A FEDERATED AGENT-BASED CROWD SIMULATION ARCHITECTURE*

Malcolm Yoke Hean LOW
Wentong CAI
Suiping ZHOU
School of Computer Engineering
Nanyang Technological University
Nanyang Avenue, Singapore 639798
E-mail: {yhlow,aswtcai,asspzhou}@ntu.edu.sg

KEYWORDS
Crowd Simulation, Agent-based Simulation, Distributed Simulation, HLA, Ontology, Behaviour Model.

ABSTRACT

Crowd control has become increasingly important in urbanized military operations such as peace keeping, riot control, disaster management, emergency evacuation, and rescue operations. In this paper, we describe an architecture for simulating virtual crowd to aid in the training, planning and decision making process in the area of military operations. Our system makes use of the light-weight agent-based simulation system, RePast, the JESS inference engine coupled with the Protégé ontology knowledge-base, the commercial-off-the-shelf game engine Unreal Tournament and the High Level Architecture to realize a distributed and extensible architecture for modeling virtual crowd.

INTRODUCTION

Crowd control has become increasingly important in urbanized military operations such as peace keeping, riot control, disaster management, emergency evacuation, and rescue operations. However, the lack of an enemy in these operations should not be confused with a lack of adversaries (Heal 2000). Economical, political and cultural factors can sometime cause crowd to turn violent if appropriate rules of engagement are not used. Given the military challenges and risks imposed by the crowds, there is an urgent need to develop a system for military personnel to get prepared for handling various situations, to formulate strategies and answer "what-if" scenarios, and to evaluate hundreds of contingency plans so as to prioritize resources and time during an operation.

One way to do so is to create a synthetic virtual environment and use Modeling & Simulation (M&S) techniques to emulate urbanized military operations. Crowd modeling and simulation is an essential component of such an environment. For such crowd simulation to be useful, the system must support the modeling of realistic individual and crowd behaviours of large number of people. While there are existing work on crowd simulation study using commercial-off-the-shelf simulation packages with built-in agent-based modeling and BDI (Belief-Desire-Intention) behaviour architecture (Shendarkar et al. 2006), modeling detailed complex human behaviours that result from interactions between tens of thousands of individuals will incur high computational cost. This approach is thus infeasible for the generation and evaluation of prompt "what-if" scenarios to handle rapidly evolving crowd situations.

In this paper, we describe our work in designing and implementing a federated agent-based distributed crowd simulation architecture. Our system makes use of the light-weight agent-based simulation system, RePast, the JESS inference engine coupled with the Protégé ontology knowledge-base, the commercial-off-the-shelf game engine Unreal Tournament and the High Level Architecture to realize a distributed, scalable and extensible architecture for modeling virtual crowd. For the rest of the paper, we will describe how these components relate to each other and discuss some of the issues in integrating these components.

RELATED WORK

Crowd simulation is an essential tool used in social studies to analyze group behaviour and norms. Increasingly, it is also being used by defense agencies all around the world to study civil-military scenarios such as riot control and peace keeping mission, as well as emergency situations such as terrorists attack and bomb blast.

Being a highly multi-discipline research areas, existing work in the field of crowd simulation focus on many different areas that include behaviour modeling, visualization, interoperability and scalability. In this section, we briefly review the work carried out by some of the research groups in these areas.

In the area of visualization, the Virtual Reality Lab (VRlab) at the Swiss Federal Institute of Technology

* This project is supported by the Defence Science Technology Agency, Singapore, under project agreement POD0613456.

Proceedings 21st European Conference on Modelling and Simulation
Ivan Zelinka, Zuzana Oplatková, Alessandra Orsoni ©ECMS 2007
ISBN 978-0-9553018-2-7 / ISBN 978-0-9553018-3-4 (CD)

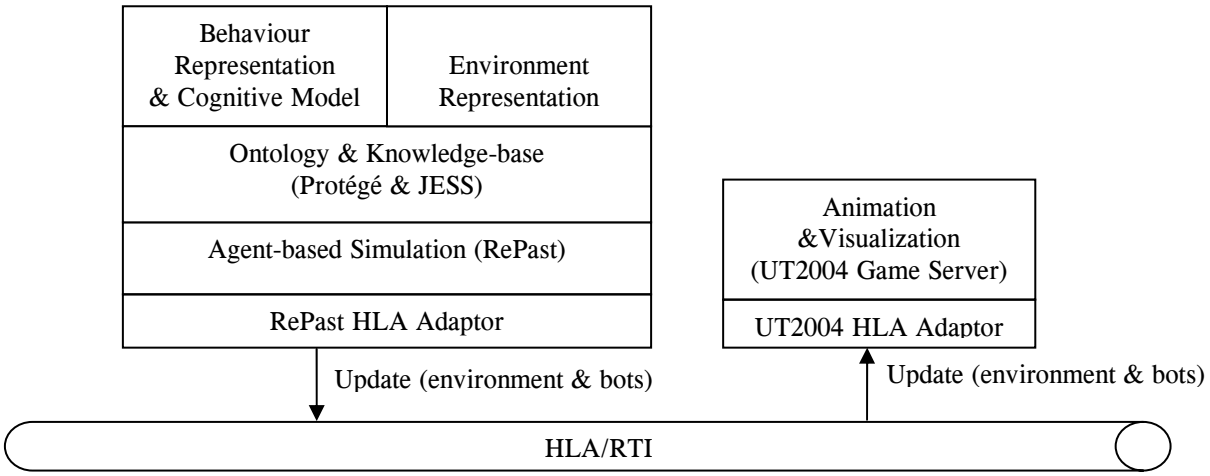

Figure 1: Federated Crowd Modeling Simulation Architecture

(EPFL) is a world leading laboratory in real-time Virtual Humans, multimodal interaction, immersive Virtual Environments, and Augmented Reality. The focus of this research group is on visualization, rendering and animation of virtual crowd (Ciechomski et al. 2005), as well as in the area of behaviour modeling (Thalmann and Monzani 2002).

The Virginia Modeling, Analysis and Simulation Centre (VMASC) focuses on investigating psychologically-based crowd model for military simulation in urban settings. This research group focuses more on the backend engine of crowd simulation, e.g. cognitive model and scenario generation (Nguyen et al. 2005), as well as interoperation between different components of a crowd simulation (e.g. between the behaviour module and the physical simulation module) using the High Level Architecture (HLA) standards (McKenzie et al. 2004). Another focus of the group is on surveying and documenting crowd scenario based on real historical events (e.g. from video clip archive and documentary) so that it can be reproduced and used as a reference scenario in crowd simulation.

With the growing interest in using intelligent agents in computer games and virtual environments, research and development in computer game has increasingly drawn on technologies and techniques originally developed in the large scale distributed simulation community, such as the IEEE High Level Architecture standard (Kuhl et al. 2002) for simulator interoperability to provide solution for interoperation as well as scalability. Distributed simulation allows an existing complex simulation model (e.g. with detailed cognitive behaviour model required for crowd simulation) to be distributed into separate smaller model to improve the execution speed. (Lees et al. 2002) described an HLA-compliant agent toolkit for building cognitively rich agents, and showed that HLA can be used to distribute an existing application with different agents being executed by different federates.

OVERALL ARCHITECTURE

In this section, we give an overall view of the crowd simulation architecture proposed in our project. The subsequent sections will further elaborate on each of the key components of the system. As shown in Figure 1, the system comprises the following five key components:

- *Behaviour Representation and Cognitive Models*
- *Ontology and Knowledge Repositories*
- *Agent-based Simulation Architecture*
- *High Level Architecture*
- *Animation & Visualization Component*

BEHAVIOUR REPRESENTATION AND COGNITIVE MODELS

Existing work on crowd behaviour modeling can be generally classified into the *microscopic* approach and *macroscopic* approach. Most computational models for crowd modeling and simulation adopt the microscopic approach where each individual agent is equipped with a set of decision rules to determine what to do in the next time step (Helbing et al. 2000). The crowd behaviours are then naturally generated as some emergent phenomena due to the interactions of the individual agents. There are two major limitations to this approach. First, it is not computationally efficient, thus it is hard for real-time simulation of a large crowd. Second, there is a gap between the (ad hoc) rules and the results from the social and psychological studies on crowd behaviours.

The macroscopic approach is mainly adopted by the sociology and psychology communities where the crowd is treated as a whole. Although there are rich observations on crowd behaviours, it is still not clear how these observations can be used to construct the computational models for crowd simulations.

In our system, a two-level cognitive model architecture is adopted. The lower level is used to model individual behaviours, and the top level model is used to represent group dynamics and crowd psychology. This two-level architecture is a natural reflection of the interaction amongst individuals, and between an individual and a crowd in real-life situations. A crowd can emerge by the interaction amongst individuals and environmental factors (e.g. a crowd of demonstrators can be formed impromptu or by an organized mobilization effort). Individuals involved in this emerging process may change their behaviours after a crowd is formed. When an individual joins a crowd, the behaviour of the individual in the crowd will be determined by both the group/crowd psychology model and individual behaviour model.

Our two-level cognitive model architecture is a step towards bridging the gap between the *macroscopic* and the *microscopic* approaches. It is a natural reflection of the interaction amongst individuals, and between an individual and a crowd in real-life situations.

In addition, the computational model of crowd behaviour will be based on careful study of the observations from existing cases and psychological/social theories on crowd behaviour. Thus, the resulting crowd behaviour will be more realistic.

In many crowd simulation systems, the crowd behaviours are scripted to allow for minimum or no human control (Musse and Thalmann 2001). These crowd behaviour models lack non-determinism and variety which are essential to human-in-the-loop simulations. These models are suitable for animations such as those used in the movie industry. However, they are not generally suitable for dynamic systems like emergency/crisis management simulations at which we are targeting. Our generic cognitive architecture will allow a user to directly control the crowd behaviour as a whole by generating some events in the environment or indirectly changing the behaviour of the crowd by controlling the behaviour of an individual in the crowd.

ONTOLOGY AND KNOWLEDGE-BASE

"An ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them" (Noy and McGuinness 2001). A knowledge repository captures instances of the concepts and relationships of the ontology and allows for knowledge updating and sharing.

One of the widely used tools for editing and managing ontology is Protégé (http://protege.stanford.edu). Ontologies developed in Protégé can be converted into Java classes and used by agent systems such as JADE (Bellifemine et al. 1999) for knowledge sharing. Instances of the concepts and relationships between concepts can also be stored in a knowledge repository through the Protégé JDBC database back-end. This allows fast and efficient updating and querying of the ontology outside the Protégé environment by different components of the crowd simulation. Ontologies developed in Protégé can also be used together with inference engine such as the Java Expert System Shell (JESS) (Friedman-Hill 2003) for rule-based reasoning (Eriksson 2003) and knowledge acquisition (Lebbink et al. 2002).

Existing work in the use of ontology in crowd simulation are mainly restricted to describing concepts in the environment (Paiva et al. 2005) as well as for path planning purpose (Yersin et al. 2005). There are also existing work in using ontology for 3D modeling and visualization of simulation (Park and Fiskwick 2004). However, each of these works only uses ontology to address a specific part of the modeling, simulation and visualization process. There is a lack of research work in integrating and using the same set of ontology for the entire process of environment representation, cognitive reasoning, simulation, and visualization.

In our system, the Protégé ontology knowledge repository is used with the JESS inference engine to keep track of the environment and the behaviours of individuals in the system. The agent-base simulation will provide updates on changes in both the environment as well as the status of agents and human players. These changes will be updated into the knowledge-base and the JESS engine will modify the behaviours of individuals accordingly based on the cognitive model.

AGENT-BASED SIMULATION ARCHITECTURE

Agent-based simulation system is an ideal choice for crowd modeling and simulation (Nguyen et al. 2005, Musse and Thalmann 2001, Pan et al. 2005). However, one important issue that is often neglected by most of the existing crowd modeling and simulation systems is scalability. For an agent-based simulation, the following operations may need to be carried out in every simulation step: perform reasoning for each agent, execute actions generated, and change agents' states and environment accordingly. A large-scale, interactive crowd simulation may consist of hundreds of virtual participants, represented by agents, and human players. Thus, it may not be possible for a sequential, agent-

based crowd simulation system to meet the real-time requirements of such simulation. The scalable, federated simulation architecture proposed in this project addresses this issue. Although a federated architecture is also proposed in other research work (Nguyen et al. 2005), it only addresses the interoperability and extensibility of the crowd federate. Scalability related problems in federated crowd modeling and simulation such as partitioning of virtual environment, agent state sharing, and agent migration are not investigated.

JADE (Bellifemine et al. 1999) is a Java-based, general-purpose middleware for the development of distributed multi-agent applications based on peer-to-peer architecture. It complies with the FIPA (The Federation for Intelligent Physical Agents) specifications so that JADE agents can interoperate with other FIPA compliant agents. JADE also provides a semantics framework to allow agents to interpret meanings of the exchanged messages according to the formal semantics specified. In addition, it also supports agent life cycle management, agent code and execution state migration, and complex interaction protocols. Although JADE has certain features required by our crowd modeling and simulation (e.g. support for agent communication, migration and semantics), it is not developed specifically for simulation (Tobias and Hofmann 2004). JADE agents are also heavy weight and not suitable for our case where hundreds of agents might be created in each simulation federate.

Swarm (Minar et al. 1996) and RePast (Collier 2003) are multi-agent software platforms for the simulation of complex adaptive systems. Both systems are developed specifically for agent-based simulation and are able to support large number of agents. However, they are not designed to support distributed multi-agent systems, and thus have minimal support for inter-agent communication and no support for agent migration. In addition, both systems also provide very minimal support for simulation model development (Tobias and Hofmann 2004).

In our system, the RePast agent simulation system is chosen to model the environment and the actions of the humans in a user-defined scenario. We adapted the RePast agent simulation system to bridge the gap between general-purpose multi-agent system and agent-based simulation system. RePast already provides a light-weight agent structure so that a large number of agents can be executed within a simulation federate. By implementing an HLA adaptor for RePast, we further added the necessary mechanisms for RePast to support event scheduling and distributed execution. Also, in order to provide support for complex behaviour modeling, the RePast agent simulation system is also linked to the JESS inference engine with behaviour models and behaviour repositories, and a semantics framework based on ontology and knowledge repositories.

HIGH LEVEL ARCHITECTURE

The High Level Architecture (HLA), developed by the US Department of Defense provides the infrastructure needed for large-scale distributed simulation. The HLA defines the rules and specifications to support reusability and interoperability of different simulators (Kuhl et al. 2002). In HLA terminology, a simulation component is referred to as a federate. A federation is then a set of federates working together to achieve a given goal. Each federate interacts with one another over the Runtime Infrastructure (RTI) (DMSO 2002). A set of simulation models developed independently can be put together to form a larger simulation (or federation). Using the HLA, each participating federate in the federation can define the objects and interactions that are shared with others in its simulation object model (SOM), but its internal behavior (and data) is completely invisible to the outside world.

In our current crowd simulation system, the HLA is used to interoperate the RePast federate with the UT2004 visualization federate. For the next phase of our work, we will also be studying the partitioning of the RePast simulation model into multiple federates and synchronizing them using HLA.

We developed an HLA adaptor for the RePast agent simulation, as well as an HLA adaptor for the UT2004 game engine. The RePast HLA adaptor converts events in the RePast simulation (e.g. creation of an individual, changes in the environment, change of movement direction of individual) into HLA object updates. The UT2004 adaptor receives HLA interactions, object creations and updates, and sends commands to the UT2004 game engine to realize the desired visualization.

ANIMATION AND VISUALIZATION

Many research work have been carried out on the generation of human avatars and human-like motions. Creating crowds for complex environment is extremely time-consuming and error-prone. While the ultimate aim of this project is to create fast and efficient visualization techniques that can render a scene based on the RePast simulation in real-time using commercial packages such as Maya, for the initial implementation phase, we choose to leverage on the visualization capabilities of the commercial game engine Unreal Tournament 2004 as the animation and visualization component of our crowd simulation architecture.

The UT2004 HLA adaptor developed in this project is based on the GameBots system, a multi-agent testbed that provides socket-based API to allow software agents

to participate as (software controlled) players in Unreal Tournament games. One of the restrictions of original GameBots system is that each socket connection allows the control of only one player in UT. For visualizing a crowd scenario using UT, it is necessary for hundreds of players to co-exist in the same game. Having a socket connection for each of the player will introduce unnecessary overhead and slow down the game engine.

We adapted the GameBots system so that each socket connection can be used to control multiple players in UT. Figure 2 shows the connection between the UT2004 HLA adaptor with the UT2004 server and the HLA. The RePast simulation will send updates for both the environment and the individual agents as HLA interactions and object updates. The HLA object interactions and updates received will be converted by the UT2004 HLA adaptor into pre-defined GameBots commands and sent to the modified GameBots module. These GameBots commands can be used to initialize or control different players in UT2004. For example, whenever a new player is created/discovered from HLA, a "Create Player *A*" command will be sent to the modified GameBots module to spawn a new player in the game. When player *A*'s position coordinate is updated to "*X*" in the simulation, a "Player *A runto X*" command will be sent. The GameBots command can also be used to effect changes to the environment. For example, when an explosion occurs at location *X* in the RePast simulation, an HLA interaction will be sent to the UT2004 HLA adaptor. This will cause a command "Create Explosion at *X*" to be sent to the UT2004 server, which will invoke a pre-defined UT script to render an explosion effect at location *X*.

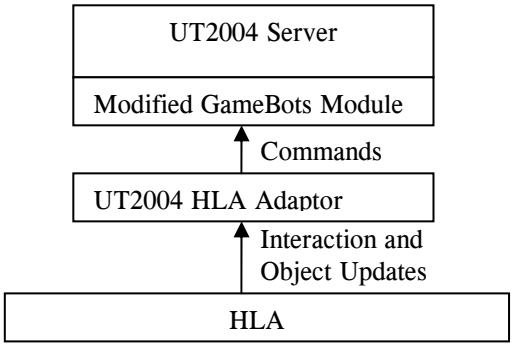

Figure 2: UT2004 HLA Adaptor

PERFORMANCE ANALYSIS - TILEWORLD

The Tileworld is a well established testbed for multi-agent research (Pollack and Ringuette 1990). It comprises an environment consisting of tiles, holes and obstacles, and agents whose goal is to score as many points as possible by pushing tiles to fill the holes. The environment is dynamic: tile holes and obstacles appear and disappear at rates controlled by the user. Tileworld has been used to study commitment strategies (i.e. when an agent should abandon its current goal and replan) and in comparisons of reactive and deliberative agent architectures.

In a crowd scenario, the action of one individual often affects the action or behaviour of many other individuals in the vicinity. For the crowd simulation to run efficiently, the inference engine used must be able to cope with inference rules that may be triggered for many agents. To test the scalability of integrating the RePast simulation with the Protégé/JESS repository, we implemented the Tileworld simulation using RePast and JESS.

The Tileworld environment is laid out in a grid structure with some of the grid cells containing either a tile or a hole. The RePast simulation is responsible for simulating the movement of agents, and a Protégé/JESS repository is used to keep track of changes to the environment as well as the individual behaviour of the agents. The RePast simulation will update the Protégé/JESS repository with the new location or action of an agent, as well as retrieve the new behaviour of the agent. Note that when an agent *A* updates its location or action, the behaviour of agent *A* or other agents in the surrounding might be changed. The JESS inference engine will automatically update the behaviours of the agents affected based on pre-defined inference rules.

Figure 3 shows a simplified finite state machine (FSM) for the behaviour of a Tileworld agent. The FSM is implemented in JESS. Each agent starts with a "Look for Tile" behaviour and carries out random walk looking for a tile in the environment. Suppose each agent has a sensor and is able to detect a tile *R* cells away from it, its behaviour will change to "Move to tile" once it comes within *R* cells of any tile. It will then proceed straight for the tile it detected.

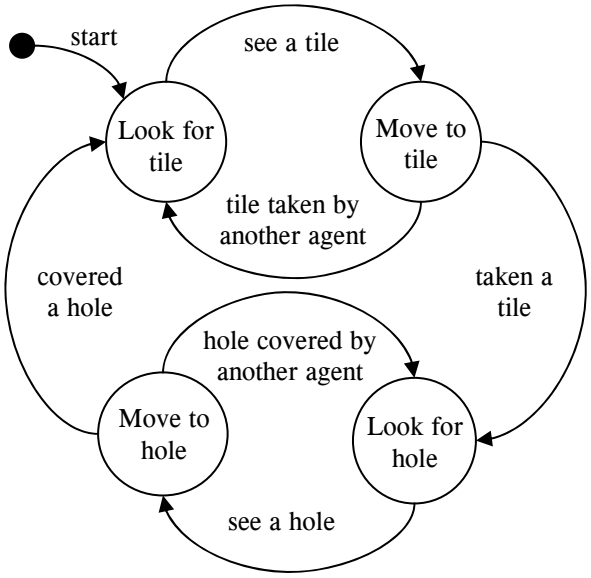

Figure 3: Finite State Machine for the Behaviour of Tileworld Agent

However, before the agent can reach the tile, the tile might have already been picked up by another agent. The JESS inference engine has to update the behaviour of those agents in the surrounding that are heading towards this tile back to "Look for tile". The similar case is true after an agent successfully covered a hole. The inference engine must update the behaviour of those agents moving towards the hole to "Look for hole".

We note that as the sensor range *R* increases, the action of an agent (in picking up a tile or covering a hole) potentially affects many other agents. We carried out an experiment to measure the effect of increasing *R* to the execution time of JESS rules. Figure 4 shows the execution times of calling JESS rules for the agent action *TakeTile()* and *CoverHole()* for different sensor ranges *R*. Our experimental results show that as *R* increases, the time to execute the JESS inference rules increases as well. This is due to the fact that the JESS inference rules have to be fired for more agent instances with larger *R*. For a simulation model with large crowd and complex cognitive model, having one simulation federate may result in long turn around time for each JESS rule evaluation. The simulation model has to be partitioned into multiple federates to ensure responsive operation.

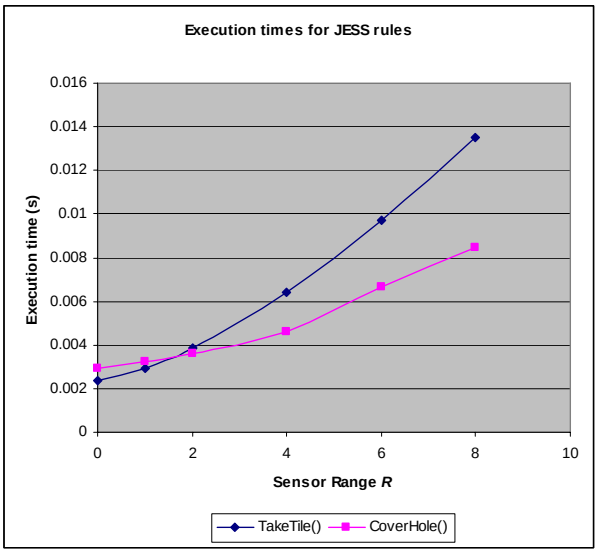

Figure 4: Execution Times of JESS Rules for Different Sensor Ranges

CONCLUSION

In this paper, we have described a federated architecture for crowd simulation. Our approach incorporates cognitive reasoning based on the JESS engine and uses the HLA distributed simulation architecture to achieve interoperability between the simulation and visualization components of the simulation.

In the next phase of our project, we will focus on the issues of partitioning the simulation into multiple federates and study how to synchronize and maintain consistency across these federates. We will also define a realistic civil-military operation scenario that will allow us to study crowd behaviours under different circumstances.

Existing work on crowd simulation typically uses a static environmental model and focuses on the interaction between individual entities and groups in the model. While these systems can be applied to short to medium term planning as well as training of personnel, they cannot be used in day-to-day operating conditions whereby the environment and state of the individual entities and groups are constantly evolving.

We will refine our proposed crowd simulation system to bridge this gap by augmenting the simulation system with interfaces for symbiotic simulation support. This will allow the simulation model to be updated based on real-time data from the knowledge repositories. It will also allow prompt what-if analyses to be carried out and any corrective actions to be quickly propagated to the physical system.

ACKNOWLEDGEMENTS

The authors would like to thank Hu Nan, Nguyen Ngoc Nam, Sean Ng Boon Kiat and Seet Yew Siang for carrying out the preliminary implementation of the crowd simulation architecture described in this paper.

REFERENCES

Bellifemine, A. P. F., A. Poggi and G. Rimassa. 1999. JADE - A FIPA-Compliant Agent Framework. In *Proceedings of Practical Applications of Intelligent Agents and Multi-agent Systems (PAAM'99)*. 97-108.

Ciechomski P., S. Schertenleib, J. Maïm, D. Maupu, and D. Thalmann. 2005. Real-time Shader Rendering for Crowds in Virtual Heritage. In *Proceedings of the 6th International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage*. 91-98.

Collier, N. 2003. RePast: An Extensible Framework for Agent Simulation. Working Paper, Social Science Research Computing, University of Chicago, USA.

DMSO. 2002. RTI 1.3-Next Generation Programmer's Guide Version 5, DoD, DMSO.

Eriksson, H. 2003. Using JessTab to Integrate Protégé and Jess. IEEE Intelligent Systems. 18(2):43-50.

Friedman-Hill, E. 2003. Jess in Action, Manning Publications Co. 480.

Heal, Sid. 2000. Crowds, Mobs and Nonlethal Weapons. Military Review. 80:45-50.

Helbing, D., I. Fakas, and T. Viesek. 2000. Simulating Dynamics Feature of Escape Panic. *Nature*. 407:487-490.

Kaminka G. A., M. M. Veloso, S. Schaffer, C. Solitto, R. Adobbati, A. N. Marshell, A. Scholer, S. Tejada. 2002. GameBots: A Flexible Test Bed for Multiagent Team Research. *Communications of the ACM*. 45(1):43-45.

Kuhl, F., R. Weatherly, and J. Dahmann. 1999. Creating Computer Simulation Systems: An Introduction to the High Level Architecture, Prentice Hall PTR.

Lebbink, H. J., C. L. M. Witteman and J.-J. C. Meyer. 2002. Ontology-Based Knowledge Acquisition for Knowledge Systems. In *Proceedings of the 14th Dutch-Belgian Artificial Intelligence Conference (BNAIC'02)*. 195-202.

Lees, M., B. Logan, G.K. Theodoropoulos. 2006. Agents, games and HLA. *Simulation Modelling Practice and Theory*. 14:752–767.

McKenzie, F.D., Q. Xu. Q. H. Nguyen and M. D. Petty. 2004. Crowd Federate Architecture and API Design. In *Proceedings of the Fall 2004 Simulation Interoperability Workshop (SIW)*. No. 04F-SIW-084.

Minar, N., R. Burkhart, C. Langton, and M. Askenazi. 1996. The Swarm Simulation System: A Toolkit for Building Multi-agent Simulations, Working Paper 96-06-042, Santa Fe Institute, Santa Fe, USA.

Musse, S. R. and D. Thalmann. 2001. Hierarchical Model for Real Time Simulation of Virtual Human Crowds, *IEEE Transactions on Visualization and Computer Graphics*, 7(2):152-164.

Nguyen, Q.H., F.D. McKenzie, and M.D. Petty. 2005. Crowd Behavior Cognitive Model Architecture Design. In *Proceedings of the 2005 Conference on Behavior Representation in Modeling and Simulation (BRIMS)*. 55-64.

Noy, N.F. and D. L. McGuinness. 2001. Ontology Development 101: A Guide to Creating Your First Ontology, In Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880..

Paiva, D. C., R. Vieira, and S. R. Musse. 2005. Ontology-based crowd simulation for normal life situations. In *Proceedings of Computer Graphics International 2005 (CGI'05)*. 221-226.

Pan, X., C. S. Han, and K. H. Law. 2005. A Multi-agent Based Simulation Framework for the Study of Human and Social Behaviour in Egress Analysis, In *Proceedings of International Conference in Civil Engineering*.

Park, M. and P. A. Fishwick. 2004. An Integrated Environment Blending Dynamic and Geometry Models. In *Proceedings of 2004 AI, Simulation and Planning In High Autonomy Systems*. 574-584.

Petty, M. D., F. D. McKenzie, R. C. Gaskins, and E. W. Weisel. 2004. Developing a Crowd Federate for Military Simulation. In *Proceedings of the Spring 2004 Simulation Interoperability Workshop (SIW)*. Arlington VA, USA.

Pollack, M.E. and M. Ringuette. 1990. Introducing the Tileworld: Experimentally evaluating agent architectures. In *National Conference on Artificial Intelligence*. 183-189.

Shendarkar, A., K. Vasudevan, S. Lee and Y.-J. Son. 2006. Crowd Simulation for Emergency Response using BDI Agent based on Virtual Reality In *Proceedings of the 2006 Winter Simulation Conference*. 545-553.

Thalmann, D. and J.-S. Monzani. 2002. Behavioural Animation of Virtual Humans: What Kind of Law and Rules? In *Proceedings of Computer Animation 2002*. IEEE CS Press. 154-163.

Tobias, R. and C. Hofmann. 2004. Evaluation of Free Java-libraries for Social-scientific Agent-based Simulation. *Journal of Artificial Societies and Social Simulation*. 7(1).

Yersin, B., J. Maïm, P. D. H. Ciechomski, S. Schertenleib, and D. Thalmann. 2005. Steering a Virtual Crowd Based on a Semantically Augmented Navigation Graph. In *Proceedings of First International Workshop on Crowd Simulation (V-CROWDS'05)*.

AUTHOR BIOGRAPHIES

MALCOLM YOKE HEAN LOW is an Assistant Professor with the School of Computer Engineering at the Nanyang Technological University (NTU), Singapore. Prior to this, he was with the Singapore Institute of Manufacturing Technology, Singapore (SIMTech). He received his bachelor and master degrees in applied science in computer engineering from NTU in 1997 and 1999, respectively. In 2002, he received his DPhil degree in computer science from Oxford University. His current research interests are in the application of parallel/distributed simulation, grid computing, and agent technology for the modeling, simulation, analysis, and optimization of complex systems.

WENTONG CAI is an Associate Professor with the School of Computer Engineering at Nanyang Technological University (NTU), Singapore, and the head of the Computer Science Division. He received his BSc in computer science from Nankai University (China) and PhD, also in computer science, from the University of Exeter (UK). He was a postdoctoral research fellow at Queen's University (Canada) before joining NTU in February 1993. He has been actively involved in the research in parallel and distributed computing for more than 10 years and has published more than 100 research papers in this area. His current research interests include parallel and distributed simulation and cluster and grid computing.

SUIPING ZHOU is currently an Assistant Professor in the School of Computer Engineering at Nanyang Technological University (NTU), Singapore. Previously, he was a postdoctoral fellow at Weizmann Institute of Science (Israel). He received his B.Eng. (1989), M.Eng. (1992) and Ph.D (1996) in Electrical Engineering from Beijing University of Aeronautics and Astronautics (P.R. China). His current research interests include: distributed interactive applications, parallel/distributed systems, and human behavior representations for virtual training systems. He has published more than 40 technical papers in international journals and conferences in these areas.