An Explainable Artificial Intelligence System for Small-unit Tactical Behavior

Michael van Lent
The Institute for Creative Technologies, The University of Southern California
13274 Fiji Way, Marina del Rey, CA 90292 USA
vanlent@ict.usc.edu

William Fisher, Michael Mancuso
Quicksilver Software, Inc.
Irvine, CA USA
bfisher@quicksilver.com, mmancuso@quicksilver.com

Abstract

As the artificial intelligence (AI) systems in military simulations and computer games become more complex, their actions become increasingly difficult for users to understand. Expert systems for medical diagnosis have addressed this challenge though the addition of explanation generation systems that explain a system's internal processes. This paper describes the AI architecture and associated explanation capability used by Full Spectrum Command, a training system developed for the U.S. Army by commercial game developers and academic researchers.

Introduction

Full Spectrum Command (FSC) is a "commercial platform training aid" developed by the USC Institute for Creative Technologies and Quicksilver Software, Inc. for the U.S. Army. A commercial platform training aid is a computer game designed not for entertainment, but as a training tool to achieve a targeted training objective. Like most video games and training simulations, Full Spectrum Command includes a significant artificial intelligence (AI) system that controls the behavior of the user's subordinate units and the opposing forces. This complex AI system, which controls up to 200 entities and uses 60% of the CPU's processing cycles, is an excellent example of game industry AI technology, a branch of AI little studied by the academic research community. In addition, Full Spectrum Command includes an Explainable AI (XAI) feature that is the result of an academic research effort motivated by previous work in systems such as Mycin (Shortliffe 1976), Debrief (Johnson 1994), and the Explainable Expert System (Swartout, Paris and Moore 1994). The goal of this paper is to describe the XAI system and its application in Full Spectrum Command. However, this requires an understanding of the AI system in FSC that controls the Non-Player Characters (NPCs) (i.e. the computer controlled soldiers). This NPC AI is interesting in its own right as an example of a game industry AI system, including the methodologies and technologies employed by game developers.

Video games are probably the largest commercial applications of artificial intelligence today. Many video game reviews, in magazines or on the web, use the quality of a game's AI as a major criterion for evaluation. Most game developers list "AI Programmer" as a job title and game technology conferences include numerous talks on advances in AI technology. While a number of researchers are working to apply research techniques to video games (Laird and van Lent 2001, Young et. al., 2004), few researchers have studied and reported on the AI techniques being developed by the video game industry. Describing FSC's NPC AI system will providing one example of an AI system developed by a commercial game developer.

In the military modeling and simulation community the academic AI research community is well-represented and state-of-the-art AI technologies are being used in many systems (Laird, Jones and Neilsen 1994). Interestingly, in both games and military simulations as the AI systems get more complicated and generate more complex behaviors a new problem arises. The correlation between a player or human controller's orders and the AI-controlled entity's behavior sometimes becomes less obvious as the AI becomes more sophisticated. One solution is to extend the AI systems so they can explain their behavior either during execution or after the fact. Ideally, this Explainable AI can present the user with an easily understood chain of reasoning from the user's order, through the AI's knowledge and inference, to the resulting behavior.

Full Spectrum Command includes an Explainable AI system that allows the user to click on any subordinate soldier during the after-action review phase of the game and ask that soldier questions about the current behavior of its platoon. In addition, the Explainable AI identifies key events to be highlighted in the after-action review.

The next section of this paper describes the Full Spectrum Command commercial platform training aid.

Figure 1: The execution matrix development screen from the planning phase. The column on the right is one phase of an execution matrix populated with a task for each of four platoons. A fifth platoon has no task assigned.

The following section gives a detailed overview of the game's AI system. The next section describes the Explainable AI system in more detail and fits it into the context of previous work on explanation systems. The fifth section details the deployment of Full Spectrum Command and provides some feedback and evaluation results. Finally, the last section describes a number of directions for future research that could improve the effectiveness of the Explainable AI system.

Full Spectrum Command

Unlike traditional video games, which are designed for entertainment and profitability, Full Spectrum Command is designed to train and exercise the cognitive skills involved in commanding a light infantry company. However, this doesn't mean that FSC isn't entertaining. Entertainment is used as a powerful tool to increase the application's effectiveness as a training system. Because FSC is engaging, soldiers spend more time using the training aid and even choose to train in their free time.

Full Spectrum Command is a Windows XP application that requires a 2GHz Pentium 4 processor, 512 MB of system RAM, and a GeForce4 graphics accelerator with 128 MB of video RAM. In the training aid the user is placed in the role of a U.S. Light Infantry Company Commander with the rank of Captain. The user controls about 120 soldiers who are divided into 3-5 platoons and associated elements. The FSC missions focus on urban combat against an asymmetric opponent (i.e. terrorists and insurgents who don't wear uniforms or fight as an organized military unit). The game's virtual environment recreates the physical urban combat training site located at Ft. Benning, GA. Called the McKenna MOUT (Military Operations, Urban Terrain) site, this is where light infantry company commanders conduct live training exercise in urban combat. Our virtual McKenna can be populated with up to 80 civilians and opposing soldiers and configured for daytime or nighttime missions.

Full Spectrum Command's game-play is most similar to the Real Time Strategy (RTS) game genre which includes games such as Warcraft and Age of Empires. In RTS games the player acts as a commander controlling the gathering of resources, production of units, and issuing orders to these units in battles against one or more opponent commanders. In most RTS games the player views the world from a top-down or "God's eye" view.

Full Spectrum Command differs from traditional RTS games in a number of key ways. First, FSC doesn't include the resource gathering and unit production aspects of most RTS games. In the game, as in real life, company commanders begin each mission with an assigned company of soldiers that may or may not be at full strength. A "mission" in FSC proceeds in three phases: planning, execution, and after-action review. In the planning phase the user reads the operation order (the goal and details of the assigned mission), organizes the composition of his forces, and creates an initial plan for the entire mission.

Figure 2: The mission playback screen from a Full Spectrum Command after-action review session.

This plan is represented as an "execution matrix" which includes a row for each sub-unit of the company (each platoon) and a column for each time-based phase of the mission plan (see Figure 1). "Go codes" (key words called over the radio) or pre-defined time stamps are set up to coordinate the actions of the platoons and transition between phases of execution. Once the plan is complete the user moves on to the execution phase by clicking the "Execute" button.

In the execution phase the user monitors the behavior of his soldiers as they carry out the execution matrix, gathers information about the location, composition and actions of the opposing soldiers, and modifies the initial plan by issuing "on the fly" orders called fragmentary orders or Fragos. Unlike the God's eye view of traditional RTS games, the user in FSC must monitor the battle and adjust his plan from the first-person perspective of the company commander's avatar in the game world, which is sometimes fairly far away from the action. This includes the possibility that the user's avatar will be killed which results in immediate mission failure. During the execution phase the user moves around the environment, sends and receives radio transmissions from both subordinate and superior units, monitors the actions of his soldiers and the opposition, and issues Fragos to modify his initial plan. An important part of the company commander's job is maintaining "situational awareness" – a clear mental model of what is happening in the simulated world. This includes recognizing when the mission objective has been achieved or when the company is incapable of achieving the mission.

The final phase of Full Spectrum Command is the after-action review (AAR). Both the military (Morrison and Meliza 1999) and educational researchers (Katz and Lesgold 1994) have identified the importance of post-problem reflection. During the AAR the user and an instructor are provided with a new set of user interfaces and tools designed to help uncover what happened, why it happened, and how the user could have done better. The AAR interfaces include a mission statistics screen, an operation order review screen, a screen that presents the user's initial plan for review, a similar screen that presents the opponent's plan for review, and a mission playback screen. From an AI perspective the most interesting components of the after-action review is the mission playback screen.

The mission playback screen gives the user and instructor the ability to replay the mission with controls to pause, rewind and fast forward the playback (see Figure 2). Unlike the execution phase, the user views the playback through a God's eye camera that can move freely to see what was happening at any point in the game at any time.

The Explainable AI system enhances this playback capability in two ways. First, a set of rules extracts key events from the mission log and represents these as hash marks on a mission time line. Places on the time line where these event and decision hash marks are tightly grouped are often particularly important points in the replay. The user can scroll along the time line and view some details of each hash mark including buttons to jump to the time and location of that event or decision. The second

enhancement the XAI system provides is the capability to click on any AI-controlled soldier in the playback window and access a pop-up menu of questions that can be asked of that soldier. The content of these questions and the corresponding answers, as well as the key events and decision points, will be described in more detail during the discussion of the XAI system later in this paper.

The AI System

There are two artificial intelligence systems in Full Spectrum Command. The Non-Player Character (NPC) AI controls the behavior of the player's subordinate units, the opposing force, and any civilians during the execution phase. The Explainable AI (XAI) works during the after-action review phase to extract key events and decision points from the playback log and allow the NPC AI controlled soldiers to explain their behavior in response to the questions selected from the XAI menu. In order to describe the application of the explainable AI research in Full Spectrum Command it is necessary to first provide an overview of the NPC AI which the XAI seeks to explain. A description of the NPC AI is also interesting as an example of an artificial intelligence system developed not by academic researchers but by commercial game developers. While the underlying ideas have similarities to a number of AI research areas their realization is driven by the processing, memory, and development effort constraints of industry developers.

The NPC AI in Full Spectrum Command is divided into two AI subsystems; a Control AI and a Command AI (see Figure 3). The Command AI and Control AI software modules act as cooperative, event-driven message handlers. The simulation sends a stream of event messages to the Control AI which determines if each event requires an immediate reaction (i.e. return fire or take cover). If so, the Control AI responds with an action notification. Depending on the event type the Control AI may pass the event to the Command AI which generates higher-level tactical behavior. Because the Command AI operates in a separate processing thread, it can take additional time to generate more complex behavior based on a database of task knowledge.

The Control AI is responsible for reactive behavior and low level actions of individual soldiers and, in some cases, the smallest unit or grouping of soldiers (a four soldier Fire Team). It manages all low level decision-making and reactive behaviors (e.g. path planning, find cover, return fire) for NPC Objects. The Command AI is responsible for the behavior of the higher echelon units such as platoons (about 30 to 50 soldiers in 3-5 squads), squads (typically 9 soldiers in two fire teams plus a squad leader), and the fire team behavior not covered by the Control AI. The Command AI manages all strategic level decision making functions (e.g. task selection, task assignment, and unit organization). The Control AI is integrated into the application's basic simulation loop along with standard game components such as rendering, user input and physics simulation. To maintain an acceptable frame rate the

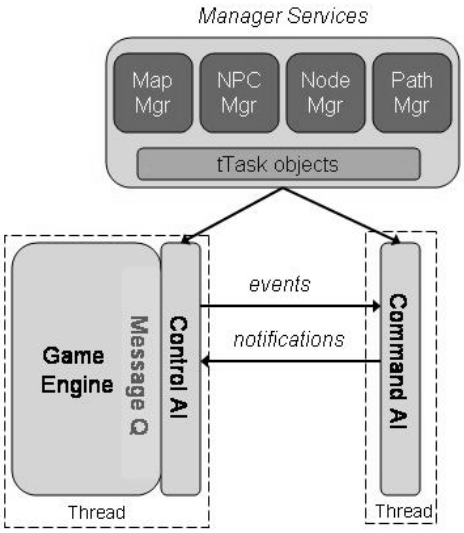

Figure 3: The structure of the AI systems in Full Spectrum Command.

simulation needs to cycle through this loop at least 30 times a second placing strict time constraints on each component of the loop. Thus the Control AI must generate NPC behaviors, not only quickly, but in a consistently predictable number of milliseconds. The Command AI runs in a separate processing thread from the simulation loop and therefore can take more time to generate higher level unit behavior without impacting the simulation loop and frame rate. Both the Control AI and the Command AI interact with a series of manager services to sense and act on the state of the simulation. These services include an NPC manager, a task manager, a map manager a node manager, and a path manager.

In each cycle through the simulation loop the Control AI is called to execute the atomic actions of the Control AI's current set of tasks. These atomic actions include moving a soldier or fire team to a new location, changing a soldier's stance (standing, kneeling, prone) or firing a soldier's weapon. The majority of the Control AI's time is spent on path planning for moving soldiers to new locations. Full Spectrum Command's map includes an embedded navigation graph stored in the map manager. NPCs moving through the environment are actually moving from node to node along the edges of this graph. The graph is carefully constructed so that an NPC that moves along the edges of the graph will never collide with a barrier in the game world. Thus, the path planning code doesn't need to know anything about the locations of buildings, trees and other barriers in the game world. All this spatial information is implicitly represented in the navigation graph. The navigation graph was developed by an automated graph generation algorithm with a great deal of fine tuning by the developers. This is a common approach to path planning in the game industry that has the advantage of very efficient path planning through an A* search over the navigation graph. The navigation graph does require some additional

storage, beyond the polygonal map representation used by the graphics renderer, and fine tuning the graph is a time consuming process. First person shooter games, such as Quake and Unreal Tournament, usually include fully automated navigation graph generation which allows users to easily develop and modify maps without having to embed their own navigation graphs. The transfer of such computationally intense tasks from run time to compile time is key to the success of this implementation strategy.

The Control AI changes the stance of an NPC soldier by simply updating the state of that NPC in the NPC manager which in turn changes how the character is rendered by the graphics engine. Weapon firing is decided when a line of sight calculation determines that an NPC has spotted an opposing soldier. The Command AI sends a message to the Control AI calling for a fire action and specifying a shooter NPC and a target NPC. The fire-at-enemy action notification is then sent by the Control AI to the simulation. The outcome of the weapons fire is determined by a statistical "combat calculator" that takes into account factors such as range, shooter and target stance, weapon type, and the target's body armor.

The Command AI, running in a separate thread, takes as input the platoon-level tasks assigned by the user, the task knowledge stored in the task manager, and a constantly updated list of events passed from the simulation thread through the Control AI to the Command AI. Full Spectrum Command tasks are organized into a task hierarchy containing N total tasks in M levels. At the top of the task hierarchy are N platoon-level tasks, such as clear-building, move-to-checkpoint, and execute-supporting-fire, and M high-level tasks used only by the opposing force (or OPFOR), such as shoot-and-scoot and ambush. As described previously, during the planning phase of FSC the user creates an execution matrix that assigns one of these platoon-level tasks to each platoon for each phase of the mission. Similarly, for each mission the mission designer creates an execution matrix assigning tasks for each OPFOR unit. The Command AI takes each top-level task and decomposes it into sub-tasks, sub-sub-tasks and so on. Each time a task is decomposed the Control AI breaks the associated unit into sub-units and assigns a sub-task to each unit.

For example, when a platoon begins a clear-building task the Command AI will decompose that task into two sub-tasks; secure-building-perimeter and assault-building. The platoon will be broken into two parts, in this case a single security squad and a multiple squad assault unit. The security squad will be assigned the secure-building-perimeter tasks while the assault unit will be assigned the assault-building tasks. The assault-building task will then be further decomposed into one or more clear-floor tasks, one for each floor of the building. The assault unit will be divided into floor clearing units (typically squads), who are then assigned to clear the floors in a specific order, one at a time. The clear-floor task for the initial floor will be divided again into multiple clear-room tasks, assigned to fire teams, with each clear-room task resulting in a set of movement orders for the individual soldiers to achieve "positions of dominance" within that room. Once the first room is cleared that fire team becomes available and can be assigned to another clear-room task on the same floor. In decomposing the tasks and forces the Command AI generally follows U.S. Army doctrine although some deviations do occur due to difficulties in getting subject matter experts to agree on an interpretation of the doctrine and constraints on the development timeline. While these differences aren't apparent when observing the resulting behavior they will have ramifications for the explainable AI.

Events in Full Spectrum Command consist of a set of pre-conditions, a set of goal conditions, and a set of task abandonment conditions. In addition, each task includes a list of child tasks in the task hierarchy and is associated with a C++ function that is called to execute the task. A task is executed when it is invoked by a parent task (or directly from the user's initial plan) and its pre-conditions are satisfied by the events passed to the Command AI. When a task is executed its C++ function is called which performs the unit and task decomposition and/or sends an action notification to the Control AI. If a task's abandonment conditions are met the Command AI stops executing the task and a report is sent to the user. For example, if too many soldiers are wounded or killed a unit may become "combat ineffective" in which case the remaining soldiers will seek cover and a radio report to the user will report that the unit has taken heavy casualties.

Explainable AI in Games and Simulations

As the previous description makes clear, the behavior of the individual NPCs in Full Spectrum Command starts with the user's platoon-level orders but is the result of a number of unit and task decomposition mixed with the reactive behavior of the Control AI. Other video games and military simulation systems include similarly complex processes to generate entity level behavior from orders to larger groups or units. As these AI systems have become more complex their inner workings have become less obvious to users. In training simulations, this can result in users who feel their orders don't really affect the NPC's behavior or users who may try to place blame for failure on the AI systems rather than their own orders.

Related Work

Interestingly, the problem of explaining the internal processing of an AI system has been previously considered by the research community only in a different context. Medical diagnosis expert systems, such as Mycin (Shortliffe, 1976) used a complex set of rules to diagnose illness and suggest treatments based on patient statistics and test results. Early on the developers of these systems realized that doctors weren't willing to accept the expert system's diagnosis on faith. As a result these systems were designed not only to diagnose an illness but also provide

explanations of how the diagnosis was generated. Explanation systems have also been applied to educational applications such as teaching LISP programmers to improve their code (Swartout, Paris and Moore 1994) and to assist knowledge engineer with debugging description logics (McGuinness and Borgida 1995).

One previous effort in the military simulation field is the Debrief system (Johnson, 1994). Debrief allows Soar agents to justify their actions in the TacAirSoar tactical air combat domain (Laird, Jones and Nielsen 1994). Debrief uses Soar's built in learning mechanism to continually store the agent's state during a mission. During the after action review (or debriefing) the system can recall the state of the agent at various points during the mission and experiment with different modifications to that state to determine which aspects of the state were critical to the agent's next decision. By performing this analysis Debrief can report the critical attribute values that resulted in the agent's decision. Debrief can also justify an agent's internal beliefs through a similar process of recalling an agent's state and examining how the agent's beliefs change in response to changes to state attribute values. Debrief explanations are presented through structured natural language phrases and graphical displays of aircraft positions. Debrief makes use of a number of Soar-specific mechanisms, such as chunking, and thus isn't directly applicable to systems not based on Soar. However, the underlying idea of logging an agent's behavior and examining that log during the after-action review to extract critical state features and explanations was influential in the development of the XAI system for Full Spectrum Command.

Explainable AI in Full Spectrum Command

The Explainable AI system in Full Spectrum Command logs the activities of the NPC AI system during the execution phase and uses that log during the after-action review phase. The log consists of a long sequence of AI events records each with an associated time stamp. The events that are recorded in the log include:

Weapon Fire: generated when an NPC shoots at another NPC. Records the weapon type, range, shooter and target details, chance to hit, hit result (did the shooter hit) and the effect of the shot on the target.

Unit Generation: generated when the player or NPC AI creates a new unit (i.e. the player creates a fourth platoon within the company or the NPC AI decomposes a platoon into squads). Records the unit id, parent unit's id, formation, unit purpose (security, assault, guard…), NPC count, task assignment, status (idle, waiting, active), leader id, and whether the unit is in position or on the move.

Unit Update: generated when the details of a unit are modified by the player or NPC AI. Records the aspects of a unit that could be changed which are formation, unit purpose, task assignment, status and in-position. NPC count doesn't change since wounded or killed NPC are still considered part of the unit.

Task Generation: generated when a unit is assigned a new task (i.e. clear-building is decomposed into secure-building-perimeter and assault-building). Records task id, id of the unit to which the task is assigned, task status (complete, incomplete, inactive) and parent task id.

Task Update: generated when the details of a task are modified (i.e. records that secure-building-perimeter is completed). Records the task id and task status which is the only detail of a task that can change.

A typical mission in FSC will take between 30 minutes and two hours and will generate a log consisting of thousands or tens of thousands of AI event records. These records are continually streamed to the hard drive as part of the after-action review file generated for each mission.

At the beginning of the after-action review phase the AI log is read from the hard drive and the XAI system reads through all the records. During this initial pass the XAI system gathers mission statistics and a list of important events to support the after-action review process. The mission statistics gathered includes mission duration, casualty statistics and ammo usage statistics. While gathering these statistics doesn't involve any AI technique, the information logged for the XAI is the easiest place to gather this data.

During this first pass through the logged AI events the system also gathers a list of important events that should be highlighted for the user. These events are indicated by hash marks on a mission time line displayed at the bottom of the mission playback screen (see Figure 2). The key event hash marks are color coded according to which side the event relates (blue for the user's units, red for enemy units and white for civilians). The XAI system detects nine types of key event:

- Friendly Soldier Killed in Action (KIA)
- Friendly Soldier Wounded in Action (WIA)
- Enemy Soldier Killed in Action
- Enemy Soldier Wounded in Action
- Civilian Killed in Action
- Civilian Wounded in Action
- Friendly Unit First Contact with the Enemy
- Platoon Task Started
- Platoon Task Completed

The KIA and WIA events are a subset of the logged Weapon Fire records in which the effect of the fire is a killed or wounded NPC. The platoon task started and completed events are a subset of the logged Task Generation and Task Completion records in which the unit associated with the task is a platoon. The XAI system has the capability to also include squad and fire-team level task events. However, as a Company Commander the user should be focusing on the activities of the platoons which are his immediate subordinate units. Finally, a "First Contact with the Enemy" event is generated from a Weapon Fire record that represents the first time an NPC in the associated unit has fired on the enemy.

The third job of the XAI system is to explain the behavior of the platoons during the mission playback. As the user moves through the mission playback the XAI

system moves through the logged AI records to maintain the current state of each unit and task. Similar to Debrief, the XAI system uses the logged information to recall the state of the NPC AI at any point during the mission. Unlike Debrief, the XAI system doesn't perform "what if" simulations but does allow the user to inquire about the status of tasks and units. On the playback screen the user can click on any friendly NPC to pop up a menu of questions the user can select to inquire about the current behavior of that NPC's platoon. Early versions of the XAI system included questions and explanations of squad and fire-team level behaviors. However, these were later removed in part to focus the user on the behavior of the platoons and in part because the explanations uncovered aspects of the NPC behavior that doesn't match U.S. Army doctrine. Limiting the user to a menu of pre-defined questions ensures that the system can answer the posed question.

The questions and answers are presented in English. Each question and answer has an associated sentence (or multi-sentence) template that is filled in with the details of the specific situation. The questions available to the user describe the platoon's current task, the status of that task, and the parameters that might affect task execution. These questions and the associated explanations are presented in English. These questions are:

What is the platoon's mission? The answer describes the platoon's top-level task and how that task is decomposed into the next level of sub-tasks. An example answer might be "2nd platoon's mission is clear building. This mission consists of two steps: secure building perimeter and assault building."

What is the platoon's mission status? The answer describes the status of the platoon's top-level task (active, complete, waiting) and the status of each sub-task. An example answer might be "2nd platoon's clear building task is active. 1st squad's secure building perimeter task is active. 2nd squad's assault building tasks is waiting."

How is the platoon task organized? The answer describes how the platoon is decomposed into the next level of sub-units and the number of soldiers in each sub-unit. An example answer might be "2nd platoon is task organized into 3 squads. 1st squad has 9 soldiers. 2nd squad has 9 soldiers. 3rd squad has 7 soldiers."

How many soldiers are in the platoon? The answer describes the total number of soldiers in the platoon and how many of these soldiers have been wounded or killed. An example answer might be "2nd platoon has 25 soldiers. Of these 2 soldiers are KIA and 1 soldier is WIA."

What is the platoon's ammo status? The answer describes the ammo status of the platoon as green (less than 75% of ammo has been used), red (more than 75% of ammo has been used), or black (no ammo remaining) for each weapon type. An example answer might be "2nd platoon is ammo status black for heavy weapons and ammo status red for light weapons."

Preliminary Deployment and Evaluation

Full Spectrum Command has been deployed for evaluation purposes in the Infantry Captain's Career Course (ICCC) at the Infantry School at Ft. Benning. Starting in the summer of 2002 beta versions were used in classroom exercises to teach concepts such as execution matrix development and battlefield synchronization. The final software was delivered in February 2003 and has been in use since. In addition, Full Spectrum Command was used in Afghanistan by U.S. soldiers tasked with training Captains in the newly created Afghan National Army.

In 2003 the Army Research Institute conducted an evaluation of the pedagogical effectiveness of Full Spectrum Command. Although the evaluation wasn't designed to explicitly test the NPC AI or XAI, some of the results provide indirect feedback on these systems. Soldiers who used FSC in the context of the ICCC were asked to rank the "relative importance of FSC fidelity" in a number of different areas. The most important area of fidelity was "tactical blue force" or the NPC AI of the soldier's subordinate units. The second most important area of fidelity was "tactical red force" or the NPC AI of the opponent units. Overall 88% of the soldiers indicated that the fidelity of FSC was "excellent" or "adequate." Taken together these results suggest that the soldiers were watching the NPC AI closely and most found it to be acceptable. Overall 60% of the soldiers felt FSC has high training value while 18% of the soldiers felt it has little or no training value. Unfortunately, the XAI was not explicitly mentioned in the results of the evaluation. As discussed in the following section, a separate evaluation is planned that will focus on the effectiveness of the XAI in answering user's questions and determining how the system should be extended to better support FSC's training objectives.

Currently Full Spectrum Command is used by approximately 20% of the instructors who teach the ICCC. This number is likely to increase as more computers with the appropriate graphics accelerator cards are obtained and more instructors are trained to use the training aid. An expanded and improved version of Full Spectrum Command, called FSC 1.5, is currently under development. This second round of development is being funded by the Singapore Armed Forces in collaboration with the U.S. Army.

Future Work

The current NPC AI and XAI systems have a number of limitations that provide fertile ground for future work. Probably the greatest limitation with the NPC AI is the difficulty in modifying the map and internal behaviors. This has become particularly apparent during the development of FSC 1.5 as we attempt to modify the NPC AI to follow the doctrine of the Singapore Armed Forces.

A more modular NPC AI system that includes an automated spatial reasoning system that can automatically generate a navigation graph would make adding new maps, tactics and doctrine much easier. To this end we are in the process of developing a standard software interface between the simulation system and key components. This standard interface will modularize these components and allow different AI systems to be swapped in and out. This effort will start with the Command AI, the map/navigation graph manager, and adding a new strategic AI module described below. Ideally, this will result in a version of Full Spectrum Command that serves as a research testbed in which different ideas and approaches can be compared and evaluated with respect to fidelity, flexibility, processing demands, and ease of development.

For the XAI, an immediate next step is to evaluate the current explanation capability to determine if it meets the needs of the users. This evaluation should include both testing of the FSC XAI component and an exploration of common types of explanations that occur in after-action reviews of live and simulated training exercises. The fundamental limitation of the XAI is the depth of the explanations it can provide. The current XAI system is limited to explaining how the pre-existing task knowledge was applied. Unfortunately, this task knowledge defines how each task should be carried out, but doesn't include any deeper knowledge about why each task should be approached in the specified way. For example, the clear-building task indicates that the secure-building-perimeter sub-task should be completed before starting the assault-building sub-task but no information on why this constraint is necessary. The NPC AI performs these steps by rote without any knowledge of why it is important to secure the perimeter. As a result the XAI system can't answer questions such as "Why do I need to secure the perimeter before I assault the building?"

Previous research into explanation systems have identified this problem and suggested a number of solutions. The obvious solution, encoding additional knowledge into each task providing the underlying motivation, is unwieldy. It requires an additional, and not insignificant, step each time a task is developed or modified. A more promising solution is to encode the domain's motivating first principles and use an automated planning approach to develop behaviors in response to specific task or mission goals. Because the system is compiling behaviors from first principles, the deeper "why" questions can be answered by retracing the behavior generation process. Although this approach is more complex it has a number of advantages. Behavior knowledge and explanations are generated by the same process from the same source and are guaranteed to match. Once a domain's first principles are encoded they are likely to apply to a wide range of goals and behaviors. It is easier to validate the behaviors once the first principles are defined. Finally, the automated planner might find a range of behaviors that can achieve the same task goals increasing the variability of NPC behavior.

Acknowledgments

This paper was developed with funds of the Department of the Army under contract number DAAD 19-99-D-0046. Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the Department of the Army.

References

Johnson, W. L., 1994. Agents that learn to explain themselves. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pgs. 1257—1263.

Katz, S. and Lesgold, A. 1994. Implementing post-problem reflection within coached practice environments. In *Proceedings of the East-West International Conference on Computer Technologies in Education*. Crimea, Ukraine. Pgs. 125-130.

Laird, J. E., Jones, R. and Nielsen, P. E. 1994. Coordinated behavior of computer generated forces in TacAir-Soar. In *Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation*. Orlando, FL.

Laird, J. E. 2000 It Knows What You're Going to Do: Adding Anticipation to a Quakebot. AAAI 2000 Spring Symposium Series: Artificial Intelligence and Interactive Entertainment, AAAI Technical Report SS00 -02.

Laird, J. E. and van Lent, M. 2001. Human-Level AI's Killer Application: Interactive Computer Games. *AI Magazine*, Volume 22 (2), pgs. 15-25.

McGuinness, D. L. and Borgida, A. 1995. Explaining Subsumption in Description Logics. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, August 1995.

Morrison, J. E. and Meliza, L. L. 1999. Foundations of the After Action Review Process. U.S. Army Research Institute for the Behavioral and Social Sciences Special Report #42.

Shortliffe, E. H. 1976. *Computer-based Medical Consultations: MYCIN*. Elsevier, New York.

Swartout, W. R., Paris, C. L., and Moore, J. D. 1994. *Design For Explainable Expert Systems*. IEEE Expert, Volume 6, Number 3, pages 58-64.

Young, R. M., Riedl, M., Branly, M., Jhala, A., Martin, R. J. and Saretto, C. J. 2004. An architecture for integrating plan-based behavior generation with interactive game environments, *The Journal of Game Development*, Volume 1 (1).