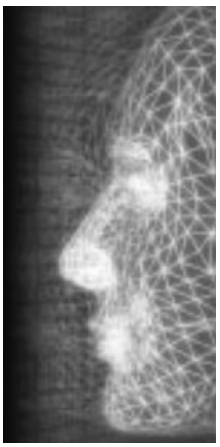
COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2008; **19**: 505–514
Published online 6 August 2008 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/cav.262

Snap: A time critical decision-making framework for MOUT simulations



By Shang-Ping Ting and Suiping Zhou†*

Deliberative reasoning based on the rational analysis of various alternatives often requires too much information and may be too slow in time critical situations. In these situations, humans rely mainly on their intuitions rather than some structured decision-making processes. An important and challenging problem in Military Operations on Urban Terrain (MOUT) simulations is how to generate realistic tactical behaviors for the non-player characters (also known as bots), as these bots often need to make quick decisions in time-critical and uncertain situations. In this paper, we describe our work on Snap, a time critical decision-making framework for the bots in MOUT simulations. The novel features of Snap include case-based reasoning (CBR) and thin slicing. CBR is used to make quick decisions by comparing the current situation with past experience cases. Thin slicing is used to model human's ability to quickly form up situation awareness under uncertain and complex situations using key cues from partial information. To assess the effectiveness of Snap, we have integrated it into Twilight City, a virtual environment for MOUT simulations. Experimental results show that Snap is very effective in generating quick decisions during time critical situations for MOUT simulations. Copyright © 2008 John Wiley & Sons, Ltd.

Received: 23 June 2008; Accepted: 24 June 2008

KEY WORDS: time critical decision making; virtual environments; Military Operations on Urban Terrain (MOUT)

Introduction

> ''Speed is the essence of war. Take advantage of the enemy's unprepared-ness; travel by unexpected routes and strike him where he has taken no precautions.''

The ancient statement of Sun Tzu[1] is especially true for Military Operations on Urban Terrains (MOUT).[2,3] Even modern armies may lose their advantages in superior air power and technologies when faced with street-by-street urban warfare.[4] Going into MOUT operations without extensive preparations may result in substantial casualties.[5] Soldiers often need recognize the current situation with incomplete information and to make rapid decisions under time pressure, uncertainty, high stakes, and changing conditions.[6–9] As pointed out by Klein,[10] human rely much more on experiences rather than deliberative rational analysis of all possible alternatives to make time-critical decisions.

In MOUT simulations, the virtual urban environments are populated by various characters. While some of these characters are controlled by human players (i.e., the trainees), most of them are non-player characters (or bots) which are usually represented by AI-driven agents. For an MOUT simulation to be effective, it is important for these bots to demonstrate some human-like tactical behaviors. Although different approaches may be used to this end, we believe that human-like behaviors should be generated by human-like decision-making processes. In terms of realism of behavior model, our philosophy is that a human behavior model should not only be able to generate seemly realistic behaviors in some given situations, it should also work like a human brain in the sense that the decision-making process of an agent should be similar to that of a human-being. That is, we emphasize on not only the end-result

*Correspondence to: S.-P. Ting, School of Computer Engineering, Nanyang Technological University, Singapore 639798, Singapore. E-mail: ting0021@ntu.edu.sg
†Member, IEEE.

realism of the behavior model, but also the structural and procedural realism.

Therefore, in this paper, we propose a time critical decision-making framework for the bots in MOUT simulations. The novel features of the framework include case-based reasoning (CBR) and thin slicing. CBR is used to make quick decisions by comparing the current situation with past experience cases. CBR is the process of solving new problems based on the solution of similar past problems. It has been argued that CBR is not only a powerful method for machine reasoning, but also a pervasive way for humans to make decisions in everyday situations.[11]

In the proposed framework, thin slicing is used to model human's ability to quickly form up situation awareness under uncertain and complex situations using key cues from partial information. In his book *Blink*, Gladwell[12] discussed how humans are able to achieve rapid situation awareness using only a narrow slice of data. This technique is called thin slicing. Using thin slicing, humans can focus on the most significant information (or key cues) about a situation, and make judgment quickly. As argued in his book, Gladwell suggests that collecting more information merely reinforces our judgment but does not help to make it more accurate. We adopt thin slicing technique to imitate human's rapid situation recognition capability in complex and uncertain situations.

Now let us use a simple example to illustrate how these two features (CBR and thin slicing) of the proposed framework could help a bot to make decisions in time-critical situations in MOUT simulations. Figure 1 shows a soldier bot and a militant bot in a close combat situation.

Figure 1. Close combat.

The soldier bot, Ted, needs to make a rapid decision to fight with the militant. Using the CBR process, Ted will retrieve past experiences that are similar to the current situation and reuse the previous solutions. If needed (i.e., the current situation is new), the proposed solution can be revised to adapt to the current situation. Once the revised solution had been successfully adapted for the new situation, it will be retained in the memory of the Ted for future engagement. The matching of the current situation with past experiences is done by picking up key cues from the partial information gathered within the limited time, i.e., by thin slicing. These cues may include the location of the militant, whether the militant is in open area or hiding behind a wall, the weapon being used and strength of the fire, etc.

The remainder of this paper is organized as follows. The next section describes the related work. The design of the framework is described in ''Design of Snap'' section. ''Integrating Snap into Twilight City'' section describes how the framework is integrated into Twilight City. Experimental results are discussed in ''Experimental Results'' section. ''Conclusions and Future Work'' section concludes the paper.

Related Work

As an interesting and challenging problem, how to generate human-like tactical decisions for the bots in time-critical and stressful situations has attracted many researchers and developers in the military training and game AI communities. As an influential figure, Laird and coworkers[6,13] have done much work in capturing and encoding human variability in human behavior models for military simulations. Their models mainly focus on the deliberative reasoning processes, which we feel are not adequate to model human's decision-making process in time critical situations.

The recognition-primed decision (RPD) model[10] proposed by Klein emphasizes the role of experiences in human's decision-making process in various time critical situations. The model also suggests that human make quick decisions in such situations by matching the current situation with past experiences and selecting solutions to similar situations. Similar ideas were also advocated in Refs. 11,14,15.

In terms of how to represent human's experiences in our proposed behavior modeling framework, we borrow ideas from some social and cognitive psychology

DOI: 10.1002/cav

studies. It has been observed that human beings possess categorical scripts in their memory to interpret and predict the world situations and new information is processed according to how it fits into these scripts called schema.[16] Since these schemas are context specific, they are dependent on an individual's experience with and exposure to a subject area rather than simply some ''raw intelligence.''[17,18] In the propose framework, we explore the concept of schema to organized the experiences of a bot in MOUT simulations.

Gladwell[12] observed that humans often perform thin slicing, that is, they reply on some key cues of a complex and uncertain situation to achieve rapid situation awareness and to make intuitive decisions. He also pointed out that collecting more information may not help to make the intuitive decisions more accurate. Consistent with this idea, based on numerous case studies, Evans emphasized that military commanders must not be overloaded with information during time critical situations.[19] We believe that thin-slicing is an important aspect of human decision-making in time-critical and complex situations. To the best of our knowledge, our current work is the first to incorporate this aspect in modeling human decision-making process for MOUT simulations.

Design of Snap

In this section, we describe Snap, a time critical decision-making framework for MOUT simulations. It aims to generate human-like tactical behaviors for bots in a complex virtual urban warfare environment. The two major features of Snap are the CBR decision-making process and thin slicing.

As shown in Figure 2, the Snap framework consists of five main components: Goal, Observe, Situation Awareness, Experience Repository, and Action. The Goal component defines the goals of the bots. It will affect determine which information is relevant for situation awareness (more details will be given in ''Rapid Situation Recognition with Thin Slicing'' subsection). The Observe component collects information about the environment and sends the collected information to the Situation Awareness component. The components inside the dash line box form the CBR process. By matching the current situation cues with the experience cases in the Experience Repository, a solution will be proposed for execution by the Action component. The Observe component will monitor the situation and update the Action component on the results of the proposed solution. Depending on its result, the proposed solution may be revised and the updated solution

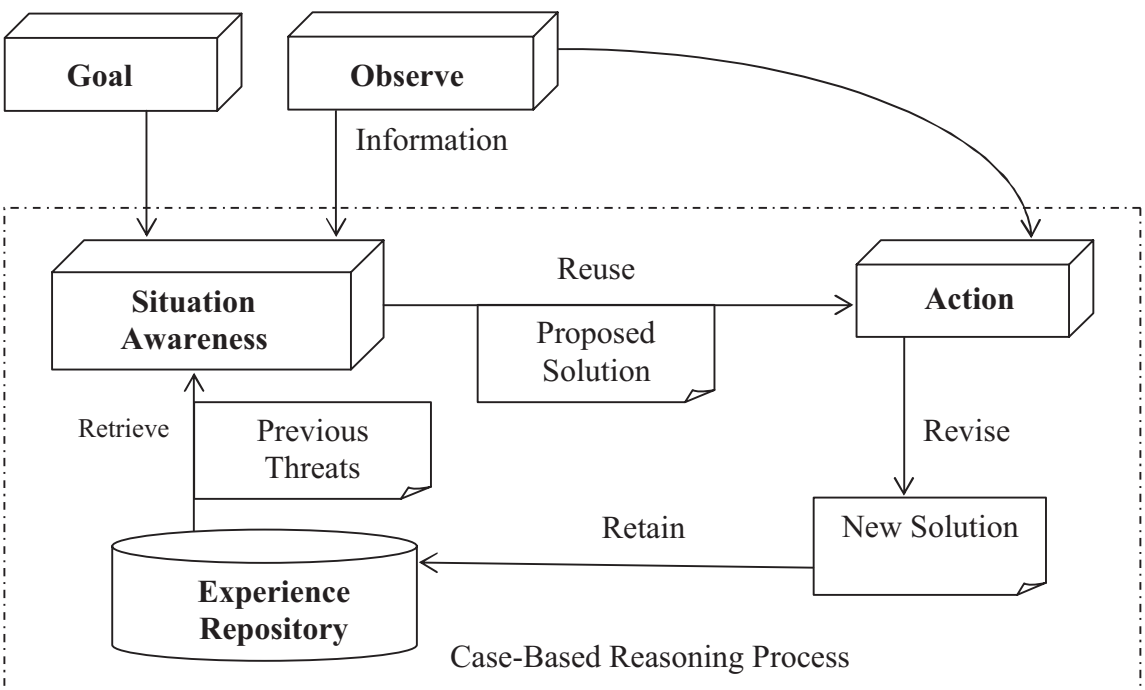


Figure 2. Decision-making framework.

DOI: 10.1002/cav

will be retained in the experience repository as a new experience case for future usage.

CBR Process

The CBR process consists of four steps: retrieve, reuse, revise, and retain. Given the goal and observations from the environment, to form up situation awareness, a bot first needs to retrieve past experiences from its Experience Repository. In our implementation, experience cases are represented by a set of <threat, solution> pairs as show in Figure 3. The threat of an experience case is represented by the precondition cues which act like a pattern (or schema) for the bot to recognize the threat. In Snap, these precondition cues mainly consist of the levels of some qualitative descriptions about the situation, e.g., whether the enemy is close or far away, whether the strength of a sniper's fire is high or low, etc. Qualitative measures are used here simply because humans reply more on qualitative rather than quantitative measures to make sense of a situation. Then, situation awareness is formed by matching the evidence cues of the current situation with the precondition cues. We will further explain this in ''Rapid Situation Recognition with Thin Slicing'' subsection.

When the evidence cues match the precondition cues of an experience case, then the corresponding solution of the case will be reused by the Action component. Each solution is represented by a sequence of actions. Each action is attached with some post-conditions which represent the expected results of the actions. The Observe component monitors the post-conditions while the actions of a solution are executed. If the post-conditions of an action are not met, the next action will not be executed and the proposed solution will be considered as unsuccessful. In this case, the proposed solution needs to be revised. In our current implementation, the revision is done by human experts. These experts will analyze the situation and provide a new solution for the bot. Subsequently, the updated solution will be retained in the Experience Repository as a new case of experience for future use.

Now let us use a simple example to further illustrate the main features of the CBR process of Snap. Suppose that Ted, a soldier bot, is patrolling a street. He is able to identify a sniper-assault threat by observing a militant bot firing a sniper gun. In this case, spotting the militant bot and the sniper fire are the two precondition cues. Thus, the precondition set for sniper-assault threat is:

{Bot(A, Militant, Male, Sniper gun),
Weapon_Fire(Sniper gun)}

Now suppose that Ted's current evidence set (we will explain how it is formed in ''Rapid Situation Recognition with Thin Slicing'' subsection) is:

{Bot(A, Militant, Male, Sniper gun),
Weapon_Fire(Sniper gun)}

The evidence set states that Ted had spotted a militant bot and also observed sniper fire. As the evidence cues match the precondition cues of the sniper-assault threat, the experience case for the sniper-assault threat is retrieved from the Experience Repository. The attached

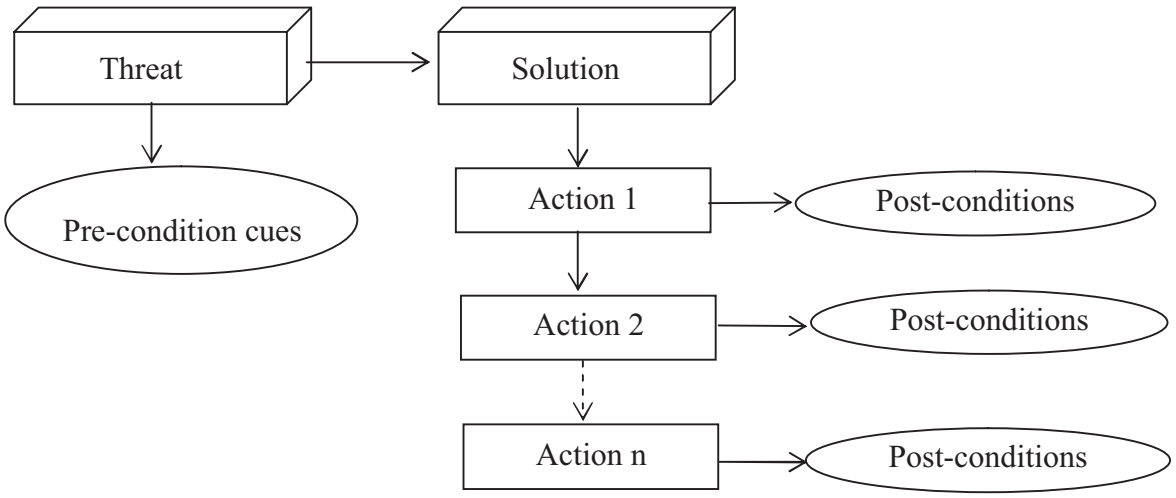


Figure 3. An experience case.

DOI: 10.1002/cav

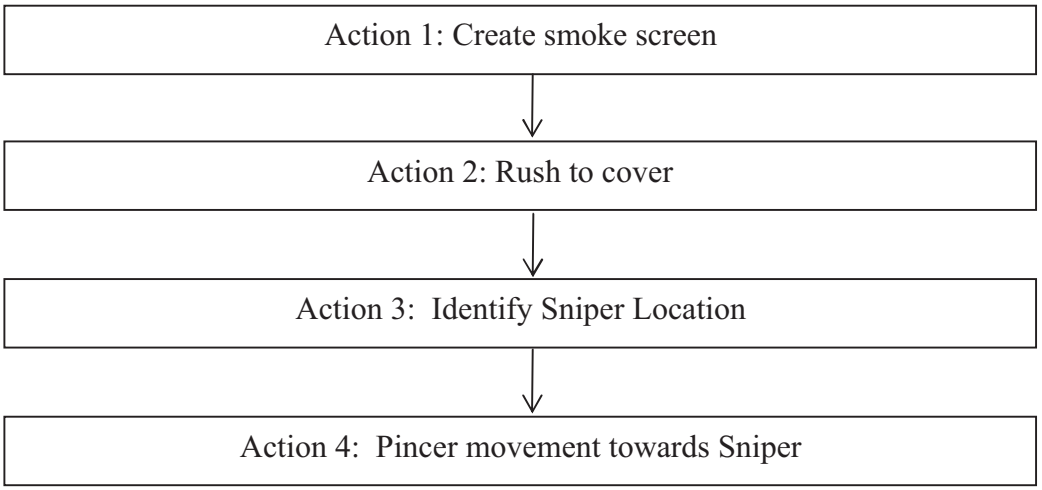


Figure 4. An experience case.

solution will be reused by the Action component, i.e., the sequence of actions contained in the solution will be executed. In our current implementation, the normal tactic to counter the sniper threat consists of the action sequence as shown in Figure 4. Note that with this tactic, we assume that Ted has some other team members to they can approach the sniper with pincer movement.

The first action is to create a smoke screen by throwing a smoke grenade. This will cause the enemy sniper to lose his aiming precision. With the smoke screen, the soldier agents can then rush to a cover before identifying the sniper location. Finally, the soldier agents will approach the sniper with a pincer movement which is a military strategy to attack the flanks of the enemy simultaneously in a pinching motion.[20]

As each action in the sequence is executed, the Observe component of Snap will monitor if the post-condition cues of the action are satisfied. For example, the post-condition set for Action 2 is:

$$\{\neg\text{Weapon\_Fire}(\text{Sniper Fire}) \hat{} \neg\text{Hit}(\text{Bot})\}$$

It states that there should not be sniper fire near the bot and that the bot should not be hit. When the observation cues satisfies the post-condition cues, the action is deemed to be successful and the next action in the sequence will be carried out.

Rapid Situation Recognition with Thin Slicing

The thin slicing technique allows the bots to recognize the current situation quickly based on some key cues from the partial information about the environment. It is achieved by the Goals, Observe, and Situation Awareness components of the Snap framework. First, the Goal component determines the constraint set of a bot according to the mission and roles of the bot. Second, the Observe component monitors the virtual environment and forms the observation set. Finally, Situation Awareness component uses the observation set to detect any violation of the constraint set. The observations that violate the constraints will be used to form the evidence.

For example, consider a soldier bot Ted in a patrolling mission. The constraint set of Ted can be expressed as:

$$\{\neg\text{Weapon\_Fire}(\text{All}), \neg\text{Vehicle}(\text{Militant}), \neg\text{Bots}(\text{Militant})\}$$

This constraint set means that Ted, as a soldier patrolling on the street, will not allow any type of weapon fire, enemy vehicles, and militants to appear. These constraints are expressed with some qualitative cues, which will be used to compare with the observation cues. The Observe component monitors the virtual environment. Suppose the observation set of Ted about the current situation is:

$$\{\text{Bot}(\text{A, Militant, Male, Sniper gun}),$$
$$\text{Weapon\_Fire}(\text{Sniper gun}), \text{Vehicle}(\text{Friendly}),$$
$$\text{BotGroup}(\text{Civilian, 50}), \ldots\}$$

It means that Ted has seen that ''bot A who is a male militant with a sniper gun, the militant is shooting with his gun, there is a friendly vehicle coming by, and a group of civilians are around, and…'' This set of observation cues is used to compare with Ted's constraint set. The comparison reveals that the observations

DOI: 10.1002/cav

''Bot(A, Militant, Male, Sniper gun)'' and ''Weapon_Fire(Sniper gun)'' violate the constraints ¬Bots(Militant) and ¬Weapon_Fire(All), respectively. Thus, the following evidence set is formed:

{Bot(A, Militant, Male, Sniper gun), Weapon_Fire(Sniper gun)}

This evidence set is then used by the Situation Awareness component to match with the precondition cues of Ted's experiences as earlier described in ''CBR Process'' subsection.

Integrating Snap into Twilight City

In this section, we first give a brief overview of Twilight City, a virtual environment that we built for MOUT simulations, and then describe the integration of Snap into Twilight City.

Overview of Twilight City

Twilight City aims to provide a high fidelity simulation platform for MOUT simulations. It is built on top of the Unreal Tournament (UT) engine[21] with various modifications.[22,23] Twilight City simulates urban warfare in an area of approximately 20 km × 20 km. There are more than 30 buildings in the virtual environment. We have made various modifications to the UT engine to enhance the behavior of the bots in Twilight City for MOUT simulation. Figure 5 shows some screenshots of Twilight City.

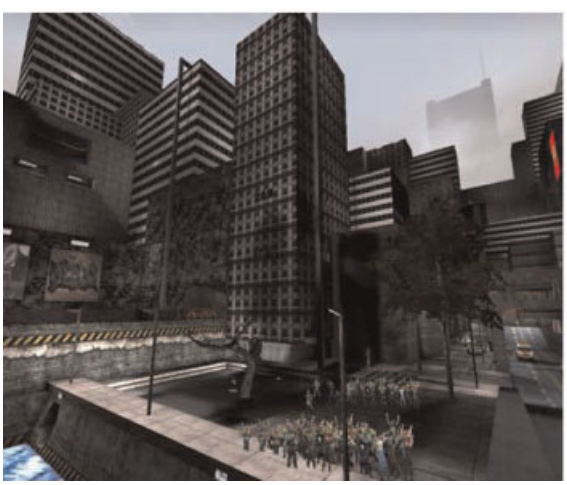


Figure 5. Twilight City.

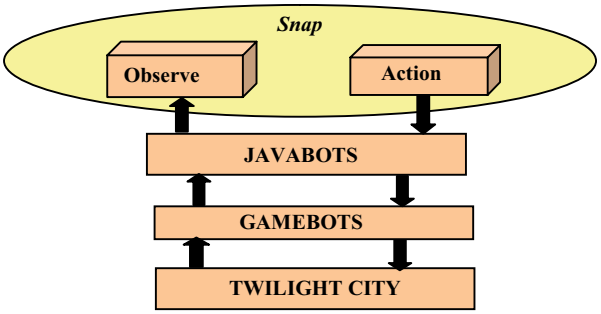


Figure 6. Integrating Snap into Twilight City.

Snap in Twilight City

The existing bots in UT have very limited decision-making skills, which is inadequate to generate realistic tactical behaviors for MOUT simulations. Thus, we use Snap enhance the decision-making capability of the bots. As shown in Figure 6, Snap interacts with the bots in Twilight City via Gamebots.[24] Gamebots is a modification to UT that opens a socket to the Unreal Engine during run time. Snap is Java-based. For the ease of communication and implementation, Snap interacts with the Gamebots via Javabots which is a middleware for external applications to communicate with UT bots via Gamebots.[25]

As illustrated in Figure 6, the Observe component will collect information about the environment for Snap to recognize the current situation and make decisions. The decisions made by Snap will send to the Twilight City through the Action component.

DOI: 10.1002/cav

MOUT situation	Experience case
Sniper Assault	Counter Sniper
Air Strike	Counter Air Strike
Close Combat Fire	Hasty Attack, Retrograde
Ambush	Counter Ambush
Bomb	Counter Bomb
Night Assault	Night Mode

Table I. Experience cases

In Twilight City, there are mainly three types of bots: soldier bots, militant bots, and civilian bots. Typical scenarios in Twilight City include the close street combat between soldiers and militant groups, and the fighting between soldiers and militant sniper, etc. Currently, our focus is the modeling of the tactical behaviors of the soldier bots. As shown in Table 1, seven experience cases for soldier bots have been implemented. These cases reflect the typical situations a soldier may face in urban warfare. The cues and solutions of these experience cases are extracted from various sources including military doctrines and interviews with experts. For instance, the Close Combat Fire situation refers to soldier bots being attacked at close range by enemy bots. This situation requires two experience cases. When the enemy bot is in open area, the Hasty Attack experience case will be retrieved so that the soldier bot will return fire at the enemy. However, when the enemy is firing under concealment, the soldier bot should retrieve the Retrograde experience so that it will quickly move out of the killing zone of the enemy bot.

Experimental Results

To evaluate the performance of Snap, we have conducted some experiments. In this section, we summarize the major results of these experiments. These experiments were conducted on a computer with Intel T2500@2 GHz processor and 2 GHz RAM. The results shown are the average of the results from 10 simulation runs.

The testing scenarios are performed within Twilight City. They include engaging a set of 20 soldier bots with various threats such as Sniper Assaults, Close Combat Fires, Bomb Assaults, and Air Strikes. The objectives of the experiments are to test the behaviors of the soldier bots under different situations and the impact of different experience cases on the soldier's behaviors. In particular, the behaviors of soldier bots during the Sniper Assault threat and the Close Combat Fire threat are compared. Three sets of 20 bots were used for this experiment. They are S(A) bots which only have Counter Sniper Experience, S(B) bots which only have Hasty Attack and Retrograde Experience, and UT bots which have no experiences but are equipped with the default tactics in UT. The average results from 10 runs are shown in Figure 7.

	Close Combat Fire			Sniper Assault		
	S (A)	S(B)	UT	S(A)	S(B)	UT
Stand Around	2.1	2.1	3.1	1.3	3.9	**17.9**
Smoke Screen	0	0	0	**12.1**	0	0
Return Fire	**15.7**	3.9	**16.9**	0	1.0	0
Take Cover	2.2	**14.0**	0	6.6	**15.1**	2.1

S(A) – Bots with Counter Sniper Experience Only

S(B) – Bots with Hasty Attack and Retrograde Experience Only

UT – Unreal Tournament bots

Figure 7. Impact of Quartz on path planning efficiency.

DOI: 10.1002/cav

The results in Figure 7 show how a type of soldier bots reacted to in different tactical situations. For example, the data in the first column show that for the 20 soldier bots of type S(A), under Close Combat Fire situation, 2.1 of them just stood around, none of them made smoke screen, 15.7 of them returned fire immediately and 2.2 of them took cover first. In the Close Combat Fire situation, we let the militant bot to shoot at the soldier bot from behind the walls. For experienced human soldiers, the natural reaction is to take cover first. In this situation, returning fire at the militant will only expose the soldiers to unnecessary danger as there is no clear line of fire at the militant. It can be observed from the experimental results that the 15.7 of the S(A) bots and 16.9 of the UT bots started returning fire to the militant immediately. 14.0 of the S(B) bots took cover. UT bots returned fire as they are designed to retaliate upon being attacked. The S(A) bots did not have the relevant experience case to handle the Close Combat Fire situation, so they had to rely on the basic reaction of retaliation with fire. However, these behaviors are unrealistic in this situation. The S(B) bots, on the other hand, behaved more like real soldiers. They were able to retrieve the Retrograde experience when there is no clear line of fire to the militant bot.

During the Sniper Assault threat, 12.1 of the S(A) bots created a smoke screen to before moving to a cover, whereas 15.1 of the S(B) bots simply moving to a cover without creating a smoke screen. This is because the S(A) bots are equipped with the Counter Sniper experience, as a result, they were able to act with the correct Counter Sniper tactics. For the 20 UT bots, 17.9 of them just stayed in the open area and searched for enemy, since they could not differentiate between a Close Combat Fire and a Sniper Assault. This behavior is extremely unrealistic as the bots remain exposed to sniper fire.[20] Figure 8a shows a soldier bot being attacked by a sniper. Figure 8b shows the behavior of a soldier bot of type S(A) upon being attacked by the sniper – the soldier bot created a smoke screen to prevent him from being hit easily by the sniper.

Another set of experiments was conducted to compare the mortality rate between Snap bots and normal UT bots during a 20 min simulation. The average results of 10 simulation runs are shown in Figure 9. Again, three types of soldier bots were used for these experiments: Snap(7) bots are soldier bots with all seven experiences in Table 1, and Snap(4) bots are soldier bots with the first four experiences (i.e., Counter Sniper, Counter Air Strike, Hasty Attack, and Retrograde) in Table 1. UT bots are the default bots in UT.

As it can be observed, the mortality rate of the Snap bots is lower than the UT bots. The number of UT bots that remain alive started decreasing in a faster rate than

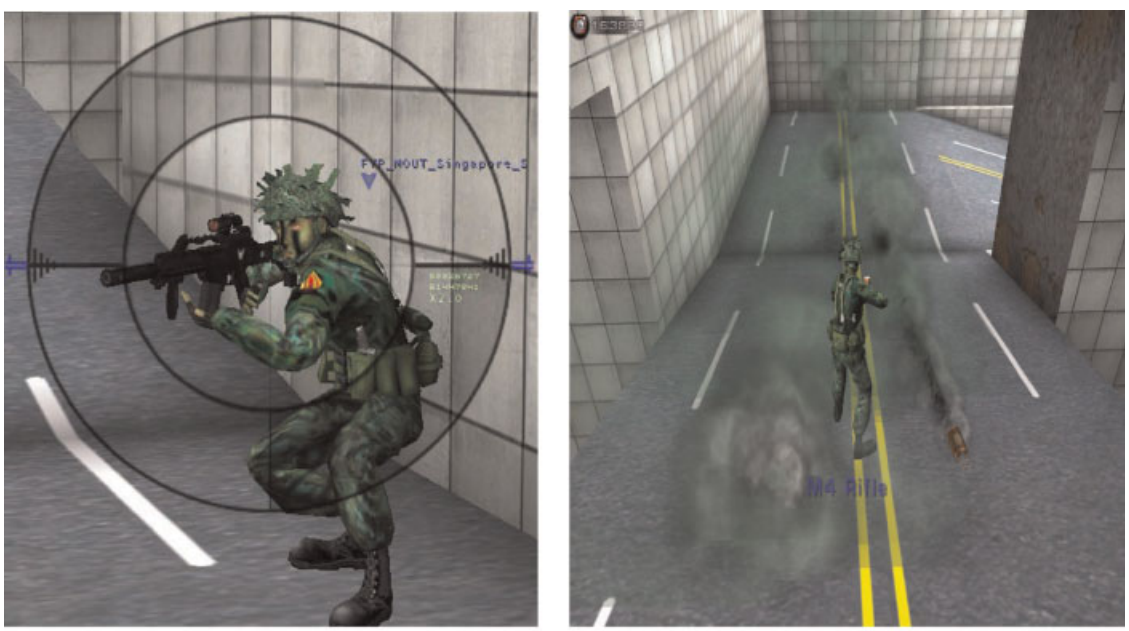


Figure 8. Sniper Assault situation in Twilight City. (a) Sniper Assault and (b) Smoke screen.

DOI: 10.1002/cav

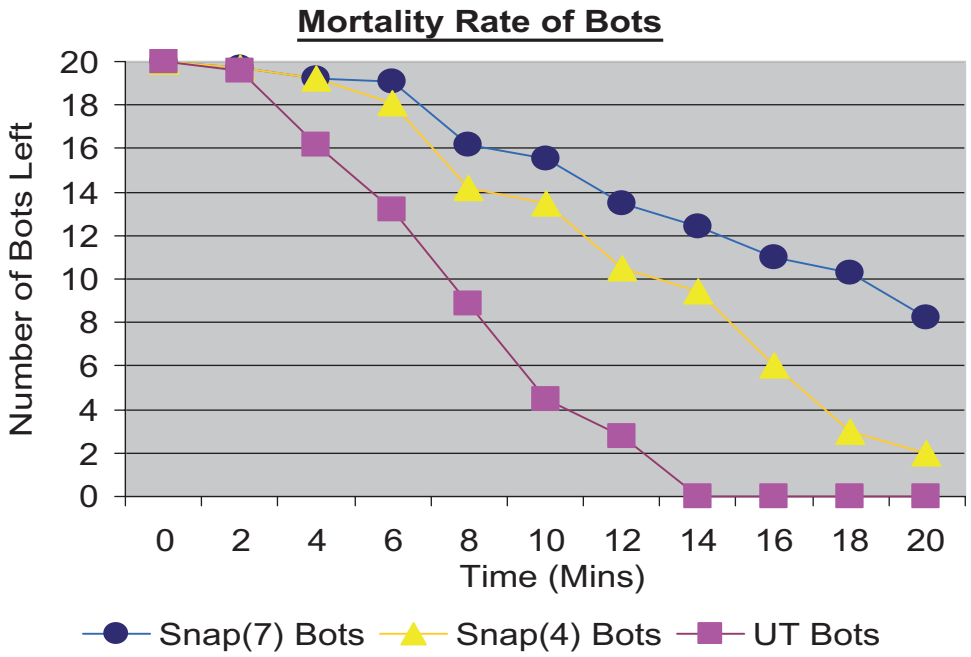


Figure 9. Mortality rate of soldier bots.

the Snap bots after 2 minutes. In 14 minutes, UT bots are totally destroyed by the threats whereas an average number of 12.4 Snap(7) bots were left. After the simulation terminated at 20 minutes, an average number of 8.2 Snap(7) bots survived. The results show that Snap bots are able to handle time critical threats better than the UT bots. Between the Snap bots, the Snap(7) bots performed significantly better than the Snap(4) bots, which is consistent with our expectation that the mortality rate of the bots decreases as the number of experience cases in the bots increases.

Conclusions and Future Work

Time critical decision-making models are important to generate realistic behaviors for the intelligent agents in virtual training systems and computer games. Formal rational rules do not produce realistic behaviors as humans may not have sufficient time to rationalize their decisions during time critical situations. Instead, past experiences may have a dominant role in determining how a human will behave in such situations.

The Snap decision-making framework that we discussed in this paper aims to imitate how human make decision in time-critical tactical situations for MOUT simulations. It uses the CBR cycle to enable the bots to make decisions in such situations with past experience cases. It also uses the thin-slicing technique for rapid situation recognition with partial information about the current situation. This is important since it is usually unlike for a human to have a complete knowledge about a situation. That is, people usually make sense of a complex situation in a timely manner based only on some key cues about the situation. Our experiment results demonstrate that Snap framework can provide realistic agent behaviors during MOUT simulations.

We will continue to work on the proposed time critical decision-making framework for more complex situations. In particular, more experience cases will be investigated and added into the Experience Repository of the bots.

References

1. The Art of War, translated by Lionel Giles, Project Gutenberg, 1910.
2. FM 90-10, Military Operations on Urbanized Terrain (MOUT). Headquarters, Department of the Army, Washington, D.C., 1979.
3. FM 3-06, Urban Operations. Headquarters, Department of the Army, Washington, D.C., 2003.

DOI: 10.1002/cav

4. Stacey CP. *Official History of the Canadian Army in the Second World War Volume III*. Queen's Printer: Ottawa, 1960.
5. Ballard JR. *Fighting For Fallujah. A New Dawn for Iraq*. Greenwood Publishing Group: Connecticut, USA, 2006.
6. Wray RE, Laird JE, Nuxoll A, Stokes D, Kerfoot A. *Synthetic Adversaries for Urban Combat Training* In Proceedings of the 2004 Innovative Applications of Artificial Intelligence Conference, San Jose, CA, July 2004. AAAI Press. Also published in AI Magazine 2005, **26**(3): 82–92.
7. Barrett LF. *Science of Emotion: What People Believe, What Evidence Shows and Where to Go From Here. Human Behavior in Military Contexts*. The National Academies Press: Washington, USA, 2007.
8. Schultz DP. Panic in The Military. Technical Report, University of North Carolina, Storming Media, January, 1971.
9. McDermott P, Battaglia DA, Phillips J, Thordsen M. *Military Operations in Urban Terrain (MOUT): Decision Making in Action*. US Army Research Institute: Fort Benning, April, 2001.
10. Klein G. *Sources of Power: How People Make Decisions*. MIT Press: Cambridge, Massachusetts, 1998.
11. López R, Mcsherry D, Bridge D, *et al*. Retrieval, reuse, revision and retention in case-based reasoning. *The Knowledge Engineering Review 2005*; **20**: 215–240.
12. Gladwell M. *Blink: The Power of Think without Thinking*. Little, Brown and Co.: New York, USA, 2005.
13. Wray RE, Laird JE. Variability in Human Behavior Modeling for Military Simulations. In Proceedings of Behavior Representation in Modeling and Simulation Conference, Scottsdale, AZ, 2003.
14. Azuma R, Daily M, Furmanski C. A Review of Time Critical Decision Making Models and Human Cognitive Processes. In Proceedings of the 2006 IEEE Aerospace Conference, MT, March 4–11, 2006.
15. Osinga F. *Science, Strategy and War: The Strategic Theory of John Boyd*. Routledge: Abingdon, UK, 2007.
16. Bartlett FC. *Remembering: An Experimental and Social Study*. Cambridge University Press: Cambridge, 1932.
17. Brewer WF, Treyens JC. Role of schemata in memory for places. *Cognitive Psychology 1981*; **13**: 207–230.
18. Widmayer SA. Schema theory: An introduction. http://chd.gse.gmu.edu/immersion/knowledgebase/strategies/cognitivism/SchemaTheory.htm
19. Evans K. *What Squad Leaders Want to Know in Battle*. US Army Research Institute: Fort Benning, GA, 2006.
20. Karagosian JW. Streetfighting: The Rifle Platoon in MOUT. *Infantry Magazine 2000*.
21. Epic Games, Unreal Engine, 1998; http://unrealtechnology.com
22. Zhou SP, Ting SP, Shen ZQ, Luo LB. Twilight City – A virtual environment for MOUT. *International Journal of Computer and Applications in press.*
23. Zhou SP, Ting SP. Qualitative Physics for MOUT, *39th Annual Simulation Symposium*, Huntsville, AL, USA, April 2–6, 2006.
24. Adobbati R, Marshall AN, Scholer A, Tejada S. GameBots: A 3D virtual world test bed for multiagent research. In Proceedings of the Second International Workshop on Infrastructure for Agents, MAS, and Scalable MAS, Montreal, Canada, 2001.
25. Marshall AN. JavaBot for Unreal Tournament, 2003; http://utbot.sourceforge.net

Authors' biographies:

Shang-Ping Ting is currently a proprietary trader in Nyenburgh (Singapore). He received his B.Eng. and M.Eng. in Computer Engineering from Nanyang Technological University (Singapore). Currently, he is reading his Ph.D. in Nanyang Technology University. His current research interests include: time critical decision making, qualitative physics, reusable software architectures and human behavior representation in modeling and simulation.

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 behavior representation in modeling and simulation. He has published more than 50 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.

DOI: 10.1002/cav