A Framework of Intelligent Environment with Smart-Active Objects (IESAO) for Flexible and Efficient Crowd Simulation

Anson Yuanxi Liang, Malcolm Yoke Hean Low, Michael Harold Lees, Wentong Cai, and Suiping Zhou
Parallel and Distributed Computing Centre
School of Computer Engineering
Nanyang Technological University
Singapore, 639798
Email: {yxliang, yhlow, mhlees, aswtcai, asspzhou}@ntu.edu.sg

Keywords: Agent-based simulation, intelligent environment, smart and active object, human behavior modeling, crowd simulation

Abstract
Agent-based crowd simulation has been widely applied in the analysis of evacuation safety under disastrous and terrorist circumstances. In crowd simulation, the virtual environment plays an important role in influencing human behavior and defining the scenario. The concepts of informed environment and smart objects have been adopted to improve the realism of the simulation by embedding semantic information in the environment. However, there is no formal approach of using these concepts in crowd simulation. In this paper, we propose a flexible and efficient framework of Intelligent Environment with Smart and Active Objects (IESAO). We properly define different types of entities and interactions, and describe the approach of modeling environment-related behavior in the intelligent environment. This framework gives a new perspective of modeling human behavior in the design phase. An implementation and a case study are discussed to show the flexibility and efficiency of this framework.

1. INTRODUCTION
Crowd simulation has become an important tool used in the military and civil-defense domain to analyze group behavior and evacuation safety under disastrous and terrorist circumstances. Agent-based modeling is one of the approaches for crowd simulation [1]. In this approach, the agent behavior model is the research focus and different techniques have been applied for the modeling of individual and group behavior in the crowd [2].

A virtual environment is the base of the simulation that defines the scenario and plays a significant role in influencing agent’s behavior. The Informed Environment [3] is one of the attempts to provide semantic information in the environment so as to simulate more realistic human behavior. Moreover, Smart Object [4, 5] is another technique used to embed human behavioral logic into the environment.

Traditionally, researchers focus more on the agent’s behavior model than the virtual environment model. However, many of the agent behavior are directly related to the environment. Our objective in this paper is to describe a framework which is able to model intelligent environments. Smart and Active objects are used in this framework as a mechanism by which the environment can direct agent’s behavior. The framework provides a new perspective in the modeling of environment-related behavior of agents. As illustrated in Section 3, under this approach, agents do not need to have scenario and environment specific logic and hence may be deployed to different environments and scenarios. Moreover, by transferring reasoning from the agents to the objects we can gain execution efficiency when there are more agents than objects. The remainder of this paper is organized as follows. Section 2 gives an overview of related work in agent-based crowd simulation, the informed environment, and smart objects. Section 3 describes the IESAO framework and discusses related issues. In Section 4, the implementation of the framework is described, while a case study is discussed in Section 5. Finally, Section 6 gives conclusions and future work.

2. RELATED WORK
2.1. Agent-based Crowd Simulation
In agent-based simulation, each human is modeled as an agent which is able to act autonomously. Under the sense-think-act paradigm, the agent perceives the surrounding environment including the surrounding agents, then decides the actions to be performed in order to achieve the goal, and finally performs the actions accordingly [6]. For the modeling of behavior and decision-making process, Finite-State Machine (FSM) model is usually adopted. Different states are defined within the agent, and decisions can be made by examining the conditions of each state, and then behavior are performed in the transition between states [2].

In recent research in crowd simulation, a generic framework based on experience and knowledge base was designed by combining the essentials of the sense-think-act paradigm and the FSM model [7, 8]. The emphasis of this
approach is the role of human experiences in decision making. In summary, the modeling of human behavior solely in the agent is the most common approach adopted for crowd simulation.

2.2. Informed Environment and Smart Objects

The Informed Environment created in [3] has urban embedded knowledge in association with human perception and analysis in the context of urban life. This information attached to the environment is meant for the simulation of virtual humans as it provides all details to direct virtual humans with coherent behavior.

An Informed Environment may consist of a set of Environment Entities. An Environment Entity (ENV) represents a surface or volume and has associated semantic information. For example, Mobile Entities such as pedestrians and cars may use the information embedded in the surfaces (ENV) for calculating displacement. Furthermore, one ENV can be composed of different kinds of objects such as objects associated with obstacles, and smart objects used for specific interactions.

Smart Objects mentioned in the Informed Environment provides some action knowledge and states some human interaction specifications mainly for animation purposes. In the game “The Sims” [4], a Smart Object broadcasts its functionality (also known as advertising data) to actors in the game, and stores a list of possible actions and what purposes they fulfill. Moreover, the concept of smart object is also adopted in [5] for the animation of low-level manipulation actions.

However, there is no unique and formal definition for smart objects and this term has various meanings in different applications. Smart objects are mainly used for the modeling of detailed interactions. In the next section, we will introduce a framework with smart-active objects that combines the advantages of the traditional agent behavior model, the informed environment, and smart objects.

3. THE IESAO FRAMEWORK

3.1. Definition of Entities

In traditional agent-based simulation, the agent is usually defined as an "autonomous entity which observes and acts upon an environment and directs its activity towards achieving goals" [9]. For crowd simulation from a modeling perspective, it can be helpful to distinguish two types of entities: the crowd agent and the environmental object.

3.1.1. Crowd Agent

A crowd agent refers to an individually situated, dynamic entity in the crowd that has some type of human control (e.g. human, car, etc.). It is able to observe the surrounding environment, make decisions and behave autonomously in order to achieve its goals. Moreover, it not only has the ability to change its behavior upon the changes in the environment, but also has the ability to communicate and interact with other crowd agents and environmental objects in the model and change their attributes if necessary.

Different types of people in the crowd, e.g. civilians, police and etc. are modeled as crowd agents. However, for other human-controlled entities, e.g. a car driven by human, we need to examine the role of the entity in the simulation in order to decide whether to model it as a crowd agent or an environmental object. If an entity is defined as part of the environment and is dependent on the scenario, it makes sense to model it as an environmental object. For instance, if in a scenario that a car is just an element in the background environment, then the car should be modeled as an environmental object. However, if in a scenario that several cars are moving together with the crowd and are not part of the background environment, then the car should be considered as a crowd agent.

3.1.2. Environmental Objects

Environmental objects are entities defined as part of the environment and are typically static and scenario-dependent. For instance, different static entities in a building, e.g. entrance, exit and etc. are modeled as environmental objects. However, for some cases, the environmental objects can also be moveable, for example, the car in the background as we mentioned in section 3.1.1.

We then further classify the environmental objects according to two properties, Intelligence and Initiative. Intelligence refers to the reasoning ability of an environmental object to change its attributes and status. Initiative refers to the ability of an environmental object to initiate an interaction with the crowd agent.

First of all, based on the property of intelligence, we classify environmental objects into two types, Smart Objects and Normal Objects. A smart object is able to observe the surrounding environment and take inputs from the environment, and update its attributes according to the changes in the environment, i.e., has some reasoning power. It has the reasoning ability in order to perform the updates by itself. On the other hand, a normal object is not able to change its attributes autonomously regardless of any changes in the environment.

Secondly, based on the property of initiative, we classify environmental objects into Active Objects and Passive Objects. An active object is able to initiate interaction with crowd agents and other environmental objects by providing information and changing their attributes actively. Conversely, passive objects are not able to either interact with crowd agents or change their attributes or status actively.

In summary, based on the two properties we defined, we have four types of environmental object in total, which
are smart-active objects, smart-passive objects, normal-active objects, and normal-passive objects. Table 1 shows their properties and differences. Specifically, the smart-active object is similar to the crowd agent functionally in the sense that both of them are able to observe the surrounding environment and act autonomously. Let us consider modeling an entrance as these four types of environmental object. Firstly, if the entrance is modeled as a normal-passive object, then the entrance is only observed by the crowd agents, and it is the crowd agent that retrieves the position of the entrance. Secondly, if the entrance is a normal-active object, then it can actively tell the surrounding crowd agents its position. Thirdly, if the entrance is a smart-passive object, then it can change its status, for example, between “open” and “close” according to certain logic. Lastly, if the entrance is modeled as a smart-active object, then it can update its status and tell the surrounding crowd agents about its changes if any.

Table 1. Categories of Environmental Objects

<table>
<thead>
<tr>
<th>Active</th>
<th>Passive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart</td>
<td></td>
</tr>
<tr>
<td>1. Can observe the surrounding environment</td>
<td></td>
</tr>
<tr>
<td>2. Has reasoning ability</td>
<td></td>
</tr>
<tr>
<td>3. Can change both its attributes and other entities’ attributes</td>
<td></td>
</tr>
<tr>
<td>4. Actively interacts with other entities, especially crowd agents</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>1. Can observe the surrounding environment</td>
<td></td>
</tr>
<tr>
<td>2. No reasoning ability</td>
<td></td>
</tr>
<tr>
<td>3. Can only change other entities’ attributes</td>
<td></td>
</tr>
<tr>
<td>4. Actively interacts with other entities, especially crowd agents</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Normal-Active</th>
<th>Normal-Passive</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Cannot observe the surrounding environment</td>
<td></td>
</tr>
<tr>
<td>2. No reasoning ability</td>
<td></td>
</tr>
<tr>
<td>3. Cannot change its or others attributes</td>
<td></td>
</tr>
<tr>
<td>4. The interactions with crowd agents are initiated from crowd agent’s side</td>
<td></td>
</tr>
</tbody>
</table>

3.2. Intelligent Environment

In our framework, the intelligent environment consists of different types of environmental objects and is designed to have two intelligent capabilities based on smart objects and active objects. Firstly, the intelligent environment is able to autonomously self-update its status and attributes by having smart objects. Secondly, it is able to interact with crowd agents and provide information and guidance actively by having active object.

The first capability of an intelligent environment allows modelers to model a dynamic and adaptive virtual environment while the scenario-dependent logic can be coded into smart objects. The environment is no longer static and is self-updateable upon the changes in the crowd agents or according to certain pre-defined logic. Moreover, it provides an easy way to model the dynamic entities in the environment, for instance, the dynamic LCD signboard.

The second capability of an intelligent environment is that it enables a new approach to model environment-related behavior and navigation plans of the crowd agents. This new approach embeds the environment-related behavioral logic of the crowd agents into smart and active objects, so that they guide the crowd agents to navigate in the environment and achieve their goals. Under this approach, crowd agents may not have scenario and environment specific logic and hence may be deployed to different environments and scenarios.

3.3. Interactions between Intelligent Environment and Crowd Agents

Figure 1. Illustration of Different Interactions between Intelligent Environment and Crowd Agent

The traditional categorization of interaction in multi-agent simulation (MAS) as communication and regular action is designed based on the viewpoint of the agent, and hence, it is not sufficient to support the IEASO framework. We need to also consider from the viewpoint of the...
intelligent environment, and further define the interaction between the agent and the environment based on two properties of the interaction.

First of all, based on the direction of the interaction, there are two types of interaction between the intelligent environment and crowd agents, one is from the crowd agent to the passive object, and another is from the active object to the crowd agent. The first type of interaction is the same as the traditional approach that agents retrieve environmental information through dynamic sensing. The second type of interaction is the opposite that the information is injected into the agents.

Secondly, based on the contents of the interaction, we further define two categories of interaction, which are informative interactions and modifying interactions. Informative interactions only involve information passing, while modifying interactions only involve states changing. These two types of interaction are not mutual exclusive, and they can happen at the same time between an environmental object and a crowd agent. Figure 1 illustrates different types of interactions and examples between intelligent environment and crowd agents involving active and passive objects.

3.4. Modeling Human Behavior through Intelligent Environment

Based on the interactions between active objects and crowd agents, we are able to have a new approach to model human behavior by placing the environment-related behavioral logic into smart and active objects. In order to understand the new approach better, we first clarify the meaning of environment-related behavior. All the behavior that requires information associated with the environment in order to be performed is considered as environment-related behavior. In our framework for crowd simulation, there are two basic types of behavioral interaction, one is between crowd agents, and another is between a crowd agent and an environmental object. All interactions between crowd agents and environmental objects are considered environment-related behavior since they require the information of the environment in order to be performed. For instance, “walking to the exit” and “using the ticket machine” are environment-related behavior, because they require the positions of the exit and ticket machine which are the information associated with the environment. On the other hand, the interactions between agents do not require the information from the environment so that they are not considered as environment-related behavior. For instance, “walking closely in a group” and “helping injured people” are not environment-related behavior because they require only information from crowd agents themselves but not from environmental objects.

In the traditional approach, if the crowd agent is modeled using the sense-think-act paradigm, then the agent

![Figure 2. Illustration of FSM-Based Modeling and Intelligent-Environment-Based Modeling of Environment-Related Behavior of Agents](image-url)
first perceives the surrounding environment, retrieves the information about the environmental objects, decides the actions to be taken, and then performs the actions. The interactions between crowd agents and passive objects reflect part of the sensing process of the agent in the traditional approach. The navigation plan of the agent is environment-related, and can be considered as a FSM with states of environment-related behavior. In order to model the navigation plan and other environment-related behavior in different environments, we have to implement different FSMs for each environment accordingly.

In the new approach, the process of sensing surrounding environmental objects by each crowd agent is replaced by the interactions with active objects so that the agent does not need to sense the environment. Moreover, smart objects are able to host certain environment-related decision making logic which is originally designed in a crowd agent. Hence, the navigation plan of agent can be embedded into the environment. The states of environment-related behavior of the original FSM are now modeled inside smart and active objects. As a result, the agents with only non-environment-related behavior can be deployed into different environments while the environment-related behavior and the navigation plan are handled by smart and active objects in the environment.

Figure 2 illustrates the differences between the FSM-based modeling and the intelligent-environment-based modeling of environment-related behavior of agents. In this simple example, there are three states in the FSM model, and each state is associated with an object in the environment. The agent examines the logic in each state and performs associated actions accordingly. For instance, an agent around Door B needs to first examine whether it has passed the door, and then decide the next action whether to pass the door or go to Door C. In the intelligent-environment-based model, the navigation plan is now embedded in the environment, and each active object directs the agent to achieve its goal according to the status of the agent. In the same situation as the previous example, the agent around Door B does not need to perform any sensing and decision making, but Door B which is modeled as the smart-active object will examine the status of the agent, and then direct the agent to perform the correct actions accordingly.

The new approach of modeling environment-related behavior in the intelligent environment gives four advantages over the traditional approach. Firstly, it allows the possibility of modeling an independent and reusable agent model that can be deployed to different environments, as long as the environment-related behavior are modeled inside smart and active objects.

Secondly, it may reduce the complexity of modeling the environment-related behavior of agents in the design phase. If the environment is complex and has many objects that influence the behavior of the agents, designing the FSM of the agent for the environment-related behavior model may be difficult, because the modeler needs to think from the point of view of the agent and plan the scenario as a whole by defining different states and the relationship between states. On the other hand, if we model the environment-related behavior using smart and active objects, then the modeler’s point of view is based on the interactions between the object and the agent. This point of view for modeling environment-related behavior may simplify the design phase since each active object may represent a state naturally and the modeler does not need to think in terms of the concept of the FSM paradigm.

Thirdly, by transferring reasoning from the crowd agents to the environmental objects, it may reduce the execution overhead of the same reasoning process on the assumption that there are more agents than objects in the environment. For instance, let us consider the scenario that there are a number of entrances, and agents will choose the less crowded one to pass. In the traditional approach, each agent must calculate the density at each entrance. When the number of agents is large, the total overhead spent on the density calculation is considerable large since it is repeated by every agent in the same time step. In the IESAO approach, each entrance can be modeled as a smart-active object. Each entrance only needs to calculate the density once, and then pass the information to the agents around. Therefore, the number of times of calculating the density can be reduced. In the new approach, we can gain execution efficiency by reducing the execution of the same reasoning process.

Lastly, using the new approach in multi-agent simulation (MAS) may reduce the memory usage during run time if the number of agent is far more than the number of objects in the environment, and an agent is designed in the way that each agent carries an instance of its own FSM. When the number of agent is large, and the FSM is complex in term of memory usage, then it will require a lot of memory during run time. In the new approach, since the environment-related behavior is modeled in the intelligent environment and there is always only one instance of the whole environment in the simulation, the memory consumption can be minimized. This has the same benefit as using a shared FSM instance for all agents.

4. IMPLEMENTATION

4.1. Rule Engine

In order to model the reasoning logic inside smart and active objects, we use a rule engine called Drools [10] which is a platform originally designed for business logic integration and allows users to write logic in an easy-understandable manner. Since in general the logic inside smart and active objects is simple and can be written in i-
then conditional expressions, using the rule engine would make the job of programming the logic easier for modelers. Moreover, the rules are written in an external file which will automatically be compiled during run time, so that modelers are able to change the configurations of smart and active objects by changing the rules but without recompiling the whole application. The rule engine of Drools contains three core modules, the production memory, the working memory, and the inference engine that implements the Rete algorithm [11, 12]. In our case, the production memory contains the rules which are the logic to be performed by smart and active objects. The working memory contains the facts which are the entities including all crowd agents and different kinds of environmental objects in the simulation. The inference engine matches the rules and the facts so that it applies the logic to the entities according to the rules specified. Figure 3 illustrates the architecture of Drools rule engine in our case.

![Figure 3. Architecture of Drools Rule Engine](image)

4.2. Architecture

The architecture for modeling the intelligent environment with smart and active objects is designed based on the hierarchy of entities in the framework. Figure 4 shows the class diagram for the architecture. First of all, every entity is extended from FactObject which is an instance of fact that is compatible to Drools rule engine. Secondly, we define CrowdAgent and EnvironmentalObject under FactObject for modeling different types of entities. Under EnvironmentalObject there are four sub-classes. Except Normal_Passive_Object, all the other three sub-classes include a shared instance of RuleEngine which is a wrapper of Drools rule engine and for handling the associated logics.

During run time, each smart and active object first retrieves the facts in the surrounding environment, for example, crowd agents and environmental objects around it. Then it inserts the facts into the rule engine and process the logic accordingly. Finally, it removes all facts from the rule engine so that the engine is ready for used by other smart and active objects.

![Figure 4. Class Diagram for Crowd Agent and Environmental Object](image)

5. CASE STUDY

5.1. Scenario and Model

For case study, we defined a scenario in a multi-level subway station under standard conditions (i.e., not emergency). The whole environment consists of three levels, the entrance level (L1), and the platform levels (B1 and B2). Figure 5 shows the maps of the three levels in the subway station. In the subway station, we define different types of entities including entrance, ticket machine, passenger center, fare gate, escalator, train, train door, and signboard. There are two types of crowd agents, with two different plans, going into the station and leaving from the station.

For the first plan, the agent who intends to take train enters the environment at the entrance of the station, and first buys a ticket at the ticket machine, and looks for assistance at the passenger center if necessary. Then it passes the fare gate and proceeds to the platform level through the escalator. Finally, it waits besides the train door until the train arrives. Once the train arrives the agent enters the train at which point that agent leaves the environment. For the second plan, the agent who has got off the train enters the environment at the train door, and then goes to L1 through the escalator. Then it passes the fare gate and exits the station at the entrance. In both plans, signboards provide information that helps the agents navigate through the station.

Since all entities in the environment have direct influence on agent’s behavior, in this case study, we modeled all environmental objects as active objects, and some of them are also modeled as smart-active objects. The crowd agents are kept very simple in this example to illustrate how the smart and active objects can be used to generate complex behavior. All the environmental behavior is handled by smart and active objects in the environment.
Table 2 shows the description of each environmental object in the model including the type and the brief descriptions and their actions. These are the basic logics for modeling the environmental objects which reflect the plans of the agents and the fundamental interactions with the agents. If the scenario is modeled using a FSM, the modeler would need to define many states with various conditions for the agent, and also the associated relationships between states. Compared to the traditional approach, the IESAO framework provides a new perspective for modeler by building the scenarios based on the environmental objects.

Based on the maps and the description of each environmental object, we generated a model of the environment from an application that was developed for environment specification in our project. Then we defined the rules for each smart and active object, and modeled crowd agent and the interactions between them accordingly.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Descriptions and Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entrance</td>
<td>Normal-active</td>
<td>* Directs agents to enter or leave from the station.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* Removes agents whose goal is leaving station from the simulation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* Generates new agents into the simulation.</td>
</tr>
<tr>
<td>Ticket machine</td>
<td>Smart-active</td>
<td>* Directs agents to buy ticket.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* Updates its status between “available” and “occupied”.</td>
</tr>
<tr>
<td>Train door</td>
<td>Smart-active</td>
<td>* Directs agents to wait for the train.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* Directs agents to get into or off the train.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* Updates its status between “closed” and “opened”.</td>
</tr>
<tr>
<td>Signboard</td>
<td>Smart-active</td>
<td>* Directs agents to the destination according to the agent’s status.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* Changes its instructions according to the situations in the environment.</td>
</tr>
</tbody>
</table>

5.2. Observations and Measurements

The simulation was setup and run on an Intel Core2 computer. Based on the non-congested situation where agents are able to move around the station, we configured the model such that the total number of agents in the simulation can reach a steady state between 100 and 550.

Firstly, we observed that the agents behaved as expected and were all able to complete their plans. We did not find any irregular movement or behavior on the agents due to the usage of active objects, such as the livelock movement between two objects. Figure 6 shows an example of the outcomes of using the smart-active object. Since the signboard is modeled by using smart-active object, it has the ability to give instructions to the agents based on the situations in the environment. In this example, both fare gate A and B are entrance to the station, but gate B is less crowded than gate A. Therefore, the signboards are aware of the situation and direct the incoming agents to the less crowded gate B.

In addition, Figure 7 shows a scenario where the ticket machine which is modeled by using smart-active object instructs the agent to wait while it is occupied by another agent. Initially, the agents move toward to the ticket machine as shown in the left screenshot of Figure 7. However, the machine is occupied by another agent,
therefore, the ticket machine instructs the agent to wait until the current agent left.

Secondly, we run the simulation multiple times and collected the elapse time per simulation step, while the total number of agents in the simulation remains constant between 120 and 530. Figure 8 shows that the elapse time per simulation step increases linearly with the increase of the number of agents.

Figure 6. An Example of Signboard Directing Agents to the Less Crowded Fare Gate

Figure 7. An Example of Ticket Machine Directing Agents to Wait

Figure 8. Average Elapse Time per Simulation Step

6. CONCLUSION AND FUTURE WORK

In this paper, we describe a framework of intelligent environment with smart-active objects for crowd simulation. We define two categories of entities in simulation as crowd agents and environmental objects. Under environmental objects, we further classify them according to two properties, Intelligence and Initiative. Hence, we have smart and active objects which enable the modeling of intelligent environment. Based on smart and active objects, we are able to model a dynamic and adaptive environment, and to model environment-related behavior in a new approach respectively. The new approach of modeling environment-related behavior using smart and active objects has two major advantages compared to the traditional approach. Firstly, it allows the reuse of agent model, that is, the same agent model that can be deployed to different environments. Secondly, it can simplify the design process of modeling environment-related behavior. In summary, the IESAO framework provides some advantages in terms of flexibility and efficiency from the modeling perspective. It is suitable for scenario-based simulation especially when agent’s behavior is tightly coupled to the scenario and environment.

Currently the case study in this paper only shows how the IESAO framework can replace a FSM-based model while the crowd agent model is very simple. However, in practice the crowd agent model may be far more complex than the one used in the case study. Hence, future work will integrate the intelligent environment with a complex agent model (e.g., the agent model under the experience and knowledge based cognitive framework [7, 8]). In addition, we will further test the efficiency of using the intelligent environment and the scalability of the implementation.

Acknowledgments

This research is supported under Singapore’s Defense Science and Technology Agency (DSTA) grant POD0613456 and is a joint work with Technical Support Working Group (TSWG).

References


Biography

ANSON YUANXI LIANG is currently a Ph.D. candidate at University of North Carolina at Chapel Hill, USA. Prior to this he was a Project Officer in the Parallel and Distributed Computing Centre at Nanyang Technological University (NTU) of Singapore. He received a B.Eng. in Computer Engineering with First-Class Honors from NTU in 2007. His current research interests include computer vision and image analysis, parallel and distributed simulation, and multi-agent system.

MALCOLM YOKE HEAN LOW is an Assistant Professor in 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 of Applied Science in Computer Engineering from NTU in 1997 and 1999 respectively. He was awarded a Gintic (now SIMTech) Postgraduate Scholarship in 1999. In 2002, he received his D.Phil. degree in Computer Science from Oxford University. His current research interest is in the application of parallel and distributed computing for the modeling, simulation, analysis and optimization of complex systems.

MICHAEL LEES is currently an Assistant Professor at Nanyang Technological University (Singapore). He completed his PhD at Nottingham University (UK) in 2006 and went on to post-doctoral positions at Nottingham University and Birmingham University. He received his BSc from Edinburgh University (UK) in Artificial Intelligence and Computer Science. His research interests include: Modelling and simulation, Agent-based systems, simulation validation, large-scale distributed simulation and high-performance computing.

WENTONG CAI is a Professor in the Division of Computer Science, School of Computer Engineering at Nanyang Technological University, Singapore. He is also the Director of the Parallel and Distributed Computing Centre. He received his Ph.D. in Computer Science from University of Exeter (UK) in 1991. His current research interests include: Parallel & Distributed Simulation, Multi-agent Systems, and Grid and Cluster Computing. He is an associate editor of the ACM Transactions on Modeling and Computer Simulation (TOMACS), and editorial board member of the Multiagents and Grid Systems – An International Journal.

SUIPING ZHOU is currently an Assistant Professor in the School of Computer Engineering at Nanyang Technological University (Singapore). Previously, he worked as an engineer in Beijing Simulation Center, China Aerospace Corporation, and then joined Weizmann Institute of Science (Israel) as a Post-doctoral fellow. He received his B.Eng., M.Eng. and Ph.D in Electrical Engineering from Beijing University of Aeronautics and Astronautics (P.R. China). He is a member of IEEE and his current research interests include: large-scale distributed interactive applications (e.g., MMOGs), parallel/distributed systems, and human behaviour representation in modeling and simulation. He has published over 70 peer reviewed articles in these areas. He is currently an associate editor of the International Journal of Computer Games Technology. He has served as technical program committee member of many international conferences and workshops in computer games and virtual environments.