Defence Research and Development Canada Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Supporting uncertainty reasoning in SIMulated Operators for Networks (SIMON)

Ruibiao J. Guo

Brad Cain

Pierre Meunier

Defence R&D Canada – Toronto

Technical Report

DRDC Toronto TR 2005-268

December 2005

Canada

Supporting uncertainty reasoning in SIMulated Operators for Networks (SIMON)

Ruibiao J. Guo

Brad Cain

Pierre Meunier

Defence R&D Canada – Toronto

Technical Report

DRDC Toronto TR 2005-268

December 2005

Author

Ruibiao J. Guo, Brad Cain and Pierre Meunier

Approved by

Keith Hendy

Acting Head, Simulation & Modelling for Acquisition, Rehearsal & Training

Approved for release by

Kathy Sutton

Chair, Document Review & Library Committee

Abstract

This report introduces a knowledge representation and reasoning scheme that can accommodate uncertainty in simulation of military personnel. The Integrated Performance Modelling Environment (IPME) simulation engine is used with the approach to demonstrate how advanced reasoning models can be integrated with conventional task network modelling to provide greater functionality and flexibility when modelling operator performance. This approach is capable of using multiple reasoning methods, including first-order logic, fuzzy logic and probability reasoning, and supporting reuse of knowledge attributes.

Résumé

Ce rapport introduit une représentation de la connaissance et du raisonnement qui peuvent accommoder l'incertitude lors d’une simulation de personnel militaire. Le moteur de simulation de l'Environnement Intégré de Modélisation de la Performance est utilisé de pair avec l'approche pour démontrer comment des modèles de raisonnement avancés peuvent être intégrés dans un réseau de tâches conventionnel pour fournir un caractère fonctionnel plus grand ainsi qu’une plus grande flexibilité lors de la modélisation d'un opérateur. Cette approche permet l'utilisation de méthodes de raisonnement de multiples, y compris la logique de premier ordre, la logique floue et le raisonnement de probabilité, et peut soutenir l'héritage d’attributs de connaissance.

This page intentionally left blank.

Executive summary

Human behaviour representation refers to computer-based models that mimic either the behaviour of a single human or the collective actions of a team of humans. The project SIMON – SIMulated Operators for Networks – is to develop an architecture for modelling military personnel in Computer-Generated Forces (CGF) by extending IPME through re-configurable components, including knowledge acquisition, perception and operator state, performance, cognition, emotions and diagnosis.

A knowledge representation approach is proposed that provides reasoning with multiple methods, such as first-order logic, fuzzy logic and probability reasoning in the project SIMON. This reasoning system handles various reasoning-related tasks in human behaviour modelling, and interacts with simulators or simulated tasks in IPME.

The Language of Agents for Modelling Performance (LAMP) described in this report is able to represent both deterministic and uncertainty knowledge in CGF. At a high level, LAMP consists of structures of reasoning components called *Aspect*s that can be invoked by simulators or IPME task nodes for decision-making. Each *Aspect* consists of a sense interface for acquiring environment data, a group of rules for inference and a collection of methods for data conversion between sense data and reasoning rules.

The LAMP-based reasoning software is implemented with C/C++ and Fast Light Tool Kit (FLTK) under cygwin Linux. Currently, the demonstrations of first-order logic, fuzzy logic and probability reasoning are available. The current prototypes show the capability to be used in CGF. Further work includes more testing, integrations with simulators and simulation engines, and development of more and larger applications. This report provides two examples of uncertainty reasoning in CGF: one is for battlefield reasoning with probability method and another is for helicopter control by fuzzy logic.

Ruibiao J. Guo, Brad Cain and Pierre Meunier. 2005. Supporting uncertainty reasoning in SIMulated Operators for Networks (SIMON). DRDC Toronto TR 2005-268. DRDC Toronto.

Sommaire

La représentation de comportement humain réfère aux modèles par ordinateur qui imitent soit le comportement d'un humain seul ou les actions collectives d'une équipe d'humains. Le projet SIMON – SIMulated Operators for Networks – a pour but le développement d’une architecture pour modéliser le personnel militaire dans une simulation en augmentant IPME avec des composantes reconfigurables, y compris l'acquisition de connaissance, l'état de perception de l’opérateur, l'exécution, la connaissance, les émotions et le diagnostic.

Nous proposons une approche de représentation de connaissance soutenant le projet de SIMON qui fournira le raisonnement par méthodes multiples, telles que la logique de premier ordre, la logique floue et le raisonnement par probabilité. Ce système de raisonnement contrôle diverses tâches reliées au raisonnement dans la modélisation de comportement humain, et interagit avec les simulateurs ou les tâches simulées dans IPME.

Le Language d'Agents pour Modéliser la PErformance (le LAMPE) décrit dans ce rapport peut représenter la connaissance déterministe et d'incertitude dans une simulation. Au niveau supérieur, LAMPE est composé de structures de raisonnement appelées *Aspects* qui peuvent être invoqués par les simulateurs ou les tâches de IPME pour la prise de décision. Chaque *Aspect* consiste en une interface de sens pour acquérir des données d'environnement, un groupe de règles pour l'inférence et une collection de méthodes pour la conversion de données entre les données de sens et les règles de raisonnement.

Le logiciel de raisonnement basé sur LAMPE a été implanté avec le C/C + + et le Fast Light Tool Kit (FLTK) sous cygwin Linux. Actuellement, la logique de premier ordre, la logique floue et les démonstrations de raisonnement de probabilité sont disponibles. Les prototypes actuels démontrent la capacité d’être utilisé dans les forces générées par ordinateur. Le travail ultérieur inclura plus d'essais, l’intégration avec les moteurs de simulateurs et de simulation, et le développement d’un plus grand nombre d’applications. Ce rapport fournit deux exemples de raisonnement d'incertitude dans CGF : l'un est pour le contrôle d'hélicoptère par la logique floue et l’autre est pour le raisonnement de champ de bataille par la méthode de probabilité.

Ruibiao J. Guo, Brad Cain and Pierre Meunier. 2005. Supporting uncertainty reasoning in SIMulated Operators for Networks (SIMON). DRDC Toronto TR 2005-268. DRDC Toronto.

Table of contents

List of figures

List of tables

Acknowledgements

The authors appreciate the contributions from Dr. Peter Kwantes in DRDC Toronto for helping us shape the direction and content of this research.

1. Introduction

Human behaviour representation (HBR) refers to computer-based models that mimic either the behaviour of a single human or the collective actions of a team of humans (Pew and Mavor, 1998). With a few notable exceptions, most HBR and Computer-Generated Forces (CGF) approaches follow an Artificial Intelligence (AI) approach that lacks plausible support from the human sciences. In a number of cases, these approaches have been found to be brittle and lack sufficient representational power to provide adequate performance, particularly for training simulations. Models of perception, cognition and behaviour moderator functions from the human sciences are thought to be a means of extending a pure AI approach such that it can better represent the military operators that the HBR is replacing.

Rule-based approaches are dominating the current human modelling area and used in various applications of CGF (Pew and Mavor, 1998). Some examples include the situation awareness for aircraft and pilot simulation (Tambe et al., 1995; Hill et al., 1997; Gratch and Hill, 1999; Jones et al., 1999; Ehlert et al., 2003), and threat event detection (Mulgund et al., 2000). Rule-based decision-making is an active research area in CGF, for instance, efforts for combat pilot target selection (Doyal and Brett, 2003), land forces (Burdick et al., 2003) and navy tasks (Stevens and Parish, 1996). There are also some other applications using the rule-based method for planning, such as the route selection of fire-teams (Hoff, 1996).

First-order-logic based systems are appropriate, in particular, for deterministic parameters or precise system models. Unfortunately, many parameters related to environment, cognition and moderators are uncertain, such as intentions of enemy forces, or operators' emotions. People often infer consequences with incomplete or uncertain information. For example, commanders might use linguistic labels such as "tired" and "energetic" to describe soldiers' physical status and make decisions based on internal values to which those labels correspond.

The Simulated Operators for Networks (SIMON) project (Cain and Kwantes, 2004) is attempting to extend the Integrated Performance Modelling Environment (IPME) and develop an architecture for modelling military personnel in CGF that can readily add human science information to HBRs.

A representation approach supporting reasoning in SIMON was reported (Guo, Cain and Meunier, 2005). This reasoning system handles both deterministic and uncertainty factors in human behaviour modelling, and interacts with simulated tasks in IPME. At the moment, the approach is limited to AI reasoning algorithms, however, there is nothing constraining the models from being more human-like in the reasoning process if the human-like reasoning process can be specified.

This document focuses on the uncertainty representation of virtual operators. The Section 2 analyzes uncertainty in CGF and reviews the systems with uncertainty reasoning in a variety of military applications. Then, Section 3 introduces a representation approach for uncertainty reasoning in virtual operators. Section 4 will deal with the military applications with our approach. Section 5 will conclude our efforts.

2. Uncertainty in Computer-Generated Forces (CGF)

Uncertainty exists in a variety of military simulation areas, for example in command and control, military aircraft and unmanned vehicle operation, landing safety officer (LSO) procedures, search and rescue missions, image recognition and behaviour moderators. In many cases, commanders cannot get exact and deterministic situation information, such as strength of adversary forces. It is also hard for pilots to foresee weather conditions or potential threats. For an unmanned vehicle controller, the data related to the current environment, e.g. terrain shapes, face smoothness, friction coefficients and obstacles, is often vague or non-deterministic. The decision-making process, for a LSO, is complicated by uncertainty stemming from the pilots' flying skills or performance on a given day, weather conditions, etc. There is also uncertainty in search and rescue missions stemming from an approximate knowledge of the position of a target, the time required for putting out a fire in an area, etc.

Many researchers explore uncertainty representation in or close to military simulation areas such as the ones mentioned above. First, in the simulation of command and control, fuzzy logic (Vakas et al., 2001; Looney and Liang, 2003; Burdick et al., 2003) and probabilistic methods (Brynielsson, 2004; Yu, 2003; Moffat and Witty, 2002; Mengshoel and Wilkins, 1998) are being used to support uncertainty reasoning for situational assessments and decision-making. Second, for pilot and aircraft simulations, some projects assist pilots' situation recognition and decision-making (Jeram, 2002; Mulgund et al., 1997; Ivansson, 2002; Blasch, 1997; Zeyada and Hess, 2000). Some other examples use uncertainty methods for helicopter control (Jeram, 2002; Montgomery and Bekey, 1998), and diagnosis of faults (Hamza and Menon, 2001). Third, for unmanned vehicle control, some researchers use uncertainty and fuzzy reasoning for safe vehicle speeds (Tunstel et al., 2002), obstacle avoidance and vehicle control (Cao and Hall, 1998; Kadmiry, 2002; Buskey et al., 2002; Panagiotis and Tzafestas, 2003; Fayad and Webb, 1999). Fourth, Richards' efforts demonstrate the possibility of using fuzzy logic and neural networks to predict the trajectory of a landing helicopter (Richards, 2002). Fifth, in the search and rescue area, a few researchers simulate target positioning with the Bayesian method (Abi-Zeif and Frost, 2005), and some others use fuzzy logic to plan rescue activities (Asuncion et al., 2004). Sixth, in the image recognition area, various studies show the ability of fuzzy logic and Bayesian networks used for the recognition and classification of objects (Geisler and Kersten, 2002), terrain traversability analysis (Howard et al., 2001), and shape recognition and retrieval in images (Gadi et al., 1999; Bimbo and Pala, 1997). Last, a number of performance moderators relevant to military operator simulations have been identified in the literatures (Pew and Mavor, 1998; Ritter and Avraamides, 2000). Picard proposed a theory known as affecting computing (Picard, 1995) that deals with the recognition and effect of emotions by the hidden Markov model. There are also researchers focusing on the impact of emotions on intelligent agents (El-Nasr et al., 2000) and on the drivers of autonomous vehicles (Al-Shihabi and Mourant, 2001) with fuzzy logic.

In summary, there are a great number of factors in CGF that are vague, incomplete and uncertain. Human beings often infer conclusions based on such uncertainty information. Therefore, it is necessary to develop tools for uncertainty reasoning in simulation engines, such as IPME, to support reasoning-related tasks, for instance, situation awareness, decision-making, planning, learning and action in CGF applications.

3. Representing uncertainty in virtual operators

This section introduces a representation for multiple reasoning methods with the focus on the uncertainty in virtual operators.

3.1 Introduction

IPME is a task network simulation software package for building models that simulate human and system performance. While providing a convenient means of representing operator tasks and estimating performance, IPME has no representation of higher-order cognitive functions such as memory or reasoning and it has no convenient way of developing these functions within its own modeling language. The project SIMON (Cain and Kwantes, 2004) is attempting to develop simulated military personnel by integrating component-based modules built on IPME to promote reuse of HBR models and model components both within the lifecycle of military equipment programs as well as sharing HBRs between programs.

The Language of Agents for Modelling Performance (LAMP) is used for reasoning in SIMON (Guo, Cain and Meunier, 2005). LAMP is simulated-task-oriented and is designed to support multiple reasoning methods, such as first order logic, fuzzy logic and probability methods.

3.2 Language of Agents for Modelling Performance

LAMP is a language of knowledge representation for various reasoning used in simulated task nodes. The overview of LAMP is shown in Figure 1. At the highest level, LAMP consists of structures of reasoning components called *Aspect*s that can be invoked by IPME tasks to return derived conclusions that moderate task execution and network branching. An *Aspect* is a knowledge unit containing attributes, facts and relationships, rules and procedures. LAMP and IPME currently communicate through a client-server SOCKET protocol.

LAMP *Aspect*s are reasoning units relevant to simulated tasks. Each *Aspect* schema is a 4-tuple:

AspectSchema = <MA, WM, LM, CL>,

where *MA* refers to meta-attributes; *WM* and *LM* are working memory and long-term memory respectively; and *CL* is a collaboration interface.

The following describes details of each member in the *Aspect* schema with a simple reasoning segment related to the emergency procedure prompts in a helicopter active control (Jeram, 2002). Assume that there is an IPME task node, named "*EmergencyProcedurePrompts*" that needs to call a fuzzy reasoning engine to derive the state of collective cues based on current situation data including torque percent, downward speed and forward speed. An *Aspect*, named "*EmergencyPromptsAspect*", supports the fuzzy reasoning used in the task node to determine the appropriate amount of collective to apply.

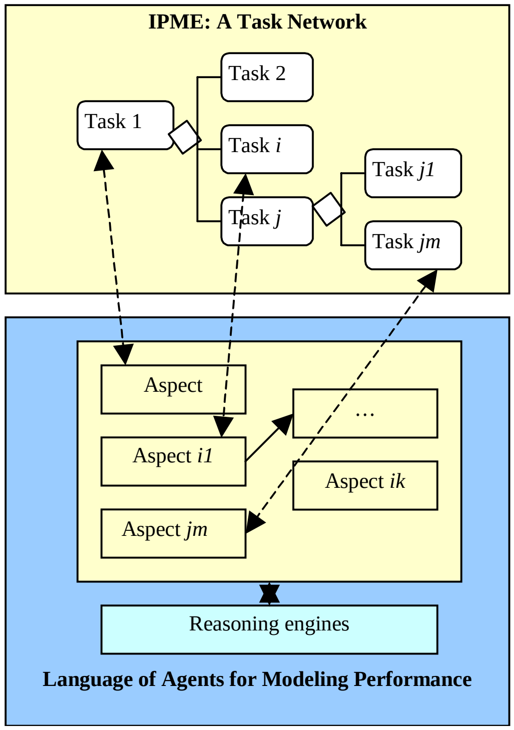

Figure 1. Overview of LAMP

Meta-Attributes (*MA*) in an *Aspect* describes this *Aspect* with *AspectName*, *AspectType*, *TaskList*, *Parent*, and *MetaSet*. *AspectType* represents the reasoning method chosen by analysts for a particular reasoning task, such as first order logic, fuzzy logic or probability. In the *EmergencyPromptsAspect*, the *AspectType* is fuzzy logic. *AspectName* is a unique identifier in system databases, such as "*InferCollectiveCue*" in this reasoning segment. *TaskList* associates this *Aspect* with simulated task nodes, in this example, "*EmergencyProcedurePrompts*". An *Aspect* is also able to inherit attributes and reasoning rules from its *Parent Aspect*. *MetaSet* contains the attributes that are global and thus shared by numerous *Aspects*, for example, available application areas.

The names Working Memory and Long Term Memory are analogous to those in psychology, but no pretence is made that the LAMP memory structures are equivalent to or even represent psychologically plausible processes. Working memory, *WM*, consists of *SenseSet*, *MiddleSet* and *OutcomeSet* that describe environment data, intermediate results and derived conclusions, respectively. In the *EmergencyPromptsAspect*, there are three environment variables: "*TorquePercent*", "*DownwardSpeed*", and "*ForwardSpeed*". Long-term memory, *LM*, is used to store persistent knowledge related to an *Aspect*. There are two kinds of long-term memories: declarative memory and procedural memory. Declarative memory is a collection of facts or relationships described by *items*. Each *item* has a name, a list of *atoms* and a *certainty* between 0.0 and 1.0. An *atom* in an *item* may be a label, a number, time or an *atom* list. An example of *items* in the *EmergencyPromptsAspect* is "[*Torque, nominal, Certainty=0.76*]". Procedural memory consists of a *RuleSet* and a *MethodSet*, which is where LAMP reasoning occurs.

RuleSet is the set of *rules* and logical associations that sustain inference. The schema of a *rule* is

RuleType RuleName {Condition$_1$, Condition$_2$, …, Condition$_m$ => Action$_1$, Action$_2$, …, Action$_w$},

where *RuleType* may be first order logic, fuzzy or probability, and *RuleName* is a unique identifier in system databases. Each *rule* consists of a group of *conditions* and a collection of *actions* derived from the *conditions*. In the *EmergencyPromptsAspect*, there are two *rules* for inferring the collective cue:

IF torque is nominal, and
the rate of descent is steep, and
horizontal speed is low,
THEN vortex ring is imminent; and

IF vortex ring is imminent and
altitude is safe,
THEN collective cue is decreasing,

in which "nominal", "steep", "low" etc. are fuzzy linguistic labels.

LAMP can also support conditional probability reasoning. For example, the probability that an alarm rings, given a burglary and an earthquake occur, can be represented by the following rule:

ProbabilityRule AlarmRings {

[Burglary, id] &&
[Earthquake, id]
=>
[AlarmRings] }

MethodSet comprises the procedures or functions used by the *RuleSet*. The *Aspect, EmergencyPromptsAspect*, needs functions to convert the environment data into fuzzy memberships. The *MethodSet* is the place for such functions used in reasoning.

Collaboration (*CL*) is a future element of *Aspect*s to represent cooperation among several *Aspect*s and meta-level controlling for reasoning process to reflect impacts of behaviour moderators. The details of collaboration will need more effort.

In summary, the *Aspect* schema in LAMP supports multiple reasoning methods, uncertainty and deterministic representation, and provides feature inheritance.

3.3 Implementation

LAMP is implemented with C/C++ and Fast Light Tool Kit (FLTK) under cygwin Linux environment. Figure 2 shows its software architecture.

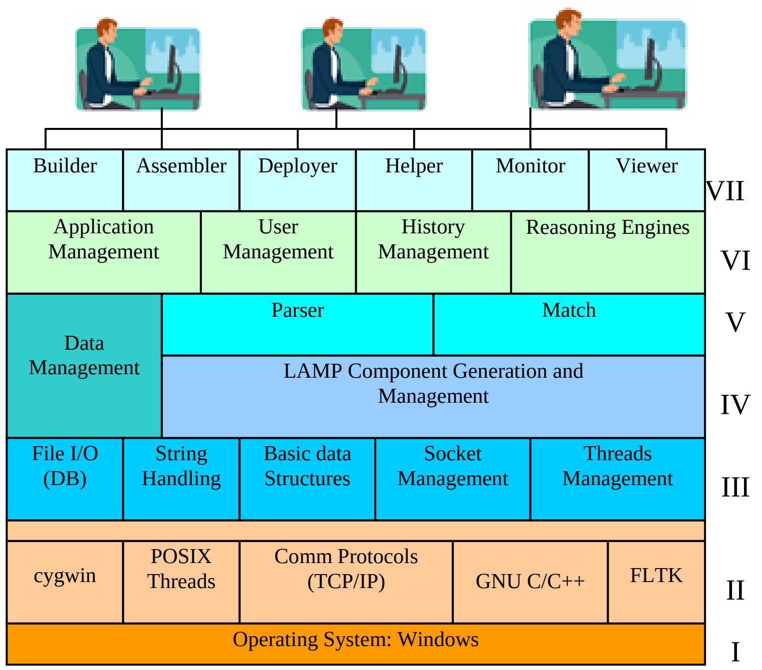

Figure 2. Software Architecture

There are three kinds of users in this system, including *Application Developers* or *Model Analysts*, *System Administrators* and *Application Users*. *Application Developers* are responsible for developing domain-specific applications with Graphic User Interfaces and their own domain expertise. *System Administrators* install, configure, deploy and maintain the reasoning system. *Application Users* make queries, provide situation data, activate reasoning engines and get conclusions from the reasoning system based on current domain applications built by *Application Developers*.

There are seven layers in the software architecture. The first two layers at the bottom (Layer I, II) are development and running environment. Currently, the operating system is Windows 2000. Cygwin provides the Linux environment under Windows. The programming language is GNU C/C++ and FLTK used for Graphical User Interface (GUI). POSIX threads and Transmission Control Protocol/Internet Protocol (TCP/IP) are used for communication between this reasoning system and the simulation engine IPME.

Two upper layers, layer VI and VII, offer user functionality. Actually, the designed goals include six tools: *Builder*, *Monitor*, *Viewer*, *Assembler*, *Deployer* and *Helper*.

The main window is shown in Figure 3, with which users can select and activate one of the six tools. The *Builder* is a development environment for *Application Developers* (shown in Figure 4), with which users are able to build applications consisting of *Aspect networks*. An application has a system name and one or more *Aspect networks*, each of which includes a group of structured *Aspects*. *Aspect Editor* is an independent window for developing *Aspects* (Figure 5). *Application Developers* fill forms in the GUI with parameters, rules and methods for domain-specific reasoning applications. The system formats the user inputs and stores them in inner files. *Monitor* (shown in Figure 6) is used for *Application Users* to provide environment data and queries, activate reasoning engines and get conclusions. *Assembler* and *Deployer* are used for administrators to configure, deploy and maintain the reasoning system. The *Helper* is a guidance manual of this software. Currently, the *Builder* and *Monitor* are available. Other tools will be developed further.

Figure 3. Main window of the reasoning system

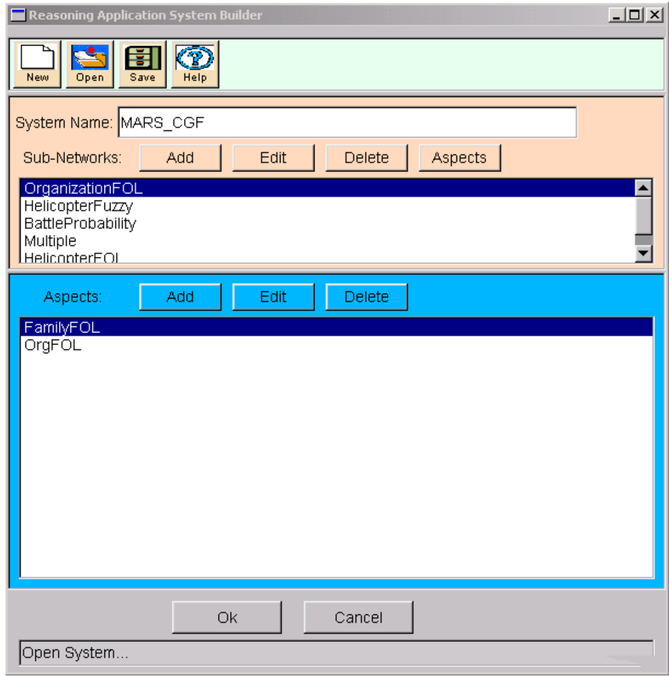

Figure 4. Builder: application development environment

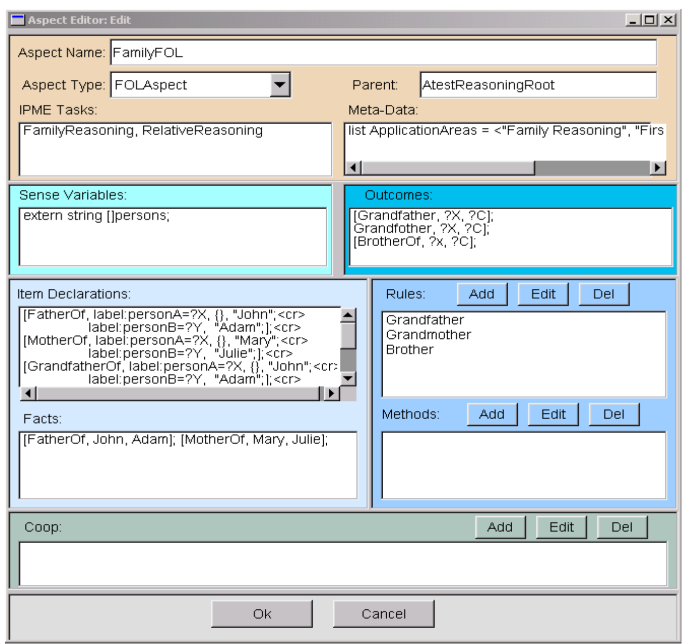

Figure 5. Aspect Editor

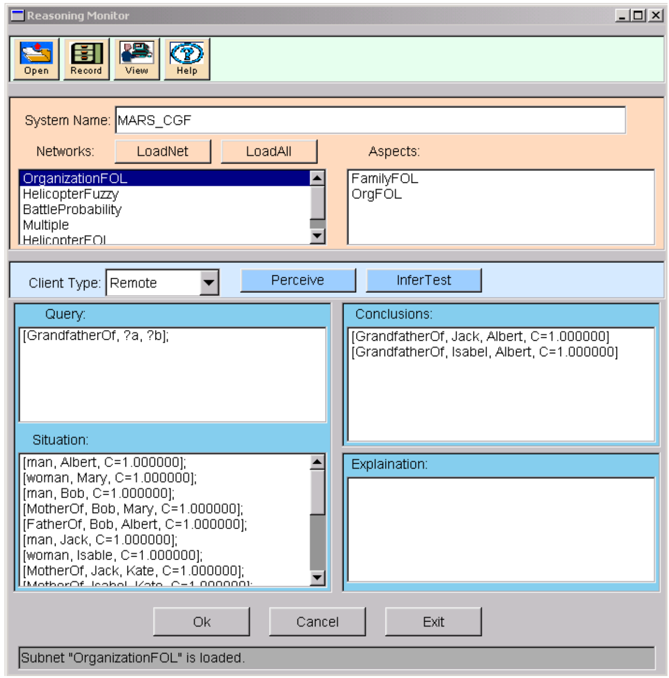

Figure 6. Monitor: displayer of reasoning results

The Layer IV and V in Figure 2 are the core part of the system with the help of the utility library at Layer III. There are a variety of LAMP components used in the system, including *atoms*, *items*, *rules*, *methods*, *aspects* and *aspect networks*. An *atom* is a character string that represents a label, a number, time or a list of labels. An *item* is a piece of minimal declarative knowledge with a name, a parameter list and a certainty between 0.0 and 1.0. An *item* is able to represents a fact, an object with attributes, or a relationship between objects, with a certainty. *Rules* in LAMP represent reasoning-related knowledge. A *rule* consists of a group of *conditions* and *actions*. Currently, LAMP supports first-order logic, fuzzy logic and probability rules. Hybrid rules will be sustained in future. *Actions* can be derived with a certainty if *conditions* in a *rule* can be instantiated with environment data. An *Aspect* includes sense variables, a *rule* set and a *method* set. A user application is a collection of *Aspect networks*.

The module *Match* in the software architecture, implemented with a Rete-like algorithm (Forgy, 1982), forms the kernel of our reasoning engines. The Figure 7 shows part of data structure used in the match algorithm. In this example, there are five conditions, labelled *C1*, *C2*, …, *C5*. Current situation data, wrapped in nine items, *W1*, *W2*, …, *W9*, is in the working memory. There are five rules with conditions above and actions, *A1*, *A2*, …, *A8*. The network in Figure 7 is called *Alpha_Beta* network, in which the nodes labelled "*AM*" are *Alpha* nodes representing current situation facts; and the "*Beta#*" nodes are the derived nodes for combinations of conditions based on the current environment data and derived middle nodes. Each *Beta* node includes matched conditions, a network of all matches and derived actions.

In this system, any combination of conditions is reusable for any other rules with same combination segment; for example, both *Beta3* and *Beta4* use *Beta2*.

Conditions:
C1: (on ?x ?y)   C2: (leftOf ?y ?z)   C3: (color ?z red) C4: … C5: …

Working Memory:
w1:(on B1 B2)   w2: (on B1 B3)   w3: (color B1 red)   w4: (on B2 table)   w5: (leftOf B2 B3)
w6: (color B2 blue)   w7: (leftOf B3 B4)   w8: (on B3 table)   w9: (color B3 red)

Rules:
P1: C1^C2^C3=> A1^A2   P2: C1^C2^C4=>A3   P3: C1^C2^C4^C5=>A4 ^A5 ^A6
P4: C1^C2^C4^C3=>A7   P5: A2^A5 =>A8

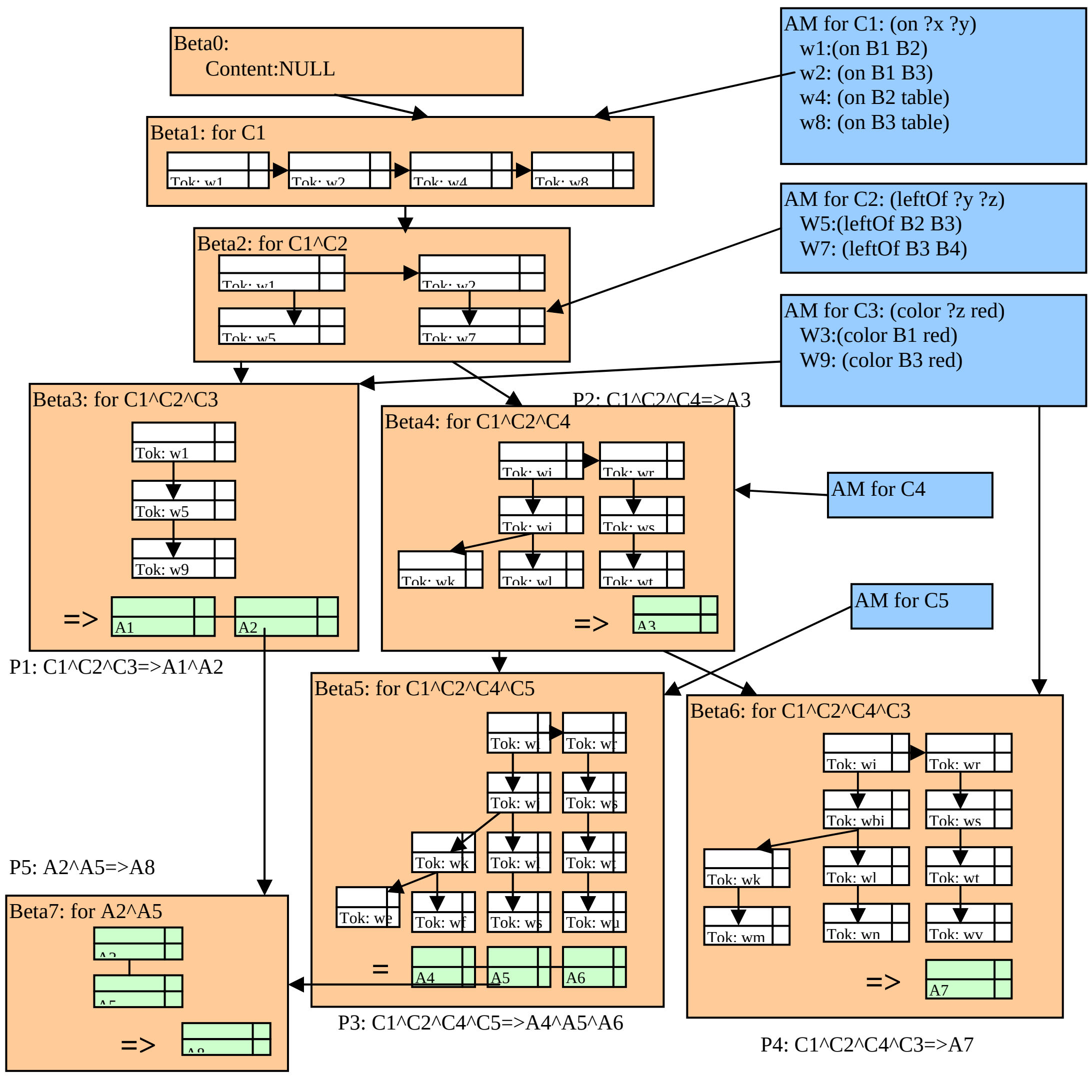

Figure 7. Alpha-Beta network: data structure used in the match algorithm

4. Supporting military applications

This section describes two military applications with the proposed approach. One is for battlefield reasoning with probabilistic method, and another is for helicopter control with fuzzy logic.

4.1 Battlefield reasoning

This part addresses probability-based battlefield reasoning for a simple example of decision-making. Commanders are attempting to make a decision for attack, barrage or defence based on several factors, including enemy's strategies, environment and strength of friendly forces. Figure 8(a) shows a sample battlefield simulator that could link to an IPME task network (Figure 8(b)). The task node "*EnemyStrategies*" needs to call the reasoning engine to infer enemy's strategies. The *Aspect* structure for such reasoning is shown in Figure 8 (c). Figure 9 shows the belief network of enemy's attack. Enemy attacks can come either as a blow-through or an artillery barrage and infantry advance. The blow-through requires sufficient armour and heavy motorized infantry that are supported by self-propelled artillery and regular infantry. On the other hand, a more cautious attack would require artillery and heavy (motorized) infantry. Each node in the belief network associates with conditional probability distributions from observation or experience.

Figure 8(a). A sample battlefield simulator

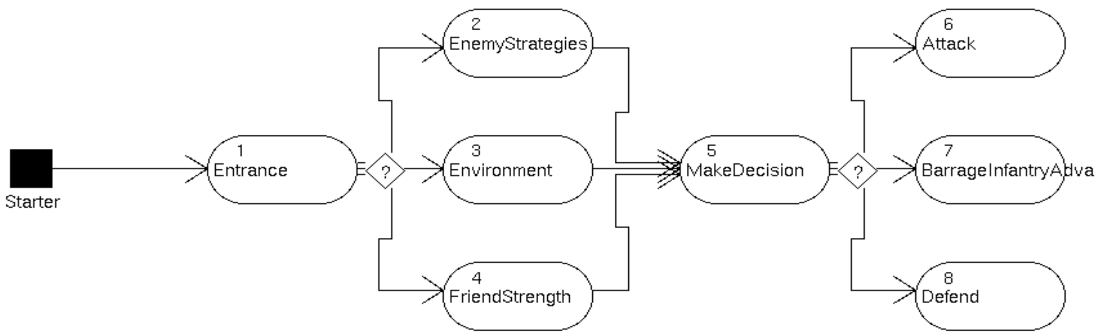

Figure 8(b). Rescue Task Network

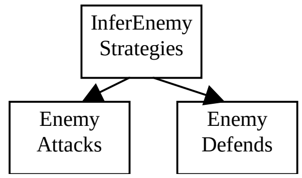

Figure 8 (c). Aspects for Inferring Enemy's Strategies

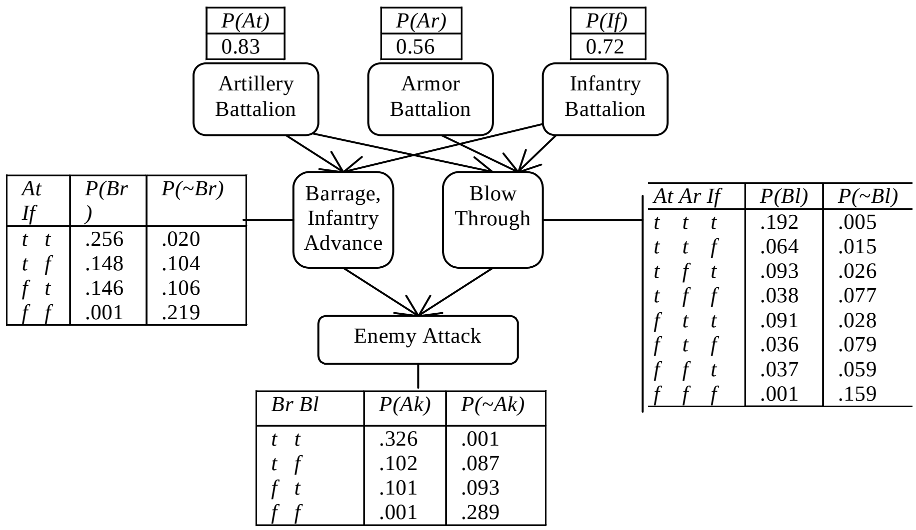

Figure 9. The Belief Network for Enemy's Attack Strategies

The *Aspect* for inferring enemy attack acquires environment information including data about *ArtilleryBattalion*, *ArmorBattalion*, and *InfantryBattalion*. Based on the belief network, the reasoning rules are as follows:

ProbabilityRule Barrage {
[ArtilleryBattalion] &&
[InfantryBattalion]
=>
[BarrageInfantryAdvance] }
ProbabilityRule Blow {
[ArtilleryBattalion] &&
[ArmorBattalion] &&
[InfantryBattalion]
=>
[BlowThrough]
ProbabilityRule Attack {
[BarrageInfantryAdvance] ||
[BlowThrough]
=>
[EnemyAttack] }

This *Aspect* associates with the IPME task "*EnemyStrategies*" and its parent *Aspect* is "Infer*EnemyStrategies*". In the *SenseSet*, there are three external variables, "*ArtilleryBattalion*", "*ArmorBattalion*" and "*InfantryBattalion*", which hold data from the environment through IPME. The screen shot of this *Aspect* is shown in Figure 10. The inference process within this *Aspect* is based on probabilistic computing, rather than true-false confirmation in first order logic.

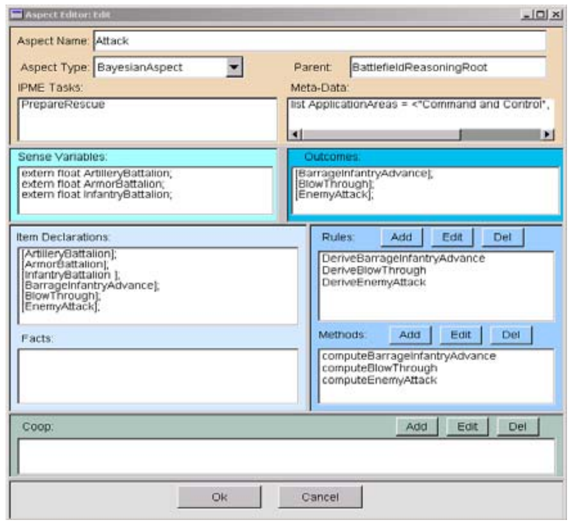

Figure 10. An Aspect for battlefield reasoning

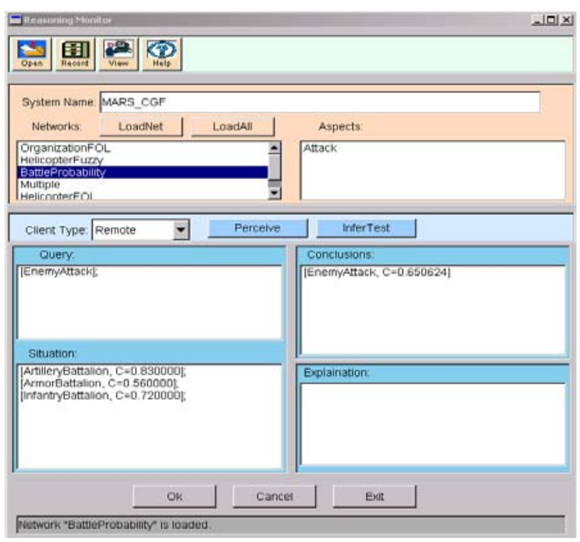

Figure 11. Reasoning result for "EnemyAttack"

When IPME task node "*EnemyStrategies*" invokes the reasoning system for enemy's most possible strategy, the reasoning engine asks IPME for situation data including information on "*ArtilleryBattalion*", "*ArmorBattalion*" and "*InfantryBattalion*". The reasoning engine, then, derives the probability of attack through rules and probabilistic computing. It, ultimately, returns the derived conclusion to the task node in IPME. An example of the reasoning results for the query “*EnemyAttack*” is shown in Figure 11.

4.2 Fuzzy reasoning used for helicopter control

A control system is an electronic or mechanical system that causes the output of the controlled system to automatically remain at some desired output (the “set point”) set by an operator. Most control systems are feedback control systems, as shown in Figure 12. When the desired system output is set and the control system is activated, a control process starts. The control system acquires the current system output and uses it as the input of the error calculator, labelled with “+/-“ in the Figure 12, to compute the error between the current system output and the set point. Based on the error and mechanisms of decision-making, the module labelled “Decision-Maker for Adjustment Amount” computes adjustment amount and sends it to the “Executive Body”. The executive body, then, responds to this adjustment requirement and creates a new system output based on the adjustment amount and system dynamics. Finally, the new system output is sent back to the error calculator, and a new control period starts. Currently, there are a variety of approaches that can be used to design a decision-making module by using control systems such as the PID (Proportional-Integral-Derivative), the fuzzy and neural network paradigms [Buskey et al., 2002].

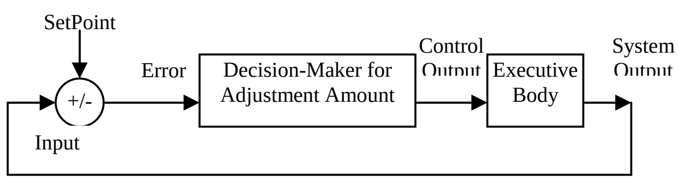

Figure 12. A typical automatic control process

Fuzzy control has some advantages over the traditional PID method. PID control is based on mathematical formulas. Given current input and set point, the PID controller calculates adjustment amount by deterministic proportional, integral and derivative calculations. Although PID control is powerful in many control systems, the tuning of proportional, integral and derivative parameters is tedious work. Furthermore, in most cases, human reasoning does not follow such deterministic mathematical calculations. Humans evaluate input from their surroundings in a fuzzy manner. For example, if the shower water gets too warm, the valve handle is turned to make the temperature go down a little. Here “too warm” and “a little” are vague information. Fuzzy logic uses such uncertain data to make decision. In another word, fuzzy-logic-based control makes use of human common sense, so that it is easier for humans to understand.

Typically, helicopter control is modelled as a task layer and a behaviour layer. In the task layer, the automation control problem is divided into tasks, such as, hover, backward, forward, fly sideways, etc. The behaviour layer, working at a lower level, consists of tasks, such as controls for heading, lateral, and longitude and height. In order to perform a task in the higher layer, one or more behaviour functions have to be activated to make adjustment for the desired output. For example, for hovering, pilots should look for small changes in the helicopter's lateral, longitudinal and height controls. Usually, behaviour functions are independent control modules, and their combinations support the functionality in the task layer.

Our example uses fuzzy reasoning to implement the helicopter's low level behaviour. In this example, a sample helicopter simulator (Figure 13(a)) could be connected to an IPME simulation environment that emulates the pilot's control input; a sample task network is shown in Figure 13(b). Figure 13(c) is the *Aspect* structure for the reasoning used in the task network, which implements the behaviour layer of helicopter control. The nodes in the task network can invoke the corresponding *Aspect*s to obtain the relevant adjustment amounts to control heading, lateral and longitudinal movement and altitude.

Fuzzy-logic-based reasoning can be used for the decision-making of control amount in the feed back control system. The following focuses on the fuzzy decision-maker for adjustment amount in longitudinal cyclic control (shown in Figure 14), which relates to the *Aspect* "*InferLongitudeAdjustment*" in Figure 13 (c). The input of the control system is the current pitch angle. Using this angle, the "*RateCalculator*" computes the *PitchAngleChangeRate* with the previous pitch angle and the time interval. The "*ErrorCalculator*", labelled with "+/-", computes the *PitchAngleError* based on the current pitch angle and the *SetPoint*. With the *PitchAngleChangeRate* and *PitchAngleError*, the *FuzzyDecision-Maker* infers the adjustment amount of the longitudinal cyclic with fuzzy reasoning.

The allowable ranges of pitch angle change rate and pitch angle error are partitioned by fuzzy sets to express the approximate nature of the measurements. They are represented by three fuzzy sets with linguistic labels: "*Negative*", "*Zero*", and "*Positive*", while the conclusion longitudinal cyclic uses five linguistic labels: "*VeryNegative*", "*Negative*", "*Zero*", "*Positive*" and "*VeryPositive*".

Figure 13 (a). A sample helicopter simulator

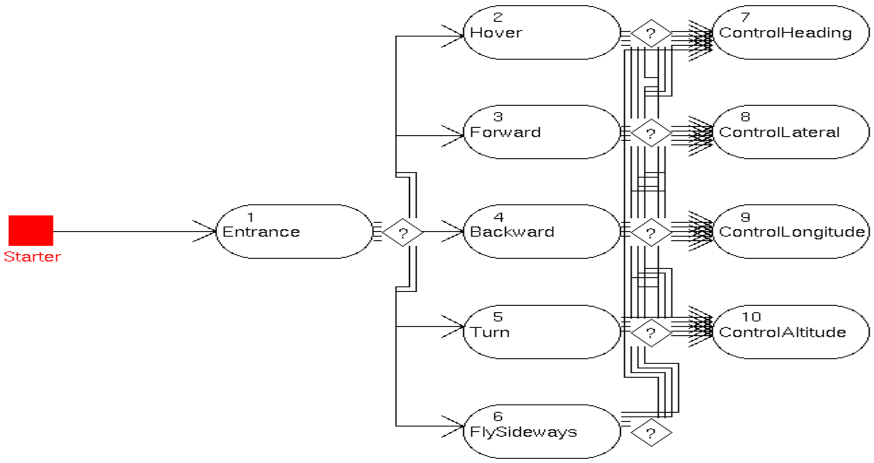

Figure 13(b). The task network for helicopter control

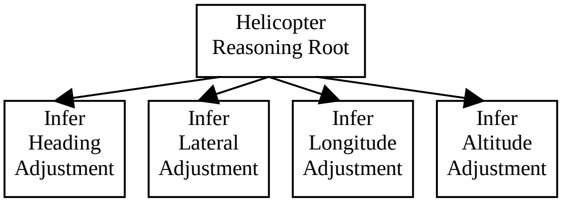

Figure 13(c). The Aspect structure for task network in helicopter control

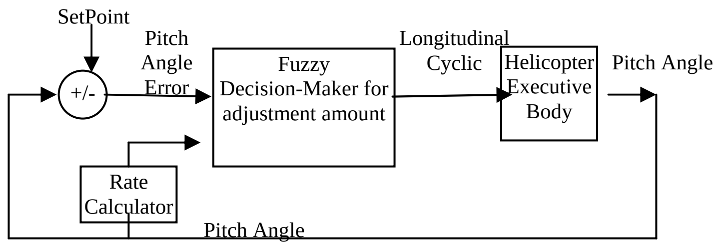

Figure 14. Fuzzy controller for helicopter longitudinal cyclic

The fuzzy membership functions for the longitudinal cyclic control are shown in Figure 15, in which the longitudinal cyclic is the output; the pitch angle change rate, *P*, and pitch angle error, *E*, are the inputs. As it is common in fuzzy logic control systems, the membership functions used to express uncertainty take on triangular or trapezoidal shapes.

The dependence relationships between longitudinal cyclic and pitch angle change rate and pitch angle error are shown in Table 1, which form the reasoning rule set.

When IPME task node "*ControlLongitude*" invokes the reasoning system for the adjustment amount of the longitudinal cyclic, the reasoning engine asks IPME for situation data including "*SetPoint*", "*CurrentPitchAngle*" and "*LastPitchAngle*". It, then, derives the longitudinal cyclic through rules and fuzzy computing. At last, it returns the derived adjustment amount to the task node in IPME for longitudinal cyclic adjustment. The interaction process between IPME task nodes and the fuzzy controller continues until the desired set point is achieved.

The following Figure 16 is the screen shot of the corresponding LAMP *Aspect* that contains three sense variables, four items, nine rules and four methods. Figure 17 shows one of the derived fuzzy conclusions. With *SetPoint* is 0.0, *CurrentPitchAngle* is –3.0 degrees and *LastPitchAngle* is –7.0 degrees, the reasoning result is "[*NumericLongitudinalCyclic, -3.846154, C=1.000000*], i.e., the instant adjustment amount for the longitudinal cyclic is –3.846154 degrees.

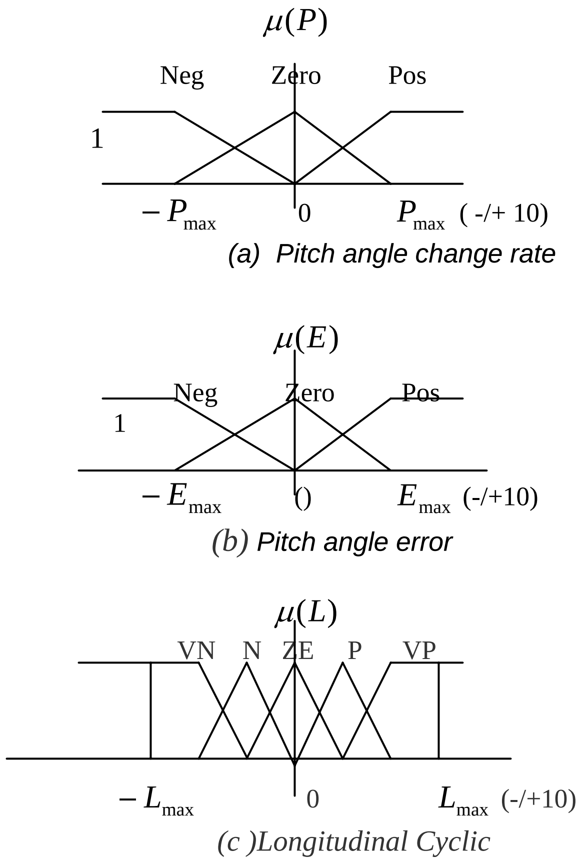

Figure 15. Membership functions for pitch angle change rate, pitch angle error and longitudinal cyclic

Table 1. Longitudinal cyclic based on pitch angle change rate and pitch angle error

ERROR / RATE	NEGATIVE	ZERO	POSITIVE
NEGATIVE	VeryPositive	Positive	Zero
ZERO	Positive	Zero	Negative
POSITIVE	Zero	Negative	VeryNegative

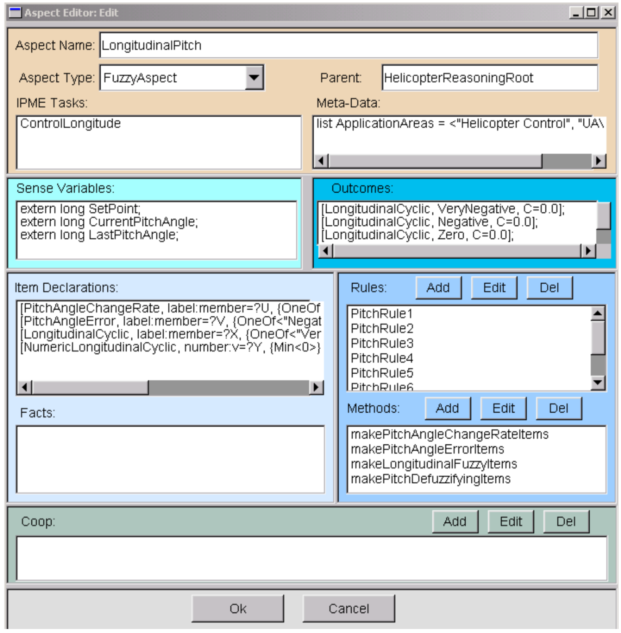

Figure 16. Aspect: LongitudinalPitch

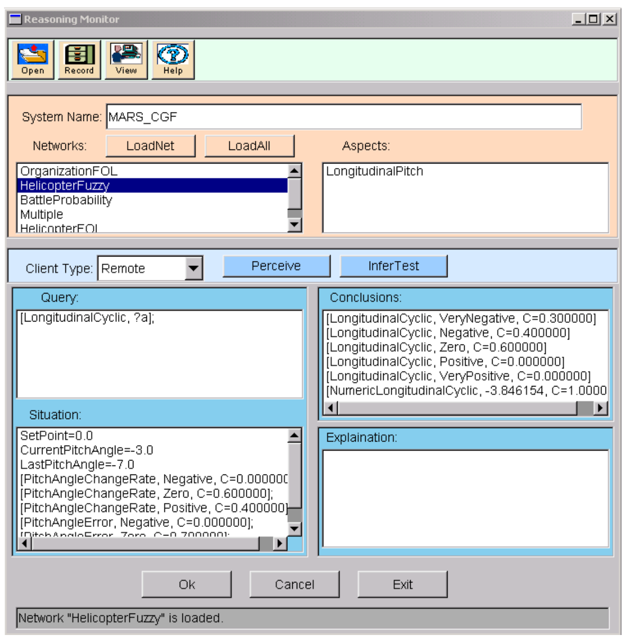

Figure 17. Reasoning results of "LongitudinalPitch"

Figure 18 shows the comparison between fuzzy control and PID control for longitudinal-pitch control (start pitch angle: –7.0; set point: 0.0). The results show that the fuzzy control curve is quite smooth, while the PID control contains large excursions from the desired state. Concretely, if the control process is divided into three phases, P1: 0-2.5 seconds, P2: 2.5-10 seconds, and P3: 10-20 seconds, the fuzzy control is much smoother than the PID control in the first phase. In the second phase, the PID's convergence speed is faster than the fuzzy control. Last, in the third phase, the fuzzy control's convergence process to set point is faster than the PID control.

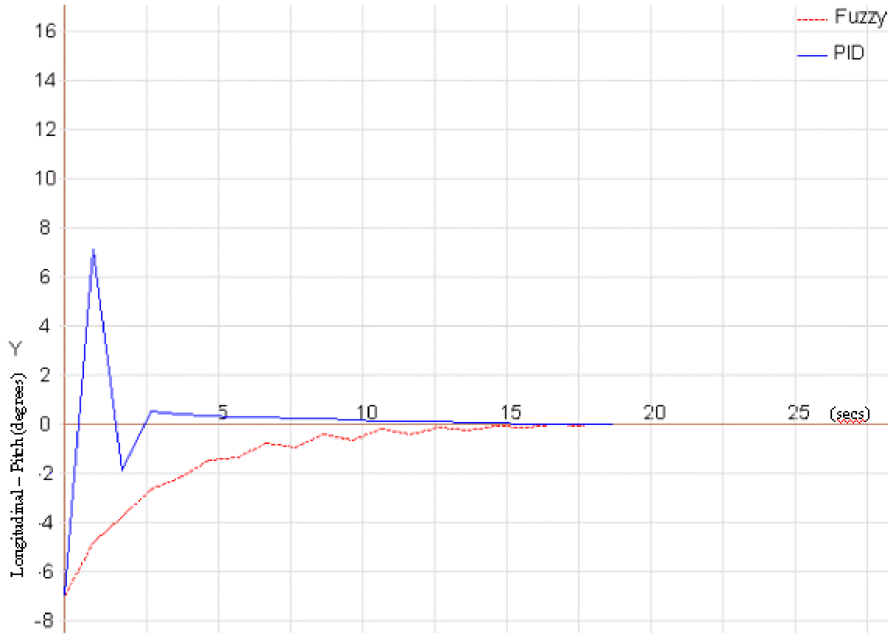

Figure 18. Comparison between Fuzzy Control and PID control

5. Conclusion

A number of tentative conclusions can be drawn based on the results reported in the various sections of this report. These results have also raised issues that may benefit from additional research. Tentative conclusions and ideas for further research, along with a summary of the limitations of the research reported previously, will be presented below.

Initially, with the literature review about uncertainty in military applications, this study found that it is often difficult to get deterministic data and models in many military simulation areas, including operations in command and control, battlefield reasoning, and military aircraft, unmanned vehicles, search and rescue, image recognition and behaviour moderators. Actually, many factors in these areas are vague, incomplete and uncertain, and human beings often assess situation and make decisions based on uncertainty information. In order to support better reasoning and decision-making in CGF, current simulation engines, such as IPME, need to be extended with reasoning mechanisms to support such uncertainty in military simulation and modeling.

The current implementation of LAMP indicated that it is possible to support both uncertainty and deterministic reasoning methods in a unified reasoning architecture. Furthermore, it is also able to integrate multiple uncertainty reasoning approaches for various simulated tasks. Currently, this software system deals with first-order-logic, and fuzzy and probability reasoning.

With the examples in various reasoning approaches, this study provided indication that the reasoning system has the potential to be used in different military simulation areas for decision-making, planning and other reasoning-related tasks. Currently, examples include the areas in battlefield reasoning, helicopter control and personnel relationship reasoning in an organization.

Regarding the integration with existing simulators or simulation engines, this system currently supports the TCP/IP based network communication. This indicated that this system is capable of working together with existing simulators or simulation engines if they sustain the TCP/IP based network communications.

A number of limitations must be considered in examining the results and conclusions reported above. First, owing to the limitation of development resource, the testing of the current software system is just limited to the process of example development. Actually, for a practical and useful software system, more testing is needed, including unit testing and functional testing. Second, the software system is implemented in GNU C/C++ and FLTK under cygwin Linux. Although it can run on Windows through the cygwin environment, it cannot run on Windows directly without the required environment. In order to use this software, the Windows users need to download and install cygwin, GNU C/C++ and FLTK. Third, the fuzzy reasoning is limited to membership functions of triangular shapes, and the probability reasoning supports only conditional probability, though it is easy to add other fuzzy shapes and probability methods.

The current studies also raised many important issues that should be considered in future research.

First, the current representation of first-order logic, fuzzy reasoning and probability reasoning could be extended further. Currently, the first-order-logic-based rules uses simple logic expression, “AND” and “OR”. For an advanced reasoning system, complex logic expressions should be supported. A method library should also be taken into account for various fuzzy reasoning and probability reasoning functions and methods, such as trapezoidal fuzzy functions, Bayesian method and Markov process.

Second, analogical reasoning and hybrid reasoning are also popular human reasoning methods. Future researches might consider supporting such reasoning, such as, case-based reasoning and neural-fuzzy reasoning.

Third, the communication protocol between this reasoning system and simulation engines is a simple one. Future considerations might contain High Level Architecture (HLA) interface and other standard communication protocols for different simulators and simulation engines.

Fourth, as mentioned-above, more efforts are needed for the collaboration component in LAMP. Its goals are to address cooperation between *Aspects*, and handle the impact of behaviour moderators.

Finally, comparing popular AI reasoning methods with human reasoning is a big topic. Future efforts might include building up more applications with various implemented methods and developing a methodology to evaluate effectiveness and performance of the integrated reasoning architecture.

6. References

1. Abi-Zeif, L. and Frost, J.R. (2005). SARPlan: A decision support system for Canadian Search and Rescue Operations. *European Journal of Operational Research*, 162(3):630-653, May 2005.

2. Al-Shihabi, T. and Mourant, R. R. (2001). A Framework for Modelling Human-like Driving Behaviour s for Autonomous Vehicles in Driving Simulators. In *Proceedings of the fifth international conference on autonomous agents*. Montreal, Quebec, Canada. 286-291

3. Asunción, M., Castillo, L., Fernandez-Olivares, J., Garcia-Perez, O., Gonzalez, A., and Palao, F. (2004). Handling fuzzy temporal constraints in a planning environment. *Annals on Operations Research, special issue on "Personnel Scheduling and Planning"*.

4. Bimbo, A.D. and Pala, P. (1997). Shape indexing by structural properties. In *Proceedings of the 1997 International Conference on Multimedia Computing and Systems (ICMCS' 97)* June 03-06, 1997. Ottawa, Canada.

5. Blasch, L.E.P. (1997). Learning attributes for situational awareness in the landing of an autonomous airplane. In *Proceedings of the Digital Avionics Conference*, San Diego, CA, October 1997, pp. 5.3.1-5.3.8.

6. Brynielsson, J. (2004). Game-Theoretic Reasoning and Command and Control. In *Proceedings of the 15th Mini-EURO Conferences: Managing Uncertainty in Decision Support Models*. Coimbra, Portugal, Sept. 22-24, 2004.

7. Burdick, C., Graf, H., Huynh, J., Grell, M., MacQueen, D., Argo, H., Bross, P., Prince, J., Johnson, H., and Blacksten, H.R. (2003). Heterogeneous Agent Operations in JWARS. In *Proceedings of the 2003 Conference on Behaviour Representation in Modelling and Simulation (BRIMS 2003). 03-BRIMS-073*. Scottsdale, Arizona.

8. Buskey, G., Roberts, J. and Wyeth, G. (2002). Online Learning of Autonomous Helicopter Control. In *Proceedings of 2002 Australasian Conference on Robotics and Automation*. Auckland, November 27-29.

9. Cain, B. and Kwantes, P.. (2004). Architectural characteristics for flexible CGF: Integrating HBR. *Internal Technical Communication*.

10. Cao, M. and Hall, E. (1998). Fuzzy Logic Control for an Automated Guided Vehicle. In *Proceedings Of the Intelligent Robots and Computer Vision XVII, SPIE Conference*. Vol. 3522, Nov. 2-3, 1998, Boston, MA.

11. Doyal, J.A. and Brett, B.E. (2003). Enhancing Training Realism by Incorporating High-Fidelity Human Behaviour Representations in the Synthetic Battlespace. In *Proceedings*

of the Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2003. December, Orlando, FL.

12. Ehlert, P.A.M., Mouthaan, Q. M., and Rothkrantz, L.J.M. (2003). A Rule-based and a probabilistic system for situation recognition in a flight simulator. In Mehdi, Q., Gough, N. and Natkin, S. (Eds.) *Proceedings of the 4th Int. Conf. on Intelligent Games and Simulation (GAME-ON 2003)*, London, Great Britain. Pp 201-207. Publisher: Eurosis.

13. El-Nasr, M.S., Yen, J. and Ioerger, T. R. (2000). FLAME – Fuzzy Logic Adaptive Model of Emotions. *Autonomous Agents and Multi-Agent Systems*. Vol. 3, Issue 3. pp 219-257.

14. Fayad, C. and Webb, P. (1999). Optimized Fuzzy Logic Based Algorithm for a Mobile Robot Collision Avoidance in an Unknown Environment. In *Proceedings of the 7th European Congress on Intelligent Techniques & Soft Computing*. Aachen, Germany, September 13-16, 1999.

15. Forgy, C.L. (1982). Rete: A Fast Algorithm for the Many Pattern / Many Object Pattern Match Problem. Artificial Intelligence, 19, 17-37.

16. Gadi, T., Benslimane, R. Daoudi, M. and Matusiak, S. (1999). Fuzzy Similarity Measure for Shape Retrieval. In *Proceedings of the Vision Interface' 99*. Trois-Rivieres, Canada, 19-21, May.

17. Geisler, W.S. and Kersten, D. (2002). Illusions, perception and Bayes. *Nature Neuroscience*. Vol. 5, No. 6. 508-510.

18. Gratch, J. and Hill, R.W. Jr. (1999). Continuous Planning and Collaboration for Command and Control in Joint Synthetic Battlespaces. In *Proceedings of the 8th Conference on Computer Generated Forces and Behaviour al Representation*, Orlando, FL.

19. Guo, R.J., Cain, B. and Meunier, P. (2005). Knowledge Representation Supporting Multiple Reasoning Methods for Simulated Operators. In *Proceedings of the 2005 Conference on Behaviour Representation in Modelling and Simulation (BRIMS 2005).* Universal City, CA, May 16-19.

20. Hamaza, R. and Menon, S. (2001). Advanced Knowledge Management for Helicopter HUMS. In *Proceedings of the DSTO International Conference on Health and Usage Monitoring*. Melbourne, Australia. February.

21. Hill, R.W., Chen, J., Gratch, J., Rosenbloom, P., and Tambe, M. (1997). Intelligent Agents for the Synthetic Battlefield: A Company of Rotary Wing Aircraft. In *Proceedings of the Ninth Conference on the Innovative Applications of Artificial Intelligence*, 1006-1012. Menlo Park, Calif.: AAAI Press.

22. Hoff, B.R. (1996). USMC individual combatants. In *Proceedings of the Defense Modelling and Simulation Office Individual Combatant Workshop*, July 1-2. Alexandria, VA: Defense Modelling and Simulation Office.

23. Howard, A., Tunstel, E., Edwards, D. and Carlson, A. (2001). Enhancing Fuzzy Robot Navigation Systems by Mimicking Human Visual Perception of Natural Terrain Traversability. In *Proceedings of the Joint 9th IFSA World Congress and 20th NAFIPA International Conference*, Vancouver, B.C., Canada, July 2001, pp. 7-12.

24. Ivansson, J. (2002). *Situation Assessment in a Stochastic Environment using Bayesian Networks.* Master thesis of Linkoping University, Sweden. http://www.ep.liu.se/exjobb/isy/2002/3267/exjobb.pdf.

25. Jeram, G.J. (2002). Open Design for Helicopter Active Control Systems. Presented at the *American Helicopter Society (AHS) 58th Forum.* Montreal, June 10-13.

26. Jones, R.M., Laird, J.E., Nielsen, P.E., Coulter, J.J., Kenny, P. and Koss, F.V. (1999). Automated intelligent pilots for combat flight simulation, *AI Magazine*, Vol. 20, No.1, pp 27-41.

27. Kadmiry, B. (2002). *Fuzzy Control for an Unmanned Helicopter.* Thesis of Linkopings Universitet. Linkoping, Sweden.

28. Looney, C.G. and Liang, L.R. (2003). Cognitive situation and threat assessments of ground battlespaces. *Information Fusion.* Vol. 4 (Issue 4, December, pp 297-308.

29. Mengshoel, O.J. and Wilkins, D.C. (1998). Filtering and visualizing uncertain battlefield data using Bayesian networks. In *Proceedings of the ARL Advanced Displays and Interactive Displays Federated Laboratory Second Annual Symposium*. College Park, MD.

30. Moffat, J. and Witty, S. (2002). Bayesian decision making and military command and control. *Journal of the Operational Research Society*. Vol. 53:709-718.

31. Montgomery, J.F. and Bekey, A. (1998). Learning helicopter control through "teaching by showing". *IEEE Conference on Decision and Control*, 4:3467 – 3652.

32. Mulgund, S., Harper, K., Zacharias, G., and Menke, T. (2000). SAMPLE: Situation Awareness Model for Pilot-In-The-Loop Evaluation. In *Proceedings of the 9th Conference on Computer Generated Forces and Behaviour Representation*, Orlando, FL, May.

33. Mulgund, S., Rinkus, G., Illgen, C., and Friskie, J. (1997). OLIPSA: On-Line Intelligent Processor for Situation Assessment. *Second Annual Symposium and Exhibition on Situational Awareness in the Tactical Air Environment*, Patuxent River, MD.

34. Panagiotis, G.Z. and Tzafestas, S.G. (2003). Motion control for mobile robot obstacle avoidance and navigation: a fuzzy logic-based approach. *System Analysis Modelling Simulation.* Vol. 43, Issue 12. pp 1625-1637.

35. Pew, R.W. and Mavor, A.S., (1998). *Modelling Human and Organizational Behaviour : Application to Military Simulations.* The National Academy Press. Washington, D.C.

36. Picard, R.W. (1995). Affective computing. M.I.T. Media Laboratory *Perceptual Computing Section Technical Report* No. 321.

37. Richards, R.A. (2002). Application of Multiple Artificial Intelligence Techniques for an Aircraft Carrier Landing Decision Support Tool. *IEEE 2002 World Congress on Computational Intelligence*. Honolulu, Hawaii, USA. May 12-17.

38. Ritter, F.E. and Avraamides, M.A., (2000). Steps Towards Including Behaviour Moderators in Human Performance Models in Synthetic Environments. *Technical Report* No. ACS 2000-1.

39. Smith, B.R., and Tyler, S.W. (1997). The Design and Application of MIDAS: A Constructive Simulation for Human-System Analysis. Presented at the *2nd Simulation Technology & Training (SIMTECT) Conference*, 17-20 March 97, Canberra, Australia.

40. Stevens, W.K., and L. Parish. (1996). Representation of unit level behaviour in the naval simulation system (NSS). In *Proceedings of the Defense Modelling and Simulation Workshop on Unit Level Behaviour* , August 7-8. Alexandria, VA: Defense Modelling and Simulation Office.

41. Tambe, M., W.L. Johnson, R.M. Jones, F. Koss, J.E. Laird, P.S. Rosenbloom, and K. Schwamb. (1995). Intelligent agents for interactive simulation environments. *AI Magazin*e (Spring): 15-37.

42. Tunstel, E., Howard, A., and Seraji, H. (2002). Rule-Based Reasoning and Neural Network Perception for Safe Off-Road Robot Mobility. Expert Systems: *The International Journal of Knowledge Engineering and Neural Networks*, Vol. 19, No. 4, September 2002, pp. 191-200.

43. Vakas, D., Prince, J. and Blacksten, H.R. (2001). Commander Behaviour and Course of Action Selection in JWARS. In *Proceedings of the 2001 Winter Simulation Conference*. Peters, B.A., Smith, J.S., Medeiros, D.J. & Rohrer, M.N. eds, pp697-705

44. Yu, L. (2003). A Bayesian Network Representation of a Submarine Commander. In *Proceedings of the 2003 Conference on Behaviour Representation in Modelling and Simulation (BRIMS). 03-BRIMS-013*. Scottsdale, Arizona.

45. Zeyada, Y. and Hess, R.A. (2000). Modelling human pilot cue utilization with applications to simulator fidelity assessment. *Journal of Aircraft*. 2000 Jul-Aug; 37(4):588-97

List of symbols/abbreviations/acronyms/initialisms

AI	Artificial Intelligence
CGF	Computer-Generated Forces
CL	Collaboration
DND	Department of National Defence
FLTK	Fast Light Tool Kit
GUI	Graphic User Interface
HBR	Human Behaviour Representation
HLA	High Level Architecture
IPME	Integrated Performance Modelling Environment
LAMP	Language of Agents for Modelling Performance
LM	Long-term Memory
MA	Meta-Attributes
PID	Proportional – Integral – Derivative
SIMON	Simulated Operators for Networks
TCP/IP	Transmission Control Protocol / Internet Protocol
WM	Working Memory

UNCLASSIFIED

DOCUMENT CONTROL DATA (Security classification of the title, body of abstract and indexing annotation must be entered when the overall document is classified)	
1. ORIGINATOR (The name and address of the organization preparing the document, Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's document, or tasking agency, are entered in section 8.) Publishing: DRDC Toronto Performing: DRDC Toronto Monitoring: Contracting:	2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.) UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification is indicated by the appropriate abbreviation (S, C, R, or U) in parenthesis at the end of the title) Supporting uncertainty reasoning in SIMulated Operators for Networks (SIMON) (U) Le support du raisonnement dans l'incertitude pour des opérateurs simulés dans un réseau
4. AUTHORS (First name, middle initial and last name. If military, show rank, e.g. Maj. John E. Doe.) Ruibiao J. Guo; Brad Cain; Pierre Meunier

5. DATE OF PUBLICATION (Month and year of publication of document.)	6a NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.)	6b. NO. OF REFS (Total cited in document.)
December 2005	39	45

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of document, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.) Technical Report
8. SPONSORING ACTIVITY (The names of the department project office or laboratory sponsoring the research and development − include address.) Sponsoring: Tasking:

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant under which the document was written. Please specify whether project or grant.)	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document) DRDC Toronto TR 2005−268	10b. OTHER DOCUMENT NO(s). (Any other numbers under which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (Any limitations on the dissemination of the document, other than those imposed by security classification.) Unlimited distribution
12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11), However, when further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.)) Unlimited announcement

UNCLASSIFIED

UNCLASSIFIED

DOCUMENT CONTROL DATA

(Security classification of the title, body of abstract and indexing annotation must be entered when the overall document is classified)

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

(U) This report introduces a knowledge representation and reasoning scheme that can accommodate uncertainty in simulation of military personnel. The Integrated Performance Modelling Environment (IPME) simulation engine is used with the approach to demonstrate how advanced reasoning models can be integrated with conventional task network modelling to provide greater functionality and flexibility when modelling operator performance. This approach is capable of using multiple reasoning methods, including first−order logic, fuzzy logic and probability reasoning, and supporting attributes inheritance of knowledge.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

(U) HBR, reasoning, human behavioural representation, simulation and modelling

UNCLASSIFIED

Defence R&D Canada

Canada's Leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

DEFENCE DÉFENSE

www.drdc-rddc.gc.ca