Intelligent bots for MOUT

Ting, Shang Ping

2011

Ting, S. P. (2011). Intelligent bots for MOUT. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/43951>

<https://doi.org/10.32657/10356/43951>

NANYANG TECHNOLOGICAL UNIVERSITY

Intelligent Bots for MOUT

A Thesis submitted to the School of Computer Engineering
of Nanyang Technological University

by **Ting Shang Ping**

for the Fulfilment of
the Degree of Doctor of Philosophy

Main Supervisor: Dr Michael Harold Lees

Co-Supervisor: Dr Zhou Suiping

Abstract

Military Operations on Urban Terrain (MOUT) are defined as military actions that are planned and conducted on a terrain where man-made construction affects the tactical options available to the commander. This type of conflicts is characterized by street-by-street, room-by-room fighting. Strategies employed in MOUT differ immensely from fighting on other types of terrain, such as large deserts or jungles. The importance of MOUT warfare is increasing due to rapid world-wide urbanisation. The limitation of urban terrains means that soldiers are expected to function in small teams without the support of air-power or tanks. To survive, soldiers rely on small squad tactics and individual situational awareness. Modern armies may suffer substantial casualties in MOUT warfare. Going into MOUT operations without proper preparation could be very dangerous to the soldiers.

Extensive training conducted with computer simulations may be able to reduce casualties in MOUT warfare. Simulations can introduce the trainees to the combat stress and shorten response time during actual combat. Although there had been considerable efforts and capital invested into this area, there are still many challenges facing existing simulation systems. One of the key challenges lies in the lack of realistic human behaviours. Human-like realism can improve the training relevance of MOUT simulations and allow the trainees to gain significant operational benefits. Human characters are represented by bots during MOUT simulations. It is important to develop intelligent bots in order to generate human-like behaviours.

Our philosophy is that an intelligent bot should not only be able to generate seemingly realistic behaviors in some given situations, it should also behave like a human in the sense that the decision-making and cognitive processes of a bot should be similar to that of a human being. Therefore, our approach for human-like behaviours focuses on achieving both procedural and end-result realism for intelligent bot behaviours. We believe that by imitating human decision making and cognitive processes under various tactical situations, human-like bot behaviours can finally be generated.

After interviewing soldiers from the Singapore Armed Forces (SAF) and engineers from the Defence Science and Technology Agency (DSTA), it is suggested that certain behavioural skills are extremely important for human soldiers during MOUT warfare. These key skills are decision making, navigation and situation awareness. If the key areas can be modelled accurately, then realistic behaviours will be generated. To generate human-like behaviours, we study the decision making and cognitive process of the soldiers in the identified key skills. Subsequently, novel features are developed to improve the realism of bot behaviours in MOUT simulation.

The novel features designed and implemented in this work includes a) a time critical decision making framework, b) a computational model for situation awareness and c) an autonomous navigation system.

This report summarizes our current work on these novel features. They are designed to support our concept of closely imitating the behaviours of humans to produce intelligent bot behaviours. We developed a time critical decision making framework for MOUT simulations called *SNAP*. As MOUT scenarios are generally time critical, the *SNAP* framework will be useful for generating realistic decisions during MOUT simulations. *SNAP* generates decision from incomplete information gathered under time constraints. The information comes from the situation awareness of the bots. A realistic computational model for situation awareness will greatly improve the quality of the information provided to *SNAP*. Therefore, we produced a computational model of situation

awareness for MOUT. With a proper computational mechanism, accurate information can be gathered for decision making. Within the complex and dynamic MOUT environment, the human soldiers are greatly affected by spatial considerations and navigation behaviours. To address the need for bot navigation, an autonomous navigation system, *Quartz*, was built. *Quartz* employs qualitative spatial representation and hierarchical spatial reasoning to enable human-like path planning. Finally, a MOUT virtual environment, *Twilight City*, was developed to demonstrate and evaluate the effectiveness of our work.

Acknowledgments

During my national service, I served as a instructor in the Officer Cadet School(OCS). Although I had never fought a war before, the experiences that I gained during my time in OCS gave valuable insights on the challenges of military operations. A common issue is the inability of squad leaders to execute the plans of higher commands effectively. The soldiers spend countless hours for mission planning. However, they may still fail to execute their plans due to stress, time constraints and changing conditions. Poor execution during MOUT warfare will lead to much casualties. In order to provide a safe and effective environment to prepare soldiers for MOUT warfare, I had been working on enhancing the value of MOUT simulations since returning to my studies in Computer Engineering. Upon my graduation, I was fortunate enough to join the Modelling and Simulation Department(ModelSim) in the Defence Science and Technology Agency (DSTA). It was then when I realized that the realism in bot behaviours falls greatly behind the graphical realism of current simulations. This is the reason that I had decided to based my Ph.D. work on developing intelligent bots to provide realistic human behaviours in MOUT simulations.

I hope to thank my supervisor, Dr Zhou Suiping, for the invaluable advice and guidance during my research. Throughout the work, he enlightened me with knowledge that I would not have comprehended. His passion for research and approachable character has injected much enthusiasm. The numerous discussions with him had helped me achieve a deeper depth in computer simulation and allowed us to provide useful contributions.

Without his constant support and patience, it will have been impossible for this experience to be so enriching and fulfilling.

I wish to thank everyone in the ModelSim of DSTA. It is a privilege to work with all of them. Nothing is more delightful than working with people who share the same passion in military simulations. I especially need to thank Dr How Khee Yin, Low Kok Seng and Daryl Lee Chin Siong for sharing their valuable insights and knowledge on MOUT simulations.

I would like to thank my parents, my wife for their understanding and support. Although there had been times when I had to give up my family time for my research, they never had any complaints and were constantly encouraging to go ahead with my work. I also need to give my loving thanks to my daughter for giving me so much joy and laughter.

Author's Publications

- Shang-Ping Ting, Suiping Zhou, Nan Hu, *Generating Situation Awareness for Time Critical Decision making*, LCNS Transactions on Computational Science, Accepted
- Shang-Ping Ting, Suiping Zhou, *Dealing with dynamic changes in time critical decision-making for MOUT simulations*, Journal of Visualization and Computer Animation, Vol 20, No. 2-3, 427-436, 2009
- Shang-Ping Ting, Suiping Zhou, *Snap: A time critical decision-making framework for MOUT simulations*, Journal of Visualization and Computer Animation, Vol 19, No. 3-4, 505-514, 2008
- Suiping Zhou, Shang-Ping Ting, Zhuoqian Shen, Linbo Luo, *Twilight City - A virtual environment for MOUT*, International Journal of Computer and Applications, Vol 30; No. 2, pages 117-123, 2008
- Shang-Ping Ting, Suiping Zhou, *Quartz: an autonomous navigation system for MOUT simulations*, Journal of Visualization and Computer Animation, Vol 18, No. 4-5, 383-394, 2007
- Shang-Ping Ting, Suiping Zhou and Nan Hu *A computational model of situation awareness in MOUT simulations*, 2010 International Conference on CYBER-WORLDS, 20-22 Oct, 2010, Singapore

- Shang-Ping Ting, Suiping Zhou, *An Architecture for Rapidly Re-configurable MOUT Simulation*, in the Proceedings of the 40th Annual Simulation Symposium, March 26 - 28, 2007, Norfolk, VA
- Suiping Zhou, Shang-Ping Ting, *A Generic Model Framework for MOUT Simulations*, in the Proceedings of the 2006 International Conference on CYBER-WORLDS, Nov 28 - 30, 2006, Lausanne, Switzerland
- Suiping Zhou, Shang-Ping Ting, *Qualitative Physics for MOUT*, in the Proceedings of the 39th Annual Simulation Symposium, April 2 - 6, 2006, Huntsville, AL, USA
- Suiping Zhou, Linbo Luo, Wee-Lit Koh, Shang-Ping Ting, *Human Behavior Representation for Crowd Simulations*, in the Proceedings of the International Conference and Symposium on Computer Games; Animation, Multimedia, IPTV and Edutainment (CGAT'08), 28th-30th April, 2008, Singapore

Contents

Abstract	i
Acknowledgments	iv
Author's Publications	vi
List of Figures	xii
1 Introduction	1
1.1 Background	1
1.1.1 Notable Urban Combats	4
1.1.2 Types of Urban Operations	5
1.2 Motivation	7
1.3 Objective	10
1.4 Challenges	11
1.5 Contributions	12
1.6 Organization of Thesis	13
2 Background	14
2.1 Evolution of MOUT	14
2.2 MOUT Training through Computer Simulations	16
2.2.1 Case Study: <i>Full Spectrum Leader</i>	17
2.2.2 Case Study: <i>Close Combat - Marines</i>	17
2.3 Behaviour Modeling in MOUT Simulations	19

2.4	Impact of Human Factors on Military Simulation	21
2.5	Behaviour Modeling for Military Simulations: Requirements for Intelligent Bots	22
2.6	Decision Making	24
2.6.1	Fuzzy Logic	25
2.6.2	Hierarchical AI	28
2.6.3	Military Decision-Making Process	30
2.6.4	Recognition primed decision	32
2.6.5	SOAR	34
2.6.6	Time Critical Decision Making	36
2.7	Situation Awareness	38
2.8	Autonomous Navigation Systems	39
2.9	Terrain Reasoning	41
2.10	Most Recent Research	43
3	<i>SNAP</i> -Time Critical Decision Making Framework	46
3.1	Motivation	47
3.2	Features of <i>SNAP</i>	47
3.3	Design of <i>SNAP</i>	53
3.3.1	Case-Based Reasoning Process	54
3.3.2	Rapid Cognition	59
3.3.3	Expectations	61
4	A Computational Model of Situation Awareness	65
4.1	Perception Stage During Situation Awareness	67
4.2	Comprehension Stage During Situation Awareness	69
4.2.1	Cue Types	70

4.2.2	Similarity measures	70
4.3	Projection Stage During Situation Awareness	74
5	<i>QUARTZ</i> - Autonomous Navigation System	77
5.1	Requirements	77
5.2	Design of <i>Quartz</i>	78
5.2.1	Why Qualitative Spatial Representation and Hierarchical Spatial Reasoning?	78
5.2.2	Qualitative Spatial Representation in <i>Quartz</i>	79
5.2.2.1	Qualitative Spatial Representation in <i>Quartz</i>	81
5.2.2.2	Map Representation for High Level Path Planning	83
5.3	Qualitative Spatial Reasoning in <i>Quartz</i>	85
5.4	Limitations of <i>Quartz</i>	86
6	The simulation testbed - <i>Twilight City</i>	88
6.1	Motivation	88
6.2	MOUT Scenarios	89
6.3	City Planning	90
6.4	Issues encountered during implementation	92
6.5	Novel techniques	95
6.5.1	Qualitative Physics Simulation	95
6.5.2	Architecture for Rapidly Re-configurable Simulations	96
6.6	Implementation Details	97
6.6.1	<i>SNAP</i> in <i>Twilight City</i>	97
6.6.2	Integrating <i>Quartz</i> into <i>Twilight City</i>	99
6.6.3	Spatial Reasoning in <i>Twilight City</i>	100
7	Experimental Results	102

7.1	Experiments results on <i>SNAP</i>	102
7.2	Evaluation of Computational Situation Awareness Model	109
7.3	Evaluation of <i>Quartz</i>	113
7.3.1	Mean Opinion Score Tests in <i>Quartz</i>	117
7.4	Mean Opinion Score Tests for <i>Twilight City</i>	119
8	Conclusions and Future Work	124
8.1	Summary	124
8.2	Conclusions	126
8.3	Future Work	128
8.3.1	Terrain Reasoning	128
8.3.2	Tactical Awareness	129
8.3.3	Applying the current work to non-military areas	129
	References	131

List of Figures

Figure 2.1	Soldiers fighting in urban environment	16
Figure 2.2	Full Spectrum Leader	18
Figure 2.3	Military Decision-Making Process	31
Figure 3.1	Close Combat	48
Figure 3.2	Enemy sniper bot taking aim at soldier bots	51
Figure 3.3	Dynamic changes during MOUT simulations	52
Figure 3.4	Decision Making Framework	53
Figure 3.5	An Experience Case	55
Figure 3.6	Descriptive Spatial Representation	57
Figure 3.7	Solution for Counter Sniper	59
Figure 4.1	Bots increasing their awareness on anticipation of impending threat	66
Figure 4.2	Maintaining Situation Awareness in <i>SNAP</i>	67
Figure 4.3	Sensory System of A Bot	68
Figure 4.4	Perception cues and their attributes	69
Figure 5.1	Squad Formation	80
Figure 5.2	Hierarchical spatial representation	81
Figure 5.3	Visibility Graphs	82
Figure 5.4	Voronoi Graphs and Regions	84
Figure 5.5	Hierarchy of qualitative spatial layers	84

Figure 5.6	Hierarchical data structure in Quartz	85
Figure 5.7	<i>QuartzPath</i> Algorithm	86
Figure 6.1	Twilight City Top View	92
Figure 6.2	Integrating <i>SNAP</i> into Twilight City	97
Figure 6.3	Experience Cases	98
Figure 6.4	Maps in <i>Twilight City</i>	99
Figure 7.1	Behavior of Soldier Bots with Different Experiences	103
Figure 7.2	Sniper Assault Situation in Twilight City	104
Figure 7.3	Mortality Rate of Soldier Bots	105
Figure 7.4	Effect of Expectations on Bots	106
Figure 7.5	Mortality Rate of Soldier Bots	108
Figure 7.6	Results of Situation Awareness	111
Figure 7.7	Frame rate analysis with Situation Awareness Model	113
Figure 7.8	Impact of Quartz on path planning efficiency	114
Figure 7.9	Frame rate analysis of Quartz	115
Figure 7.10	Tactical path planning	116
Figure 7.11	Mean Opinion Score for bot navigation	118
Figure 7.12	Conducting Mean Opinion Score Tests of <i>Twilight City</i>	119
Figure 7.13	Mean Opinion Score Tests Results	121
Figure 7.14	MOS Results for Experienced Vs Non-experienced users	123

Chapter 1

Introduction

1.1 Background

Military Operations on Urban Terrain (MOUT) is the term used to describe any type of military action where battles take place in an environment where man-made construction or high population density is the dominant feature, such as a city or town [1]. This type of conflicts is characterized by street-by-street, room-by-room fighting. MOUT warfare is not new to soldiers. However, urbanised areas and urban populations have grown significantly during the past 10 years. The urbanisation had started to exert a much stronger influence on military operations. The worldwide population is shifting from a rural to an urban environment. In 2008, the world reached an important milestone: For the first time in history, more than half its human population, 3.3 billion people, will be living in urban areas. By 2030, this is expected to swell to almost 5 billion. The urban percentage of the world's population is projected to reach 60 percent by 2030 [2]. The increase in urban population have made overcoming challenges of combat in built-up areas an urgent and important need. Future battles are expected to be fought in urban areas and much more work must be done to ensure the survival of our soldiers during urban combat.

Urban combat operations may be carried out to take advantage of certain strategic or tactical strength of the military force. Urban operations may also be executed to take

control or possession of a particular urban area to reduce the strategic advantages of the enemy. Key urban areas holds the political power and wealth of a city or country. These urban areas may come in the form of economic institutions, transportation complexes and industrial bases. The capture of these key areas may generate decisive psychological advantages. More often than not, these advantages tend to determine the success of the overall war.

Military thinkers and analyst fully understands the dangers of fighting in urban areas. As early as 500 B.C., Sun Tzu [4] advised that “the worst policy is to attack cities,”. His advice had been quoted in many military writings and doctrine since then. Despite the sensible advice, wars have been waged within cities repeatedly for many centuries.

A city is the greatest challenge to any tactical force. In cities, the space separating friendly from enemy forces compresses. The zone is often thousands of meters in open battle but only tens of meters in the urban maze of densely aggregated buildings, streets and back alleys. The traditional advantages of fighting outside the killing zone disappear as cities forces soldiers to conduct close combat. The compartmented nature of the urban environment fragments forces. Short lines of sight limit the effective ranges of weapons and disrupts the effective use of precision-guided weapons launched from aerial platforms. Compartmented urban terrain lessens to a significant degree the advantages of superior situational awareness and electronic-communications dominance. These challenges and obstacles that are not usually found on an open battlefield. Large numbers of civilians, culturally important structures, narrow streets and alleys causes huge difficulties. A soldier in MOUT must contend with all kinds of distractions and stresses as he moves through a hostile city. They include differentiating hostile targets from innocents, constantly watching for sniper fire and maintaining a state of heightened awareness at all times [7].

Urban terrain creates great difficulty for movement. Streets channelize the ground movement. As the routes are predictable, cities offer ideal environment for preparing

ambushes. The Russian Army discovered this in Chechnya. The Chechen guerrillas anticipated their armored assault into Grozny. The guerrillas then ruthlessly ambushed the Russians from their sides, behind, and above [5]. The space constraints of the narrow street immobilized the Russian armored forces. Thus, the Russians had great difficulty in moving forward or backwards and to attack or retrograde.

Urban terrain poses a great challenge for military forces to achieve a reasonable level of command, control, communications and intelligence (C3I). No other terrain types pose a greater challenge. The density of urban areas hinders navigation. Most global positioning system (GPS) navigation systems needs to connect with a minimum of three satellites to find out the coordinates of a location. This is often not possible while the soldiers are within the interior of buildings or blocked by tall buildings. Even if the ground unit managed to pinpoint their exact location in a city, it is not easy to communicate this information to their higher command. Radios require line-of-sight before transmissions can be done. Line-of-sight is normally impossible as the soldiers are obstructed by urban structures such as buildings.

As civilian casualties and collateral damage must be avoided as much as possible, modern artillery and large-scale weapons are inappropriate for urban combat. For this reason, urban fighting benefits the defender. Attackers lose the advantage of firepower and mobility. A city can ingest an invading army, paralyze it for weeks, and render it ineffective [8, 9]. Today's military forces are constantly refining their strategies for fighting in large urban environments. An invading army that needs to conduct operations in a hostile city must have a good grasp of MOUT strategies and tactics.

Another complex issue in MOUT warfare is the presence of civilians. Under time constraints, it can be difficult to distinguish civilians from militants. This is due to largely similar dress code. There is a need for the soldiers to get familiar with the characteristics of the urban population before the battle. When attacking, the attackers

also tend to become more exposed than the defender as they use the open streets more often, unfamiliar with the defenders' secret and hidden routes.

1.1.1 Notable Urban Combats

- *Stalingrad, 1942-43*: Soviet defense of Stalingrad in World War II killed many soldiers from the invading German army. They later established the conditions for a Russian counter-offensive that could be decisive. According to historical estimates, more than 1.5 million people was killed in the battle that lasted more than 30 days.
- *Berlin, 1945*: During the World War II, the Soviets took more than two weeks to take over Berlin. The casualties of this battle easily numbered more than ten thousand.
- *Manila, 1945*: The Japanese navy defended the city against the advancing US troops with poorly equipped and badly trained personnel. Despite their superiority, US forces suffered heavy losses. In addition, much of the city population and infrastructure was destroyed.
- *Jerusalem, 1967*: Israeli forces captured East Jerusalem from Jordanian control. Israel suffered an estimated 400 casualties while Jordan recorded huge numbers of deaths.
- *Hue, 1968*: This is one of the most famous battle during the Vietnam war. North Viet Cong forces and Vietnamese Army captured Hue in South Vietnam. During this battle, they defended the city for about three weeks against US and South Vietnamese soldiers. More than 400 US Marines died. An estimated 5,000 other fighters on both sides died.

- *Mogadishu, 1993*: Eighteen US servicemen in Somalia were sent to restore order and safeguard relief supplies. However, they were killed in a fire fight in the capital.

1.1.2 Types of Urban Operations

To assess the issues involved in MOUT, we need to differentiate between the different kinds of operations that the modern soldiers could be asked to performed during “urban operations”. It is also important to recognize the conditions that affects the effectiveness of various urban operations. Once the various operations and their optimal conditions is understood, the feasibility of armies dominating the enemies in urban combat can also be assessed accurately.

The first type of urban operations is known as the “policing operations”. Like peacetime policing, prevention of violence outbreak is the key goal of the policing operations. Peacekeeping operations carried out in Bosnia and Haiti are some examples of policing missions. Policing forces are normally challenged by small groups of poorly trained opposition. The common adversaries are generally irregular groups who are less coordinated than military units. As in peacetime policing, the most important objective of the mission is to maintain presence within the area of operations, concentrate strong forces to overwhelm the key insurgents rapidly and bring them away from the rest of the population quickly.

The second category of urban mission, raids, has a broad range. Raids can have many differing goals. For example, the soldiers could be tasked to evacuate an embassy, rescue hostages, arrest key enemy personnels, seizing key facilities such as ports or airfields. The typical characteristics of such missions is that

- The soldiers to be inserted rapidly into the enemy territory
- A mission to be completed at a target location (e.g. assault on enemy forces)

- The extraction of soldiers is done rapidly once the mission objectives is completed

The element of surprise is the key to success in a raid. In general, the doctrine is to insert overwhelming forces into to the target location, complete the mission objectives and extract the raid forces before the adversarial forces is able to react. Since the end of the Cold War, there had been countless number of raids. Civilians have been rescued from embassies in Sierra Leone and Albania. Raid forces took control of an airfield during the Panama invasion to achieve a key strategic advantage. American forces had also conducted raids to arrest certain notable Bosnian Serb war criminals.

The third kind of urban operations is known as sustained urban combat. The primary missions of the sustained urban combat are to defend a city, take control of a city area, or eliminate enemy forces that are hiding within a city area. Since the fighting in Hue during the Vietnam War, American forces had not fought in sustained urban combat for thirty years. The Russian assault on Grozny is the one of the recent example of sustained urban fighting.

Sustained urban combat could be fought against enemy forces with varying skill levels. These skill levels may range from untrained civilians to well-drilled military unit. As described above, it is considered as one of the toughest and costly kind of military operations. It is an arena that even untrained, irregular forces can cause substantial losses on an well-trained attacking military force in sustained urban combat.

The modern armies will send their forces into sustained urban fighting so as to achieve a wide variety of objectives. However, due to the significant costs of such missions, it is very unlikely that the military commanders would choose to fight a sustained urban combat unless the interests at stake is of paramount importance.

1.2 Motivation

Casualties can be avoided during MOUT warfare by providing the soldiers with extensive preparation. Traditional MOUT training requires trainee soldiers to go through a series of live exercises in an urban environment. With live training, it is difficult for the trainers to record the individual performance and mistakes of all the trainees. Thus, trainees could be repeatedly given the incorrect training values in terms of tactics and expectations without being spotted by the trainers. Virtual MOUT training could be used to supplement live training as virtual systems are able to capture the results of the training. Trainers can analyze and correct trainees before putting them into live training by the information collected during virtual training. In this way, the learning curve of the soldiers can be accelerated. Significant costs and time can also be saved as Virtual reality (VR) training systems is inexpensive and convenient.

VR systems may be grouped into 3 main categories [119]:

- (i) Non-Immersive (Desktop) Systems: Using the desktop system, the virtual environment is viewed through a portal or window by utilising a standard high resolution monitor. Interaction with the virtual environment can occur by conventional means such as keyboards, mice.
- (ii) Semi-Immersive Projection Systems: A semi-immersive system will comprise of a relatively high performance graphics computing system which can be coupled with either a large screen monitor or a large screen projector system or multiple television projection systems. Semi-immersive systems therefore provide a greater sense of presence than non-immersive systems and also a greater appreciation of scale. In addition, images can be provided that are of a far greater resolution than head mounted displays and this implementation provides the ability to share the virtual experience.

- (iii) Fully Immersive Head-Mounted Display Systems: An HMD uses small monitors placed in front of each eye which can provide stereo images. Stereo images are provided in a way that a slightly different image is presented to each eye. The major difference is that the two screens are placed very close (50-70mm) to the eye, although the image, will be much further away because of the HMD optical system. Fully immersive VR systems tend to be the most demanding in terms of the computing power and level of technology required to achieve a satisfactory level of realism and development is constantly underway to improve the technologies. Major areas of research and development include field of view vs resolution trade-offs, reducing the size and weight of HMDs and reducing system lag times.

For virtual MOUT training, it is important to achieve tactical immersion for the soldiers to attain situation awareness and perform the accurate response rapidly. With this goal in mind, we note that the benefits of our model improves along with the level of immersion. However, to achieve our aim of developing low-cost virtual simulation environments, this thesis will describe the implementation of techniques to develop realistic bot behaviours in non-immersive simulations. It must be emphasized that our implementation should not be regarded as the distinct boundary for future implementations. For example, it is possible to turn a desktop system into a semi-immersive system by simply adding shutter glasses and the appropriate software, or a fully immersive system by connecting an HMD.

Visiongain's findings had stated that global government spending on simulation and virtual training in 2008 reached a total of \$7.9bn [6]. There are already a large selection of commercial off-the-shelf and customised simulation tools in use for a broad range of applications. In their analysis, they found that the increasing usage of simulation by military across the globe is due to a number of factors. Military analysts are producing numerous strategies in order to benefit from the

latest technological advancements in the simulation sector. They also found that armed forces will be relying more and more on simulation to full up the capability gaps of their training programs over our forecast period. Although modern armies are able to utilize the existing technologies to develop realistic graphics, sounds and equipments for simulation training. After building the realistic virtual environment, the developers need to populate the environment with bots which may act as adversaries, soldiers or civilians. The value of virtual MOUT training on the human soldiers is greatly influenced by the realism of bots behaviours during simulations [11].

Realism can be judged by the bots' similarity to human behaviours. Bots exhibiting realistic behaviours are known as intelligent bots. This is an important area in MOUT simulations that needs much more work. Moore's law predicts that the processing power of computer to double every 12 to 18 months [10]. In foreseeable future, the processing power of the computers may reach the level of the human brain. However, human behaviours in MOUT simulations do not merely depend on processing speed. To generate realistic behaviours, we need to understand how human soldiers behave during real life MOUT warfare and develop techniques to model these behaviours. Therefore, this work seek to develop techniques and frameworks for intelligent bots behaviours for MOUT simulation. Another important issue is that we need to define a method to measure the level of realism. As realism is a matter of perception, we believe that a qualitative method will be more useful than a quantitative method to measure realism. We also note that, our bots are developed to simulate MOUT soldiers, the bots behaviours must achieve tactical realism in terms of actions and strategies. Therefore, a effective method to measure realism in our work will be to analyze the results of opinion surveys from trained soldiers who participated in our simulation experiments.

1.3 Objective

Virtual MOUT simulations seek to provide training to the soldiers in the areas of individual tactics, adversary behaviours and terrain reasoning. This research seeks to enhance the value of MOUT training by making the behaviours of the bots more realistic. Our goal is not to produce a model to generate more efficient and optimal behaviours. Instead, the focus will be on achieving human-like realism. In terms of realistic behaviours, our philosophy is that an intelligent bot should not only be able to generate seemingly 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. We believe that the end-result realism of the behaviour model ultimately relies on the structural and procedural realism.

Our objective is to develop an approach towards building realistic human behaviours for bots in MOUT simulations. The proposed approach should be designed to work during time critical and dynamically evolving situations. With this goal in mind, novel techniques will be used to develop various components of human behaviours in MOUT. The main areas to be addressed in our work is time critical decision making, situation awareness and autonomous navigation. In particular, a time critical decision making framework, a computational model for situation awareness and a autonomous navigation system will be produced. These components will work in tandem to generate realistic human behaviours in MOUT simulations.

5.

1.4 Challenges

Generally, MOUT simulations do not consider time constraints as a factor during decision making. This is unrealistic as MOUT situations is often time critical. Given the luxury of time, soldiers may be able to perform a thorough cost-benefits analysis and produce the best solution. However, soldiers in a time critical situations normally uses a different set of criteria in order to generate a satisficing solution [12]. This is an issue that need to be addressed so that realistic decision making for bots can be generated.

For decision making to be effective, the bots needs to have proper situation awareness. Situation awareness plays a crucial role for human soldiers to make decisions in Military Operations on Urban Terrain (MOUT). It is therefore important to accurately model situation awareness in order to generate realistic tactical behaviors for the non-player characters (also known as bots) in MOUT simulations. This is a very challenging problem due to the time constraints in decision-making process and the heterogeneous cue types involved in MOUT. Although there are some theoretical models on situation awareness, they generally do not provide computational mechanisms suitable for MOUT simulations.

Another area found to be lacking in current MOUT simulations that the bots are typically not able to navigate intelligently within a complex MOUT environment. This is due to the dynamic changes occurring during runtime. These dynamic changes to the environment affects the path planning of the bots and may render a previously sound path to become invalid. As human soldiers are able to reasonably adapt to these changes during missions, we seek to impart our bots with similar abilities during navigation.

To fulfill our objective, we will work towards overcoming these challenges in the areas of decision making, situation awareness and path planning of the bots.

1.5 Contributions

The major contributions of our research are the time critical decision making framework, the computational situation awareness model and the autonomous navigation system.

We designed a time critical decision-making framework called *SNAP* [13]. With *SNAP*, the bots can utilize key cues and past experiences to form solutions rapidly. The novel features of the framework include case-based reasoning (CBR) and *expectations*.

We also proposed a computational model of situation awareness for the bots in MOUT simulations. The computational model aims to form up situation awareness quickly with some key cues of the current tactical situation. It is also designed to work together with *SNAP* to produce realistic tactical behaviours.

To handle the requirements for realistic navigation, an autonomous navigation system, *Quartz*, is designed. *Quartz* is an autonomous navigation system for MOUT simulations. *Quartz* contains features such as qualitative spatial representation and hierarchical spatial reasoning which enable fast situation analysis and human-like path planning.

To evaluate the effectiveness of the proposed implementation, we adapted a commercial game engine to develop a virtual environment for MOUT simulations. This innovative approach allows us to provide a high quality immersive simulation environment at low cost. This simulation environment is known as *Twilight City*.

Twilight City is designed to support our major work by providing qualitative information required for *SNAP*. The spatial data structure of *Twilight City* is also well suited for *Quartz*. Our success in building such a game engine based training system was noted by the Defence Science and Technology Agency (DSTA) and a version of our training system was integrated into their simulation systems.

1.6 Organization of Thesis

The report is organized as follows:

Chapter 1 presents the motivation, objectives, contributions for our research.

Chapter 2 provides the background on existing MOUT simulation technologies and describes their limitations.

Chapter 3 discusses how *SNAP* can enable bots to imitate how humans make decisions in time-critical situations.

Chapter 4 goes into the computational model of situation awareness and how bots attain situation awareness during MOUT simulations.

Chapter 5 explains the fast situation analysis and human-like path planning that can be performed with *Quartz* in the dynamic environment.

Chapter 6 provides detailed description of implementing the simulation techniques and a short description on the construction of the *Twilight City*.

Chapter 7 discusses the experiments done on the proposed mechanisms and the results obtained.

Chapter 8 draws the conclusions of our research and discusses some potential directions for future work and enhancements.

Chapter 2

Background

This chapter contains the review of the MOUT simulations and relevant case studies. The key requirements for building intelligent bot behaviours for military simulations are also studied in this chapter. With these requirements in mind, we will discuss the various techniques that may be applied to generate realistic behaviours. Finally, a study is done on some key areas of our work. These areas include decision making, situation awareness autonomous navigation and terrain reasoning.

2.1 Evolution of MOUT

In the entire history of the mankind, terrain has played a crucial role in influencing the success of battles. From amphibious warfare to fighting in open terrain, to navigating through dense forested areas, each terrain contains its unique set of requirements and challenges.

In medieval times, most of the actual fighting was done on battlefields far from city centers. A city's only real military significance was its fortifications and its garrison. A city will lose its military advantage once these defenses were taken over.

However, the world is rapidly becoming more and more urbanised in modern days. Most fighting now takes place within the streets and buildings of cities. These urbanized areas are becoming hugely challenging military impediments. Apart from possibly hindering a strategic route of approach towards the enemy area, the population of a city poses major security, administrative, and logistics problems.

Urban operations is a modern kind of warfare that require an entirely new set of strategies and tactics[94]. Tactics for fighting in urban terrain are constantly evolving as tough challenges arise. In the past, armies used to be able to gain combat superiority from their sheer numbers, the fire power of their weapon systems, heavy artillery and tanks. The modern armies now find that their old principles and strategies do not apply during combat missions in urban environments. With the unique cover and concealment provided by a city environment, the adversarial forces can strike from all directions and angles and retreat into their concealment within the civilian population rapidly. The presence of civilians in cities also affects the effectiveness of airspace superiority and heavy artillery during urban operations.

Modern military forces are discovering that the most effective way to eliminate the hostile elements residing within a city is to employ small squads of well-trained and technologically superior soldiers that excel in room-to-room combat as shown in Figure 2.1. It is these carefully coordinated squads of soldiers that will determine the success in the modern urban battlefields [95]. This approach poses great danger for the attacking forces and only highly skilled soldiers functioning in well coordinated teams should be allowed for such combat missions.

Figure 2.1: Soldiers fighting in urban environment

2.2 MOUT Training through Computer Simulations

Cost-effective methods for developing soldier decision-making and leadership skills are especially needed in urban operations. One solution is to conduct a portion of this training, such as mission rehearsals, in virtual environments through the use of individual combatant simulators [14]. With effective MOUT simulations, soldiers can be introduced to the stress of combat and familiarize themselves with urban combat tactics without exposing themselves to life-threatening situations. Research has been successful in demonstrating that virtual simulation can adequately support the performance of a variety of activities and is perceived to be effective for training both individual and collective tactical behaviours [96].

2.2.1 Case Study: *Full Spectrum Leader*

Full Spectrum Leader (FSL) is a PC strategy game employed as a training tool to simulate battlefield co-ordination at the company level. The training aid spawned from a technology partnering agreement between the US Army and Singapore Armed Forces. The game is designed to develop the infantry commander's cognitive skills, tactical decision-making, resource management and adaptive thinking through various scenarios. These scenarios are focused on asymmetric threats within peacekeeping and peace-enforcement operations. It allows the user to understand the importance of terrain analysis and the potential fratricide situations that can occur during a battle.

Key Features

- Custom designed to reflect current operational tactics
- Head-to-Head multiplayer capability via LAN.
- User-level editor for scenario customization.
- Instructor evaluation mode for curriculum-based usage.

FSL is developed by the Institute of Creative Technologies (ICT) in California, US. ICT was formed by the US Army and the University of Southern California, bringing together the Hollywood film community and Silicon Valley-based electronic games developers to work on state-of-art immersive training simulation. Besides the US Army, the Singapore Army is one of the few users to use FSL.

2.2.2 Case Study: *Close Combat - Marines*

Close Combat: Marines (CSM) is the first version of Close Combat series made specifically for military training purposes. The US marines uses CSM as an inex-

Figure 2.2: Full Spectrum Leader

pensive and easy-to-use tool to teach a Marine leader the dynamics of tactics. It allows a soldier to fight hundreds of scenarios, make thousands of tactical decision, experiment with different tactics and learn from his mistakes.

Key Features

- Repetition - In order to understand and identify patterns, Marines need hundreds of simulated examples. In order to internalize lessons, Marines need to fight an active enemy and suffer from their own tactical mistakes. Through repetition, the basic lessons become so well known that advanced tactics and experimentation can be attempted. Only with the experience of fighting through a hundred enemy positions can a leader look for weaknesses in a given position and initiate creative ways to exploit that weakness. Reading

the subtle aspects of a tactical situation is a learned skill that requires far more practice than is currently available outside of a simulation.

- Efficient use of time - Schools and units schedule training time. Weekends, nights, travel time, and dead time can all be used for individual simulation training. This time is usually far more plentiful than that allocated to formal learning environments. In the operating forces, especially, opportunities for individualized learning should be maximized.
- Peer competition -Marines can fight each other on a simulated battlefield. These tactical learning experiences, heightened by professional rivalry, can serve as a catalyst for doctrinal discussions, an opportunity to build leader cohesion, and a chance to compare tactics and techniques among professionals. Close Combat simulation is a great tool while deployed either on ship, exercise, or overseas.

2.3 Behaviour Modeling in MOUT Simulations

Both FSL and CSM provides extremely immersive and engaging simulations for training. Computer games technology are used to develop FSL and CSM. Like in computer games, the strengths of FSL and CSM lies mainly in the realism of their graphical aspect and game play. Since the challenge for graphical realism had been largely addressed, we believe the next step forward is to improve the behaviours of the bots during simulations.

A behaviour is a set of responses to external events (often referred to as the “stimulus”) that is designed to reach an end goal. Each behaviour has been designed to handle different responses to changing stimuli. Properly designed behaviors will counter all possible responses in order to carry through to the desired result.

George and Cardullo did work on modelling behavioral representation in synthetic forces. These models should incorporate the spectrum of human bio-mechanical, physical, psychophysical and psychological parameters, responses and interactions. Synthetic battlespaces composed of physical (aircraft, sensors etc.), environment (weather, dynamic terrain etc.) and these human models. Currently the human models are not as sophisticated as the physics based physical and environmental models. This is in part due to the fact that human behaviour represents highly complex nonlinear and adaptable systems. Conventional approaches of state machines and expert systems have been applied to computer generated forces (CGF). The result in some cases has been synthetic force portrayals that are not totally autonomous, are unrealistic and do not support the full DoD requirements for analysis and training.

Realistic bots behaviours can increase training value during MOUT simulations. Laird and Wray explored the benefits of building intelligent bot behaviours [15]. They stated that intelligent behaviours will be able to provide variability of behaviors, tactical considerations, situation awareness and group coordination during MOUT training. Our work targets similar areas of benefits as Laird. Ourston and Reece shared their experiences that occurred during the development of the Dismounted Infantry SAF (DI SAF) portion of the Distributed Warrior Network (DWN) project [16]. They discuss some of the technical issues involved in adequately modeling artificial combatants when combined with virtual combatants in the same simulated battlefield environment. Furthermore, they made recommendations on future improvements on current MOUT simulations. These recommendations include inside-building behaviours, visual processing, movement, damage models and terrain reasoning. Although they are working on different projects, Laird and Reece had both highlighted the need for more realistic bot behaviours

in MOUT simulations. This concurred with our view that much more work needs to be done to produce intelligent behaviours in MOUT simulations.

2.4 Impact of Human Factors on Military Simulation

Human factors is a critical consideration during military simulations. Military models often prefer to concentrate on hard numbers. Although a warship can be treated as a single entity with specified parameters such as speed, armour or fire power, urban warfare often depends on the behaviours of small squads or individual soldiers. In these situations, training, morale, intelligence, and personalities (leadership) will influence the outcome of the combat. The sheer number of variable creates a big problem for urban warfare simulation. Commercial wargames had often tried to take these factors into account. A valid criticism of some military simulations is that human factors are often ignored. This could be due to the fact that they are very difficult to model accurately. Furthermore, few commanders like to accept that his troops may disobey him. To overcome this shortcoming, military analysts had considered civilian wargames as being more realistic, in their approach to military simulations. In 1980, James F. Dunnigan was invited by the Pentagon to work with Science Applications Incorporated (SAI) and Rand on wargaming simulations. He was a prominent military analyst and founder of Simulations Publications Incorporated. With this team, they developed a more realistic military simulation model known as SAS (Strategic Analysis Simulation). This model is still being used.

Military simulation models should be used carefully. They generally attempt to aid the decision-making and preparation process. Putting too much trust in inaccurate

military simulation can be very dangerous. The Vietnam War was intensively gamed between 1964 and 1969 in a series of simulations codenamed Sigma. During this period of time, great emphasis was placed in the value of military simulations. This can be attributed to the proven success of operations research during World War II and the increasing capacity of modern computers in processing enormous amount of data. When Nixon took over government in 1969, all the data relating to both countries was fed into a computer model. The information included total population, gross national product, military strength, manufacturing output, tanks and aircrafts. The computer model was used to determine when can US win the war. To the surprise of the analysts, the computer model had infamously predicted that US should had won the war at 1964.

2.5 Behaviour Modeling for Military Simulations: Requirements for Intelligent Bots

To produce intelligent behaviors, we intend to work on a few key areas. These areas include decision making, autonomous navigation, terrain reasoning, expectations and tactical behaviors.

The Defense Modeling and Simulation Office of the U.S. Department of Defense requested the National Research Council (NRC) to set up the Panel on Modeling Human behaviour and Command Decision Making: Representations for Military Simulations. The role of the panel was to review the current state of the art in the human behaviour representations for military simulations. The key emphasis of their review are in the important areas of human behaviour simulation such as cognitive and organizational behaviour. The panel was made up of experts in organizational behaviour, individual behaviors, decision making, computational

modeling, human factors and military simulations. The project was done over a period of 18 months. The panel argued for the need to create models of human behaviour, summarized a rigid methodology for ensuring the consistent development of useful models and identified the psychological process models that can help in the representation of realistic human-influenced action. The panel also did a detailed analysis of the applied and theoretical research in human behaviour modeling at the individual, unit and command levels [18].

The panel produced a final report that states that, for the modeling of individual combatants, it is important to represent the processes underlying the observable behaviour, including attention, decision making, perception, situation awareness and planning. The report further supports the need for our key areas of work and the potential benefits that they may bring.

Tambe [19] have provided some insight into building intelligent behaviours for simulation environments. Tidhar considered the insights and produced a more detail set of requirements. He identify four types of requirements necessary for a model of a human involved in military scenarios: (1) ability to interact with the environment; (2) ability to exhibit rational behaviour when reasoning about the world; (3) ability to exhibit irrational behaviour; and (4) ability to provide a good simulation environment.

Based on their work, human use a variety of techniques and methods when reasoning about the world. These include building and maintaining situation awareness, planning, pursuing multiple goals simultaneously, and interleaving pro-active and reactive behaviours. Due to these considerations, Tidhar require that the military simulation system include the following features:

- (a) Building and Maintaining Situation Awareness: The ability to analyze the model of the world and identify particular aspects that require a response.

- (b) Decision Making and Reasoning: The ability to perform complex reasoning, e.g., make decisions, plan, perform spatial and temporal reasoning, etc.
- (c) Simultaneous Goals: The ability to hold multiple goals and interleave their achievement.
- (d) Proactive and Reactive: The ability to react to the changing world and to interleave pursuing goals and reacting to the world.
- (e) Explanations: The ability to provide clear and high level explanations of the way reasoning is performed.
- (f) Levels of Knowledge: The ability to model different types and levels of knowledge (e.g., knowledge about the world and about how to behave in the world).
- (g) Real-Time Performance: The ability to perform activities in a time scale comparable to human activity.

The requirements specified by Tambe and Tidhar coincide with the requirements of this project. This further confirms our research direction of developing intelligent bot behaviours.

2.6 Decision Making

Decision making is a major component of our research. In particular, time critical decision making is important to the military domain. In this section, we will review the work done in developing human-level decision making before discussing time critical decision making. We recognize the fact that there are a lot of different decision making approaches. For the purpose of this research, we are not aiming at providing an exhaustive review on all decision making approaches. Therefore, we will focus on some influential approaches that have been applied in military simulations.

2.6.1 Fuzzy Logic

Fuzzy logic is an emerging field of theory and application that can be very important for the control of systems. This is especially useful for those systems which cannot be described mathematically or which are nonlinear in nature. The strength of fuzzy logic lies in its heuristic approach to control. Instead of requiring complex mathematical equations which describe a system's behaviour, fuzzy logic allows a set of common sense, plain-English rules to generate a desired system response. Fuzzy logic is a form of multi-valued logic derived from fuzzy set theory to handle approximate reasoning. In contrast to fuzzy logic, binary sets with binary logic hold variables should have a membership value of only 0 or 1. In fuzzy set theory with fuzzy logic the set membership values that comes from the range (inclusively) between 0 and 1. Therefore, the degree of truth of a fuzzy logic statement can range between 0 and 1 and is not constrained to the two truth values true (1), false (0) as in classic predicate logic [26].

The first paper on fuzzy logic (defined as a multivalued logic based upon set theory) was published by R.H. Wilkinson in 1963. He first proposed the concept in his 1961 Electrical Engineering master thesis [28]. He started to redefine the earlier multivalued logics in terms of set theory. His main purpose was to demonstrate how analog logic can be used to simulated any mathematical functions using hardwired analog electronic circuits. He did this by building a number of linear voltage ramps. These ramps were then fixed in a “logic block” using diodes and resistor circuits. The “logic block” simulated the maximum and minimum fuzzy logic rules of the AND and the INCLUSIVE OR operations. In 1965, Lotfi Zadeh popularized the fuzzy logic of Wilkinson's work without using the electrical circuits set-up. The term “fuzzy logic” emerged in the development of the theory of fuzzy sets by Lotfi

Zadeh [27]. To date, fuzzy logic has been applied to diverse fields from control theory to artificial intelligence. Despite its broad usage, fuzzy logic still raises many questions among statisticians who prefer probabilistic logic and control engineers who prefer traditional two valued logic.

Fuzzy provides an extremely simple way to draw definite conclusions from vague, ambiguous or imprecise information. In a sense, fuzzy logic closely imitates the human decision making process with its ability to handle approximate data and find precise solutions. Classical logic demands in-depth understanding of a particular system, mathematical formulas and precise numerical values. However, fuzzy logic incorporates an alternative manner of thinking which allows the modeling of complex systems using a higher level of abstraction based from expert knowledge and experience. Fuzzy logic enable designer to express this knowledge with subjective concepts such as hot, hard or very bright.

Fuzzy logic shows some potential for the generation of human-level AI. Humans do not normally think in absolute terms. Although drivers are taught to signal a turn precisely 100 feet before the intersection, people had never got out on the road and measure the exact distance.

The same is true with military or strategic axioms. It is “good” to get behind an enemy. It is good to cut an enemy’s lines of supply and communication. However, there are no precise mathematical definitions for the terms. It is reasonable to expect that fuzzy logic will play an important role in the human-level AI development. The Department of Defense has funded many computer war games (WARSIM, JWARS, TACOPS). However, they are generally lacking in strategic or even tactical AI.

In the simulation of the command and control area, fuzzy logic had been used for situational assessments and decision making. An example is work done by

the JWARS office to create autonomous, heterogeneous agents for the purposes of better differentiating unit behaviors at the lower levels of JWARS [105]. The agents represent a wide range of military units and use a knowledge base of facts, rules, and associated actions to reason about their own situation, the enemy situation, and the environment.

JWARS has developed a methodology using fuzzy logic for generating a rich set of heterogeneous units using agents that operate within a hierarchical organization and act cooperatively to accomplish large scale missions. Individual units can each have their own set of rules that interpret the local information, consider constraints placed on them by higher authority, assess the local situation in the context of the larger situation using their own rule sets, and generate appropriate actions or behaviors as the scenario progresses [106].

While JWARS benefits high-level strategic commanders, our work is focused on aiding the squad-level tactical commanders. The main differences in requirements include time latency during decision making, bot behaviours and heterogeneous cues representation.

There are potential benefits in applying fuzzy logic to our work. A possible area that may require fuzzy logic is for the bots to attain situation awareness during MOUT simulations. The complex MOUT environment means that our bots need to handle heterogeneous cue types and generate situation awareness from them. Fuzzy logic could aid in the comprehension stage of generating situation awareness. Some limitations of fuzzy logic includes stability and learning capabilities. Since fuzzy logic is not formal, it is difficult to ensure that a fuzzy system will function correctly. Since fuzzy systems has also no memory to learn during problem solving and cannot learn membership functions, we do not aim to impart learning capabilities in this

research. The focus of our current work is to emulate the behaviours of expert soldiers in MOUT simulations.

2.6.2 Hierarchical AI

Hierarchical approaches to developing AI systems is not a new idea. For example, Brooks subsumption architecture uses a hierarchy of behaviours within a single agent [29]. The application of hierarchies for military organisation is considerably older. Throughout human history, armies have long been organised along very strictly defined hierarchical principles. The ancient Roman army is a well-known example.

Hierarchies are also very natural control structures for use in games. They allow the development of separate AI systems for controlling high level strategic behaviours of the army and for low level tactical behaviours which can be applied to individual units [30]. Hierarchical AI can also be used more generally in other games where very large organisations are modelled [31]. Games which have small groups of units under some form of group command can also benefit from hierarchical AI [32]. In an AI hierarchy, the uppermost level will be some form of command agent with overall strategic control. The lowest level will be an agent AI responsible for the control of a individual soldier and there may be any number of intermediary layers of AI representing command over groups of individual soldiers (such as squads) and progressively larger groupings of combat units.

For control of small squads, Sterren considers a de-centralized approach whereby squad agents communicate and form emergent strategies, such as for fire-and-move behaviours. The advantages and disadvantages of the decentralized approach are illustrated using an example of having the squad wait in ambush. The key limitation

appears to be that there is no single AI responsible for the strategic deployment of the squad. A solution to this is presented in the following paper, [32]. Sterren modifies the architecture from a self-organising one into a simple hierarchical one. This presents two distinct command styles, Firstly, the authoritarian style, where the member agents always perform the actions they are commanded to do. Secondly, the coaching style, where a higher level squad AI sends requests to the bots. These bots then evaluate them according to their current circumstances before deciding which action to perform. It is suggested that a third style is a hybrid of the above two styles might be the most successful.

At the top level is a human or AI game player, which makes the highest level decisions. Below the top level, there are AIs for individual armies, divisions, brigades, companies, platoons and squads as required by the sizes of the forces involved and the complexity of the game. Finally, at the bottom level of the hierarchy are AIs to control the behaviours of individual soldiers.

Ramsey [34] presents a similar approach, less specifically focused on military organisation, which he names it the “Multi-Tiered AI Framework”, MTAIF. It contains four levels of intelligence: strategic, operational, tactical and individual unit. Within this generalised framework, the AI structure is quite similar to that advocated by Kent[30]. However, the tasks to be carried out by the different units have greater variety. They include resource gathering, city and empire building. This paper provides much discussion to solving the problems of message passing and coordination with large hierarchies. One notable aspect of these AI architectures is the absence of any form of learning. One of the rare instances of automated learning during war games simulation is seen in an online report [35]. This report presents work on developing command agents for strategy simulations using Cognitive Work Analysis and Machine Learning approaches.

A limitation of hierarchical reasoning is that the solution is not chosen based on external objective measures. In fact the solution is chosen based on the number of remaining solutions available. Therefore, the quality of the solution could drift into poorer solution due to dynamic changes in the environment. We will need to ensure our architecture can mitigate this risk. As discussed in this section, hierarchical AI is important in modelling the human-like behaviours. The later sections will discuss how hierarchical structures may be crucial in forming our frameworks for decision making or autonomous navigation.

2.6.3 Military Decision-Making Process

The military decision-making process (MDMP) is a single, established, and proven analytical process [107]. The MDMP is a tool that assists the commander and staff in developing plan and estimates. Formal problem-solving process may begin once a new mission is received. While the MDMP seeks to produce an order immediately, the analytical aspects of the MDMP continue at all levels during operations. The MDMP helps the commander study the battlefield situation and reach logical decisions. The process helps them apply thorough, sound judgment, good logic, and professional knowledge to achieve a solid decision. The full MDMP is a sequential, deliberate and time-consuming process. It is only used when adequate planning time and sufficient staff support are given to study numerous friendly and enemy courses of action (COAs). This typically occurs when developing estimates for the commander, formulating operation plans and during staff training designed specifically to teach the MDMP. The MDMP is shown in Figure 2.3.

However, the MDMP cannot produce plans and orders quickly enough for the squad level soldiers. Theoretically, the MDMP enables a commander to employ tactically sound plans leading to success on the battlefield. However, recent research reveals

Figure 2.3: Military Decision-Making Process

that the MDMP actually has the opposite outcome in most cases. The MDMP is a staff-driven planning that inadvertently isolates the ground commander from the plan formulation.

A group of research scientists from Klein Associates in Fairborn, Ohio, conducted studies of military organizations and planning. It made some surprising findings. First, with its focus on the staff process, the MDMP separated the unit commander from planning in most of the reviewed cases. Thus, junior staff officers, despite being the least experienced individuals, had to derive a workable plan. Second, contrary to conventional belief, the MDMP produced cautious and rigid plans that were not suited to the dynamic demands of the situation. Researchers attributed this to blind compliance with the doctrinal planning templates. Lastly, the MDMP slowed an organizations operational speed and affected its ability to react to rapidly changing situations. In MOUT situations, decisions must be made rapidly as the environment is constantly changing. Therefore, MDMP is not a suitable approach for MOUT simulations.

2.6.4 Recognition primed decision

The recognition-primed decision (RPD) model [40] proposed by Gary Klein emphasizes the role of experiences in human's decision-making process in various time critical situations. The model also suggests that humans 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 [41, 42, 43].

Recognition-primed decision (RPD) is a model of how people make rapid and effective decisions when faced with complex situations. In the RPD model, the human subject is expected to develop a probable course of action. The proposed course of

action is then compared to the constraints enforced by the situation and the first course of action that is not rejected will be selected as the solution. This technique is rapid but may result poor decisions in certain unique or misidentified circumstances at times. Nevertheless, this approach has its benefits and does seems to be a valid model to simulate the human decision-making process.

The RPD model identifies a possible solution to a situation as the first one that is retrieved immediately from previous experience. RPD develops a decision in two manners. Firstly, the course of action that is reasonable is recognized. Secondly, the course of action is evaluated through imagination to determine if the necessary actions are reasonable.

In the RPD model, experience plays an important role in the decision-making processes. RPD reveals a critical difference between how experts and novices handle recurring situations. Experienced individuals can typically recognise the situation better and generate quicker decision. This is because the situation may match a previous situation that they had been through before. Without the benefit of such experiences, novices are forced to consider a huge range of possibilities. Generally, they tend to select the first course of action that should be able to work. The inexperienced individuals also have to apply the method of using trial and error with their imagination.

Recognition primed decision making closely match the decision making process of the commanders in various organizations that are involved in time critical situations such as fire fighting, search and rescue missions, police raids and other emergency operations. The RPD model can be used to simulate the decision making processes of both the experienced and the inexperienced personnel. The recognition primed decision life threatening situations. The studies developed from the model can be

used as samples for the organizations to predict the possible outcomes of important operations.

Although RPD shares similar goals as the current research, it is not ideal for MOUT simulations. Firstly, RPD is not built to handle the time constraints during MOUT simulation. Secondly, there is no computational model for RPD in MOUT simulations. A model that can address these weaknesses is needed.

2.6.5 SOAR

Soar is a symbolic cognitive architecture, created by John Laird, Allen Newell and Paul Rosenbloom at Carnegie Mellon University [112]. It provides a method to develop realistic cognition through a computer programming architecture for human level AI. Since its beginnings, it has been commonly applied by AI researchers to develop various aspects of human behaviors.

The Soar project aims to be able to manage the full suite of capabilities of an intelligent bot. With the proper capabilities, the bots should be able to handle not only the routine tasks but also the challenging open-ended problems. In order to achieve the goal, it needs to be able to generate data representations and use the relevant information to generate the necessary knowledge in their correct form (such as procedural, declarative, episodic). Soar should also address the group of mechanisms representing the process of the human mind. An important principle for the Soar architecture is the idea that general intelligence is best simulated through a symbolic system. This is called the physical symbol system hypothesis. The principles of cognition guiding Soar architecture can be linked to the psychological theory proposed by Allen Newell in Unified Theories of Cognition [113].

In Soar, all long-term knowledge is encoded as production rules, while the current situation (perception, situational awareness, mission information, and goals)

is encoded in declarative data structures comprising its state. Problem solving is basically an exhaustive process of searching through a pre-defined problem space (the repository of various states that can be accessed by the system at a given time) to find a goal state (which holds the solution to overcome the current problem). This is achieved by constantly searching for the states that can help to bring the system nearer to its intended goal gradually. Each search is made up of a decision cycle containing an elaboration phase. The elaboration phase consists of a variety of knowledge on the problem that exist on Soar's working memory and a decision procedure. This decision procedure weighs the findings from the previous phase and assigns preferences to decide the correct action to be taken. The rules in Soar match against the state and propose operators to apply to the current situation. Primitive operators send commands to the motor system. Complex operators are dynamically converted to subgoals that are then pursued by once again having rules select and apply operators in the subgoals, eventually resulting in operators that execute primitive actions.

Soar seeks to achieve general intelligence for agents in various situations. However, there is no claim that this ultimate goal has already been achieved. Researchers who are familiar with the system recognize that Soar is still lacking in some key aspects of intelligence. Some examples of missing capabilities include automated generation of new data representations during simulation and application hierarchical clustering to enable rapid cognition. Furthermore, the application of SOAR in MOUT simulations is still limited at the moment. Although Wray employed the SOAR architecture to create the MOUTBots, his main focus was to build bots that can exhibit variable behaviours as a result of a random probability function. While his aim was to generate variable behaviours, our work is to generate realistic behaviours. His work also did not consider the requirement of handling the rapidly changing environment and time critical decision making during MOUT combat.

2.6.6 Time Critical Decision Making

Soldiers often need to recognize the current situation with incomplete information. After situation recognition, they have to make rapid decisions under time pressure, uncertainty, high stakes and changing conditions [36, 37, 38, 39]. As pointed out by Klein in [40], 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.

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. Laird and Wray had done much work in recording and encoding the variability in human behaviour models for military simulations [11, 36]. 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.

Humans often makes assessment on their current situations and generate certain expectations on the impending events in their situations. Schmitt and Klein emphasized the impact of stress and uncertainty on decision making [82]. It can be observed that expectancies plays a key role in influencing the judgment and, therefore, the decisions of soldiers. Warwick incorporated expectancies in his design of

the memory representation in RPD computational models [81]. The expectations produced can be used to select the course of actions in his framework. Ji developed a fuzzy logic-based RPD model that attempts to employ expectancies for the verification of the accuracy of judgments. Adjusting in the course of actions may be required if the expectations conflicts with the observed facts [83].

To improve the intelligence of his bots, Laird added anticipation to Quakebots [104]. His bot is distinguished by its ability to build its own map as it explores a level, use a wide variety of tactics based on its internal map, and in some cases, anticipate its opponent's actions. Laird found that instead of trying to encode behaviors for each specific situation, a better idea is to attempt to add a general capability for anticipating an opponent's actions. From an AI perspective, anticipation is a form of planning; a topic that researchers in AI have studied for 40 years. The power of chess and checkers programs comes directly from their ability to anticipate their opponent's responses to their own moves. Anticipation for bots in first-person shooters (FPS) has a few twists that differentiate it from the standard AI techniques such as alpha-beta search.

In terms of challenges and end results, our work on modeling expectations is similar to Laird's efforts to add anticipation to his bots. However, while his goal is to produce intelligent bots to win FPS games, our goals are to simulate the end-result and procedural realism of a human soldier in MOUT warfare. The aim of including expectations is to further streamline the decision making process of our model towards the human brain. The expectations will serve to influence the course of actions and verify judgments.

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

It has been observed that humans 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 [44]. Since these schema are context specific, they are dependent on an individual's experience with and exposure to a subject area rather than simply some "raw intelligence" [46, 47]. In the proposed framework, we explored the concept of schema to organized the experiences of a bot in MOUT simulations.

Gladwell observed that humans often perform *thin slicing*, that is, they rely on some key cues of a complex and uncertain situation to achieve rapid situation awareness and to make intuitive decisions [48]. He also pointed out that collecting more information may not help to make 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 [49]. We believe that handling time critical decisions is an important aspect of human decision-making in time-critical and complex situations. To our best knowledge, our current work is the first to incorporate this aspect in modeling human decision-making process for MOUT simulations.

2.7 Situation Awareness

Endsley's framework of situation awareness is a widely accepted abstract framework of situation awareness [97]. According to Endsley, there are three main stages for the formation of situation awareness which include perception, comprehension and projection. In the perception stage, key cues from the environment are picked up by a persons sensory and attention system. These cues will then be used for the person to understand the current situation in the comprehension stage. A person

also has the capability to predict how situation may evolve (i.e., projection into the future). It should be noted that Endsley’s framework describe situation awareness at high and abstract level, thus it leaves much room for a modeler when developing computational models of situation awareness for specific applications.

Like in the perception stage of the Endsley’s model, Hill attempts to model the perceptual attention in virtual humans [98]. Similarly in [99], Herrero et al. introduced human-like hearing perception for intelligent virtual agents. With similar goals in mind, we adapted the sensory functions of the Unreal game engine to create the sight and hearing senses for our bots [100]. The comprehension stage of the Endsley’s framework requires the transformation of the perceived information into relevant values for situation understanding. McCarley et al. [101] and Warwick et al. [102] separately developed computational models to support situation awareness. With their models, they seek to achieve greater simulation realism. However, their models are not catered towards the specific needs of MOUT simulations. For instance, McCarley’s model does not work with the case based reasoning process, and Warwick’s model can not handle heterogeneous cue types which are typical for MOUT.

For evaluation purpose, Endsley designed the Situation Awareness Global Assessment Technique (SAGAT) [103]. SAGAT provides an objective measure of situation awareness based on queries made in simulation freeze. We will apply SAGAT methods to evaluate the effectiveness and realism of our proposed model.

2.8 Autonomous Navigation Systems

In MOUT simulations, the virtual urban environments are populated with characters and buildings. While some characters are controlled by human players, most of

the characters are non-player characters (also know as bots) which should demonstrate some level of human-like tactical behaviours, e.g., tactical path planning and movement. Therefore, autonomous navigation systems are crucial to MOUT simulations [50].

Autonomous navigation system has been studied extensively by researchers in the entertainment industry and the academia. Existing path finding algorithms such as the A* algorithm [51] and the Dijkstra's algorithm [52] are essentially aimed at finding the shortest path in static environments. These algorithms are not adequate to generate realistic movements for the bots in MOUT simulations mainly due to their high computational cost in large and dynamic environments.

Penn and Turner proposed an exosomatic visual architecture to model pedestrian walking behaviours [53, 54]. However, we feel that their model is not adequate for tactical movements. In [55], Cohn suggested that it is relatively easy to translate common sense knowledge into some qualitative terms as compared to quantitative terms. This implies that qualitative spatial reasoning may be more suitable for generating realistic and human-like behaviours. How to efficiently generate human-like tactical behaviours for the bots attracts many researchers particularly in the game industry as an interesting and challenging problem. Reece, Kraus and Dumanoir proposed a multi-step path-finding process for tactical movement planning [56]. However, their approach is inadequate to deal with dynamic environments.

Voronoi diagrams have been proposed for spatial analysis in urban environments [57]. Schussman and Bertram further refined Voronoi diagrams to construct a hierarchical data structure for gridless, scattered data [58]. However, their approach was not used for path planning algorithms.

2.9 Terrain Reasoning

Terrain reasoning is the capability of the bot to consider the impact of terrain during its planning, decision-making, action selection and team communications. For the bot behaviours to be intelligent during simulations, terrain reasoning is essential [85].

Consider the following scenario. A bot begins with an arbitrary position and orientation in an out-door environment. There is a goal location. The bot moves toward the goal avoiding obstacles, ducking under low branches, climbing over objects, avoiding difficult terrain where feasible and squeezing through tight spaces where necessary. In some situations the bot tries to avoid the sensory field of one or more hostile agents.

The aim of terrain reasoning is not to tell the bot how to achieve its goal. Instead, we make the bot aware of its environment and associate a set of actions with different perceptions. a decision is made as to where the agent should move. This decision is made incrementally after each step the agent takes so that a dynamically changing environment can be supported [120]. For example, chasms may open up, trees may fall down, pits or craters appear or the goal may change.

Terrain reasoning systems are generally made up of the below components:

- Terrain representation
- Functions that can query, construct or manipulate the terrain representation

Terrain reasoning should be addressed with custom adaptations to suit the specific situation that the bot is in. This will ensure that the terrain reasoning can be fine tuned for the specific needs of the AI and characteristics of the simulations.

A commonly used method to describe terrain are waypoints. Similar ideas such as cells or grids had also been widely used [56, 59]. Waypoints represent the subset of the accessible terrain for various entities in the simulation. Viable movement are denoted by the connections between these waypoints. The resulting graph formed by the waypoints provides the valid paths for the bots to traverse during simulation. The waypoints can become part of more abstract representations such as areas and portals. Therefore, waypoints are a robust and flexible terrain representation. Other than the shortest path from one point to the other, waypoint graphs contains much more information. The combination of waypoints can represents the shape of the simulated terrain. When added with line-of-fire and line-of-sight data, waypoint graphs can be even be utilized to express various actions that combine to make up tactics.

The relationship between the waypoint graph and tactical behaviours allows models to form meaningful properties. These properties can be eventually used for the computation required to generate the tactical behaviours from the bots.

Every waypoint has three characteristics that should be considered:

- Local environment. Waypoints describe, for example, the lighting of the location, the movement types required (squatting, swimming, climbing), whether a door is present.
- Membership(s) of higher-level terrain concepts. Waypoints can be organised for the representation of complex terrain such as a room, a river, or the army camp.
- Relations with other waypoints. There are alot of relations between waypoints: is there a possible line-of-sight, time taken to travel from one waypoint to the other.

Bot with good terrain reasoning should be able to understand terrain and relevant concepts such cover, movement time, terrain zones. In MOUT simulations, AI and terrain play a particularly big role. Effective terrain reasoning results in a richer bot behaviours. Terrain reasoning is made up of terrain representation and related algorithms. These algorithms help the bots to create, query and manipulate the terrain representation. Apart from providing an effective method for navigation, waypoint graphs had been proven to be a convenient means for terrain reasoning and representation [121].

Many military tactics places great considerations on the terrain. In military simulation, the waypoint information can help the bots to determine the effectiveness of these tactics. With the relevant equations to express the waypoint graph properties, the tactical terrain concepts such as sniping points, ambush zones and bomb attacks can be rapidly computed and easily identified by the bots.

The reasoning about tactical concepts can be easily extended to include player and bots behaviour local to the waypoint. This activity information allows the bots to adapt its tactics during simulation. Adaptation is a result of re-enforcement learning. The greater understanding of the tactical significance of various locations also help the bots to adapt their tactical behaviours in the simulations.

The waypoint based terrain reasoning are used in real-time. As an example, they can be used to predict threat locations and select the proper tactics. Terrain reasoning can also be used to pre-process terrain information in order to allow the simulation to off-load certain data that can assist the run-time bot behaviours.

2.10 Most Recent Research

In this section, we include the latest work published in related areas of our research. Some of these work had been published after the submission of this thesis.

Luo had been doing ongoing work on modeling agents with human-like decision making and behavior execution capabilities in crowd simulation [87, 88]. He aimed to provide a generic framework that reflects the major cognitive and physical processes as observed from human behaviors in real-life situations. The design of the framework is based on some basic assumptions and related cognitive theories on human behaviors in various real-life situations. Recently, Hu presented a novel pattern-based framework to address the challenge in modeling lower-level pedestrian navigational behaviors [89]. In this framework, an agents perceived information and subjective judgment on the situation are summarized as spatial-temporal patterns. An agent will match the perceived patterns with some prototypical cases in its experience. That is, an agents decision-making process is essentially treated as a pattern-matching process. Both Linbo and Hu's work shares similar goals in generating human-like behaviours. We will study their work and find out if they can be used to improve human-like decision making in MOUT context.

In 2009, Gemrot and his team presented Pogamut 3, an open source platform for rapid development of behaviour for virtual agents embodied in a 3D environment of the Unreal Tournament 2004 videogame [90]. Pogamut 3 is designed to support research as well as educational projects. Pogamut is a useful toolkit for developing educational scenarios concerning orientation in urban areas. These improvement make Pogamut 3 applicable beyond the domain of computer games. With similar goals in mind, Gorman produced QASE API to offer a general purpose library for the creation of game agents, in the hope that the availability of such software would help stimulate further interest in the field [91]. Though geared towards machine learning, the API would be flexible enough to facilitate multiple forms of artificial intelligence, making it suitable for application in research and in undergraduate

courses centering upon traditional AI and agent-based systems. Such contributions will facilitate and encourage greater research in bot behaviours modeling and enhance the value of our work.

Hingston designed a Turing test to differentiate between bots and human players in computer games [93, 92]. In his new version of the bot Turing test, the judges are simply game players and judging is an inherent part of the game. His method is obviously a bot version of the famous Turing Test, first proposed by Alan Turing as a thought experiment in which a judge has a chat session with a human and a computer program, and then has to nominate which was which [24]. A major difference is that Turing's test was proposed as a test of the computers intelligence, while the his method is a test of the bots ability to give the appearance of being human. It is noted from his results that the opinions of human subject matter experts is a better measure of human realism as compared to other quantitative means. His method is mainly useful in computer games. In our experiments, we will adopt a similar approach to validate our model for human-like realism in MOUT simulations.

Chapter 3

SNAP -Time Critical Decision Making Framework

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 MOUT simulations are 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 section, 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 rapid cognition. CBR is used to make quick decisions by comparing the current situation with past experience cases. Rapid cognition 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.

3.1 Motivation

The key to producing realistic behaviours lies largely in the decision making process of the bots. Currently, conventional game AI techniques had been used to design bots in MOUT simulations. Although game AI is able to provide challenging opponents for gamers, they may not be realistic for simulation training. Building realistic behaviours for simulations require special considerations for the situations involved. An important factor that differentiates the behaviours of human in MOUT situations and normal life is the time constraints. As human are likely to make different decisions given different time frames. An optimal decision can be made with a long time frame and a satisficing decision is produced under time constraints. Although there is already a lot of work done in producing optimal decisions in computer simulations, the research on producing satisficing behaviours in MOUT simulations is still lacking. Producing optimal behaviours under time constraints is unrealistic and serves little operational benefits to the soldiers. Our work seeks to bridge this gap by producing a time critical decision making framework, *SNAP*, for bots in MOUT simulations.

3.2 Features of *SNAP*

The novel features of the framework include case-based reasoning (CBR) and rapid cognition. 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 an effective method for machine reasoning. It is also a pervasive way for humans to make decisions in everyday situations [41].

In the proposed framework, rapid cognition 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* [48], Gladwell discussed how humans are able to achieve rapid situation awareness using only a narrow slice of information. Using this limited information, humans can focus on the most significant information (or key cues) about a situation, and make judgments 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 a similar approach 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 rapid cognition) of the proposed framework could help a bot to make decisions in time-critical situations in MOUT simulations. Figure 3.1 shows a soldier bot and a militant bot in a close combat situation.

Figure 3.1: 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 rapid cognition. 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.

Another important feature of *SNAP* is *expectations*. Real MOUT situations are highly dynamic, thus the decision making process of a soldier is never a one-step process. A solution may become invalid as the situation evolves. In this case, a soldier may re-assess the current situation and attempt to find new solutions. We believe that expectation plays an important role in this regard. In MOUT warfare, by matching the current situation cues with his/her experience cases, a human soldier may have a quick assessment of the current situation and find a solution to deal with the current situation. In the meantime, the soldier will also form up expectations on the likely future events. As the situation evolves, if the expectations are not fulfilled or violated, the soldier may infer that his/her initial solution may be invalid due to dynamic situational changes. Then the soldier may abort the previous solution and find a new solution. Without a mechanism to assess their solutions during execution, the bots can only know the validity of the solution after the execution of the complete solution. This could result in the bots blindly executing the initial solutions even though they are no longer effective.

Therefore, we propose to handle the dynamic situational changes with expectations. In this work, we consider expectations as some events that are likely to occur in the near future. Bots continuously observe the ongoing situation after a solution is selected. If the observations are consistent with expectations, then the solution will be enhanced. Observations that do not match expectations may lead to the invalidation of the solution. In this case, the bots re-assess the situation to find new solutions and form up new expectations. The feature of expectations is designed to work in tandem with the existing features of CBR and rapid cognition. Rapid cognition will pick up key cues from partial information and pass these cues into the *CBR process*. CBR will match the cues with past experiences to retrieve the relevant experience cases. *Expectations* are formed up along with the past experiences. The bots will use these *expectations* to assess the validity of the solutions in the face of rapidly changing situations.

Now let us use a simple example to illustrate how these features of the *SNAP* framework could help a bot to make decisions in time-critical and dynamic MOUT situations. Figure 3.2 shows a sniper bot aiming at a soldier bot.

The soldier bot needs to make a rapid decision to counter the sniper. Using the *CBR process*, it 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 to the new situation, it will be retained in the memory 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. These cues may include the location of the sniper, whether the sniper is in open area or hiding behind a wall, the weapon being used and strength of the fire, etc.

Figure 3.2: Enemy sniper bot taking aim at soldier bots

Figure 3.3 shows the simulation time line of a group of soldier bots being attacked by a sniper and their subsequent actions. It is likely that the enemy sniper is attacking the soldier from a building. According to doctrine, a human soldier will attempt to take cover and form up with his squad before performing a counter sniper operation to neutralize the enemy threat. During the counter sniper operation, the soldier is expected to overpower the enemy through speed and superior fire power. If there are more enemies or stronger firepower than the soldier expects, then the initial solution to perform a counter sniper operation may become invalid. The soldier bots select the solution to perform a counter sniper operation at T_1 . With this in mind, they hold a set of expectations on the enemy behaviour, e.g., the enemies are adopting a hit-and-run tactics and will not be waiting at the sniper location. They do not expect to face heavy enemy fires during the counter sniper operation. At T_2 , an unexpected threat of *Close Combat Fire* is observed. This new threat violates the current expectation and therefore invalidates the counter sniper solution. The

Figure 3.3: Dynamic changes during MOUT simulations

violation of the expectation implies that there might be an ambush waiting for the counter sniper squad and a new solution will need to be generated.

3.3 Design of *SNAP*

Figure 3.4: Decision Making Framework

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

As shown in Figure 3.4, 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 determine which information is relevant for situation awareness (more details will be given in Section 3.3.2). 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 will be retained in the *Experience Repository* as a new experience case for future usage.

3.3.1 Case-Based Reasoning 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 $\langle threat, solution \rangle$ sets as shown in Figure 3.5. 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 rely 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 Section 3.3.2.

The threat in an experience case holds pre-condition cues which act like a pattern (or schema) for the bot to recognize the threat. In *SNAP*, these pre-condition cues mainly consist of some descriptions about the situation. The cue values can be nominal, quantitative or qualitative. Similarity matching is then conducted by matching the evidence cues of the current situation with the pre-condition cues.

The key cues picked up by the bots are used to form an *evidence set*. When the evidence cues match the precondition cues of an *experience case*, the corresponding solution of the case will be used. The solution generally activates a new mission.

Figure 3.5: An Experience Case

Each mission is organized into several phases to represent the temporal relationships between different parts of the mission. In each phase, there are some *expectations* providing constraints to the mission. The *expectations* are monitored while the mission is being executed. If the *expectations* are not met, the next phase may not be executed and the current mission could be under threat. In this case, the proposed solution may need 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:

$$[Bot(A, Militant, Male, Snipergun, VeryFar) \wedge Weapon_Fire(Snipergun)]$$

Now suppose that *Ted*'s current evidence set (we will explain how it is formed in Section 3.3.2) is:

$$[Bot(A, Militant, Male, Snipergun, VeryFar) \wedge Weapon_Fire(Snipergun)]$$

The evidence set states that *Ted* had spotted a militant bot at a “very far” distance and also observed sniper fire. The bot uses a spatial representation based on human spatial cognition. The bots can represent things in two fundamental ways: where something is relative to the agents location, or egocentrically (e.g., something is to my left); or where something is in absolute terms relative to a world coordinate system, or allocentrically (e.g., something is at a specific latitude/longitude).

This combination of egocentric and allocentric representations has previously been used successfully in systems such as TAC-AIR Soar [114]. The egocentric representation of an item includes both the distance to the item and its relative bearing. Distance and bearing are generally represented qualitatively. Qualitative representations provide certain advantages for applying reasoning and logic to planning and decision-making. In detail, distance is represented as descriptive distance to the target. Studies had shown that soldiers tend to demonstrate the ability to generate numerical distance estimates that scale linearly with increasing distance within the ranges typically encountered in MOUT [115].

Figure 3.6: Descriptive Spatial Representation

A descriptive distance, as seen in Figure 3.6, is how distant something is relative to the current visual horizon, and ranges across “here” (within $1/8$ the distance to the horizon), “near” (between $1/8$ and $1/4$ the distance to the horizon), far (between $1/4$ and $1/2$ the distance to the horizon), and “very far” (beyond $1/2$ the distance to the horizon). The nearest distance, ‘here’, is used to indicate that a particular location is within the immediate proximity of the bot, corresponding to the region described by as personal space. The second two distances, “near” and “far”, divide the space used for local navigation. “Very far” is generally used for very distant objects and architecture.

In the current example, the enemy sniper is described to be firing at a distance corresponding a sniper assault. 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 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 3.7. Note that with this tactic, we assume that *Ted* has some other team members so that 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 pincer movement [60].

As each action in the sequence is executed, the *Observe* component of *SNAP* will

Figure 3.7: Solution for Counter Sniper

monitor if the expectation cues of the action are satisfied. For example, the expectation set for Action 2 is:

$$[\neg Weapon_Fire(SniperFire) \wedge \neg Hit(Bot)]$$

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 expectation cues, the action is deemed to be successful and the next action in the sequence will be carried out.

3.3.2 Rapid Cognition

Rapid cognition 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. Secondly, 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:

$$[\neg Weapon_Fire(All) \wedge \neg Vehicle(Militant) \wedge \neg Bots(Militant)]$$

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:

$$[Bot(A, Militant, Male, Sniper gun) \wedge Weapon_Fire(Sniper gun) \wedge Vehicle(Friendly) \\ BotGroup(Civilian, 50) \wedge \dots]$$

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 observations $Bot(A, Militant, Male, Sniper\ gun)$ and $Weapon_Fire(Sniper\ gun)$ violate constraints $\neg Weapon_Fire(All)$ and $\neg Bots(Militant)$ 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 Section 3.3.1.

3.3.3 Expectations

When a threat is identified, expectations will tell the bots what is likely to happen next. As seen in Figure 3.5, each experience case contains *expectations*. The *expectations* consists of a group of events associated with the identified threat. Based on past experiences, the bots expects these events to occur in near future. Each event in an *expectation* is represented by its corresponding cue set (called *Event Set* in this paper). The event set contains key cues that the bots should monitor in order to confirm their expectations.

Now let us use a simple example to further illustrate how expectations work together with other modules 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:

$$[Bot(Militant, Snipergun) \wedge Weapon_Fire(Snipergun)]$$

Now suppose that *Ted*'s current evidence set is:

$$[Bot(A, Militant, Male, Snipergun) \wedge Weapon_Fire(Snipergun)]$$

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 solution will be reused by the *Action* component, i.e., the sequence of actions contained in the solution will be executed.

During the execution of the solution, the expectation frame is monitored. This is done by monitoring the cues from the event sets of the *expectations*

Solution Validation with Expectations

The *Observe* component monitors the evolving situation and forms the observation set. The observation set is made up of a set of key cues picked up by the bot. These key cues are used for matching with the cues in the event sets that form up the *expectations*. The successful matching of the cues means that the expected events had occurred and the expectation is therefore confirmed. For example, suppose that *Ted* is currently approaching the sniper in the *Counter Sniper* operation. The current *expectation* of *Ted* contains *Event Set A* and *Event Set B* as we discussed before. The *Event Set B* of *Ted* can be expressed as:

$$[\neg Weapon_Fire(All) \wedge \neg Vehicle(Militant) \wedge \neg Bots(Militant)]$$

This event set means that *Ted* does not expect to be attacked by any type of weapon fire, enemy vehicles and militants during his movement. The *Observe* component monitors the virtual environment for key cues and forms observation sets. Suppose that the observation set of *Ted* about the current situation is:

$$[Bot(A, Militant, Rifle) \wedge Weapon_Fire(Rifle) \wedge BotGroup(Civilian, 50), \dots]$$

It means that *Ted* had been engaged by ‘bot A who is a male militant with a rifle, rifle fire, and a group of civilians are around, and ...’. This set of observation cues is used to compare with *Ted*’s *Event Set B*. The comparison reveals that the observations $Bot(A, Militant, Male, Rifle)$ and $Weapon_Fire(Rifle)$ violate *Event Set B* cues of $\neg Bots(Militant)$ and $\neg Weapon_Fire(All)$ respectively. This tells *Ted* that its expectation is not fulfilled and the current solution is likely to be invalid. This is consistent with the fact that the movement route of *Ted* had been compromised and the enemy may attempt to ambush *Ted* before he reaches the sniper’s location.

Dynamic Expansion of Evidence Sets

When a solution is deemed invalid, the observation cues that violate the event set cues will be combined with the current evidence set to form an expanded evidence set. This allows the bots to adapt to new situation rapidly. The *Situation Awareness* component will then retrieve new experience cases with the updated evidence set. The newly retrieved experience case will be sent to the *Action* component for processing. Consider the case when *Ted* had identified a *sniper-assault* threat and is currently performing the *Counter Sniper* solution. The current evidence set is

$$[Bot(A, Militant, Male, Snipergun) \wedge Weapon_Fire(Snipergun)]$$

New observation sets will continue to be monitored by the Observe component. As discussed before, suppose that the new observation set is:

$$[Bot(B, Militant, Rifle, Near) \wedge Weapon_Fire(Rifle) \wedge BotGroup(Civilian, 50), \dots]$$

The two new observations $Bot(B, Militant, Male, Rifle)$ and $Weapon_Fire(Rifle)$ violate *Event Set B* of *Ted*'s expectation frame. These new observations are added to *Ted*'s existing evidence set. The dynamically expanded evidence set is thus changed to:

$$[Bot(A, Militant, Sniper, Far) \wedge Weapon_Fire(Sniper) \wedge Bot(B, Militant, Rifle, Near) \wedge Weapon_Fire(Rifle)]$$

The expectation violation will make *Ted* terminate the execution of the current solution and to re-assess the evolving situation. Using the new evidence set, *Ted* will identify a *Close Combat Fire* threat. This new threat is of immediate concern and takes precedence over the previous solution. Thus, the *Close Combat Fire* experience case is retrieved and *Ted* will perform the corresponding solution to this situation.

Chapter 4

A Computational Model of Situation Awareness

Although much work had been done on the *CBR process* in *SNAP*, the *Situation Awareness* component of *SNAP* is still rather primitive. A realistic situation awareness component has yet to be developed to provide useful information for the decision making process, which motivates us to work on a computational model of situation awareness. The computational model should work together with the major features that we have built into *SNAP* such as *CBR process*, *qualitative spatial representation* and *expectations*.

Our computational model is based on Endsley's influential framework on situation awareness [97] which consists of three stages: *perception*, *comprehension* and *projection*.

Now let us use an example of a soldier bot on a patrol mission to illustrate how situation awareness is formed up in our model in dynamic MOUT situations. First, the bot perceives the current situation through the cues that fall within its hearing and vision range. The bot comprehends the current situation by comparing the perceived cues with its experience cases. The experience case with the highest

similarity value to the current situation will be selected. The corresponding action of this case will be chosen to deal with the current situation.

Being on a patrol mission, the bot has certain expectations (i.e., projection) on how the situation may evolve. The perceived cues are also used to compare with the bot's expectations. For example, the bot may expect that the street should be populated with at least fifty civilians since the street is normally crowded. If the street is too quiet, this could mean that local people already anticipated an impending ambush. Figure 4.1 shows the initial hearing and sight radius of the bot which are represented as H and R , respectively.

Figure 4.1: Bots increasing their awareness on anticipation of impending threat

When the expectation on the size of the crowd is not met, the bot suspects an impending ambush. Thus, the hearing and sight radius are expanded (by h and r , respectively) which will allow the bot to pick up more cues in anticipation of a possible attack. This is consistent with the fact that a soldier tends to be more alert when he senses danger.

Our current work seeks to improve the *Situation Awareness* component in *SNAP*. Figure 4.2 shows how our model generates situation awareness in *SNAP*. Following Endsley's abstract framework of situation awareness, we also consider 3 levels of

Figure 4.2: Maintaining Situation Awareness in *SNAP*

situation awareness in our model. During the *Perception* stage, the *Sensory System* will pick up the external information from the virtual environment and produce observation cues. These observation cues will be used during the *Comprehension* stage for evaluation with the *expectations* of the bots. The mission can continue if cues do not violate the *expectations* of the bots. Violations will be used to identify the probable threat. When a new threat is identified, a new mission will be activated to neutralize the threat.

4.1 Perception Stage During Situation Awareness

Figure 4.3 illustrates the sensory system that we have implemented for our bots to perceive the environment in MOUT.

The solid sector in Figure 4.3 represents the field of view of a bot. This sector is characterized by the *field of view angle* and the *SightRadius* to simulate the *Sight* function of the bot. The dotted circle surrounding the bot shows the coverage of the *Hearing* function of the bot. The *Hearing Threshold* is used to control the hearing radius of the bot.

Although there may be many cues occurring within the MOUT environment, only cues that fall within the coverage of a bot's *Sight* or *Hearing* function are picked

Figure 4.3: Sensory System of A Bot

up. As shown in Figure 4.3, only Cue A and Cue C are picked up by the *Hearing* and *Sight* functions of the bot respectively. Cue B will be missed out by the bot. Such cues may have different types such as weapon, weapon fire or enemies. Cues are gathered and organized into *observation* sets for further evaluation during the *Comprehension* stage. An example observation set is shown as follows:

$$[\text{Bot}(\text{Militant}, \text{Rifle}) \wedge \text{Gun_Fire}(\text{Rifle}) \wedge \text{Grp}(\text{Civilian}, 50), \dots]$$

which essentially means that the bot has observed 1) “a militant bot with a rifle”; 2) “rifle fire”; 3) “50 civilians”; and “...”.

The context a bot experiences depends completely on the senses it is provided with to perceive relevant cues. Therefore, the bot’s reality is the internal representation of those senses. The *SNAP* framework currently provides for visual, auditory, and tactile senses. Some of the cues obtained through these senses and their attributes are listed in Figure 4.4 below.

Figure 4.4: Perception cues and their attributes

4.2 Comprehension Stage During Situation Awareness

In this stage, the *Situation Awareness* component receives the *observation sets* from the *Observe* component. With the cues from these *observation sets*, a bot determines if its current mission is under threat. If so, it will generate an *evidence set* and match with a list of probable threats. Once the threat is identified, the corresponding *experience case* will be retrieved from the *Experience Repository* and the corresponding solution will be used.

Each bot also has a set of expectation cues. It will monitor the expectation cues with the *Sensory* system as discussed in the *Perception Stage*. The expectation cues are further classified as *invariant* or *variant*. Violation of the *invariant expectations* means that the mission is under threat, therefore it should be terminated and a new mission may be started. In contrast, violation of *variant expectations* may raise certain concern of the bot (for instance, the bot may become more alert) but are insufficient to show that the current mission is under threat.

4.2.1 Cue Types

In *SNAP*, cues observed from the environment can be qualitative, nominal or quantitative. In our implementation, quantitative cues (for instance, number of civilians) are represented using integer values. A nominal cue is a discrete cue whose values are not necessarily in any order (e.g, weapon type). The qualitative cues are used for qualitative spatial reasoning.

4.2.2 Similarity measures

Similarity measures are used to assess the degree of resemblance between the current situation and past experiences. We assume that the cues in current situation and the cues in the past experiences have the same set of features.

Suppose that the *Evidence Set*, E_c , of the current situation and the past *Experience Case*, E'_c , are represented as:

$$\text{Evidence Set, } E_c = (C_1, C_2, \dots, C_i, \dots, C_n)$$

$$\text{Experience Case, } E'_c = (C'_1, C'_2, \dots, C'_i, \dots, C'_n)$$

where C_i and C'_i can be nominal, quantitative or qualitative values. To assess the similarity between E_c and E'_c , a global similarity value is computed by aggregating the local similarity between cue pairs C_i and C'_i . The following heterogeneous similarity computation method is proposed to calculate the local similarity between C_i and C'_i :

$$S_L(C_i, C'_i) = \begin{cases} 0 & \text{if } C_i \text{ or } C'_i \text{ is unknown} \\ N_Match(C_i, C'_i) & \text{if } C_i \text{ is nominal} \\ Quant_Similar(C_i, C'_i) & \text{if } C_i \text{ is quantitative} \\ Quali_Spatial(C_i, C'_i) & \text{if } C_i \text{ is qualitative} \end{cases}$$

If either C_i or C'_i is unknown, the cue similarity between them is set to 0. For the nominal values, the similarity is either 0 or 1 as determined in the following way:

$$N_Match(C_i, C'_i) = \begin{cases} 1 & \text{if } C_i = C'_i \\ 0 & \text{otherwise} \end{cases}$$

Assuming the bot picks up the cue, $C_1 = \text{Weapon_Fire(Rifle)}$, and compares it with $C'_1 = \text{Weapon_Fire(Any)}$. $N_Match(C_1, C'_1)$ will give a result of 1. This is because C'_1 includes all types of weapon fire.

The similarity value between a pair of quantitative cues are determined as follows:

$$Quant_Similar(C_i, C'_i) = 1 - \frac{|C_i - C'_i|}{\Delta_i}$$

where $\Delta_i = \text{Max}_i - \text{Min}_i$ is used to normalize the cue difference, and Max_i and Min_i are the maximum and minimum values for cue i , respectively. When the *experience case* contains $C'_2 = \text{Group(Civilian, 50)}$ and the bot observes $C_2 = \text{Group(Civilian, 12)}$, $Quant_Similar(C_2, C'_2) = 1 - \frac{|12-50|}{50-0} = 0.24$. In this case, the low similarity value between C_2 and C'_2 implies that the current cue does not match closely to the *experience case*.

It has been suggested that humans rely more on the qualitative description than the quantitative description of their environment [76]. In our previous work, we

had organized the virtual environment into hierarchical zones and added qualitative descriptions [62] to facilitate efficient path planning in dynamic environments. Using the qualitative spatial information, $\text{Quali_Spatial}(C_i, C'_i)$ computes the similarity values for qualitative spatial cues. A typical qualitative cue is the combat zone where the conflict is taking place.

The model activates the recursive algorithm below to check if the soldier bot and the enemy have the same parent zone:

```

Query_Zone(zone a, zone b){

    //Let parenta & parentb be parents of a & b

    if(parenta = parentb)

        return parenta;

    return Query_Zone(parenta, parentb);

}

```

The Query_Zone algorithm terminates when a common parent zone is found. This parent zone is the lowest level in the spatial hierarchy that contains both the soldier and its enemy. Thus, the parent zone is considered as the combat zone.

$\text{Quali_Spatial}(C_i, C'_i)$ will compare the qualitative spatial relationship in the following way:

$$\text{Quali_Spatial}(C_i, C'_i) = \begin{cases} 1 & \text{if } C_i = C'_i \\ 0 & \text{otherwise} \end{cases}$$

For example, the *Close Combat Fire experience case* has $C'_3 = \text{Combat_Zone}(\text{Room})$ stating that the threat can be considered *Close Combat Fire* only if the combat take places within a room. However, the observed cue, $C_3 = \text{Combat_Zone}(\text{StreetA})$, indicates that the soldier is attacked on *Street A*. Thus, C'_3 and C_3 does not match and $\text{Quali_Spatial}(C_3, C'_3) = 0$. This result implies that the soldier is not engaging his enemy within the close proximity of a room.

After obtaining the similarity values of all cue value pairs, the next step is to integrate them into a global similarity value, $S_G(E_c, E'_c)$, using the following formula:

$$S_G(E_c, E'_c) = \begin{cases} 0 & \text{if } w_i = 1, S_L \leq \beta \\ S_p + \frac{\sum_{i=1}^n w_i S_L(C_i, C'_i)}{\sum_{i=1}^n w_i} & \text{otherwise} \end{cases}$$

where $\beta \in [0, 1]$ is a threshold parameter determined by the user. This threshold parameter is determined by analyzing data gathered from human experts and the threshold can be calibrated according to the requirements of the human expert running the simulation. It is used to determine whether C_i and C'_i is close enough. $w_i \in [0, 1]$ is the weight of the cue i , which represents the relative significance of the cue i and is also assigned by the user. Often, there are some important cues or cues that are required to be satisfied before the past experience can be reused. That is, if the value of a required cue in the current situation is not close to the value of the same cue in a past experience, this *experience case* cannot be used to solve the current problem no matter how similar the other cues are. For example, spotting sniper fire is a required cue to assess for the *Sniper Assault* threat. For required cues, we will set their weight to be 1. If a required cue in the *evidence*

set is not close enough to the corresponding value in the *experience case*, we shall assign 0 to $S_G(E_c, E'_c)$.

In our implementation, each mission will have a pre-determined list of possible threats. Thus, the global similarity values are computed for the threats in the pre-determined list and the threat with the greatest similarity will be identified.

Certain threats are more common in some missions. S_p modifies the similarity value according to the current mission. For example, patrolling soldiers are common targets of enemy snipers. When violation of *expectations* occur during a patrol, it is likely that a sniper assault had taken place. When performing matching with the *Sniper Assault experience case*, the S_p adds 0.10 to $S_G(E_c, E'_c)$. The addition enhances the likelihood of a sniper assault in the patrol mission. With S_p , the decision bias of the bots towards certain threats during specific missions can be modeled.

4.3 Projection Stage During Situation Awareness

Expectations help the bots to anticipate certain future events and pick up key cues rapidly. They are used during the *projection* stage to monitor the status of the mission and detect key cues.

We have introduced two types of *expectations*: *invariant expectations* and *variant expectations*. *Invariant expectations* must be fulfilled at all times or the current mission will be considered to be under threat. Unlike the *invariant expectations*, the *variant expectations* need not be fulfilled at all times. The violation of the *variant expectations* merely alerts the bots that certain threats may be forming.

Now let us use a simple example to further illustrate how *expectations* work. Suppose that *Ted*, a soldier bot, is patrolling a street. His current *invariant expectations*

are:

$$[\neg \text{Gun_Fire}(\text{Any}) \wedge \neg \text{Vehicle}(\text{Militant}) \wedge \neg \text{Bot}(\text{Militant})]$$

This *expectation* means that *Ted* does not expect to see any type of weapon fire, enemy vehicles and militants during his movement. Suppose that the *observation set* about the current situation is:

$$[\text{Bot}(\text{A}, \text{Militant}, \text{Rifle}) \wedge \text{Gun_Fire}(\text{Rifle}), \dots]$$

It means that *Ted* has been engaged by “bot A who is a militant with a rifle, and rifle gun is observed, and ...”. This set of observation cues is used to compare with *Ted*’s *invariant expectations*. The comparison reveals that the observations $\text{Bot}(\text{A}, \text{Militant}, \text{Rifle})$ and $\text{Gun_Fire}(\text{Rifle})$ violate *expectations* of $\neg \text{Bot}(\text{Militant})$ and $\neg \text{Gun_Fire}(\text{Any})$ respectively. This tells *Ted* that its *invariant expectations* is not fulfilled and the mission is under threat. The cues violating the bot’s *expectations* are used to form the *evidence set*. This *evidence set* is then used to generate similarity values with the *experience cases*. In this case, the *evidence set* should state that *Ted* had spotted a militant bot and also observed sniper fire. The current *evidence set* is:

$$[\text{Bot}(\text{A}, \text{Militant}, \text{Sniper gun}) \wedge \text{Gun_Fire}(\text{Sniper gun})]$$

A *variant expectation* for the patrol mission could be $[\text{Grp}(\text{Civilian} \geq 10), \dots]$. This *expectation* states that the soldier bot expects to see at least ten civilian on the streets. It is not normal for the streets to be deserted. In real life situation, this could mean that the locals sense an impending attack and are staying away from the danger zone. The violation of this *expectation* alone is not sufficient to determine any threat. Yet, this cue is significant enough to be included in our *evidence set*.

When *variant expectations* are violated, the perception senses could be sharpened by increasing the *SightRadius* and *Hearing Threshold* of the bots. This reflects that the bots are expecting danger and increasing their alertness.

Chapter 5

***QUARTZ* - Autonomous Navigation System**

Autonomous navigation systems are important to MOUT simulations for generating realistic tactical behaviours for the non-player characters (or bots). In this paper, we describe our work on Quartz, an autonomous navigation system for MOUT simulations. Novel features of Quartz include qualitative spatial representation and hierarchical spatial reasoning which enables fast situation analysis and human-like path planning in a dynamic environment. In MOUT simulations, the virtual urban environments are populated with characters and buildings. While human players control some characters, most of the characters are non-player characters (also known as bots) which demonstrate some level of human-like tactical behaviours, e.g., tactical path planning and movement. Therefore, autonomous navigation systems are crucial to MOUT simulations.

5.1 Requirements

The basic requirements of an autonomous navigation system in MOUT simulations are *effectiveness*, *efficiency*, *scalability* and *adaptability*. *Effectiveness* requires the

navigation system to be able to generate realistic tactical movements for the bots in a complex environment. *Efficiency* requires the paths to be computed in real-time. *Scalability* requires the system to be able to accommodate a large number of bots. *Adaptability* requires the navigation system to be able to deal with the dynamic environments.

5.2 Design of *Quartz*

Quartz is designed to be an autonomous navigation system that is able to quickly generate human-like tactical movements for a large number of bots in a complex and dynamic virtual urban warfare environment. The two major features of *Quartz* are the qualitative spatial representation and hierarchical spatial reasoning framework.

5.2.1 Why Qualitative Spatial Representation and Hierarchical Spatial Reasoning?

It has been suggested by some researchers that humans rely more on the qualitative description than the quantitative description of their environment when doing path planning in daily life [64, 66]. Typical qualitative information includes location names, relative sizes, regions, relative relations among objects and relative distances, etc. [67]. Although there are a number of path-finding algorithms for robots and agents, they are generally based on the accurate quantitative description of the environment. We believe it is necessary to model the human cognitive process in spatial reasoning in order to generate human-like behaviours for the bots in a complex virtual environment. In addition to using qualitative spatial descriptions, it seems to us that humans plan their paths in an incremental and hierarchical manner.

As an example, consider how a person describes the way to office from home. It is unlikely that the person knows exactly how many meters to move along which exact directions. In fact, the person is more likely to describe the way in terms of a number of key points and their relations, e.g., “first, I will go to the tall building at street S1, then from there, I will go to the subway station at street S2,..., finally I reaches my office at location X!”. The key spatial points and their spatial relationships can be easily remembered and analysed by humans. They essentially inform a person the rough path between two locations. However, when actually moves from home to the office, the person still needs to plan his/her detailed path along the way, e.g., how to move from home to the tall building at street S1, etc. Note that a detailed path is often planned only when the person is on that specific section of the rough path. This incremental and hierarchical path-planning approach helps to reduce the amount of information that a person needs to remember and help him/her deal with the dynamic environment.

5.2.2 Qualitative Spatial Representation in *Quartz*

Qualitative spatial representation describes the virtual environment by a number of regions, connections between different regions, and key points, etc. As humans rely more on the qualitative relations among various objects rather than the exact quantitative descriptions of the world in their daily-life spatial reasoning, qualitative spatial representation is believe to be more suitable to efficiently generate human-like movements for the bots [64]. For MOUT simulations, it is crucial to enable tactical awareness and fast terrain reasoning when planning paths for the bots. This is a challenging task which often involves many factors, e.g., the 3D environment, limited fields of views, team formations, cover and concealment [65, 68, 69]. As

Figure 5.1: Squad Formation

an example, Figure 5.1 shows the formation of a squad that is firing at the enemy from different directions.

Quartz adopts a hierarchical spatial reasoning framework. At the higher level, only some key points connecting different regions are determined using the qualitative description of the map. At the lower level, the detailed path within a region is determined as the bot moves into the region with the heuristic A* search algorithm [51]. With this hierarchical spatial reasoning framework, *Quartz* is able to cope with dynamic environments efficiently. For example, if a path node is blocked along the detailed path in a region, only this detailed path needs to be re-computed, the general direction regarding how to move region by region from the starting point to the destination remains unchanged. Thus the computational cost of path planning can be greatly reduced, which help to improve the scalability of the navigation system. With its current implementation, *Quartz* can support up to a few hundred bots on a high performance personal computer.

In *Quartz*, we model the human cognitive process in spatial reasoning with a hierarchical multi-resolution spatial representation. As shown in figure 5.2, different resolution levels are used to store the key points between A and B. A point at a higher level represents a group of points at the level below it, i.e., the resolution of the spatial description decreases as the level increases.

Figure 5.2: Hierarchical spatial representation

As path-finding within a dynamic environment is a computation-intensive task especially when the two locations are far apart, a hierarchical approach helps to divide the complicated problem into some manageable sub-problems, thus allows *Quartz* to handle long paths efficiently and elegantly. In *Quartz*, a virtual environment is described with a detailed map and a hierarchy of qualitative maps. Different representation and spatial reasoning techniques are used for these two kinds of maps. The detailed map uses visibility graph to link different points with edges, and it is used for low-level path planning. The qualitative maps use Voronoi graphs to segment the environment into regions, and they are used for high-level spatial reasoning.

5.2.2.1 Qualitative Spatial Representation in *Quartz*

As shown in figure 5.3, the detailed map consists of a large number of points that represent various positions in the virtual environment. An edge will be formed

linking two points if there exists a line-of-sight between the two points and they are within a reasonable distance. This map is used by some low level path-finding algorithms such as A* to equip the bots with the low level navigation capability, such as moving from point to point.

5.3.a: Points

5.3.b: Points with edges

Figure 5.3: Visibility Graphs

5.2.2.2 Map Representation for High Level Path Planning

At the high level, the map is organized as a hierarchy of spatial regions using Voronoi graphs, as shown in figure 5.4 and figure 5.5. A region in the map is represented by a point known as the region node. In *Quartz*, the hierarchy of regions is built with a bottom-up manner. The Voronoi graph-based regions form the base layer (level 1) of the hierarchy. The next layer (level 2) in the hierarchy is built on the base layer by grouping the base regions into region clusters. Each region cluster is then treated as a new region and is represented by its region node. Figure 5.5 demonstrates the aggregation of the lower level regions to form higher level regions. In figure 5.5, region 3A and 3B are aggregated to form their parent region 4A. To represent this parent-child relation between regions, a hierarchical data structure is adopted in *Quartz*. As shown in figure 5.6, the dashed line connecting two region nodes (at different levels) means the parent-child relation between the two regions. A solid line connecting two region nodes means that the two regions are adjacent to each other and they belong to the same parent region. A region node contains the information about how to move across its child regions. This hierarchical data structure provides important information for qualitative spatial reasoning.

5.4.a: Region Nodes

5.4.b: Regions

Figure 5.4: Voronoi Graphs and Regions

Level 1 Regions

Level 2 Regions

Level 3 Regions

Level 4 Regions

Figure 5.5: Hierarchy of qualitative spatial layers

Figure 5.6: Hierarchical data structure in Quartz

Note that while the higher level regions are suitable for analysing long paths rapidly, the lower level regions can find short paths more accurately.

5.3 Qualitative Spatial Reasoning in *Quartz*

Quartz has a hierarchical qualitative spatial reasoning framework based on the hierarchical spatial representation as described in Section 5.3.2.2. During path planning, *Quartz* first considers those paths between key spatial points if these points are connected by either the dashed lines or solid lines as shown in figure 5.3. As a parent region node stores the information about how to move across its child regions, path planning is straightforward if the current region and the destination region belong to the same parent region. As an example, consider the path planning between a current location in region 3C and a destination in region 3D (see figure 5.6). The path information on the possible routes between region 3C and 3D can be found from their parent region node 4D. To move across the child regions (at level 2 in this case) of region 3D, the path information on the possible routes leading towards the level 2 region containing the destination can be found from region node 3D. This process is repeated along the hierarchical levels (in descending order) until

the destination point is found. However, if the current and destination regions are not in the same parent regions, the path planning needs to move to the next higher level.

```
path QuartzPath(region a, region b){  
  
    //Let parent_a & parent_b be parent region of a and b;  
  
    if (parent_a=parent_b)  
        return a + parent_a + b;  
    return a + QuartzPath(parent_a , parent_b) + b;  
}
```

Figure 5.7: *QuartzPath* Algorithm

Figure 5.7 illustrates the *QuartzPath* path-planning algorithm in *Quartz*. The recursive algorithm first check if the two child regions have the same parent region and add the two child regions into the path plan if no common parent region is found. The *QuartzPath* algorithm terminates when a common parent region is found and returns the path plan.

With the hierarchical qualitative spatial reasoning framework, bots can incrementally plan their paths. This helps to reduce the computational cost of path planning. In addition, as bots do not commit to long paths, the impact of dynamic obstacles is greatly reduced.

5.4 Limitations of *Quartz*

Some of the limitations in *Quartz* include the variability of the bot navigations. To allow hierarchical path planning, the bots need to have spatial information on the target location. The variability of the bots path planning is only affected by the mission type and the force composition. If all the bots received the same data

of spatial information, they will put the spatial data through the same algorithms. These algorithms will generally return the same paths. If all the bots generate the same paths, the realism of the simulation will be affected. However, this issue can be mitigated by varying the spatial information given to various bots.

Another challenging issue for *Quartz* is the dynamic changes in the MOUT environments. With a potentially rapidly changing environment, it is likely that alot of re-planning is necessary for the bots. The re-planning involves more than just analyzing the new terrain. It also involves understanding the mission type, enemy strength or weapon threat. Therefore, for *Quartz* to work, an appropriate decision making model is required to handle the situation awareness and threat identification of the bots.

Chapter 6

The simulation testbed - *Twilight City*

This chapter describes the implementation of the *Twilight City* [70] as the simulation testbed. Minor details like setting the virtual environment up will not be covered in detail as it will be a tedious procedure. The main emphasis of this chapter will be implementing the part of this system as a simulation platform for MOUT.

6.1 Motivation

The goal of our *Twilight City* project is to create high-fidelity simulation testbed for MOUT. A typical application scenario is the special squad operation for saving hostages held in a building by a group of terrorists. Various operation tactics need to be investigated before the real squad operation. The constructed virtual environment needs to resemble the real operation environment not only in terms of human's visual and audio perceptions but also in terms of the behavior of the non-player characters (i.e., AI bots). Creating such a high-fidelity virtual environment is a challenging task. Although there is some work on the application of FPS engines

to military training, the design requirements may be different for different training tasks and scenarios. As pointed out by Lewis and Barlow in [71], “The games must have sufficient fidelity to provide valid results. This fidelity does not have to be complete in all aspects, just enough in the variables of interest.” In the following sections, we identify the major considerations of *Twilight City* before explaining the implementation of *SNAP* and *Quartz* into *Twilight City*

6.2 MOUT Scenarios

Military operations in Panama, Somalia, Kuwait and Iraq highlighted the critical requirements for modern military forces to be able to operate effectively in an urban environment. Combat operations within an complex urban environment will pose great difficulties for the soldiers. Our MOUT training systems should be able to support MOUT operations. These operations are generally stability operations and support operations. Some of the MOUT operations include Combating Terrorism, Peacekeeping Operations, Noncombatant Evacuations, Civil Disturbance Operations, Nation Assistance and Humanitarian Assistance [72]. To battle in urban areas, ground commanders and key leaders at every level must be able to comprehend the nature of various environments. This understanding of enemy forces, urban terrain and friendly forces can be achieved with MOUT simulations [73].

To build a good MOUT simulation system, it is not enough to describe the general characteristics of an urban area. The number of floors and rooms in a building are important factors to consider when trying to determine the force allocation during a MOUT mission. A commander can only be able to provide sufficient resources to capture a key objective when he can analyze the MOUT environment at a very high level of detail. Troops need to react well towards surprise attacks, poor visibility,

movable objects, smoke and noise. Apart from dealing with environmental effects, they must manage their voice communications, night vision, weapons and other equipments.

6.3 City Planning

The construction of the city environment took a lot of planning before the network of interconnected buildings and streets were finally built. Various technical and design considerations were necessary to produce the simulation system for the MOUT environment.

More than any other battle environment, the urban battlefield is highly dynamic. There are the many types of factors that make military operations in urban environments very dynamic. The soldier must understand the five military aspects of terrain (OAKOC) to be able to comprehend fully the unique aspects of urban terrain [74]. Terrain analysis in an urban environment differs from that of an open terrain in many ways. There are five key aspects of terrain analysis for military forces. They are as listed below:

- (a) Obstacles
- (b) Avenues of approach
- (c) Key terrain
- (d) Observation and fields of fire
- (e) Concealment and cover

Depending on the layout pattern of the streets, the combatants can improvise or create man made obstacles quickly to obstruct narrow streets. These obstacles may

not cause a significant concern in situations with wider streets. Natural obstacles arguably pose less of a problem in urban terrain than in open terrain. Rubble resulting from direct or indirect fire may hinder all types of troop movement during urban combat. In extremely rare situations, certain rubble could actually help to facilitate troops movement. For example, if a building collapses across a waterway, the rubble could provide the structure to enable access across the waterway.

Analysis of cover and concealment is also vital to success on the urban battlefield. Infrastructure characteristics, brick, wood and glass can all provide different levels of cover from enemy observation and weapon fire.

Some of the important structures often involved in MOUT operations are listed below with some supporting reasons.

- Sewer and subway systems can provide infiltration routes.
- Elevated railways or mass transit routes that can provide mobility; if the combat operations disable these facilities, huge congestion will happen.
- Utilities such as gas, electrical or water facilities are all considered strategic targets.
- Forces cannot attack hospitals and clinics when not under use for military purposes. The medical personnel in the hospital or clinic could become a source of medical support for all groups and elements.
- Stadiums, parks, and sports fields may serve as enemy prisoner of war facilities, holding areas or landing and pickup zones.

The city layout was planned with numerous interconnected buildings. A network of roads shall be placed in the city map to allow vehicular movement during the MOUT operations. Lifts will be placed in various buildings to allow the soldiers

an alternative movement route. Besides underground levels, a sewage system is placed into the environment as well. Following the principles of MOUT, some of the buildings are designed to act as a fire base or reconnaissance point.

Figure 6.1: Twilight City Top View

The environment seeks to provide as much flexibility as possible to the potential users in terms of the types of MOUT operations that can be simulated within the environment. Some of the MOUT operations are hostage rescue, terrorists search, ambush and patrol. Taking the nature of the identified MOUT operations into account, high-rise buildings are designed to allow sniper attacks and reconnaissance posts. Buildings often complicate military operations in urban areas. Their composition, frontages, size, and window locations affecting troops positioning and weapons deployment considerations. Sewers should be built to facilitate the sewerage sabotage by the terrorists.

6.4 Issues encountered during implementation

The work of building the virtual environment leads to many issues. In this section, we will briefly describe a few issues faced during implementation. Greater details can be found in [117].

To enhance the realism of *Twilight City*, special effects are needed. These special effects can help to provide an immersive environment for the MOUT simulation. More importantly, the special effects such as smoke, shattering glass, water bodies and rain can help to the testing of different MOUT situations. The various possible scenarios of MOUT operations are analyzed and the custom special effects, which will improve the realism and accuracy of the simulation, are identified.

It is observed that voice control capabilities are lacking in the presently available MOUT environments. This function will be significant in the testing of the bots as user can verbally control the bots and get them to move just by moving their mouths. When users are doing testing of bots across different maps or maps on different computers being integrated together through TCP/IP interface, the users can use this speech function to activate events without having to control two or more keyboards. Modifications to the Unreal Engine are necessary to accommodate to the voice control capability

For *Twilight City* to act as an efficient simulation tool, some features must be added to the systems. For example, the game engine can be modified to test the efficiency of an evacuation plan during a gas attack during a concert in Singapore Indoor Stadium. These features can be added in the form of modifications to the game engine. Some of the useful modifications to game rules and bots AI are identified below.

Poison Gas Volume is made for the purpose of simulating the effects of a gas attack. The Bots Visualiser displays the locations of all the soldiers activated within *Twilight City*. Night Vision is a modification to the HUD to allow the testing of night mission with Night Vision Goggles. Parachute is a mutator that allows the simulation of airborne operations with parachutes in MOUT. Improved Bots Movements

is a modification to the game engine to change the FSM structure of the bots logic in the game engine. This modification will result in a marked improvement in bots movement. The bots will become smarter and will demonstrate better movement such as increased dodges and jumps.

One of the major considerations in building a virtual MOUT training environment is the fidelity of the virtual environment. As movable objects such as bottles, boxes and chairs, etc. are quite common in real-life, it is important to incorporate these objects in the virtual environments. However, in the existing MOUT simulations, besides the avatars of the soldiers and some vehicles, most objects are static. The reason for this is that modeling and implementing movable objects is a challenging task in terms of computational costs and mathematical complexities. Traditional methods to implement movable objects are based on exact physics models and numeric analysis, thus are very time-consuming. The responsiveness of the system may be greatly affected by the behavior of the movable objects. To reduce the computational cost of simulating moveable objects in *Twilight City*, a framework for qualitative behaviour simulation has been proposed and integrated into our simulation engine.

Building high resolution virtual environments at a low cost is insufficient to fulfill the requirements of a modern simulation system. New generation simulation systems should be agile and rapidly reconfigurable due to a growing emphasis to reduce the development time. Such a system is possible if development efforts can be minimized by composing existing models from a model repository into new virtual environments. To achieve reusability and composability, the simulation systems must have interoperable models. Presently, model interoperability is a huge challenge due to the application specific nature of existing models. These models

are built with a stovepipe development approach. As a result, replicates of existing models are continually being rebuilt for different applications. This needless reinvention causes costly and time consuming development iterations. To address these issues, an elegant architecture is needed to quickly develop virtual environments and support reuse from a common repository of interoperable and extensible models.

6.5 Novel techniques

In order to adapt the *Unreal* game engine into a simulation test bed, a number of innovative techniques are explored. The following sub-sections discuss our contributions to adapt *Twilight City* for MOUT simulations.

6.5.1 Qualitative Physics Simulation

Movable objects can increase realism during simulation. However, most simulated objects are non-movable due to the need to limit the computational overheads. To overcome this limitation, we applied qualitative physics to provide acceptable physics behaviors at a low computation cost [75].

Traditional physics-based simulation is based on the kinematics equations of an object. The behaviour of the object is then generated by numerically solving these equations, which may need huge computational resources for real-time simulations. Thus, it may not be feasible to perform such precise simulations at a large scale in real-time applications like MOUT simulations. In most cases, achieving a rapid estimate of the behaviour can provide a lightweight yet good-enough alternative as compared to the precise solutions of the kinematics equations.

Qualitative physics aims to perform behaviour simulation at certain acceptable levels of realism and accuracy so as to fulfill some operational requirements while keeping the computational load low. In essence, qualitative physics applies our knowledge of the qualitative (causal) relations of various factors in the physical world to the behaviour modelling of the objects in the virtual environment [76]. Human beings are more sensitive to the qualitative (causal) relations among various objects and factors rather than on the exact physics of the objects. They rely on these relations in their reasoning about the environment and make decisions. Therefore, qualitative physics focuses on the changes in the environment, e.g., changes in water level, position changes, orientation changes, etc. In qualitative physics, changes in the environment are described in terms of process. Examples of processes are movement of water from a high ground to a low ground, heat transfer from an object with higher temperature to another object with lower temperature, etc. Such processes are activated due to the relationships that exist among the objects in the environment. A process is determined to be active or inactive by assertions of some preconditions and quantity conditions. The process specifications allow the system to automatically infer a set of active processes. An active process will then perform the necessary changes through its influences.

6.5.2 Architecture for Rapidly Re-configurable Simulations

Unreal engine is mainly a visualization engine, which can be supported by external applications. These applications can be qualitative physics engines to provide augmented reality. The external engines can be used to monitor bots actions or perform voice controls as well. Work should be done on the software architecture of the system so that *Twilight City* can demonstrate good extendibility and cooperation with external modules. Therefore, we came up with effective architecture

designs on which the virtual environments are developed [61, 78]. The main objective that our architecture seeks to fulfill is to support rapid development of future environments according to the changing requirements of modern MOUT warfare. These architecture setups will serve as templates and guides for future researchers.

6.6 Implementation Details

After developing *Twilight City*, the *SNAP* framework and the *Quartz* navigation system was implemented into the simulation testbed. We will cover the implementations below.

6.6.1 *SNAP* in *Twilight City*

Figure 6.2: Integrating *SNAP* into *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 6.2, *SNAP* interacts with the bots in *Twilight City* via *Gamebots* [79]. *Gamebots* is a modification to UT that opens a socket to the *Unreal Engine* during runtime. *SNAP* is Java-based. For the ease of communication and implementation,

SNAP interacts with the *Gamebots* via Java bots which is a middleware for external applications to communicate with UT bots via *Gamebots* [80].

As illustrated in Figure 6.2, 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.

In *Twilight City*, there are mainly three types of bots: soldier bots, militant bots, and civilian bots. Typical scenarios in *Twilight City* include 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.

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

Figure 6.3: Experience Cases

As shown in Figure 6.3, 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.

6.6.2 Integrating *Quartz* into *Twilight City*

6.4.a: Regions

6.4.b: Centre of Twilight City

6.4.c: Visibility Graphs

Figure 6.4: Maps in *Twilight City*

Quartz is used as the navigation system for the bots in *Twilight City*. As discussed in Section 5.3.2, a hierarchy of qualitative maps is used to describe the virtual environment. Voronoi graphs are used to form the base layer of the qualitative spatial regions. There are more than 200 spatial regions at the base layer of the *Twilight City* map. Each parent region holds between two to four children regions, thus five layers are used in the hierarchical map structure. It is noticed from figure 6.4.a that the center of the map has the highest concentration of regions. This corresponds to the large number of the buildings at the center of *Twilight City* as shown in Figure 6.4.b. Figure 6.4.c shows the visibility graph network that consists of nodes and edges. *Twilight City* contains more than 650 qualitative spatial points that act as nodes in the visibility graph network. These maps are used by *Quartz* for spatial reasoning and path planning.

6.6.3 Spatial Reasoning in *Twilight City*

Quartz's hierarchical spatial reasoning framework utilizes the hierarchical map structure and the information contained in the maps to build paths for the bots. For tactical path planning, apart from containing spatial information, qualitative spatial points also contain some tactical information such as whether the point is a sniping point, ambush zone, friendly zone or enemy zone. Path information stored in the region nodes allows the bots to navigate rapidly towards child regions. These pre-computed data greatly speed up the path planning in *Twilight City*. In the current implementation, the number of levels in the map hierarchy is five. Thus, with the *QuartzPath* algorithm, the maximum number of qualitative spatial points in any path is only 9. Without using the hierarchical map structure, with most existing path finding algorithms, the number of spatial points in a path can easily exceed 30. As we mentioned before, long paths containing a large number of spatial

points are computationally expensive and more susceptible to the changes in the environment.

Chapter 7

Experimental Results

This chapter discusses the experimental results of implementing *Quartz* and *SNAP* in *Twilight City*. Apart from performing experiments on *SNAP* and *Quartz*, we also use subjective perception tests and performance testing in our efforts to test and benchmark the effectiveness of *Twilight City*.

7.1 Experiments results on *SNAP*

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 behaviours. 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 7.1. The results in Figure 7.1 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.

	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 7.1: Behavior of Soldier Bots with Different Experiences

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[60]. Figure 7.2.a shows a soldier bot being attacked by a sniper. Figure 7.2.b 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.

7.2.a: Sniper Assault

7.2.b: Smoke Screen

Figure 7.2: Sniper Assault Situation in Twilight City

Another set of experiments was conducted to compare the mortality rate between *SNAP* bots and normal UT bots during a 20 minutes simulation. The average results of 10 simulation runs are shown in figure 7.3. Again, three types of soldier bots were used for these experiments: *SNAP*(7) bots are soldier bots with all 7 experiences in Table 1, and *SNAP*(4) bots are soldier bots with the experiences

(i.e., *Counter Sniper*, *Counter Air Strike*, *Hasty Attack*, and *Retrograde*). UT bots are the default bots in UT.

Figure 7.3: Mortality Rate of Soldier Bots

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 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.

To evaluate the performance of *SNAP*, particularly its effectiveness in dealing with dynamic changes in time-critical tactical situations, 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@2GHz processor and 2GHz RAM. 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 expectations on the soldier's behaviours. In particular, the behaviors of soldier bots during the *Sniper Assault* threat are compared. Three types of soldier bots were used to this end. They are S(A) bots which have expectations, S(B) bots which have no expectations, and UT bots which are equipped with the default tactics in UT. The S(A) bots and S(B) bots are also equipped with similar experiences to deal with various threats, such as *Sniper Assaults*, *Close Combat Fires*, etc. In the testing scenario, the bots were first engaged by enemy sniper in *Sniper Assault* and subsequently hunted by enemy bots in *Close Combat Fire*. The average results from 10 runs are shown in figure 7.4.

	Upon Sniper Assault			Upon Close Combat Fire		
	S (A)	S(B)	UT	S(A)	S(B)	UT
Stand Around	2.1	3.1	15.1	1.3	0.5	0.3
Counter Sniper	10.6	11.1	0	3.1	15.1	0
Close Combat Fire	0	0	0	9.1	1.0	0
Take Cover	3.3	4.2	1.2	3.6	1.1	2.1
Return Fire	4.0	1.6	3.7	2.9	2.3	17.6

S(A) – Bots with Expectations

S(B) – Bots without Expectations

UT – Unreal Tournament bots

Figure 7.4: Effect of Expectations on Bots

The results in figure 7.4 shows how each type of soldier bots reacted in different

tactical situations. For example, the data in the first column show that for the 20 soldier bots of S(A) type, under *Sniper Assault* threat, 2.1 of them just stood around, 10.6 of them initiated a *Counter Sniper* operation, none of them initiated a *Close Combat Fire* operation, 4.0 of them returned fire immediately, and 3.3 of them took cover first. In the *Close Combat Fire* threat, the militant bots chase and attack the soldier bots. For experienced human soldiers, when the threat has changed from *Sniper Assault* to *Close Combat Fire*, the natural reaction is to adapt their operation immediately. In this situation, continuing with the *Counter Sniper* operation may be useless since the movement of the soldiers are exposed. It is also more important to deal with the immediate threat of *Close Combat Fire*. It can be observed from the experimental results that the 9.1 of S(A) bots changed their operation into *Close Combat Fire* upon being contacted while only 1.0 of the S(B) bots changed their operation. This is because the S(A) bots are equipped with expectations that allow them to adapt their operations to deal with the dynamic changes. S(B) bots had already committed themselves to the *Sniper Assault* operation and will continue the operation even when attacked by enemies. They have no experience on how to deal with the threat of *Close Combat Fire*. 17.6 of the UT bots simply returned fire as they are designed to retaliate upon being attacked. Compared with the S(A) bots, the behaviours of S(B) bots are unrealistic in this situation. The S(A) bots, on the other hand, behaved more like real soldiers. They were able to retrieve the *Close Combat fire* experience when the expectation violation invalidates the original solution.

Another experiment was conducted to compare the mortality rate between S(A), S(B) and default UT bots during a 20 minutes simulation. The simulation exposes the bots to *Sniper Assault*, *Close Combat Fire*, *Ambush* and *Bomb* threats in rapid succession. The average results of 10 simulation runs are shown in Figure 7.5.

Figure 7.5: Mortality Rate of Soldier Bots

As it can be observed, the mortality rate of the S(A) and S(B) bots are much lower than the UT bots. The number of UT bots that remain alive started decreasing in a faster rate than the *SNAP* bots (i.e., S(A) bots and S(B) bots) after 2 minutes. In 14 minutes, UT bots are totally destroyed by the threats whereas an average number of 13.1 S(A) bots were alive. After the simulation terminated at 20 minutes, an average number of 8.7 S(A) 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 S(A) bots performed significantly better than the S(B) bots. We also observed two major steep declines for S(B) bots at around 8 minutes and 14 minutes. These timings coincide with the activation of new threats and information. The majority of the S(B) bots were destroyed during the activation of new threats as they could not adapt fast to the new threat. The superior performance of the S(A) bots and the steep decline in number of S(B) bots upon activation of new threats are

consistent with our expectation that the mortality rate of the bots decreases when *SNAP* is equipped with expectations to deal with dynamic situational changes.

It should also be noted that we do not claim our SNAP approach is better than other approaches since in general it is meaningless to compare different decision making approaches regarding which one can generate more realistic decisions. It is hard to compare due to many reasons: different implementations, objectives, etc. Here, the experiments are used to illustrate how experiences could help in making better decisions. Our philosophy is that a intelligent bot should not only be able to generate seemingly realistic behaviors in some given situations, it should also behave like a human in the sense that the decision-making and cognitive processes of a bot should be similar to that of a human being. Therefore, our approach for human-like behaviours focuses on achieving both procedural and end-result realism for intelligent bot behaviours. We believe that by imitating human decision making and cognitive processes under various tactical situations, human-like bot behaviours can be finally be generated. Our model offers advantages such as being able to take the experiences of expert soldiers and include them in the bot's reasoning. Moreover, current models did not take into account of the time criticality of MOUT. Unlike current models, our model is focused on generating time-critical decisions and not optimal solutions.

7.2 Evaluation of Computational Situation Awareness Model

To evaluate the performance of our situation awareness model, we have conducted some experiments. In particular, we wish to verify its realism and efficiency during time-critical situations.

In our work, we focus on achieving both procedural and end-result realism. In fact, the procedural accuracy to real-life human situation awareness is more important than the end result. To verify our procedural realism, the *SAGAT* approach is adopted. A group of 7 infantry section commanders are put through the experiments individually. The test subjects are instructed to recognize threats in the virtual environment. Sudden freezes will be activated during runtime and the subjects will answer a set of questions. Although the subjects were informed that freezes would occur, they were not informed of the timing. The questions are structured such that a proper understanding of the perception, comprehension and projection stages during situation awareness can be judged. Subsequently, the results of the human subjects are compared with the results of the bots simulation.

A total of three test scenarios are conducted. All three scenarios start with a patrol mission. In Scenario A, a single enemy bot will be used to engage the subject at a close range. A group of enemy bots will ambush the subject with heavy fire in Scenario B. Finally, in Scenario C, an enemy sniper will attack the subject from a distance. The questions for the human subjects are listed below:

- (a) What is the current mission?
- (b) What are the possible threats?
- (c) What is the current threat identified?
- (d) List the cues that identify the current threat
- (e) What is the solution?

For each question, a similarity measure, S_t , is computed to judge the closeness of the answers with the actual situation.

$$S_t = \frac{\text{Number of matches}}{\text{Total number of cases}}$$

For example, if there are a total of 5 threats in *Question 2* and the subject identify 3 of them accurately, a score of $\frac{3}{5} = 0.60$ is given for answers to this question. The average scores of all the subjects are then taken for comparison with the scores of the bots as shown in Figure 7.6.

Scenario A - Close Combat Fire					
	Current Mission	Possible Threats	Identified Threat	Cues spotted	Solution
Humans	1	0.89	1	0.77	1
Bots	1	1	1	0.89	1

Scenario B - Ambush					
	Current Mission	Possible Threats	Identified Threat	Cues spotted	Solution
Humans	1	0.89	0.87	0.67	1
Bots	1	1	1	0.91	1

Scenario C - Sniper					
	Current Mission	Possible Threats	Identified Threat	Cues spotted	Solution
Humans	1	0.89	0.55	0.4	1
Bots	1	1	0.8	0.86	1

Figure 7.6: Results of Situation Awareness

As all 3 scenarios starts off with a *Patrol* mission, the results for *Current Mission* and *Possible Threats* are similar in all 3 scenarios.

We can observe that, in Scenario A, the scores for spotted cues differ by only 0.12 between the human subjects and the bots. All the subjects identified the *Close Combat Fire* threat accurately.

In Scenario B, the difference on spotted cues is increased to 0.24. However, both the humans and the bots are able to identify the threat accurately. Upon further investigation, we found that the drop in the human subjects' score on cue spotting is due to the fact that most human subjects pick up the key cues of "heavy fire" and "a group of enemies". They make their judgment of an ambush threat and

stop picking up the rest of the cues. This explains why threat identification remains relatively accurate even though the number of observed cues dropped.

In Scenario C, the human subjects' score for the observed cues and threat identification dropped to 0.4 and 0.55 respectively. The main reason for the drop is that the human subjects are not familiar with the visual and audio effects of the sniper fire in the virtual environment. Upon clarification that they had witnessed sniper fire, all the human subjects could identify the threat accurately.

As the score of the solutions are only considered if the subjects could identify the correct threat, all solution generation scores achieved a maximum of 1.0. We found that the solution generation are very accurate once the threat is identified correctly. This is in fact consistent with Klein's observation that experts can generate rapid solutions through proper situation awareness and relevant experience cases.

In all three experiments, the cues picked up by the bots are more than the humans. This is because the bots are able to pick up all cues in their field of view. The humans can miss out certain cues in their field of view if they are overly excited with certain prominent cues. Moreover, the human subjects are not familiar to the virtual environment yet. This is evident in their failure to recognize the gun shot from a sniper rifle in Scenario C. More exposure to the virtual environment can help the human subjects to improve their recognition of cues..

To further evaluate the effectiveness of our model in supporting real-time simulations, we have also conducted the frame rate analysis using the same scenarios. Although the frame rate can be affected by a number of factors, the decision making process (which includes the situation awareness model) may incur significant computation overhead thus may impair the frame rate greatly.

We have conducted a series of experiments. Two different types of bots are used in these experiments for comparison. As shown in Figure 7.7, the *Normal* bots are

the default bots provided by the *Unreal* engine which have very primitive cognitive capability and are supposed to consume little computation resource. The *SNAP* bots are equipped with our decision making and situation awareness model. The results shown in Figure 7.7 are the average of 10 runs.

Figure 7.7: Frame rate analysis with Situation Awareness Model

It can be seen that the frame rates for both implementations decrease as the number of bots increases. The results show that the performance of our *SNAP* bots are comparable to the *Unreal* bots in terms of simulation frame rate. This suggests that the decision making process (including the situation awareness model) in *SNAP* does not incur significant computation overhead to the simulation.

7.3 Evaluation of *Quartz*

To evaluate the performance of *Quartz*, we have conducted a series of experiments. In this section, we summarize the major results of these experiments. These ex-

periments were conducted on a computer with Intel T2500@2GHz processor and 2GHz RAM. The data shown in the results are the average of 10 runs. The time for the ten soldier bots to reach their destination with and without (i.e., simply using the A* algorithm and the detailed map) the *Quartz* system are recorded. In Figure 7.8, the normal case means the implementation without using *Quartz*.

Figure 7.8: Impact of Quartz on path planning efficiency

As expected, the time taken for the squad to reach the destination increases as the number of civilian bots (thus the total number of bots) increases. However, it can be clearly seen that the recorded time for the implementation with *Quartz* increases much slower than the implementation without *Quartz*. This indicates that *Quartz* helps to improve the efficiency of the autonomous navigation system, thus also improves the scalability of *Twilight City*.

To further evaluate *Quartz*'s capability in supporting real-time simulations, we have also conducted the frame rate analysis using the same scenario. Although the frame rate can be affected by a number of factors, the autonomous navigation system can

significant affect the frame rate. The results of the frame rate analysis are shown in Figure 7.9.

Figure 7.9: Frame rate analysis of Quartz

It can be seen that the frame rates for both implementations decrease as the number of bots increases. When the number of civilian bots increases beyond 64, the frame rate of the implementation without *Quartz* drops much faster than the implementation with *Quartz*. When the number of civilian bots hits 228, the frame rate of the implementation with *Quartz* is maintained at round 80, while the frame of rate of the implementation without *Quartz* drops to around 30. The results show that *Quartz* helps to reduce the computational overheads of the autonomous navigation system, and the benefit of *Quartz* becomes more evident as the number of bots increases. This makes *Quartz* particularly useful for large-scale MOUT simulations such as crisis management involving a large crowd in urban areas.

Figure 7.10: Tactical path planning

Figure 7.10 shows a soldier bot searching for some qualitative spatial points as defined by the maps of *Twilight City*. With the information stored in various spatial points, the bot is able to move to suitable positions to execute various tactical commands such as attack, defend, snipe, etc. It is also interesting to observe that some bots might get stuck at some locations without using *Quartz*,

whereas there is no such case when *Quartz* is used. This indicates that the spatial information contained in the hierarchy of maps helps *Quartz* to plan path more effectively.

7.3.1 Mean Opinion Score Tests in *Quartz*

In order to validate the realism of the bot navigation in *Quartz*, further experiments are required. The work on validation of human crowd simulation by Pelechano [116] supports the use of feedback from human subjects to validate the realism of human behaviours in simulations. Lessons are taken from their work and use Mean Opinion Score (MOS) tests to judge the realism of the bot navigation with *Quartz*. 14 subjects (aged between 20 to 40 years old) were asked to watch two videos of bot navigation in *Twilight City*. The subjects' opinions on the realism of the bot navigation were collected. All the subjects have at least 2 years of military training. One of the video shows the bot navigation with *Quartz* and the other video shows the bot navigation with A* algorithm. To prevent biased opinions, subjects are not informed that the bots in the two videos are running on different navigation models. The subjects were asked to give their opinions on the realism of the bot navigation by giving a score (5-Excellent, 4-Good, 3-Fair, 2-Poor, 1-Bad) Figure 7.11 summarizes the results of these tests. Video1(*Quartz*) achieved a MOS score of 3.67 while Video2(A*) achieved a score of only 1.18. Generally, the subjects agreed that the realism of the bot navigation is Video1(*Quartz*) is better than the realism in Video2(A*). They noticed that, in Video2(A*), bots sometimes run across open areas which is very dangerous in a real MOUT scenario. They also noted that Video2(A*) bots do not use buildings or walls for cover and concealment. Instead these bots naively tries to traverse the best path from their start to end point. In comparsion, Video1(*Quartz*) bots avoids killing areas during navigation and shown a preference to move close to walls and buildings for cover and concealment.

Figure 7.11: Mean Opinion Score for bot navigation

7.4 Mean Opinion Score Tests for *Twilight City*

To further evaluate the performance of *Twilight City*, we have installed it on some PCs (DELL with Pentium IV 2.4GHz CPU) in the Parallel and Distributed Computing Center (PDCC) at Nanyang Technological University and the Modeling and Simulation Department of the Defense Science and Technology Agency of Singapore. 26 subjects (aged between 20 to 40 years old) were asked to play with *Twilight City* for about half an hour (see Figure 7.12). Then the subjects' opinions on various aspects of *Twilight City* were collected. As the system is meant mainly for military purpose, all the subjects have more than 2 years of military training experiences. The military experience of the human subjects will help to validate the accuracy of the behaviours simulation in *Twilight City*. Besides that, 16 of the subjects also had experience with military simulation systems before.

Figure 7.12: Conducting Mean Opinion Score Tests of *Twilight City*

It should be noted that the emphasis of these tests is not on the behavior of the AI bots that we have developed. Though the bots in *Twilight City* have demonstrated a certain level of tactical intelligence, the emphasis of these tests is on a subject's perception of the visual effects of the virtual environment and various artifacts that we have introduced. The subjects were asked to give their opinions on the following questions:

- (a) How do you feel the responsiveness of *Twilight City*?
- (b) Is *Twilight City* a good urban warfare simulation tool?
- (c) How do you feel the voice module (accuracy and responsiveness)?
- (d) How do you feel the behavior (realism and responsiveness) of the movable objects?

The subjects answer these questions by giving a score (5-Excellent, 4-Good, 3-Fair, 2-Poor, 1-Bad) to the performance of *Twilight City* on each of these aspects. Figure 7.13 summarizes the results of these tests.

Figure 7.13: Mean Opinion Score Tests Results

The subjects are satisfied with the responsiveness of *Twilight City* and the behavior of the movable objects that we have introduced. They also feel that these movable objects greatly enhance the realism of *Twilight City* as compared to other FPS games they have played. However, subjects suggested that movable objects should be modeled to show damages after collisions. For example, a part of the box should be chipped off upon colliding with the wall. Though some subjects suggest that the voice module has noticeable delays and failed to produce correct results in some cases, the results show that the voice module is generally acceptable by most subjects. In general, the subjects feel that *Twilight City* is a good prototype system for MOUT simulations. The movable objects, various artifacts and the voice module are important to MOUT simulations. However, more voice commands need to be added, and more training of the voice module are also needed.

During the data collection, it is stated that 16 out of 26 subjects had experience with simulation systems. Further analysis is done to find out how different are the results from this 16 subject with respect to the rest. In the evaluation of *Twilight City* as a realistic urban simulation tool, the subjects with simulation experience gave a MOS score of 4.375 against the MOS score of 4.4 from the subjects without simulation experience. In the evaluation of the responsiveness of *Twilight City*, the subjects with simulation experience gave a MOS score of 4.625 against the MOS score of 4.7 from the subjects without simulation experience. In the evaluation of the responsiveness of the voice module, the subjects with simulation experience gave a MOS score of 4.4 against the MOS score of 4.55 from the subjects without simulation experience. From the MOS score comparison It is noted that experience of simulation systems does not affect the MOS score by a significant amount. We believe that this is a positive sign which shows that the subjects understand the objectives of the experiment and are unbiased. The results are shown in Figure 7.14

Figure 7.14: MOS Results for Experienced Vs Non-experienced users

Chapter 8

Conclusions and Future Work

8.1 Summary

Time critical decision-making models are important to generate realistic behaviours for the intelligent agents in virtual training systems and computer games. Formal rational rules do not produce realistic behaviours 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 discussed in Chapter 3 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 unlikely 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.

To deal with the dynamic changes in MOUT simulations, we incorporated expectations into *SNAP*.

The expectations serve to

- (a) Validate the initial solution
- (b) Provide dynamic situation assessment.
- (c) Enable bots to adapt their behaviour to changing situations.
- (d) Allow bots to re-use past cues during experience

Our experiment results demonstrate that *SNAP* framework can provide realistic agent behaviours during MOUT simulations.

In Chapter 4, we highlighted that recent findings in decision making processes show that human experts often make decisions by comparing the current situation with their past experiences. This is very different from the goal-oriented or rational reasoning of conventional models. To enhance the effectiveness of *SNAP*, we produced a computational model of situation awareness. Our experimental results demonstrate that the computational model can help to generate realistic behaviours for bots under time pressure. The computational model follows the three stages of the Endsley’s model. In the perception stage, the bot senses are limited and can pick up heterogeneous cues. During the comprehension stage, similarity measures were computed and experience cases are retrieved. Finally, the novel feature of expectations is applied during the projection stage. Our experimental results also shows that the computational model closely models the situation awareness of human soldiers. Further frame rate analysis also shows that the computational cost is not high.

Chapter 5 describes our work on *Quartz*. The *Quartz* navigation system explores the human cognitive process in spatial reasoning. *Quartz* adopts qualitative spatial representations and hierarchical spatial reasoning. These two features make

Quartz more close to the way that humans handle daily-life spatial problems. As humans rely more on the qualitative relations among various objects rather than the exact quantitative descriptions of the world in their daily-life spatial reasoning, we believe that qualitative spatial representation and reasoning are more suitable for generating human-like movements for the bots.

Chapter 6 touched on the virtual environment built for MOUT simulations, *Twilight City*. The main emphasis of the chapter is to discuss the implementation of time critical decision making framework and autonomous navigation in a MOUT virtual environment.

Chapter 7 presented our experiment results in *Twilight City*. *Twilight City* performed well with the MOUT operations objectives implemented. Such tests increased the confidence level on the effectiveness of *Twilight City* and gave us a better understanding of its limitations as well. Tests done on autonomous navigation and time critical decision making provided firm evidence regarding the impact of this research on building intelligent bots behaviours for MOUT simulations.

8.2 Conclusions

In this project, we seek to provide intelligent bots behaviours for MOUT simulations. For an MOUT simulation to be effective, it is important for these bots to demonstrate some human-like tactical behaviors. 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 realism of the behavior model, but also the structural and procedural realism.

Time critical decision-making models are important to generate realistic behaviours for the intelligent agents in virtual training systems and computer games. Formal rational rules do not produce realistic behaviours 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. In our current work, a time critical decision making framework, *SNAP*, had been produced. From this framework, we intend to develop future bots. By varying the experience cases, we will be able to produce different behaviours for adversary, soldier and civilian bots.

Situation awareness provides the information for the bots to perform their cognitive tasks and decision making. The quality and relevance of the information directly affects the realism of the bot behaviours. Therefore, we proposed a computational model for situation awareness in MOUT simulations. This model is built to handle heterogeneous cue types within a dynamic environment. The computational situation awareness model works in tandem with the decision making and navigation components.

Our *Quartz* navigation system explores the human cognitive process in spatial reasoning. *Quartz* adopts qualitative spatial representations and hierarchical spatial reasoning. These two features make *Quartz* more close to the way that humans handle daily-life spatial problems. As humans rely more on the qualitative relations among various objects rather than the exact quantitative descriptions of the world in their daily-life spatial reasoning, we believe that the qualitative spatial representation and reasoning in *Quartz* are more suitable for generating human-like movements for the bots.

The *Twilight City* was produced for this project. It provided a virtual environment for MOUT simulations. Feedback from researchers showed that the proposed

virtual environment is effective for MOUT simulations. With *Twilight City*, we are able to test the realism of our bot behaviours after implementing *SNAP* and *Quartz*. The results obtained provided us with a good understanding on the merits and limitations of our work.

All in all, the research on bot behaviours had produced some significant results in the areas of decision making and autonomous navigation. We had produced a number of publications to share our current work.[13, 62, 70, 63] We hope to further extend our work to produce more significant contributions in building intelligent bots behaviours.

8.3 Future Work

Although the research made some contributions for producing intelligent bots behaviours in MOUT simulations, there is still room for more contributions. The following sections propose a number of recommendations that can be implemented to further extend our current work.

8.3.1 Terrain Reasoning

Solders' behaviours in MOUT situations are greatly affected by the terrain [84]. The selection of the low-level actions and high-level plans requires terrain reasoning. Terrain reasoning can be important in the following areas [85]

- Recognition of key terrain features and communicating about them
- Selection of plans and actions based on terrain
- Generating expectations based on locations
- Tactical considerations according to the terrain features

We seek to enhance the realism of bots behaviours by including terrain reasoning in our decisioning making framework. It will require us to work on terrain representation and corresponding functions to manipulate the terrain representation. To achieve our aim, more work on *Quartz* and *SNAP* may be required. *Quartz* can be used to provide terrain representations to provide terrain information. *SNAP* will use the terrain information to generate plans, actions and expectations.

8.3.2 Tactical Awareness

Building bot behaviours without tactical awareness serves little operational benefits for the training of soldiers. It is our goal to build bots that are able to serve either as challenging opponents or useful team members for our trainees. The bots should be able to understand the tactical considerations of various MOUT situations.

It is observed that tactical considerations are generated based on training, experience and terrain features. These considerations will influence the selection of plans and actions in bots. With a more realistic selection of plans and action, the training value of our MOUT simulations will be increased.

A number of important MOUT scenarios are selected and the tactical awareness of our bots will be tested on them. These scenarios include, ambush, room clearing and grenade throwing [86]. Once our bots are able to perform tactical reasoning in such scenarios, we can further extend our work to include a greater scope of MOUT situations.

8.3.3 Applying the current work to non-military areas

The time critical decision making model developed in *SNAP* can be applied to non-military areas. An area where this model may be applied to is in the realm of

high frequency trading(HFT). HFT is the execution of computerized trading strategies characterized by extremely short position-holding periods. In HFT, programs running on high-speed computers analyze market data, using algorithms to utilize trading opportunities that may open up for only a fraction of a second to several hours [118]. Much information happens to be unwittingly embedded in market data, such as quotes and volumes. By observing a flow of quotes, high-frequency trading machines are capable of extracting information that has not yet crossed the news screens. We believe that *SNAP* can be adapted to receive cues in the form of financial data, recognize current scenarios and generate profitable trades. We are trying to talk to potential investors who may be interested to provide funding for further exploration in this area.

Another area that the current work can be used to contribute in non-military applications is in area of computer game technologies. We can see that the techniques developed in this thesis may be used to improve some of the existing technologies in terms of decision making models, virtual environments and path planning. In the area of decision making, we believe that our model is able to generate more realistic behaviours for game characters. Such realism will provide a better in-game experience for gamers and enhance the value of the game. Furthermore, with our case-based reasoning framework, it will be easier for game developers to transfer human experiences into game content and generate believable behaviours. As movement of the bots is vital to the perception of realism, our autonomous navigation systems will improve the human-like realism of bots in games. This is true especially in games which are set in 3D urban terrains. Other techniques such as qualitative physics will also help to reduce computational costs of games while generating acceptable physics behaviours of common urban objects.

References

- [1] FM 90-10, Military Operations on Urbanized Terrain (MOUT), Headquarters, Department of the Army, Washington, D.C., 1979
- [2] United Nations Population Fund, State of World Population 2007: Unleashing the Potential of Urban Growth, accessed online at www.unfpa.org, on July 16, 2007.
- [3] Marine Corps Warfighting Laboratory, Urban Warrior: Conceptual Experimental Framework, United States Marine Corps, Virginia, United States, 1998
- [4] The Art of War, translated by L. Giles, Project Gutenberg, 1910
- [5] C. Gall, T. de Waal, Calamity in the Caucasus, New York University Press, New York, United States, 1998
- [6] Visiongain, Military Simulation and Virtual Training Market 2009-2019, Visiongain, United Kingdom, 14 Aug, 2009
- [7] FM 3-06, Urban Operations, Headquarters, Department of the Army, Washington, 2003
- [8] C. P. Stacey, Official History of the Canadian Army in the Second World War Volume III: The Victory Campaign, The Operations in North West Europe 1944-1945, 1960

REFERENCES

- [9] J. R. Ballard, Fighting For Fallujah, A New Dawn for Iraq, 2006
- [10] G. E. Moore, Cramming more components onto integrated circuits, Vision-gain, Electronics Magazine, McGraw-Hill, 19 Apr, 1965
- [11] R. E. Wray, J. E. Laird, Variability in Human Behavior Modeling for Military Simulations, 12th Behavior Representation in Modeling and Simulation Conference (BRIMS), Tennessee, USA, 12-15 May, 2003
- [12] H. A. Simon. Models of Man: Social and Rational, New York: John Wiley and Sons, Inc., pp. 279 , 1957
- [13] S. P. Ting, S. Zhou, Snap: A time critical decision-making framework for MOUT simulations, Journal of Visualization and Computer Animation, Vol 19, No. 3-4, 505-514, 2008
- [14] R. J. Pleban, D. E. Eakin, M. S. Salter, Analysis of Mission-Based Scenarios for Training Soldiers and Small Unit Leaders in Virtual Environments, ARI Research Report 1754, Fort Benning GA, Jan, 2000
- [15] J.E. Laird, R. Wray, Towards Human-like Adversaries for MOUT Training, 23rd Soar Workshop, Ann Arbor, MI, May, 2003
- [16] D. Ourston, D. Reece, Issues Involved with Integrating Live and Artificial Virtual Individual Combatants, Spring Simulation Interoperability Workshop, Orlando FL, March, 1998
- [17] G. R. George, F. Cardullo, Application of Neuro-Fuzzy Systems to Behavioral Representation in Computer Generated Forces, Proceedings of the Eighth Conference on Computer Generated Forces and Behavioral Representation, 575-585, Orlando FL, 11-13 May, 1999

REFERENCES

- [18] R.W. Pew, A.S. Mavavor, Modeling Human and Organizational Behavior: Application to Military Simulations, Final Report, National Research Council, National Academy Press. Washington D.C, 1998
- [19] M. Tambe, R. M. Jones, J. E. Laird, P. S. Rosenbloom K. B. Schwamb Building Believable Agents for Simulation Environments, in Proceedings of the AAAI Spring Symposium: Believable Agents, 1994
- [20] G. Tidhar, C. Heinze, S. Goss, G. Murray, D. Appl, I. Lloyd, Using Intelligent Agents in Military Simulation, Proceedings of Artificial Intelligence Conference, 829-836, American Association for Artificial Intelligence, Menlo Park, CA, 1999
- [21] N. Kasabov, Introduction: Hybrid intelligent adaptive systems, International Journal of Intelligent Systems, Vol.6, 453-454, 1998
- [22] J. S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach (2nd edition), NJ: Prentice Hall, 2003
- [23] S. Ginsburg, An Introduction to Mathematical Machine Theory, Addison-Wesley, 1962
- [24] A. M. Turing, Computing machinery and intelligence, Mind, 433-460, Oxford University Press, Oct, 1950
- [25] G. H. Mealy, A Method to Synthesizing Sequential Circuits, Bell Systems Technical Journal, 1045-1079, 1955
- [26] V. Novak, I. Perfilieva, J. Mockor, Mathematical principles of fuzzy logic, Dodrecht: Kluwer Academic, 1999
- [27] L. A. Zadeh, Fuzzy sets, Information and Control, 338-353, 1965

REFERENCES

- [28] R. H. Wilkinson, A method of generating functions of several variables using analog diode logic, *IEEE Transactions on Electronic Computers*, 112-129, 1963
- [29] R. A. Brooks, Intelligence Without Representation, *Artificial Intelligence Journal* 47: 139-159, 1991
- [30] T. Kent, Multi-tiered AI layers and terrain analysis for RTS games, *AI Game Programming Wisdom 2*, Charles River Media, Hingham, MA, 2004
- [31] N. Wallace, Hierarchical Planning in Dynamic Worlds, *AI Game Programming Wisdom 2*, Charles River Media, Hingham, MA, 2004
- [32] W. van der Sterren, Squad Tactics: Team AI and Emergent Maneuvers, *AI Game Programming Wisdom*. Charles River Media, Hingham, MA, 2002
- [33] J. Reynolds, Tactical Team AI Using a Command Hierarchy, *AI Game Programming Wisdom*, Charles River Media, Hingham, MA, 2002
- [34] M. Ramsey, Designing a Multi-Tiered AI Framework, *AI Game Programming Wisdom 2*, Charles River Media, Hingham, MA, 2004
- [35] P. Sweetsen, M. Watson, J. Wiles, Intelligent Command Agents for Command and Control Simulations using Cognitive Work Analysis and Machine Learning, University of Queensland, Brisbane, Australia, 2003
- [36] R. E. Wray, J. E. Laird, A. Nuxoll, D. Stokes, A. Kerfoot, Synthetic Adversaries for Urban Combat Training, *Proceedings of the 2004 Innovative Applications of Artificial Intelligence Conference*, San Jose, CA, Jul 2004
- [37] L. F. Barrett, Science of Emotion: What people believe, What evidence shows and Where to go from here, *Human Behavior in Military Contexts*, The National Academies Press, 2007

REFERENCES

- [38] D. P. Schultz, Panic in The Military, Technical Report, University of North Carolina, Storming Media, January 1971
- [39] P. McDermott, D. A. Battaglia, J. Phillips, M. Thordsen, Military Operations in Urban Terrain (MOUT): Decision Making in Action, US Army Research Institute, Fort Benning, Apr, 2001
- [40] G. Klein, Sources of Power: How People Make Decisions, MIT Press, Cambridge, Massachusetts, 1998.
- [41] R. Lopez De Mantaras, D. McSherry, D. Bridge Retrieval, D. Leake, B. Smyth, S. Craw, B. Faltings, M. L. Maher, M. T. Cox, K. Forbus, M. Keane, Aamodt, I. Watson, Retrieval, Reuse, Revision and Retention in Case-based Reasoning, The Knowledge Engineering Review, Vol. 20, Issue 3, 215-240, Sep, 2005
- [42] R. Azuma, M. Daily, C. Furmanski, A Review of Time Critical Decision Making Models and Human Cognitive Processes, Proceedings of the 2006 IEEE Aerospace Conference, MT, 4-11 Mar, 2006
- [43] F. Osinga, Science, Strategy and War: The Strategic Theory of John Boyd, Abingdon, 2007
- [44] F. C. Bartlett, Remembering: An Experimental and Social Study, Cambridge University Press, Cambridge, 1932
- [45] J. Akahani, K. Isbister, Digital City Project: NTT Open Laboratory, NTT Communication Science Laboratories 2-4, Kyoto, Japan, 2000
- [46] W. F. Brewer, J. C. Treyens, Role of schemata in memory for places, Cognitive Psychology, Vol. 13, 207-230, 1981
- [47] S. A. Widmayer, Schema theory: An introduction, <http://www2.yk.psu.edu/jlg18/506/SchemaTheory.pdf>

REFERENCES

- [48] M. Gladwell, *Blink: The Power of Think without Thinking*, Little, Brown and Co., 2005
- [49] K. Evans, *What Squad Leaders Want to Know in Battle*, US Army Research Institute, Fort Benning, GA, August, 2006
- [50] R. E. Wray, J. E. Laird, A. Nuxoll, R. M. Jones, *Intelligent opponents for virtual reality trainers*, The Interservice/Industry Training, Simulation and Education Conference (I/ITSEC), Orlando, FL, 2002
- [51] N. Nilsson, *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill, 1971
- [52] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Introduction to Algorithms*, Second Edition, 2001
- [53] A. Turner, A. Penn, *Encoding natural movement as an agent-based system: an investigation into human pedestrian behaviour in the built environment*, *Environment and Planning B: Planning and Design*, 473-490, 2002
- [54] A. Penn, A. Turner, *Space syntax based agent simulation*, *Proceeding of the 1st International Conference on Pedestrian and Evacuation Dynamics*, 99-114, University of Duisburg, Germany, 2002
- [55] A. G. Cohn, *Calculi for Qualitative Spatial Reasoning*, *International Conference on Artificial Intelligence and Symbolic Mathematical Computation*, Steyr, Austria, 23-25 Sep, 1996
- [56] D. Reece, M. Kraus, P. Dumanoir, *Tactical movement planning for individual combatants*, *9th Conference on Computer Generated Forces and Behavior Representation*, Orlando, FL, 2000
- [57] L. Chaimowicz, A. Cowley, D. Gomez-Ibanez, B. Grocholsky, M. A. Hsieh, H. Hsu, J. F. Keller, V. Kumar, R. Swaminathan, C. J. Taylor, *Deploying*

REFERENCES

- Air-Ground Multi-Robot Teams in Urban Environments, 2005 International Workshop on Multi-Robot Systems, 223-234, Washington DC, March, 2005
- [58] S. Schussman, M. Bertram, B. Hamann, K. I. Joy, Hierarchical data representations based on planar Voronoi diagrams, Visual Symposium, Amsterdam, 2000
- [59] G. Snook, Simplified 3D Movement and Pathfinding Using Navigation Meshes, Game Programming Gems 1, Charles River Media, 2000
- [60] J. W. Karagosian, Streetfighting: The Rifle Platoon in MOUT, Infantry Magazine, September, 2000
- [61] S. P. Ting, S. Zhou, Quartz: An Autonomous Navigation System for MOUT, 20th International Conference on Computer Animation and Social Agents (CASA2007), Hasselt, Belgium, 11-13 Jun, 2007
- [62] S. P. Ting, S. Zhou, Quartz: an autonomous navigation system for MOUT simulations, Journal of Visualization and Computer Animation, Vol 18, No. 4-5, 383-394, 2007
- [63] S. P. Ting, S. Zhou, Dealing with dynamic changes in time critical decision-making for MOUT simulations, Journal of Visualization and Computer Animation, Vol 20, No. 2-3, 427-436, 2009
- [64] K. Forbus, J. V. Mahoney, K. Dill, How qualitative spatial reasoning can improve strategy game AIs, 15th international workshop on Qualitative Reasoning, San Antonio, Texas, 2001
- [65] A. Kamphuis, M. Rook, M. H. Overmars, Tactical Path Finding in Urban Environments, International Workshop on Crowd Simulation, EPFL, Lausanne, Switzerland, Nov 24-25, 2005

REFERENCES

- [66] M. Knauff, R. Rauh, C. Schlieder, Preferred mental models in qualitative spatial reasoning: A cognitive assessment of Allen's calculus, 17th Annual Conference of the Cognitive Science Society, 200-205, 1995
- [67] K. Zimmermann, C. Freksa, Qualitative spatial reasoning using orientation, distance and path knowledge, *Applied Intelligence*, Vol. 6, 49-58, 1996
- [68] M. Rook, A. Kamphuis, Path Finding using Tactical Information, *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, Utrecht, 2005
- [69] S. T. Crino, Representation of urban operations in military models and simulations, *Winter Simulation Conference*, Proceedings of the 33rd conference on Winter simulation, Arlington, Virginia, 715 - 720, 2001
- [70] S. Zhou, S. P. Ting, Z. Q. Shen, L. Luo, Twilight City - A virtual environment for MOUT, *International Journal of Computer and Applications*, Vol 30, No. 2, 117-123, 2008
- [71] E. Lewis, M. Barlow, The Use of Games to Investigate Tactical Decision-making, *SimTecT 05*, Sydney, Australia, 10-12 May, 2005
- [72] FM 90-10-1, An Infantry man's Guide to Combat in Built-up Areas, Headquarters, Department of Army, Washington, DC, 12 May, 2003
- [73] A. Al Ahuja, Determining battlefield effects in an urban environment: MOUT terrain analysis - Military Operations in Urban Terrain, *Military Intelligence Professional Bulletin*, Jul-Sep, 2003
- [74] FM 34-130, Intelligence Preparation of the Battlefield, Headquarters, Department of Army, Washington, DC, 8 Jul, 1994
- [75] S. Zhou, S. P. Ting, Qualitative Physics for MOUT, *Proceedings of the 39th Annual Simulation Symposium*, 2 - 6 Apr, 2006, Huntsville, AL, USA

REFERENCES

- [76] K. D. Forbus, Qualitative process theory, *Artificial Intelligence*, Vol. 24, No. 1-3, 85-168, 1984
- [77] S. P. Ting, S. Zhou, An Architecture for Rapidly Re-configurable MOUT Simulation, 40th Annual Simulation Symposium, Norfolk, VA, 26 - 28 Mar, 2007
- [78] S. Zhou, S. P. Ting, A Generic Model Framework for MOUT Simulations, 2006 International Conference on CYBERWORLDS, Lausanne, Switzerland, 28-30 Nov, 2006
- [79] R. Adobbati, A. N. Marshall, A. Scholer, S. Tejada, G. A. Kaminka, S. Schaffer, C. Sollitto, GameBots: A 3D virtual world test bed for multiagent research, *Proceedings for the Second International Workshop on Infrastructure for Agents*, Montreal, Canada, 2001
- [80] A. N. Marshall, JavaBot for Unreal Tournament, 2003, <http://utbot.sourceforge.net>.
- [81] W. Warwick, R. Hutton, Developing Computational Models of Recognition-Primed Decision Making, *Proceedings of the Tenth Conference on Computer Generated Forces*, 232-331, Norfolk, VA, 2001
- [82] J. F. Schmitt, G. A. Klein, Fighting in the Fog: Dealing with Battlefield Uncertainty, *Marine Corps Gazette*, 62-69, Quantico, Virginia, 1996
- [83] Y. Q. Ji, H. Ying, R. M. Massanari, J. Yen, J. Ager, A Fuzzy Logic-Based Computational Recognition-Primed Decision Model, *Information Sciences*, Vol. 177 (20), 4338-4353, Elsevier, 2007
- [84] D. Pottinger, Terrain Analysis for Realtime Strategy Games, *Game Developer Conference 2000*, San Jose, CA, Mar 20-24, 2000

- [85] W. van der Sterren, Terrain Reasoning for 3D Action Games, Game Developer Conference 2001, San Jose, CA, March 20-24, 2001
- [86] W. van der Sterren, AI for Tactical Grenade Handling, CGD-AI, <http://www.cgf-ai.com/docs/grenadehandling.pdf>, Dec, 2000
- [87] L. Luo, S. Zhou, W. Cai, M. Low, M. Lees, Towards a Generic a Framework for Modeling Human Behaviours in Crowd Simulation, IAT 2009
- [88] L. Luo, S. Zhou, W. Cai, M. Lees, M. Low, Modeling Human-like Decision Making for Virtual Agents in Time-critical Situations, 2010 International Conference on CYBERWORLDS, Singapore, 22-24 Oct, 2010
- [89] N. Hu, S. Zhou, Z. Wu, M. Zhou, B. E. K. Cho, Spatial-temporal patterns and pedestrian simulation, Computer Animation and Virtual Worlds, Vol. 21, 387399, 2010
- [90] J. Gemrot, R. Kadlec, M. Bida, O. Burkert, R. Pibil, J. Havlicek, L. Zemcak, J. Simlovic, R. Vansa, M. Stolba, T. Plch, C. Brom, Pogamut 3 Can Assist Developers in Building AI (Not Only) for Their Videogame Agents, Agents for Games and Simulations, LNCS 5920, Springer, 1-15, 2009
- [91] B. Gorman, M. Fredriksson, M. Humphrys, The QASE API - An Integrated Platform for AI Research and Education Through First-Person Computer Games, International Journal of Intelligent Games and Simulations, 2007, <http://ut3bots.codeplex.com/> (27.9.2009)
- [92] P. Hingston, A New Design for a Turing Test for Bots, IEEE conference of Computational Intelligence and Games, Copenhagen, 2010
- [93] P. Hingston, A Turing Test for Computer Game Bots, IEEE Transactions on Computational Intelligence and AI In Games, Vol. 1, No. 3, 169-186, Sep, 2009

REFERENCES

- [94] Department of Army, Military Operations on Urbanized Terrain (MOUT), FM 90-10, Washington, DC, Aug, 15, 1979
- [95] J. Catagnus, Jr. Earl, B. Z. Edison, J. D. Keeling, D. A. Moon, Infantry Squad Tactics: Some of the Lessons Learned during MOUT in the Battle for Fallujah, Marine Corps Gazette, Sep, 2005
- [96] B. W. Knerr, D. R. Lampton, An Assessment of the Virtual-Integrated MOUT Training System (VIMTS), Army Research Institute for the Behavioral and Social Sciences, 2005
- [97] M. R. Endsley, Toward a theory of situation awareness in dynamic systems, Human Factors, Vol. 37 No. 1, 32-64, Human Factors and Ergonomics Society, Santa Monica, US, 1995
- [98] R. W. Hill, Modeling perceptual attention in virtual humans, Conference on Computer Generated Forces and Behavioral Representation, Orlando, US, May, 1999
- [99] P. Herrero, A. Antonio, Introducing human-like hearing perception in intelligent virtual agents, Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-agent Systems, 733 - 740, Melbourne, Australia, 2003
- [100] Epic Games, Unreal engine, <http://unrealtechnology.com>, 1998
- [101] J. S. McCarley, D. W. Christopher, J. Goh, W. J. Horrey, A computational model of attention situation awareness, Proceedings of the 46th Annual Meeting of the Human Factors and Ergonomics Society, Santa Monica, US, 2002
- [102] W. Warwick, R. Hutton, Developing computational models of recognition primed decision making, Proceedings of the Tenth Conference on Computer Generated Forces, 232-331, Norfolk, US, 2001

REFERENCES

- [103] M. R. Endsley, Situation awareness global assessment technique (SAGAT), In Aerospace and Electronics Conference NAECON, US, 1988
- [104] J. E. Laird, It knows what youre going to do: adding anticipation to a quakebot, Proceedings of Fifth International Conference on Autonomous Agents. Montreal, Canada, 2001
- [105] C. Burdick, H. Graf, J. Huynh, M. Grell, D. MacQueen, H. Argo, P. Bross, J. Prince, H. Johnson, H.R. Blacksten, Heterogeneous agent operations in JWARS, Proceedings of the 2003 Conference on Behaviour Representation in Modelling and Simulation, Scottsdale, Arizona, 2003
- [106] D. Vakas, J. Prince, H. R. Blacksten, Commander behaviour and course of action selection in JWARS, Proceedings of the 2001 Winter Simulation Conference, 2001
- [107] Department of the Army, FM101-5 The Military Decision-Making Process, US Government Printing Office, Washington, DC, 2003
- [108] T. Clancy, Red Storm Rising, Harper Collins, 1988
- [109] R. C. Rubel, The epistemology of war gaming, Naval War College Review, 2006
- [110] M. Fuchida, Midway: The Battle That Doomed Japan, Naval Institute Press, 2001
- [111] T. B. Allen, War Games: Inside the Secret World of the Men who Play at Annihilation, New York, McGraw Hill, 1987
- [112] J. Laird, P. Rosenbloom, A. Newell, J. Allen Soar: An Architecture for General Intelligence. Artificial Intelligence, 33: 1-64, 1987
- [113] A. Newell, Unified Theories of Cognition, Harvard University Press, 1990

- [114] M. Tambe, W. L. Johnson, R. M. Jones, F. Koss, J. E. Laird, P. S. Rosenbloom, K. Schwamb, Intelligent agents for interactive simulation environments, *AI Magazine*, 16, 15-40, 1995
- [115] J. M. Loomis, J. M. Knapp, Visual perception of egocentric distance in real and virtual environments, *Virtual and Adaptive Environments*, L. J. Hettinger and M. W. Haas (Eds.), 21-46, 2003
- [116] N. Pelechano, C. Stocker, J. Allbeck, N. Badler, Being a Part of the Crowd: Towards Validating VR Crowds Using Presence, *Virtual and Adaptive Environments*, Seventh International Joint Conference on Autonomous Agents and Multi-Agent Systems. (AAMAS'08), May 12-16, 2008
- [117] S. P. Ting, Virtual Training System for Military Operations on Urbanized Terrain, Masters of Engineering Dissertation, Nanyang Technological University, 2006
- [118] I. Aldridge, High-Frequency Trading: A Practical Guide to Algorithmic Strategies and Trading Systems, Wiley, 2009
- [119] R. S. Kalawsky, Exploiting Virtual Reality Techniques in Education and Training: Technological Issues, 58, The Advisory Group on Computer Graphics (AGOCG), Jul, 1996
- [120] C. E. Campbell, G. McCulley, Terrain reasoning challenges in the CCTT dynamic environment, *AI, Simulation, and Planning in High Autonomy Systems*, Proceedings of the Fifth Annual Conference on Distributed Interactive Simulation Environments, 55-61, 7-9 Dec, 1999
- [121] B. D. Reich, H. Ko, W. Becket, N. I. Badler, Terrain reasoning for human locomotion, *Proceedings of Computer Animation 1994*, IEEE Computer Society Press, 1994