designs. The links will allow the system requirements to be verified against the proposed implementation. Verification could then be done in an automated fashion, prior to committing to the build phase. This rapid prototyping concept would increase the chance of finding serious design errors prior to flight test.

Currently, some system design tools have become commercially available, but they do not address the needs of flight-crucial systems and only create conventional databases called data dictionaries. The actual H/W and S/W implementation information is not an integral part of these tools.

4. DESCRIPTION OF THE KNOWLEDGE-BASED SYSTEM DESIGN/INFORMATION TOOL

The following section will review the work accomplished to date and show how it applies to the larger problem outlined above. The methods for capturing system design knowledge, examples of what can be done using this knowledge, and an overview of the structure of the knowledge-based system (KBS) will be discussed. In related work, a good approach to design knowledge capture for the space station can be found in Wechsler (ref. 5).

4.1 Focus

The effort is focused on the development of a generic knowledge capture system (KCS) for digital FCSs, which utilizes mature AI technology. The KCS is being used to capture design knowledge for the NASA high-angle-of-attack research vehicle (HARV), a modified F/A-18A with a thrust-vectoring capability. The primary efforts to date have been focused on the development of an intelligent documentation system for the system and H/W design realms. Examples of expert analyses that can be accomplished once the design knowledge is captured are described in the sections on the spin recovery system and nosewheel steering behavioral model.

4.2 The Knowledge Capture System

A major portion of the effort for the KCS has been devoted to the development of a knowledge representation (KR) which is tailored to the specific needs of the FCS problem domain. Four domains of knowledge have been identified within the FCS problem domain: system design, H/W design, S/W design, and utilities. These four domains provide the flight test engineer with the diverse kinds of information needed and the relationships which exist between them. Each domain possesses its own unique KR.

4.3 System Design Realm

The structured analysis methodology is used to describe the system design (ref. 6). This methodology is based on a top-down hierarchical decomposition of system requirements using an extremely graphical user interface. The decomposition continues until the requirements are given with an adequate degree of detail. This design methodology creates a cleaner, more understandable design.

The tool creates linked hierarchical trees of data flows, processes, and externals. Each node in the process tree represents a process and is provided with a process description and other unique attributes which are stored in slots. To support flight control system design, the tool stores and tracks requirements for failure probability and mission criticality. In addition, external agencies and data flow objects are identified in the KB. All of this information is depicted graphically in data flow diagrams (DFDs). Figure 5 depicts a level 0 DFD.

The concepts of a process, an external, and a data flow, as defined by structured analysis, are identified here as graphical objects and individually represented as frames. The properties of the process, external, data store, and data flow objects are stored in the slots of the individual frames associated with each of these objects. The name of the process, failure probability, and data flow inputs are all examples of slots. The nature of the slot values can draw from the full spectrum of the paradigms supported by the Knowledge Engineering Environment (KEE™). Namely, they may be simple values, pointers to other frames, inherited values, active values, rules, and so forth. This KR will allow the users to perform various expert analyses of the system design. It is intended that the pointers, which are stored as slot values, will provide access to the related H/W, S/W, and utilities implementation knowledge stored elsewhere.

A hierarchical representation scheme is used for each of the three types of objects (processes, externals, and data flows). Each of these three hierarchies forms an individual, linked KB. In each case, the hierarchy is used to allow properties to be inherited and to identify the natural linkage between individual objects. These individual KBs are linked with pointers.

In a typical application of the structured analysis methodology, the data flow diagrams are viewed as an end object. In this KBS, the data flow diagrams are primarily viewed as a graphical front end for the KCS. Every data flow diagram image is mouse sensitive and possesses its own menu for entering and accessing knowledge. Now the test engineer can graphically see the relationships between systems, rather than trying to infer them from stacks of written text.

4.4 Hardware Design Realm

The knowledge representation for the H/W design is based upon the hierarchical block diagrams typically used in this problem domain. The nature of the representation is similar, although not identical, to the structured analysis methodology. The H/W objects are represented graphically as blocks, and these objects are decomposed in a hierarchical fashion until they have been described to an adequate degree of detail.

The connectivity between these H/W objects is indicated graphically with lines. These lines may represent various H/W connection abstractions such as data buses, a flow of information, or a form of control. In any case, this connectivity can be represented in the form of objects of a specific type and may possess a hierarchical characteristic.

The concepts of H/W and their connectivity are identified in the KBS as being objects and are individually represented as frames. The properties of these objects are stored in the slots of the individual frames. The nature of the slot values and the hierarchical relationship of the frame representation is the same as that provided for the system design. The H/W block diagrams serve as a graphical front end for the H/W design KB. Figure 6 depicts a top level H/W block diagram.

4.5 Software Design Realm

The structured analysis methodology is also used to describe the S/W design. The long-range plans include placing the KCS on a network with a workstation that has the flight code. This would give the KCS access to the flight code so that it would be possible to pull up listings of the flight code relevant to the objects defined in the S/W design data flow diagrams.

4.6 Utilities Design Realm

The utilities consist of the electrical power, hydraulic power, and environmental control systems which form an infrastructure for the FCS, and any embedded avionics system for that matter. The KR for this realm will utilize the structured analysis methodology to encode the design knowledge. It will also utilize the block diagram representation described previously for the H/W design realm. This representation will be used to encode the implementation knowledge.

4.7 Authoring and Browsing Mechanisms

The authoring mechanisms allow the user to create, delete, connect, and locate the user interface graphical images with mouse and menu commands. These ordinary capabilities provide typical graphical interface features. More importantly, the authoring mechanisms allow the user to *properly* create, delete, rename, and connect objects in the semantic network. The objects and their linkage within the semantic network are highly structured and canonical. A failure to conform to this formal structure would destroy the inheritance, browsing, and reasoning functionality. The authoring mechanisms also serve to enforce the methodologies which have been deemed appropriate for the individual realms within the KB.

The authoring mechanisms provide mouse and menu commands for inserting slot values for the properties of the objects in the KB. These mechanisms include an editor window for entering text descriptions and popup selection menus for properties whose slot values are restricted to a specific set of legal values. In some cases, the entry of slot values is monitored by demons. These demons actively monitor the knowledge entry and warn the user if the new value lies outside an envelope which is known to be valid. A demon is used to verify that failure probability requirements are kept as the design is decomposed.

The browsing mechanisms allow the user to display the text description, hierarchical relationships, and properties for the objects which have been entered into the semantic network. This information is accessed through the graphical, menu-driven, and mouse-sensitive user interface. This interface supports a random access to the KB. The knowledge representation supports a browsing strategy similar to the way we, as humans, pursue problem solutions. In this case, the network structure tends to guide the user in exploring the KB.

4.8 A Decomposition of the Spin Recovery System

The HARV flight tests will include research flight work with an angle of attack greater than 55°. For safety of flight in this regime, a spin chute recovery system has been added to the F/A-18A research aircraft. The following describes a decomposition which has been performed on the spin chute recovery system using the KCS.

Figures 7 and 8 show the hierarchical decomposition of the primary system that will deploy a spin chute for a spin recovery. These figures depict two of the many H/W diagrams involved in the decomposition of the spin chute recovery system. Figure 7 includes a partial display of the hierarchy. Figure 8 indicates the use of dual abstractions.

The box/line graphics provide an abstraction which follows directly from the top level H/W diagram graphical user interface. These box/line abstractions are linked directly to the objects in the hierarchical H/W design

KB. The bit map graphical depiction of the circuit diagram has been added to clarify the user interface at the component level.

The two abstractions are tied together with bit-mapped hot spots. Authoring mechanisms allow the user to link the box/line abstractions (and correspondingly their H/W and signal flow objects) to as many hot spots on the bit map abstraction as may be desired. Browsing mechanisms allow the user to mouse on a hot spot or a box/line abstraction. Highlighting indicates the correspondence between a selected hot spot and its box/line abstractions. Similarly, highlighting indicates the correspondence between a selected H/W object (or signal flow object) and the relevant hot spots.

Further work in this area will be concentrating on the ability to take detailed H/W diagrams and perform failure modes and effects analysis of the systems they represent.

4.9 A Behavioral Model for the Nosewheel Steering System

The KBS includes dynamic behavioral models to aid the test engineer and also provides for rapid prototyping of the system. These models will be used to describe the system operation as a function of its operating modes. The models will permit the user to interactively enter mode commands and explore their impact on FCS operation.

The behavioral model described here permits the user to issue cockpit mode commands to a nosewheel steering (NWS) model which indicates the response and changes in the aircraft state. The model is based upon a rule set and a forward chaining paradigm. A dynamic display of the rules and their execution is available to dynamically document the system operation.

The NWS is a secondary control system within the FCS which is only operable on the ground. It provides nosewheel angular deflection proportional to pedal force when engaged. There are three modes of operation: off, low gain, and high gain. The desired mode is selected by the pilot, with switches located on the control stick grip, and is a function of several inputs, such as wing fold and weight on wheels. The NWS switch is used for NWS engagement and mode control, while the autopilot switch is used for NWS disengagement on the ground.

A dynamic interactive display (fig. 9) is provided to control and display the control stick switch commands, the NWS system status, the NWS system block diagram, and the relevant F/A-18A aircraft status. The display window of the control stick switches includes a control stick and KEE™ active images for the NWS and autopilot switches. This window, which is mouse sensitive, will accept switch commands in an identical fashion to those issued by the pilot by way of the actual aircraft control stick. The KEE™ active images, which depict the NWS switch and the autopilot switch, are mouse sensitive. It is possible to issue a momentarily depressed, held depressed, or released command with these images.

The display of the aircraft status is also mouse sensitive. It is possible to explore the NWS logical operation as a function of aircraft power, touchdown status, wing fold status, and launch bar status by mousing the appropriate active image. As these parameters are changed, the appropriate operational mode is dynamically updated and displayed in the F/A-18A operation mode window. The NWS-related aircraft operational modes are: power off, wings folded, taxi, takeoff (T/O), launch, in flight, and landing.

A rule-based system implements the NWS mode logic. These rules are activated by the control stick switch commands and by changes in the aircraft status variables. The NWS system response is displayed by highlighting the appropriate mode in the NWS control mode status window. The NWS system response is also displayed by highlighting the control path in the NWS system block diagram window. It is possible to trace the rule execution in a KEE™ dynamic forward chaining execution window. The rule displays are mouse sensitive and permit the user to display a selected rule.

4.10 Knowledge-Based System Structure

The KBS is coded in Common Lisp, utilizes an expert system shell called the knowledge engineering environment, and is currently under development on a Symbolics machine (fig. 10). This rapid prototyping environment has been utilized for the development of a KCS, which is tailored to the needs of the FCS problem domain. The KCS can be ported to any platform which is supported by KEE™. These platforms currently include the Symbolics and other major computing environments.

The KCS includes authoring mechanisms which enable the user to build a semantic network uniquely appropriate to a particular FCS. The KCS also includes browsing mechanisms which provide access to the semantic network knowledge. Rule-based models perform their reasoning on the objects defined in the semantic network.

The semantic network is composed of four realms of knowledge: the FCS system design realm, the FCS H/W design realm, the FCS S/W design realm, and the FCS utilities design realm (fig. 11). Each of these realms is implemented with linked hierarchical networks of objects. The KBS semantic network is formed by linking the hi-

erarchical networks of the four realms. The objects are individually represented with a frame-based representation. Authoring mechanisms enable the user to define a semantic network of FCS objects and their properties.

The semantic network of FCS objects are defined in an environment which includes an inference engine. Reasoning functions are under development which will enable the user to view and analyze the objects defined. Three kinds of models are to be developed:

1. behavioral models
2. failure mode and effects models
3. fault tree analysis models

The behavioral models are to provide a dynamic presentation of how designated objects behave as a function of user commands, FCS state variables, and FCS modes. The failure mode models are to indicate the loss of functionality associated with component failures. The fault tree models are to provide a diagnostic capability for the loss of FCS functionality. This diagnostic capability will allow the user to identify the possible causes and to help isolate the actual cause of the loss of FCS functionality.

5. CONCLUDING REMARKS

This project has proven to be an ambitious one. The roughly three man-years of effort have yielded a prototype which promises to fulfill the objectives, stated earlier, for a useful flight-crucial segment of the high-angle-of-attack research vehicle flight control system.

So far, the promises of artificial intelligence have been fulfilled. It has been possible to develop a knowledge capture system that captures the flight control system knowledge in a form which is tailored to the problem domain and is accessible to the user in a friendly fashion. Furthermore, the modeling capability has proven the value of defining the flight control system objects in an environment with an inference engine.

The system and hardware design realms now have a working functionality. In the remaining one man-year of effort, it is projected that a working prototype, capturing knowledge in all four of the realms, will be implemented. In terms of computer resource requirements, the response time is generally adequate, and less than 10 megabytes on the hard disk have been required to date for the design knowledge.

Looking to the future, it is projected that this prototype provides an infrastructure upon which a full-scale, fully operational knowledge capture system will be built that includes design aid capabilities. Longer range visions include such growth possibilities as postflight fault diagnosis, real-time ground support of flight tests, and real-time monitoring in the cockpit. The complexity of today's advanced aircraft demand that tools such as this be developed and utilized throughout the life cycle to assure safe and efficient flight operations.

6. REFERENCES

1. Neubauer, Chris, Radloff, B., and Rhodes, D., *Optimizing Systems Engineering Teams, Methods, and Tools*, AIAA Paper 88-3870, October 1988.
2. Mackall, Dale A., *Development and Flight Test Experiences With a Flight-Crucial Digital Control System*, NASA TP-2857, 1988.
3. Chin, J., Chacon, V., and Gera, J., *X-29A Flight Control System Performance During Flight Test*, AIAA Paper 87-2878, September 1987.
4. Evans, Martha B. and Schilling, L.J., *The Role of Simulation in the Development and Flight Test of the HiMAT Vehicle*, NASA TM-84912, April 1984.
5. Wechsler, D.B. and Crouse, K.R., *An Approach to Design Knowledge Capture for the Space Station*, NASA TM-89272, 1987.
6. DeMarco, T., *Structured Analysis and System Specification*, Prentice-Hall, 1973.

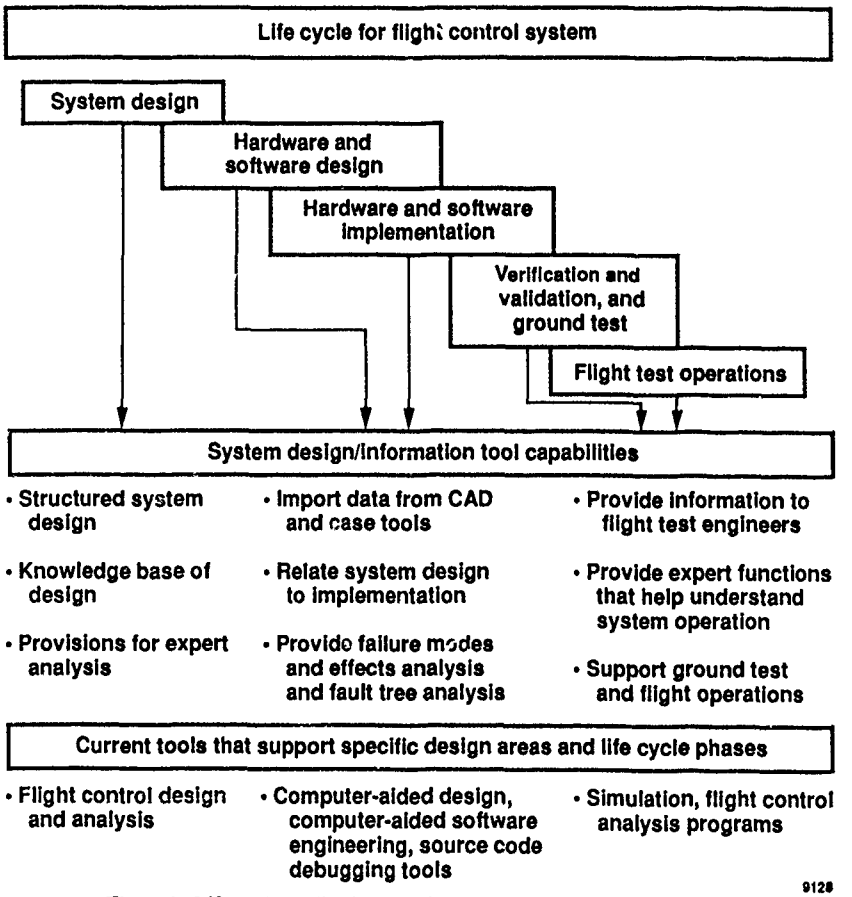
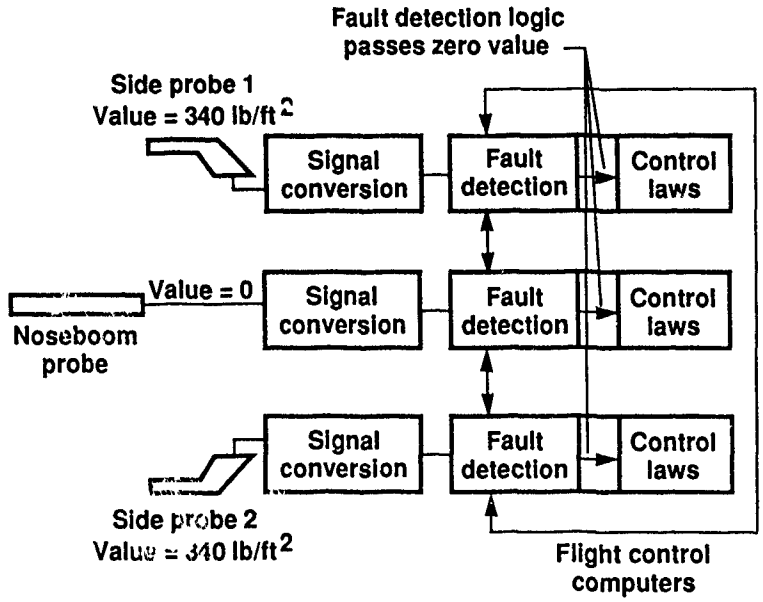


Figure 1. Life cycle applications for the system design/information tool.

Hardware Diagram and Failure Condition



Fault Detection Logic

If noseboom value - side probe 1 < fault detection level and noseboom value - side probe 2 < fault detection level, then use noseboom value else declare noseboom failed and use average of side probes

Fault detection level = 354 lb/ft^2, 5 in. of mercury

9129

Figure 2. Summary of X-29 failure condition and fault detection logic.

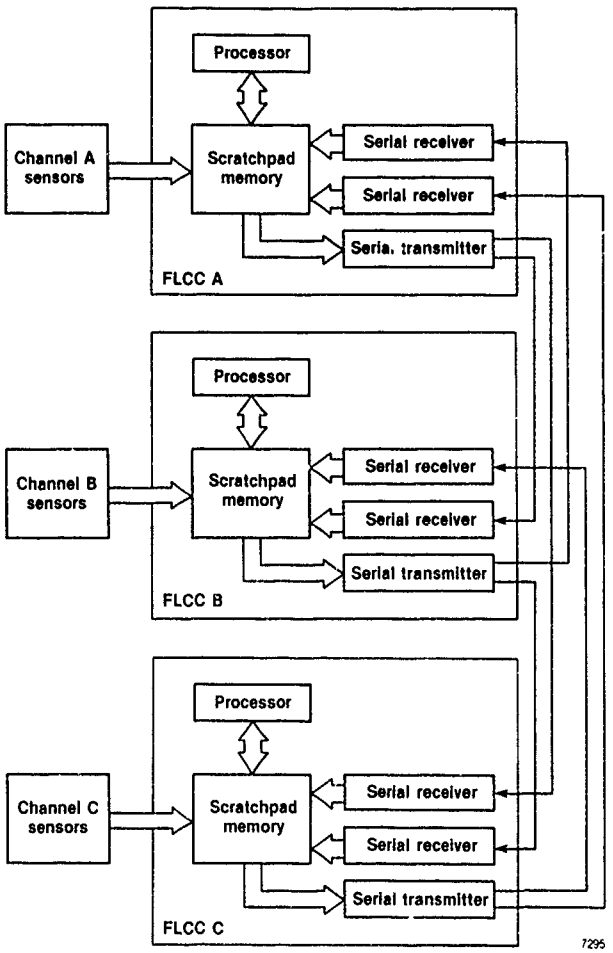


Figure 3. AFTI F-16 cross-channel monitoring uses inform. tion sent on digital links.

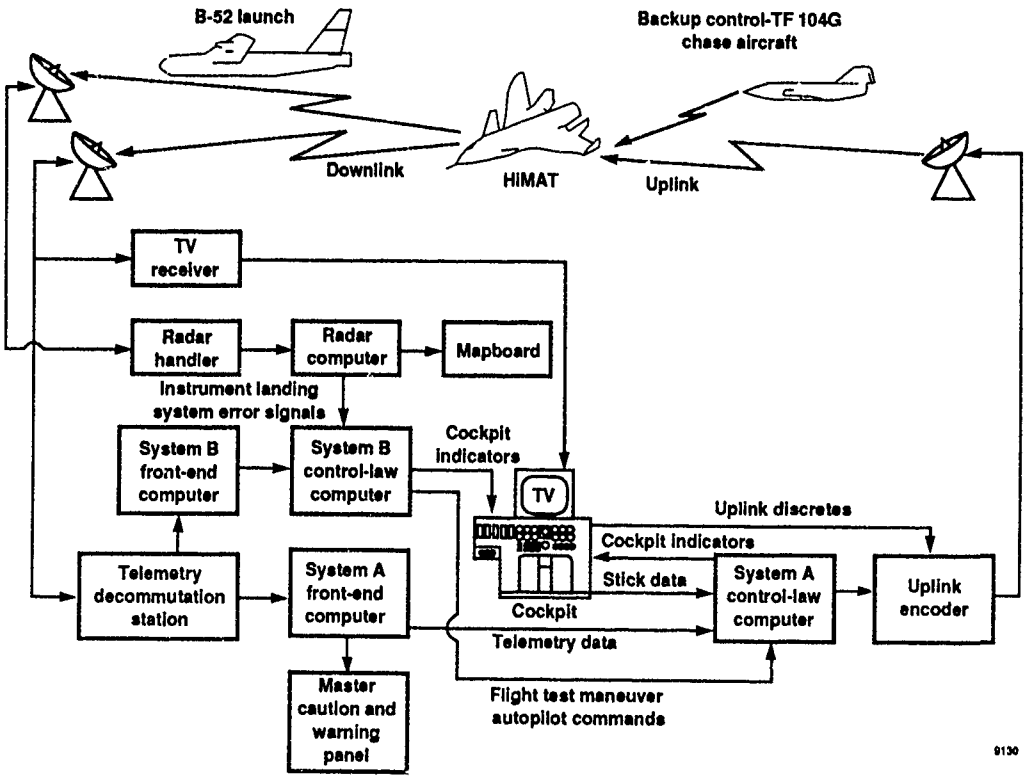


Figure 4. HiMAT control system.

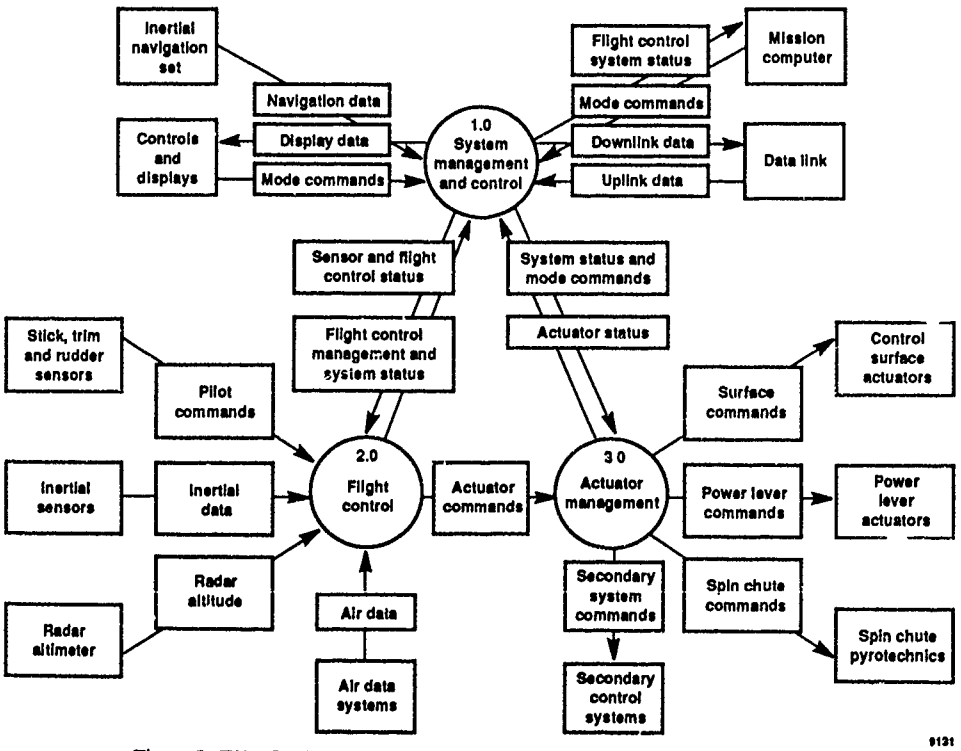


Figure 5. F/A-18A flight control system—top level data flow diagram (level 0 DFD).

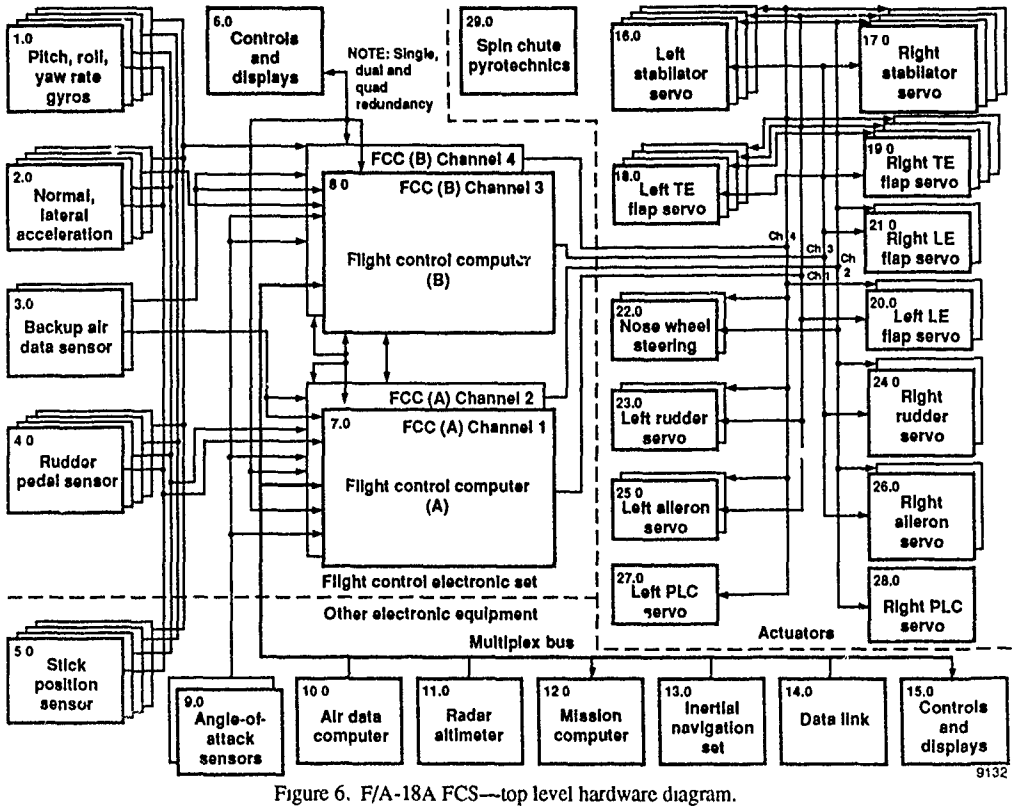


Figure 6. F/A-18A FCS—top level hardware diagram.

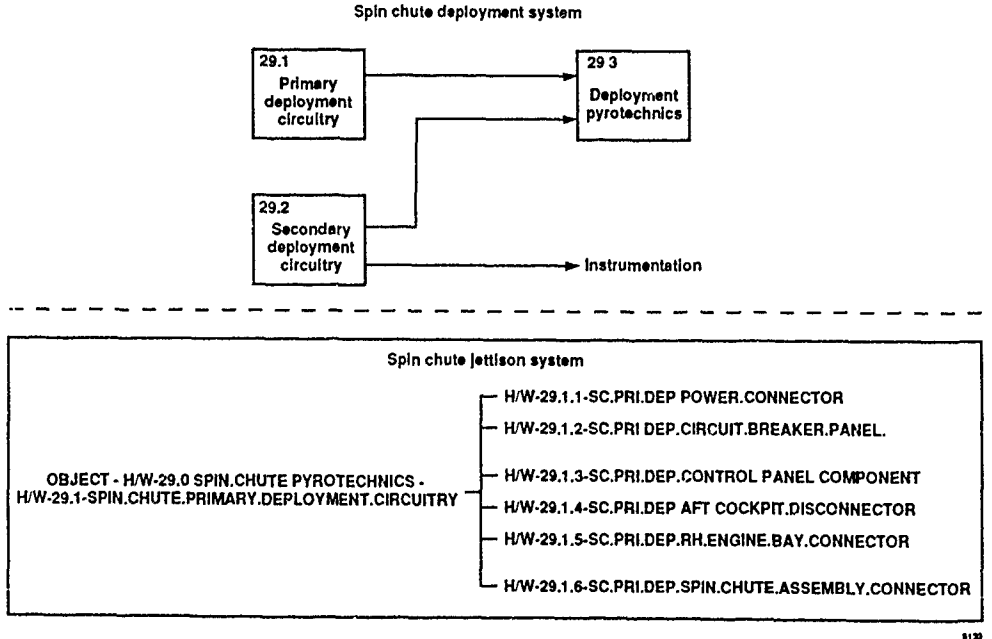


Figure 7. F/A-18A flight control system—spin chute deployment system hardware diagram (level 1 HWD).

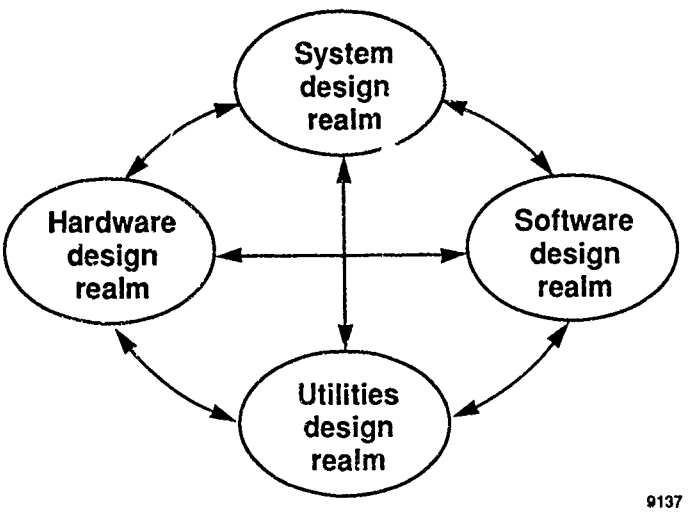


Figure 10. The layered knowledge-based system architecture.

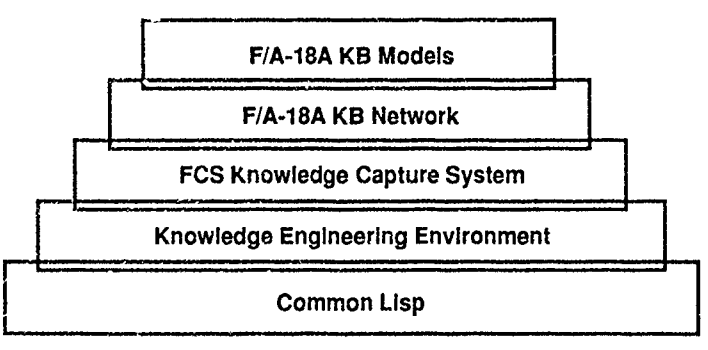


Figure 11. The knowledge realms and their linkage.

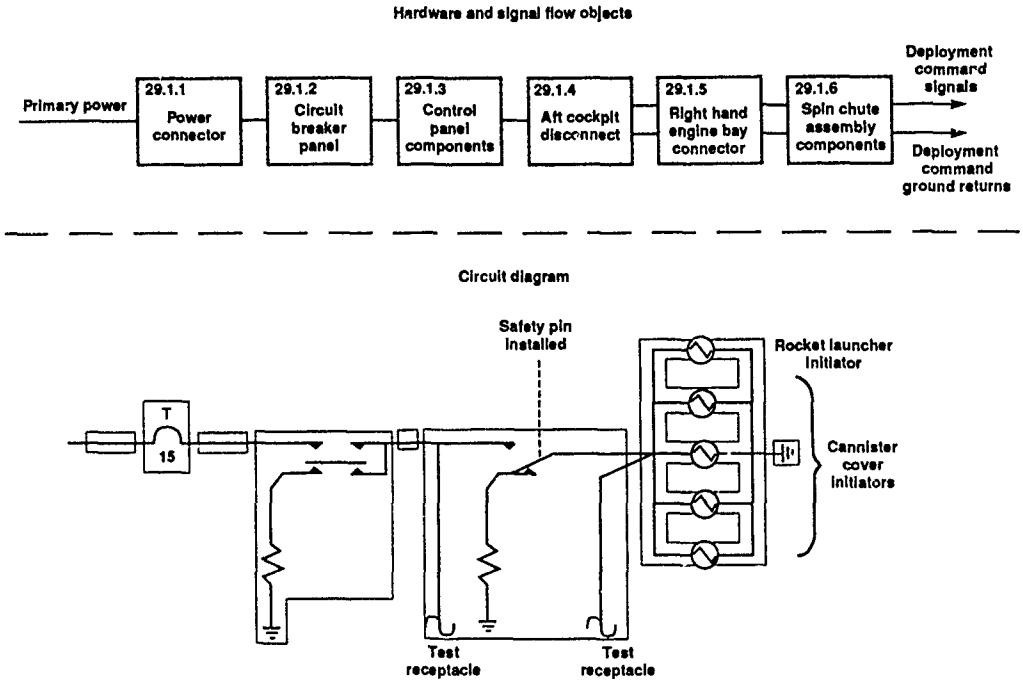


Figure 8. F/A-18A flight control system—primary spin chute deployment circuit hardware diagram (level 2 HWD).

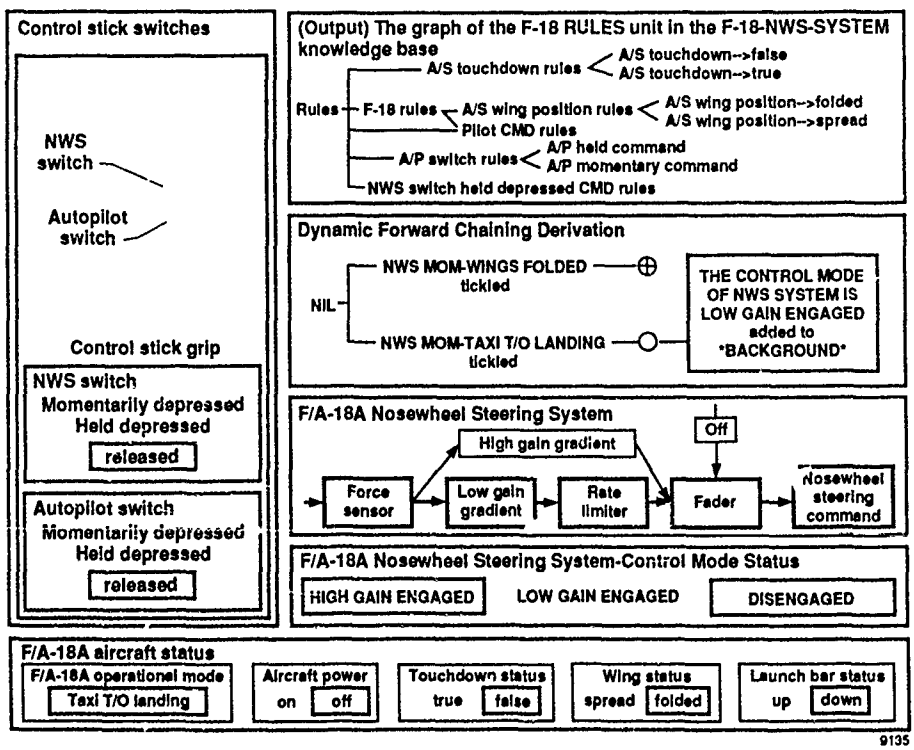


Figure 9. F/A-18A FCS—nosewheel steering behavioral model display.

A STUDY OF AN INTEGRATED IMAGE AND INERTIAL SENSOR SYSTEM

by

R. Koch, R Bader and W. Hinding

Bodenseewerk Geraetetechnik GmbH

Intelligent Systems Division

Nussdorfer Str.- D - 7770 Überlingen

FRG

SUMMARY

The target approach over large distances of a cruise missile type vehicle, which is equipped with an imaging infrared sensor aided inertial navigation system, is examined in simulation. The study is based on the idea of using the same image sensor for navigation update and target recognition with subsequent tracking. Knowledge based methods are found to play a key role in solving the difficult image interpretation task for real world scenery. The final extraction of navigation data from the processed and interpreted IR-image information, and their combination with the inertial sensor data is based on conventional optimization and filtering techniques. The study shows that the combined information leads to an improvement of navigation data. Filtering techniques are found to be capable of quantitatively estimating major error sources inherent to the gyros and accelerometers

1. INTRODUCTION

The position and the course of a missile could in principle be determined by using only one navigation system. Examples are inertial, ground based or satellite based navigation. However by combining the data of several navigation systems, the overall precision can be improved [1]

Besides precision, other qualities like passiveness (no emitting sources) and independence of external aids are desirable too Inertial navigation systems (INS) are autonomous in this sense. But the components of an INS, the gyros and accelerometers, are subject to a number of error sources This leads to a time-dependent decrease of accuracy for location and orientation measurements Therefore, prior to the termination of the mission updates based on different sources are required from time to time It is even possible to calibrate the INS during the mission.

An aid for the INS which preserves the autonomy and which is passive and thus not detectable, is navigation based on infrared images of landmarks. Combining infrared image based navigation and inertial navigation is interesting because it will also represent an economic solution. The same image sensor is used both for navigation update in a midcourse guidance phase, and target recognition with subsequent tracking in a terminal homing phase, where it is needed anyway

Technically, inertial and image based navigations are complementary in their error behaviour The short term stability of inertial systems and the long term stability of the image based navigation complement each other. A plot of the typical time-dependences of the position and velocity errors of an aided INS is shown in fig 1. In this figure the updates lead to substantial reductions of the errors The magnitude of the effect depends of course on the precision of the aiding navigation source

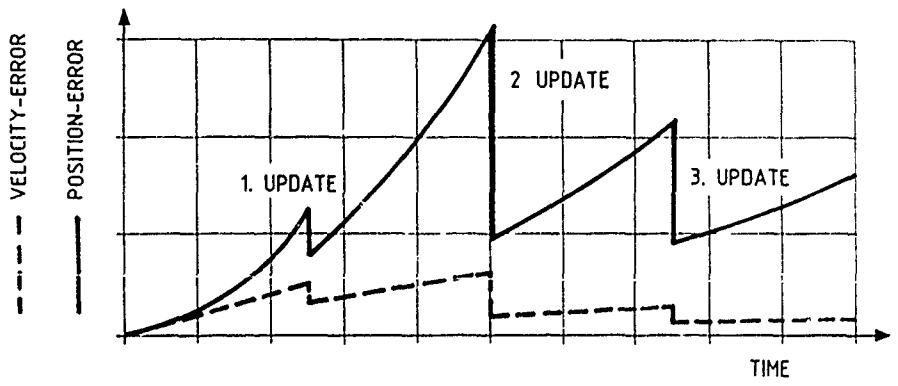


Fig.1 Error propagation of an aided navigation system

The aim of the present study was to investigate methods for extracting navigation relevant information from infrared images and to determine how this information can be used to produce calibration effects comparable to those shown in fig. 1

2. TASK DESCRIPTION

A prerequisite for using images as navigation aids is a map Based on the map, one can calculate the position of certain structures in the image plane. This calculation is done by using an estimate for the location and orientation of the image sensor, resp of the missile itself. From the difference of the expected and real positions of the structure in the image, corrections for the location and orientation of the missile can be derived This comparison becomes possible if the structure has been separated from the background. In fig 2 the expected and extracted real positions of a road structure are plotted on top of the IR-image. The structure is symbolically stored in terms of straight lines and their intersections

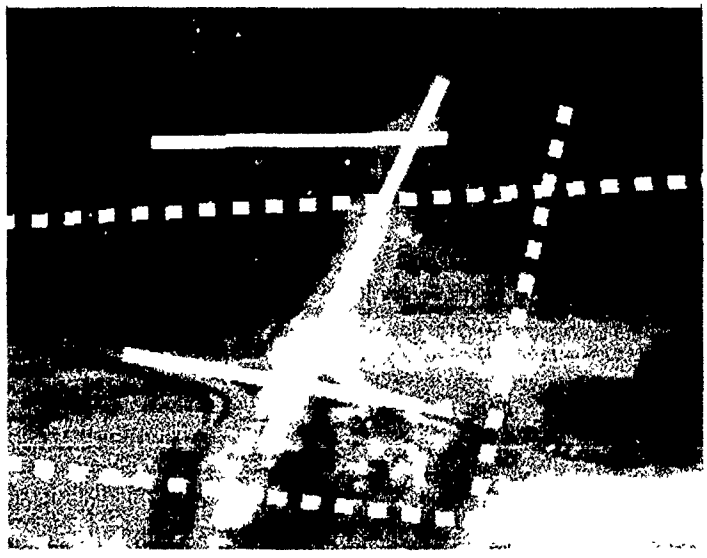


Fig 2: Extracted (full lines) and expected (dashed lines) road structure The difference in location and orientation reflects the INS navigation error

The extraction of navigation relevant objects from images relies on todays abilities of image interpretation Solutions in this area are only applicable in very restricted domains [2] If one considers a cruise missile type carrier flying at low altitude, additional difficulties, as the visibility of objects, arise. Objects can be partly or fully hidden due to hills and vegetation Besides that, perspective distortions and the influences of the daytime and the weather must be handled Therefore the recognition algorithms need a high interpretative power. In order to achieve the interpretation task, certain concepts being presently applied in artificial intelligence models have been analyzed, adapted and implemented In particular, knowledge based search techniques turned out to be most helpful tools.

By an appropriate use of such methods in combination with adapted preprocessing techniques, the difficult image interpretation task can be solved

Fig.3. An example image to be interpreted in terms of navigation relevant data

3. THE KNOWLEDGE BASED IMAGE INTERPRETATION CONCEPT

Some necessary properties of the image interpretation system can be derived from the application requirements The system should be:

- adaptable to different types of targets and objects,relevant to navigation
- robust against interference factors as well as daytime influences and weather
- capable of real time execution on a suitable hardware.

Our analysis shows that the following procedure responds to the demands

3.1 Preprocessing

Figure 3 shows an infrared scene that is well suited for navigation update. It has a digitized size of 512x512 pixels. The aim of preprocessing, as we use it, is the extraction and recording of object edges Edge extraction techniques allow the processing of low contrasts Furthermore edge extraction is independent of daytime, during which the contrast can switch. As far as noise is concerned, edge extracting operators can be conceived such that local continuity is used as a hint for the existence of an edge [3]. On the whole, these techniques seem to be essential to preserve robustness at the early processing stages

Edges are recorded as objects with as many attributes as possible. We think that it is crucial for the application of knowledge based interpretation techniques that as little information as possible is lost during preprocessing. On the other hand, the results must be represented in terms of data structures, that can be used as input to the interpretation programs. Therefore the result of image preprocessing is not another filtered image but a series of edge elements from which the original image could approximately be reproduced. In fig. 4 the edge elements of fig. 3 are shown. They are represented as straight line segments. In fig. 5 edge segments from the boundaries of contrast stripes have been combined [4]. Thus we get a representation of "lines" that are present in the image In fig 5 lines that are brighter than the surrounding background are shown The dark lines are also recorded. The road that runs through the village can already be located

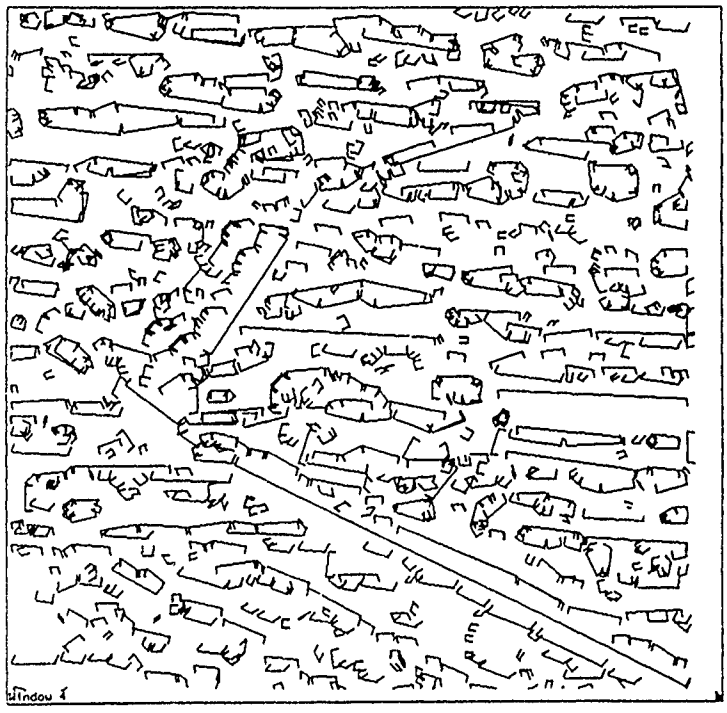


Fig 4· Edge elements of part of the image shown in fig 3 The straight line segments lie on the boundary of positive contrast domains

3.2 Interpretation

The method for interpreting the preprocessed image data in terms of meaningful objects, is based on abstract models of these objects [5] The shape of an extended object is our primary criterion for recognition This is however not the only possible interpretation paradigm Texture could be used as well. "Shape recognition" turns out to be a very important method, because of the robustness requirement Objects may be partly invisible and still show some critical shape features Shape models are implemented with the help of rule based programming. It is easily possible to switch between different rule subsets and interpret images in terms of different expected structures.

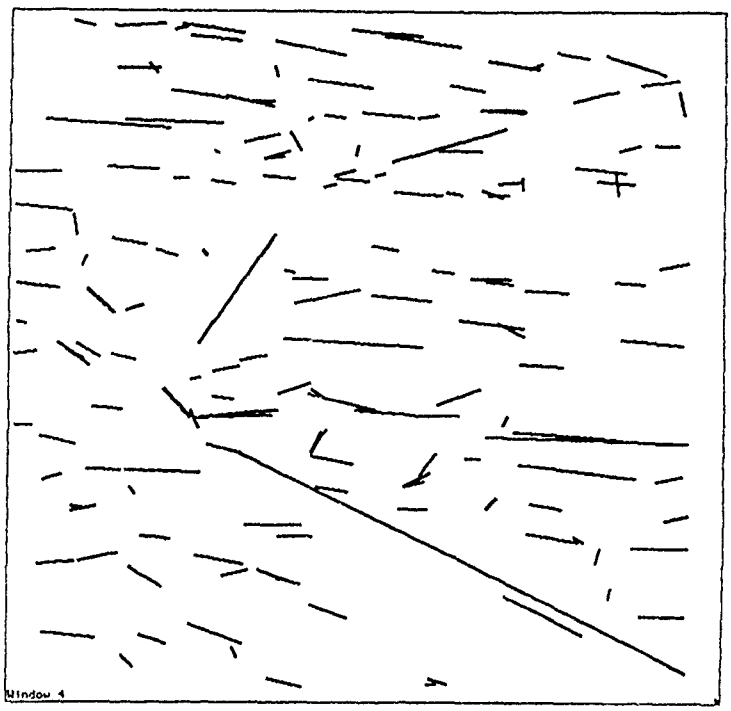


Fig 5 Symbolic representation of the lines of part of the image of fig 3

Due to the fact that objects appear differently on the map and in the image, a modelling, which is in principle independent of the perspective and other distortions is required

Therefore the interpreting rules represent invariants in the relations between object parts Such invariants can be related to proximity, number, symmetry etc

A simple example should illustrate the basic method. Figure 6 shows an abstract model of the prominent road structure of fig 3 The aim is to identify certain straight line segments of fig 5 as belonging to the road

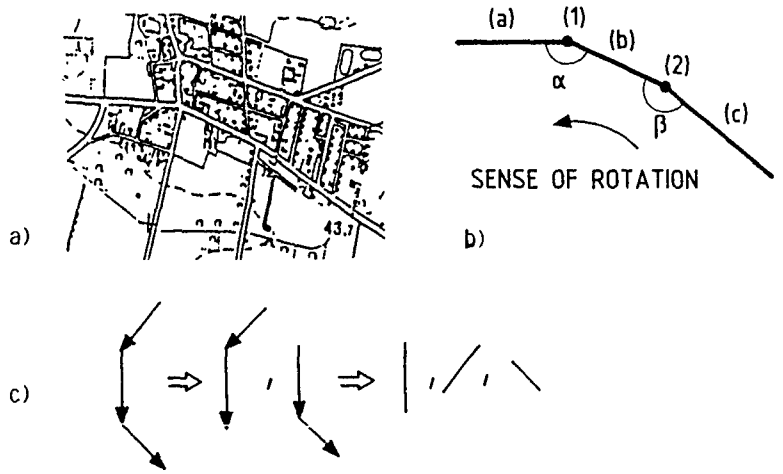


Fig.6. a) Map section; b) Object model, c) Object decomposition and definition of search levels

In this example we shall use an important characteristic that is not changed by the transformations from the map to the image plane. It is the convexity of the example structure If we define a sense of rotation as indicated in fig 6b, distinctive properties for the three straight line segments can be derived. Taking the line segments two by two, there is always an incoming and an outgoing line (fig. 6c). One can only differentiate between incoming and outgoing lines in the light of the predefined rotation sense and the fact that two lines form a corner that is part of the structure. This very simple example ought to show how certain shape properties can be described in terms of relations between

parts It is seen from figs. 6b and 6c that several interpretation levels must be introduced. Beginning at the line segment level corners are built which are subsequently combined to the 3-line convex structure, which is part of a polygon

The stepwise approach to interpretation is similar to the "blackboard" procedure [6] and to dynamic programming. The efficiency of the search relies on the introduction of the steps and the use of adequate knowledge which acts as a filter for the creation of objects on the next higher level. This approach is different from a hypotheses-creating-and-testing approach. It is more efficient and uses less memory.

However using shape invariants for object recognition is not sufficient. On the highest levels of interpretation there may still exist several similar objects, which cannot be discerned by the above procedure. If this is the case, knowledge about the expected approximate size can be used to discriminate between the candidates. This knowledge is derived from the navigation parameters given by the INS. It is vague in the sense that the system must be aware of the inaccuracy of the INS data.

Fig. 7 shows 8 remaining candidates for the road structure discussed above. Some vague information on the expected orientation of the structure has been used. Further knowledge, concerning the approximate size, already leads to a unique solution. It turns out that the candidate, which is largest in size is most compatible with the expectations. This last step illustrates an important feature of our method. The INS can cooperate with the knowledge based interpretation system.

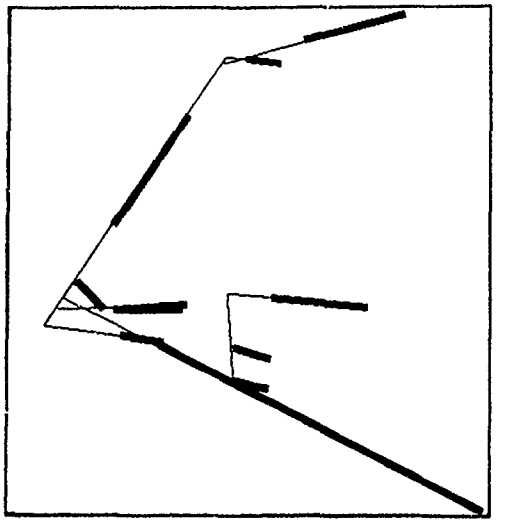


Fig.7 Remaining candidates after applying the convexity constraint and vague orientation knowledge

4. DERIVING NAVIGATION DATA FROM IMAGE OBJECTS

Isolated points, intersection points and straight line orientations are the quantities that are mainly used for deriving navigation data. Surface moment parameters could also be considered The observed difference between the expected and measured values for these quantities can be expressed in terms of corrections to the INS navigation parameters. These parameters consist of the location and orientation in space This leads, among others, to six parameters to be determined. In the example case of chapter 3 it is possible to determine these 6 parameters from the information obtained in one image frame. In general filtering techniques have to be applied, because the relative motion of the missile between frames must be taken into account. It is obtained from the INS, as the error that accumulates during the time elapsed between successive frames, can be neglected. Non-linear filtering techniques have been applied in order to derive the navigation parameters.

5. UPDATING THE INS

The objective of an inertial navigation system update is to improve the navigational data such as position and velocity. Furthermore, the updates can be used to estimate inertial sensor errors If sensor error compensation is applied, the accuracy of the system can be improved even more. The navigation updates are supplied by the image analysis system (IAS) The update information consists of a current position vector of the missile, which is made available to the INS

The update procedure is based on error models of the INS and IAS. An optimal filter, constructed with the help of those error models, combines the continuous navigation data from the INS with the position fixes from the IAS in order to calculate the navigation and sensor errors. A correction algorithm applies sensor calibration and resets the filter parameters (see figure 8).

5.1 INS error sources

In the study we considered a strapdown inertial system. The accuracy of such a system is influenced by

- the initialization errors,
- the computational errors,
- the direct sensor errors,
- sensor errors resulting from the dynamic environment

The initialization errors relate to the uncertainties of the location, velocity and orientation at the start. These uncertainties are negligible if the missile is launched from the ground. Numeric precision is particulary important with strapdown systems. The three orientation angles must be continuously recalculated from the signals of the gyros. The accuracy of the result depends on the numeric procedure. In general one tries to minimize the relative influence of this error source.

By direct sensor errors one understands the imperfections of the inertial instruments (gyros and accelerometers). Their influence can be disastrous, because they are integrated over time. Position errors due to the gyro drifts can evolve with the third power of time. It is possible to estimate the sensor errors using Kalman-filtering and to apply compensations.

In strapdown inertial systems some errors depend on the dyn- - environment. For example the errors induced by the gyro scale factors increase with increasing angular rates of the missile body Depending on the magnitude of the angular rates, it is also possible to estimate scale factor errors

The influence of the different kinds of error sources has been determined in simulation. As a result the error propagation is based on a simplified model. It turns out that the dominant role is played by the accelerometer bias.

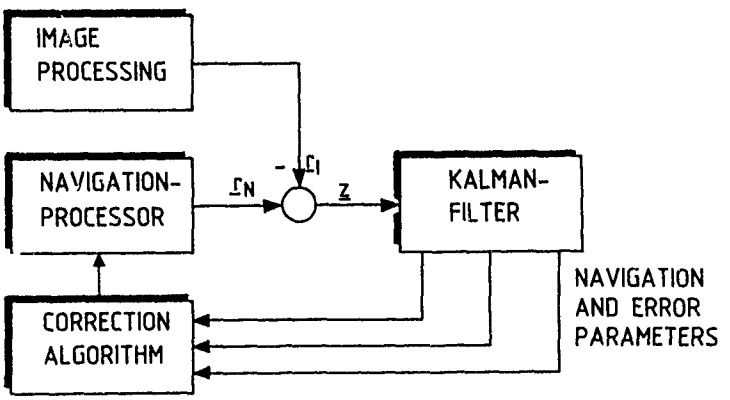


Fig 8 Data flow diagram for the INS update

5.2 Estimating the INS errors

In the block diagram of fig 8 the INS updating procedure is shown The image analysis along the flight trajectory provides a location vector r_I. The difference between the INS-value r_N and r_I is the measurable quantity which is fed into the Kalman-filter. Here a new state consisting of location error, velocity error and accelerometer bias is estimated. Via a correction algorithm the filtering results are adapted for the navigation processor and the filter itself

6. RESULTS AND CONCLUSIONS

Computer flights have been performed using real world scenery For this purpose an object oriented database for the storage of generic object data, as shown in fig 6b, has been designed. In this database the navigation relevant features of objects and structures are stored. It must not be confounded with the knowledge base used for recognition, although one could think of combining the two kind of data in a later stage.

From the simulated flights the following conclusions could be drawn: Using knowledge based techniques, it is possible to implement an image processing concept, which allows the extraction of navigation relevant data The accuracy of the data is sufficient for supporting the inertial navigation system. The integration can be done by using filtering techniques Orientation data can also be derived from the image data, but it has not yet been taken into account. Real time execution seems possible on a suitable processor hardware

References

1. Britting J K , "Inertial Navigation Systems Analysis", John Wiley & Sons, New York, 1971

2. See for instance Proc. of the DARPA Image Understanding Workshop, Los Angeles California 1987 and successive workshops.

3 J F. Canny, "Finding Edges and Lines in Images", thesis MIT, June 1983

4. Nevatia, R. and Babu R., "Linear Feature Extraction and Description", Computer Vision, Graphics and Image Processing, Vol. 13, 1980, pp. 257-269

5 Kanade T., in "Pictorial Data Analysis edited by R.M Haralick, NATO ASI Series, Series F: Computer and Systems Sciences No. 4, Springer-Verlag 1983

6 R Engelmore, T. Morgan eds., "Blackboard Systems", Addison-Wesley Publishing Company 1988.

REPORT DOCUMENTATION PAGE			
1. Recipient's Reference	**2. Originator's Reference**	**3. Further Reference**	**4. Security Classification of Document**
	AGARD-CP-474	ISBN 92-835-0610-3	UNCLASSIFIED

5. Originator	Advisory Group for Aerospace Research and Development North Atlantic Treaty Organization 7 rue Ancelle, 92200 Neuilly sur Seine, France
6. Title	KNOWLEDGE BASED SYSTEM APPLICATIONS FOR GUIDANCE AND CONTROL
7. Presented at	the Guidance and Control Panel 51st Symposium held at the Instituto Nacional de Industria, Madrid, Spain from 18th to 21st September 1990.

8. Author(s)/Editor(s)	**9. Date**
Various	April 1991
10. Author's/Editor's Address	**11. Pages**
Various	266

12. Distribution Statement	This document is distributed in accordance with AGARD policies and regulations, which are outlined on the back covers of all AGARD publications.

13. Keywords/Descriptors

AI	Pilot decision aiding
Combat automation	Sensor fusion
On-board tactical battle management	Tactical information management

14. Abstract

This volume contains the 21 unclassified papers, and the Keynote Address, presented at the Guidance and Control Panel Symposium held at the Instituto Nacional de Industria, Madrid, Spain from 18th to 21st September 1990.

The papers were presented covering the following headings:
— Representative applications;
— Design concepts and synthesis techniques,
— Related methods and techniques;
— Information processing and system architecture;
— Mechanization and integration issues.

AGARD Conference Proceedings No 474
Advisory Group for Aerospace Research and Development, NATO
KNOWLEDGE BASED SYSTEM APPLICATIONS FOR GUIDANCE AND CONTROL
Published April 1991
266 pages

This volume contains the 21 unclassified papers, and the Keynote Address, presented at the Guidance and Control Panel Symposium held at the Instituto Nacional de Industria, Madrid, Spain from 18th to 21st September 1990.

The papers were presented covering the following headings.

P.T.O.

AGARD-CP-474

AI
Combat automation
On-board tactical battle management
Pilot decision aiding
Sensor fusion
Tactical information management

AGARD Conference Proceedings No.474
Advisory Group for Aerospace Research and Development, NATO
KNOWLEDGE BASED SYSTEM APPLICATIONS FOR GUIDANCE AND CONTROL
Published April 1991
266 pages

This volume contains the 21 unclassified papers, and the Keynote Address, presented at the Guidance and Control Panel Symposium held at the Instituto Nacional de Industria, Madrid, Spain from 18th to 21st September 1990

The papers were presented covering the following headings

P.T.O.

AGARD-CP-474

AI
Combat automation
On-board tactical battle management
Pilot decision aiding
Sensor fusion
Tactical information management

AGARD Conference Proceedings No 474
Advisory Group for Aerospace Research and Development, NATO
KNOWLEDGE BASED SYSTEM APPLICATIONS FOR GUIDANCE AND CONTROL
Published April 1991
266 pages

This volume contains the 21 unclassified papers, and the Keynote Address, presented at the Guidance and Control Panel Symposium held at the Instituto Nacional de Industria, Madrid, Spain from 18th to 21st September 1990.

The papers were presented covering the following headings:

P.T.O

AGARD-CP-474

AI
Combat automation
On-board tactical battle management
Pilot decision aiding
Sensor fusion
Tactical information management

AGARD Conference Proceedings No 474
Advisory Group for Aerospace Research and Development, NATO
KNOWLEDGE BASED SYSTEM APPLICATIONS FOR GUIDANCE AND CONTROL
Published April 1991
266 pages

This volume contains the 21 unclassified papers, and the Keynote Address, presented at the Guidance and Control Panel Symposium held at the Instituto Nacional de Industria, Madrid, Spain from 18th to 21st September 1990.

The papers were presented covering the following headings:

PTO

AGARD-CP-474

AI
Combat automation
On-board tactical battle management
Pilot decision aiding
Sensor fusion
Tactical information management

— Representative applications, — Design concepts and synthesis techniques; — Related methods and techniques, — Information processing and system architecture; — Mechanization and integration issues ISBN 92-835-0610-3	— Representative applications; — Design concepts and synthesis techniques; — Related methods and techniques, — Information processing and system architecture; — Mechanization and integration issues ISBN 92-835-0610-3
— Representative applications, — Design concepts and synthesis techniques, — Related methods and techniques, — Information processing and system architecture, — Mechanization and integration issues ISBN 92-835-0610-3	— Representative applications; — Design concepts and synthesis techniques; — Related methods and techniques, — Information processing and system architecture; — Mechanization and integration issues ISBN 92-835-0610-3

AGARD

NATO OTAN

7 RUE ANCELLE · 92200 NEUILLY-SUR-SEINE

FRANCE

Téléphone (1)47.38.57.00 · Télex 610 176
Télécopie (1)47.38.57.99

DIFFUSION DES PUBLICATIONS
AGARD NON CLASSIFIEES

L'AGARD ne détient pas de stocks de ses publications, dans un but de distribution générale à l'adresse ci-dessus. La diffusion initiale des publications de l'AGARD est effectuée auprès des pays membres de cette organisation par l'intermédiaire des Centres Nationaux de Distribution suivants. A l'exception des Etats-Unis, ces centres disposent parfois d'exemplaires additionnels, dans les cas contraire, on peut se procurer ces exemplaires sous forme de microfiches ou de microcopies auprès des Agences de Vente dont la liste suite.

CENTRES DE DIFFUSION NATIONAUX

ALLEMAGNE
Fachinformationszentrum,
Karlsruhe
D-7514 Eggenstein-Leopoldshafen 2

BELGIQUE
Coordonnateur AGARD-VSL
Etat-Major de la Force Aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Directeur du Service des Renseignements Scientifiques
Ministère de la Défense Nationale
Ottawa, Ontario K1A 0K2

DANEMARK
Danish Defence Research Board
Ved Idraetsparken 4
2100 Copenhagen Ø

ESPAGNE
INTA (AGARD Publications)
Pintor Rosales 34
28008 Madrid

ETATS-UNIS
National Aeronautics and Space Administration
Langley Research Center
M/S 180
Hampton, Virginia 23665

FRANCE
O N E R A. (Direction)
29, Avenue de la Division Leclerc
92320, Châtillon sous Bagneux

GRECE
Hellenic Air Force
Air War College
Scientific and Technical Library
Dekelia Air Force Base
Dekelia, Athens TGA 1010

ISLANDE
Director of Aviation
c/o Flugrad
Reykjavik

ITALIE
Aeronautica Militaire
Ufficio del Delegato Nazionale all'AGARD
3 Piazzale Adenauer
00144 Roma EUR

LUXEMBOURG
Voir Belgique

NORVEGE
Norwegian Defence Research Establishment
Attn: Biblioteket
P O. Box 25
N-2007 Kjeller

PAYS-BAS
Netherlands Delegation to AGARD
National Aerospace Laboratory NLR
Kluyverweg 1
2629 HS Delft

PORTUGAL
Portuguese National Coordinator to AGARD
Gabinete de Estudos e Programas
CLAFA
Base de Alfragide
Alfragide
2700 Amadora

ROYAUME UNI
Defence Research Information Centre
Kentigern House
65 Brown Street
Glasgow G2 8EX

TURQUIE
Milli Savunma Başkanlığı (MSB)
ARGE Daire Başkanlığı (ARGE)
Ankara

LE CENTRE NATIONAL DE DISTRIBUTION DES ETATS-UNIS (NASA) NE DETIENT PAS DE STOCKS DES PUBLICATIONS AGARD ET LES DEMANDES D'EXEMPLAIRES DOIVENT ETRE ADRESSEES DIRECTEMENT AU SERVICE NATIONAL TECHNIQUE DE L'INFORMATION (NTIS) DONT L'ADRESSE SUIT.

AGENCES DE VENTE

National Technical Information Service
(NTIS)
5285 Port Royal Road
Springfield, Virginia 22161
Etats-Unis

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris
France

The British Library
Document Supply Division
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
Royaume Uni

Les demandes de microfiches ou de photocopies de documents AGARD (y compris les demandes faites auprès du NTIS) doivent comporter la dénomination AGARD, ainsi que le numéro de série de l'AGARD (par exemple AGARD-AG-315) Des informations analogues, telles que le titre et la date de publication sont souhaitables. Veuiller noter qu'il y a lieu de spécifier AGARD-R-nnn et AGARD-AR-nnn lors de la commande de rapports AGARD et des rapports consultatifs AGARD respectivement. Des références bibliographiques complètes ainsi que des résumés des publications AGARD figurent dans les journaux suivants:

Scientifique and Technical Aerospace Reports (STAR)
publié par la NASA Scientific and Technical
Information Division
NASA Headquarters (NTT)
Washington D.C. 20546
Etats-Unis

Government Reports Announcements and Index (GRA&I)
publié par le National Technical Information Service
Springfield
Virginia 22161
Etats-Unis

(accessible également en mode interactif dans la base de données bibliographiques en ligne du NTIS, et sur CD-ROM)

Imprimé par Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ