Towards Building an Interactive, Scenario-based Training Simulator

Brian Magerko
John Laird
University of Michigan
1101 Beal Ave.
Ann Arbor, MI 48109

734-936-3874, 734-647-1761

magerko@umich.edu, laird@umich.edu

Keywords:
training, automated direction, synthetic characters, user modeling, interactive drama

ABSTRACT: *This paper presents problems with non-interactive training simulators, and presents the design for an interactive system that will address these problems. The focus of this system is an omniscient Director agent that monitors the actions of the Trainee and other characters involved in a training scenario. Once the Director notices when the training goals of the scenario are not being met, it then dynamically modifies the state of the world to make reaching the training goals more likely. The pedagogical effectiveness of the training simulation should be increased through this observation and action combination.*

1. Introduction

Realism is an important value to uphold when creating military training simulations. The environment should be immersive and three-dimensional, complete with realistic opponents and teammates. To further provide a realistic simulation in this environment, the Trainee should have as much of a free range of action as possible; his actions should not be constrained by what the designers envision the Trainee would possibly do. We define the Trainee's ability to act as he desires in the world as *user flexibility*. If the only goal of building a training simulation system were to present an absolutely realistic world, then we would worry about maximizing this user flexibility and nothing more. However, given that a training simulation is a pedagogical tool, there will be specific training goals for a given scenario. In order to achieve those goals, the Trainer needs the representational flexibility to formalize the goals for the training scenario, to script out in some abstract manner what should happen, and have a way to dynamically modify the scenario to ensure the training goals are achieved. We define the ability of the Trainer to specify the exact scenario he wishes as *writer flexibility*. This paper proposes a system that attempts to find a balance between user and writer flexibility.

Finding the balance between user flexibility and writer flexibility lies in the monitoring of the Trainee's actions and the intelligent modification of the environment. As the User executes actions in the world, those actions may begin to lead him down a path that diverges with the training goals that the Trainer has specified. Monitoring the Trainee and recognizing this possible divergence from the training scenario is a key element of improving the efficiency and efficacy of immersive training. Once a divergence is recognized, the Trainee must be encouraged in some unobtrusive, realistic manner to follow a path more consistent with the pre-defined scenario. How the world should change, as well as which global actions to execute to create this change, is the other side to our problem. Once something has gone awry, how do we reconcile the current state of the world with the desired one in both a realistic and non-obtrusive fashion?

Inspiration for building the system we are proposing in this paper has come mainly from work done on interactive drama systems [1], [2], [3], and [4]. Laurel's definition has come to aptly characterize this yet-to-be-realized art form [5]:

> An "interactive drama," then, is a first-person experience within a fantasy world, in which the User may create, enact, and observe a character whose choices and actions affect the course of events just as

> they might in a play. The structure of the system proposed in the study utilizes a playwriting expert system that enables first-person participation of the User in the development of the story or plot, and orchestrates system- controlled events and characters so as to move the action forward in a dramatically interesting way.

This definition brings up several important points, the most important being that in an interactive drama, *the User is the character*, much akin to the experience we desire to offer a Trainee. Offering a virtual world that gives the Trainee a free range of action, while at the same time ensuring that certain pedagogical goals are met, is a similar problem to creating an interactive drama, though most certainly a more constrained one. The main difference between the two types of systems is the overall goal of each: interactive drama's main goal is for entertainment, the other is for training.

2. The Problem

When working alongside other human trainees in a virtual environment, or alongside synthetic teammates, there are possible problems or situations that may arise that would harm the pedagogical success of the training scenario. The root of all of these problems can be traced to actions taking place in the world that harm the possibility of the training goals being achieved and ways of correcting them. We have classified these problems into three different categories: human teammate error, trainee error, and multi-agent direction.

2.1 Human Teammate Error

In training scenarios, a trainee is rarely alone. Rather, he works with a group of teammates to accomplish the mission goals given to them by their Trainer. In the course of going through a mission with other trainees, fulfilling the training goals for this mission could be hindered by the mistakes made by the human soldiers on their side.

For example, consider an example where Trainee Tom is a soldier in a platoon of troops. His platoon has been given the mission goal of taking over and securing a church on the edge of a jungle. The Trainer specifies the training goal for this scenario is to put Tom in a situation where he must take command of the platoon and that the other trainees must respond to this change in command. Now, the most logical way of this to happen is for the acting commander (which may be a synthetic character) of the platoon to be killed in battle, thus forcing the duties of command onto Tom. Furthermore, to encourage this possibility, the initial state of the world includes a sniper in the bell tower of the church; ideally, if the platoon is sneaking up to or storming the building, the sniper would be able to take the commander out.

Given that we desire to offer a rich, realistic environment, it is possible that the commander will not die because of some actions taken by Tom's teammate. For example, 1) the team member in charge of navigation could lead the platoon to the wrong location, thus avoiding the building altogether, 2) one of the platoon members could spot the sniper and shoot him first, 3) the commander could establish a base on the perimeter while other members of the platoon forged ahead, 4) one of the characters could dive and take the bullet meant for the commander, 5) or even some other character behavior that the Trainer couldn't foresee that would render the sniper useless. With the precondition that the first-in-command has to die, thus rendering Tom the commander of the platoon, the world must somehow change to insure that, despite the initial setup's failings, the training goal will be met.

We propose the use of a *Director* agent to have a global, real-time view of the environment. The first responsibility of the Director is to recognize that the scenario is diverging from the training goals specified by the Trainer. Once the agent recognizes that there is a problem, it must then alter the world in order to correct for this problem. As in our example above, the Director must be able to handle any of the situations we have listed and more.

It also should be noted that some of the behaviors above are actual mistakes made by the teammates, while others are proper actions according to military doctrine, yet they still hinder Tom reaching his training goals. Making a distinction between the two is important; improper behavior by any human participant should be negatively trained. However, it is difficult to do so without sacrificing the experiences of the other human trainees participating. For instance, if the navigator trainee leads the platoon astray, he should not be rewarded for this and have the church appear on whichever path he chooses. This a design issue that arises when we consider using this system for training more than one trainee at a time; how do we constrain the system so we can allow the training of a group of trainees at once? How do we avoid certain states of the world while still offering negative reinforcement for mistakes made by trainees? While the answer is not obvious, we feel that this is an issue best answered after we better understand how to build systems for training a single Trainee.

2.2 Trainee Error

Just as the Trainee's human teammates may, the Trainee himself can make mistakes that can have an effect on the pedagogical effectiveness of the training scenario.

Typically an error in behavior means a deviation from the accepted behavior of a trainee in the given situation. Knowing when the trainee is following incorrect procedure, even though it may not directly affect the mission outcome, is key to teaching the Trainee proper behavior in a given training scenario. If the director knows that the Trainee has made a mistake, it can alter the state of the world to negatively reinforce that behavior. For instance, if the Trainee is assigned to check a room in a hallway that the platoon is sweeping through, he may not check as thoroughly as he should; therefore, the Director agent may notice this mistake and place an enemy in one of the spots the Trainee overlooked. We see here a chance for the Trainee to learn by example, a viable learning technique for students, to avoid making costly mistakes as well as keeping them in line with the scenario [6]. Through this type of response, the Director can be used directly as a pedagogical tool for trainees in an interactive environment.

2.3 Multi-Agent Direction

Our third classification refers to the flip-side of these problems: the necessity to direct groups of characters to work together to encourage an action to happen in the world that is difficult for a single NPC (Non-Player Character) to achieve. In our framework, the training knowledge is encapsulated in the Director; we do not need to distribute the training knowledge to all of the NPC's involved in the scenario. Because of this design choice, it is logical to use the Director as the coordinating agent when a group must work together to solve a training goal. Let's return to our sniper example. If the sniper is unable to get a clear shot at the platoon's commander, he made need cooperation from other NPC's to help him achieve this important goal. Those NPC's could create a diversion that attracts the platoon into a better position for the sniper, or even possibly attack the platoon with a goal of killing the commander before retreating.

The need for a Director agent here is most apparent when noticing that there is a distinct difference between characters acting to support their inherent military goals compared to acting in supporting the training goals. In a normal situation, the synthetic characters, enemy and teammate alike, would have standard military doctrine to fall back on. However, when the scenario begins moving away from the desired training goals, the characters need to be given some goals or behavior that will help bring the scenario back to the desired condition. The Director agent, which has the pedagogical knowledge, can alter a group of characters to move away from specific doctrine and get the scenario back on track. Once this group of characters has been given their new goals, it is up to them to carry out these goals correctly as a group. Executing these goals introduces a new problem of flexible coordination and communication in the teamwork involved. We plan on investigating an approach developed by Tambe based on the *joint intentions* theory, which is called STEAM [7]. STEAM incorporates team synchronization, monitoring and repair abilities to enable a team of agents to maintain a coherent and correct view of the teams' goals and plans. By identifying the need for this type of system, we hope to address the concern we have about what a group of agents do once they are given orders by the Director to fulfill a goal together.

3. System Design

We have designed an architecture, currently in the later stages of construction, as a first step in creating a complete interactive training simulator [8]. We have rooted our system in the Unreal Tournament 3D engine, a commercial graphics engine that provides tools for constructing a 3D, first-person view experience. This provides us with an environment to design and create a three-dimensional map, to populate it with synthetic characters, and a communications system to use between these characters. Aside from the environment, we also have several other modules that make up the entire system design, shown in Figure 1 below. There are some components that warrant further explanation, which are also described below.

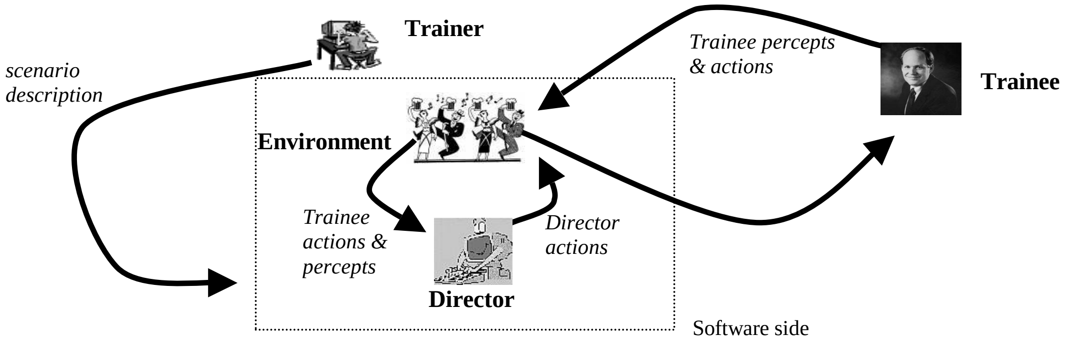

Figure 1: The train simulator architecture.

3.1 Scenario Description

Given that we desire the Trainer to be able to specify a training scenario as a separate module from the environment, we need to provide a representation for describing the scenario. We have opted towards a representation akin to a POP-style planning language, with an initial state, operators with preconditions and effects, and training goals to be fulfilled for each scene in a scenario. POP, also known as partial-order planning, can be seen as a specialized search process, attempting to find a path from an initial state of the world to the goal states using these operators.

Scenes are then linearly ordered to provide a temporal description of how each set of goals may be achieved, as shown in the Figure 2. We will soon begin experimenting with our representation in order to get a better understanding if this is a proper representation choice for this kind of problem.

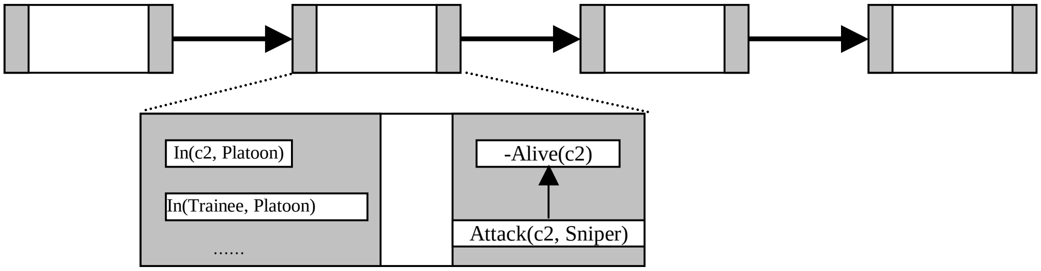

Figure 2: An example of scene ordering in a scenario description. The close-up is a view of how the scene is defined in terms of an initial state and required events. The other elements discussed, temporal and content constraints, as well as background knowledge, are not depicted. They are inputted into the Director without any processing (such as inputting them into a planning system to construct a plan).

3.1.2 Initial World State

The initial state of the world for any training scenario will include: the external (e.g. what items they have, where they are, etc.) and internal (e.g. explicit knowledge, goals, etc.) states of the synthetic characters, the initial state of the environment, the User's beginning state, and the pre-scripted scenario for the given training exercise. While it would be to our advantage to build a system that we could specify all of this in the scenario description, at this point in time we are focusing solely on the scenario content being available for description, leaving the rest to other components (such as placing knowledge directly into the agents). That having been said, we still need to specify exactly what the initial state of the world is for this example, regardless of where the knowledge originates from in the system. Figure 2 illustrates the layout of the building and the initial positioning of the characters involved, including the Trainee.

3.1.3 Operators

A typical operator representation in planning involves the name of the action being performed, the preconditions for this action to successfully begin, and the effects that this action has on the world. For example, the operator GoTo(x) may have a precondition of being at some location y, where y is not equal to x. The effects of that operator would be not being at y and being at x. Using these operators to construct a plan representation for each scene, we still need a manner of sharing information across scenes. In our system, the operators contain variables that may be locally or globally defined. The global definition of a variable across scenes allows us to ensure that the content will be consistent. For example, if the Trainer constructs a scenario that requires the platoon member that saves the Trainee's life earlier winds up being injured later on under heavy fire, then we want to be sure that whichever variables should represent that soldier are all bound to the same character. Therefore, we see the need for allowing global variables, in both the required events as well as in all other components, to be shared between scenes.

3.1.4 Training Goals

The purpose of using an interactive training simulator is to provide a means of answering the question "How do we get the Trainee to experience the training goals?" Take our example scenario from above. There is one distinct training goal that the Trainer wishes to have fulfilled, namely *to have the Trainee engaged in combat with a hidden enemy*. The environment has been set up to make this event a distinct possibility (as opposed to offering the Trainee a completely empty building);

however, it is possible that the Trainee can take unforeseen actions (e.g. they are unorthodox, against protocol, more intelligent than the Trainer had foreseen, etc.) which would possibly lead the Trainee to avoid fulfilling the goals of the exercise.

We may also constrain the partial ordering of these goals with *temporal constraints*. Temporal constraints on goals indicate that certain goals must be fulfilled before others, just as a POP-style planning language may allow. The *content constraints* limit the binding of the variables that are used in the plan. For the moment, we are not allowing disjunctive constraints for simplicity in the representation. However, we will consider them further down the road as we explore the problem further.

3.2 The Director

The Director is the component that takes the plot representation as input and then observes the world, making sure that the training goals will be fulfilled. It's purpose is to assure that the Trainee's experience is not hindered by mistakes made in the world, either by him or other characters involved. If he does veer from the path intended by the Trainer, it is up to the Director to dynamically alter the world so that this new course will lead to the desire goals. As noted in our sniper example above, the Director has two main problems when monitoring a scenario: recognizing when there is a problem and then acting on it.

3.2.1 Recognition and Action

Thus far, our efforts in recognition have focused on maintaining both a physiological and mental model of the Trainee. We intend to use this model to give the Director a hypothesis about what the Trainee is trying to accomplish, as well as a more detailed picture of their actual physical state. It should be pointed out that we are still working on building a system to experiment with this design, that this paper is a description of the problem we are tackling and the blueprints we have laid out to tackle it with.

Recognition can be broken into two separate questions: *Is the Trainee in a disagreeable state?* and *is the Trainee on a course of action that will put him in a disagreeable state?* We define a disagreeable state to be a state of the world that it is either very improbable or impossible for the training goals to be fulfilled without intervention by the Director. Understanding if the Trainee is on his way to being in one is a very different problem compared to recognizing if he already is. To approach the former problem, we intend to apply a user model as a predictive mechanism for Trainee behavior. The latter is a problem addressed in planning research.

3.2.2 User Model

Our approach for behavior prediction takes inspiration from a variety of sources, mostly from role-playing adventure games (e.g. Dungeons and Dragons) and user-modeling research done in Algebra tutors [9]. If we presume to have a good, cognitively-motivated hypothesis of the User's goals, his motivations in a given scene, then we can make a more informed decision about what should happen next in the scenario.

A similar approach to behavior prediction can be seen in Laird's work on anticipation in the QuakeBot project [10]. An agent running around in a battleground-type computer game environment would attempt to predict it's opponent's actions by running an internal simulation of what it thinks the enemy would do given its current state. The first prototype for our current project has been designed without experimental User data; however, we plan on creating a more rigorous model in the future. This gives us an opportunity to have a realistic and plausible model for what a User is actually attempting to do.

3.2.3 Planning

As we have mentioned above, our scenario description is based in a POP-style planning language. Each scene is a partial-order plan, and the linear ordering of these scenes represents the complete scenario. This allows us to borrow from current planning techniques to aid us in recognizing possible errors in the behavior of the trainee and of other characters. An example of relevant work is research done by Onder and Pollack [11]. They presented a unified algorithm in probabilistic planning to identify failure points in plans, as well to refine plans to prevent failures. While we have yet to experiment with this or other possible failure-recognition approaches, it is important to recognize this problem and to search for viable solutions.

3.2.2 Action

Once the Director desires a state S, it must then execute some action to encourage the User to enter that state. We have yet to design an intelligent method of generating these actions and therefore have put this task into the hands of the Trainer as part of the scenario description. As mentioned before, our representation language has obvious parallels to a planning language. We specify an initial state, operators with preconditions and effects (background knowledge), and goals (required events). We specify a partial ordering of the operators and required events with our temporal constraints. Given a particular scene, one can imagine using a POP planning system such as Graph-plan or SAT-plan to create a feasible partial-order plan (or a set of plans) with our constructs. A resulting plan would represent the space of possibilities for what the Trainee could do in the world and how the world would change. Given this representation, we can now create a relationship between Trainee behavior and direction by annotating our plan with directions that will encourage the various operators to be executed.

3.3 Variability

The ideal interactive training simulator would have a richly-defined world, with an infinite (or at least sizably large) amount of actions available to the Trainee, just as in the real world. No matter what actions the Trainee executed, they would wind up reaching situations that would fulfill the training goals. Each time that the Trainee started the system anew, different interactions would lead to varied experiences, though every time encountering the types of situations that the Trainer desires.

Putting aside possible variance due to Director actions, our design philosophy focuses on separating interactivity into two types of variability: temporal variability and content variability. We define *temporal variability* as allowing time to be the key variable for the flexibility in an interactive experience. Who? What? Where? and Why? are all static aspects of the story; the only variable is when scenes actually occur. Pete Weyhrauch's PhD thesis [12] at Carnegie Mellon is a prime example of introducing temporal variability into a first-person dramatic experience to produce an interactive system. We define *content variability* as allowing scenes to be defined in an abstract fashion, giving flexibility into what specifically happens in that scene. The allegiance of a village….which of the Trainee's teammates breaks under pressure….who aids the Trainee at the end….all of these things are examples of abstract concepts in a scenario that deal with content.

Note that we have decided to address the problem of content variation before concerning ourselves with temporal variability. Scene ordering is specified by the Trainer and is static in this representation. In order to make progress in either of these types of variability, we will focus solely on one of them and leave the other as a control variable in our experimentation.

4. Discussion

In this paper, we have presented the motivations and design for an interactive training simulator. Building such a system is a sizeable task, though one that we have already made progress on in building our environment and initial agent designs. What are the major issues we will face in construction and experimentation using this design? We will be focusing our main efforts on building the Director agent and experimenting with different approaches to the manner in which is recognizes problems and then addresses them. Given the large size of such a system, we will be narrowing our current research to addressing specific questions and problems that will benefit not only the construction of this system, but in the design of future interactive systems as well.

Approaching user modeling will be the most daunting task in the construction of the Director. Experimenting with several different approaches, such as a rule-based method as mentioned earlier or perhaps a Bayesian approach [13], will be necessary to validate our choice as well as to examine the effectiveness of using such a model. It is our hypothesis that trainee prediction will improve the effectiveness of the Director and thus the experience of the trainee. However, we cannot be certain of this without first building several models, integrating them into the Director's decision processes, and then conducting experiments to understand what the best approach is out of our choices and why.

Along with the cognitive modeling of the Trainee, we are also considering modeling the Trainee's physiological state. By monitoring something as simple as heart rate and giving that as another input to the Director, we could possibly even further tailor the scenario to the Trainee at any given time. For instance, if we wanted a training goal to be that the trainee must experience something surprising when they are calm, the Director could monitor when the trainee's heart rate, find when it has reached an acceptable low level, and then create a platoon of enemies hidden nearby for a surprise attack. While we have yet to delve into this possibility further, it may be a very useful cousin to a User model in providing the Director with a rich hypothesis of the Trainee's current state.

It is unclear what kinds of content variability are possible, both for our system and in general. Given the nature of our representation, we can be assured that the "Who?" part of a scene may be variable for instance, as we've shown in our examples. What about deeper features of the plot though? Is this representation sufficient enough to model variability in complex relationships between characters? in complicated sequences of actions? Or are these aspects of the scenario perhaps too complicated to allow much flexibility? We believe that once we have built a functional, working system, we will be able to vigorously experiment with both our representation and our approach to direction without compromising our architectural design.

What types of Director actions should be possible? Up until this point, we have mainly focused on Director actions changing some aspect of the states of a given synthetic character. Our main focus thus far in our design has been altering the synthetic characters' goal states and knowledge base. By allowing the Director to alter the internal states of the synthetic characters, we give the Director an indirect yet powerful tool for altering the state of the world. However, one can also imagine a situation where an auditory or visual cue in the environment may yield the desired effect of encouraging or discouraging the User to enter a particular state.

What types of experimentation should we conduct? Simply put, how do we in fact know when we're done? What makes an interactive simulator more successful than other approaches to training simulation? than other iterations of the same approach? To fulfill our artistic goals, we can only assume that user enjoyment has to play a key part in assessment of the overall success of the project. The variables we may consider in experimenting are the many possible technical alterations we can make to the system's design: how much autonomy is given to the characters, how the scenario description language is specified, what kind of direction the Director could enact, etc. After constructing a fully-functional prototype, we will be approaching these issues and evaluating how they affect User enjoyment, as well as system load, communication bandwidth between agents, and other technical concerns. Designing a methodology for user testing will be an important step in showing that a system that fits our definition of an interactive drama is actually a "successful" example of one.

How do directable, intelligent characters fit into this framework? Given that there are intelligent agents running around the environment, each with their own beliefs, goals, and knowledge, what sort of information needs to be communicated between the agents and the Director? How does this affect the kinds of direction the Director is capable of? How do they believably switch to a new goal when given one by the Director? These questions will be approached as both our research on direction as well as Assanie's work on directable agents progresses [14].

5. References

[1] Magerko, B. A Proposal for an Interactive Drama Architecture. Artificial Intelligence and Interactive Entertainment: Papers from the 2002 AAAI Spring Symposium. Menlo Park, Calif.: AAAI Press, 2002.

[2] Mateas, M. and Stern, A, Towards Integrating Plot and Character for Interactive Drama. In Working Notes of the Social Intelligent Agents: The Human in the Loop Symposium. AAAI Fall Symposium Series. Menlo Park, CA: AAAI Press, 2000.

[3] Sgorous, N. M. "Dynamic generation, management, and resolution of interactive plots." Artificial Intelligence 107, pp. 29-62, 1997.

[4] Weyhrauch, P. "Guiding Interactive Drama," PhD thesis, Computer Department, Carnegie Mellon University, CMU-CS-97-109, Jan. 1997.

[5] Laurel, B. Toward the Design of a Computer-Based Interactive Fantasy System. PhD diss., Drama Department, Ohio State University, 1986.

[6] Simon, H.A. and Zhu, X. Learning Mathematics from Examples by Doing. Cognition and Instruction, 4:137-66, 1987.

[7] Tambe, M. Agent architectures for flexible, practical teamwork. In Proceedings of the National Conference on Artificial Intelligence (AAAI). August, 1997.

[8] Laird, J., As-Sanie, M., Bachelor, B., Benninghoff, N., Enam, S., Stokes, D., Jones, B., Kerfoot, A., Lauver, C., Magerko, B., Sheiman, J, Stokes, D., Wallace, S. A Test Bed for Developing Intelligent Synthetic Characters. Artificial Intelligence and Interactive Entertainment: Papers from the 2002 AAAI Spring Symposium. Menlo Park, Calif.: AAAI Press, 2002.

[9] Anderson, J.R., Boyle, F. and Resier, B. J. "Intelligent Tutoring Systems," *Science*, New Series, Vol. 228, Issue 4698, Apr. 26, 456-462, 1985.

[10] Laird, J. E., "It Knows What You're Going to Do: Adding Anticipation in a Quakebot," AAAI 2000 Spring Symposium Series: Artificial Intelligence and Interactive Entertainment, March 2000.

[11] N. Onder and M. E. Pollack, "Conditional, Probabilistic Planning: A Unifying Algorithm and Effective Search Control Mechanisms,"Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99) , July, 1999.

[12] Weyhrauch, P. "Guiding Interactive Drama," PhD thesis, Computer Department, Carnegie Mellon University, CMU-CS-97-109, Jan. 1997.

[13] Albrecht, D. W., Zukerman, I., Nicholson, A. E. and Bud, A. "Towards a Bayesian Model for Keyhole Plan Recognition in Large Domains," User-Modeling: Proceedings of the Sixth International Conference, Vienna, New York: Springer Wien New York, 1997.

[14] Assanie, M. Directable Synthetic Characters. Artificial Intelligence and Interactive Entertainment: Papers from the 2002 AAAI Spring Symposium. Menlo Park, Calif.: AAAI Press, 2002.

Author Biographies

BRIAN MAGERKO is a Ph.D. candidate at the University of Michigan, focusing in Intelligent Systems. He received his B.S. from Carnegie Mellon in 1999. His research interests include interactive drama, user modeling, and artificial intelligence and interactive entertainment.

JOHN LAIRD is Professor of Electrical Engineering and Computer Science at the University of Michigan. He received his B.S. from the University of Michigan in 1975 and his Ph.D. from Carnegie Mellon University in 1983. He is one of the original developers of the Soar architecture and leads its continued development and evolution. From 1992-1997, he led the development of Tac-Air-Soar, a real-time expert system that flew all of the U.S. fixed-wing air missions in STOW-97.