Journal of Computing in Higher Education
Spring 2001, Vol. 13(1), 52-69.

A Client-Server Computational Tool for Integrated Artificial Intelligence Curriculum

Lawrence B. Holder and Diane J. Cook
Department of Computer Science and Engineering
University of Texas at Arlington

ABSTRACT

THE GOAL OF THIS PROJECT is to increase students' interest in Artificial Intelligence (AI), as well as to promote learning of the topics that comprise the subject. We describe the development of a Web-based multimedia delivery method to accomplish this goal. The highlight of the course material is an integrated simulation environment that allows students to develop and test AI algorithms in a dynamic, uncertain, visual environment.

In this paper we describe the multimedia approach to teaching AI classes that we have developed and introduce the associated the simulation tool. We present a series of homework assignments and projects that make use of the tool and evaluate the effect of the simulation environment on students' interest level and confidence in the class material. (*Keywords: agent design, Java application, multimedia delivery, simulation environment, collaborative environment*)

INTRODUCTION

ARTIFICIAL INTELLIGENCE is a field of study that draws from many disciplines including Computer Science, Mathematics and Information Theory, Cognitive Psychology, and Philosophy. Teaching the subject to students from just one of these disciplines therefore presents a challenge. Ensuring that students can apply the ideas presented in class poses an additional challenge.

In the Department of Computer Science at the University of Texas at Arlington, we have developed a multimedia delivery method for the foundational classes in AI. Our class organization centers on the demonstration of AI techniques in an integrated visual simulation environment. In this paper, we will present the integrated simulation tool, describe the development of projects and experiments that use the tool, and evaluate the effectiveness of the multimedia approach toward increasing students' interest and understanding of the course materials.

BACKGROUND

THE FOUNDATIONAL CLASSES in AI at the University of Texas at Arlington are divided into Artificial Intelligence 1 and Artificial Intelligence 2. These classes are offered each year to upper-class undergraduate students as well as graduate students. Students who are interested in AI must take these classes before enrolling in advanced AI classes. Students intending to pursue related Computer Science disciplines also take these classes as breadth requirements in order to learn about the AI techniques that pervade Computer Science disciplines and applications.

Material in the AI classes is introduced by describing the idea of a rational agent, and then presenting techniques for creating such these agents (Russell, 1995). A rational agent is one that uses intelligence to interact with the world in order to maximize expected performance.

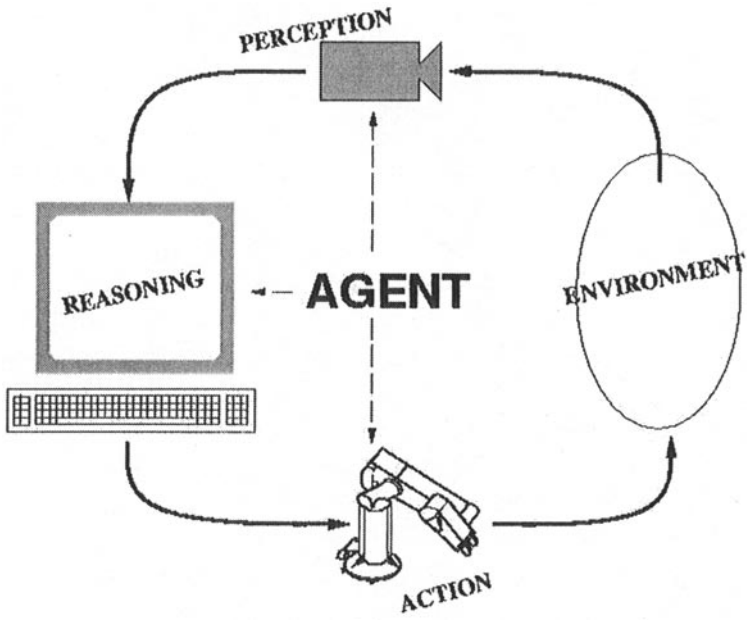

Figure 1. Rational agent design

As shown in Figure 1, rational agents operate by receiving a percept, or set of sensory information that the agent can detect about the environment. A percept represents the explicit values of external sensors, and therefore does not necessarily provide all of the information an agent would like in order to make decisions. A rational agent must employ reasoning techniques to understand the environment and formulate an appropriate action, and change the environment by executing a selected action.

Students in the AI classes learn about the goals of rational agents, and learn current techniques for creating these agents. In particular, students need to learn methods of internally representing information about the external world. They also learn methods for problem solving by searching through large spaces containing possible environment states, and for generating plans (or sequences of actions) to achieve desired goals. Natural language processing, uncertainty reasoning, computer vision, and speech processing are all essential com-

ponents of rational agent design, and machine learning techniques are introduced so that the agent can improve performance on any task with experience.

Many studies have been conducted that indicate the value of computer-based multimedia and simulation tools to learning (Clark, 1997; Graham, 1998; Reinhardt, 1995). This is perhaps no more important than in the Computer Science curricula, where students perform a majority of their work in front of a computer screen. In response, we are developing an environment in which students can electronically access all class material. We introduce a tool in this environment, which simulates and tests ideas presented in lectures in a dynamic, competitive, and complex simulated environment.

Figure 2 shows the home page that students use to read homework problems and solutions, view online lecture notes and demonstrations, and access the rational agent simulator. The focal point of the class material is an object-oriented Wumpus World Simulator, which can be retrieved from http://www-cse.uta.edu/~holder/wumpus. The Java simulator can be easily modified to produce different types of worlds. Using this simulator, students acquire hands-on experience with AI techniques and discover their usefulness for intelligent tasks. Because one common environment is used for a majority of student homework assignments and projects, students can compare the performance of an intelligent agent with and without each of the techniques introduced in class. Furthermore, because the environment is object-oriented in design, each student project actually adds to the capabilities of the tool and adds to the features that can be learned by students in other classes.

Other simulation environments exist which also support AI research and teaching. For example, the TileWorld system (Pollack, 1990) supports simulation of agent design in a dynamic two-dimensional grid world in which objects can appear and disappear at specified rates. The Truckworld simulator developed at the University of Washington (Hanks, 1993) uses a graph structure to represent connections between locations along which trucks can execute delivery plans and respond to unforeseen situations. The Michigan Intelligent

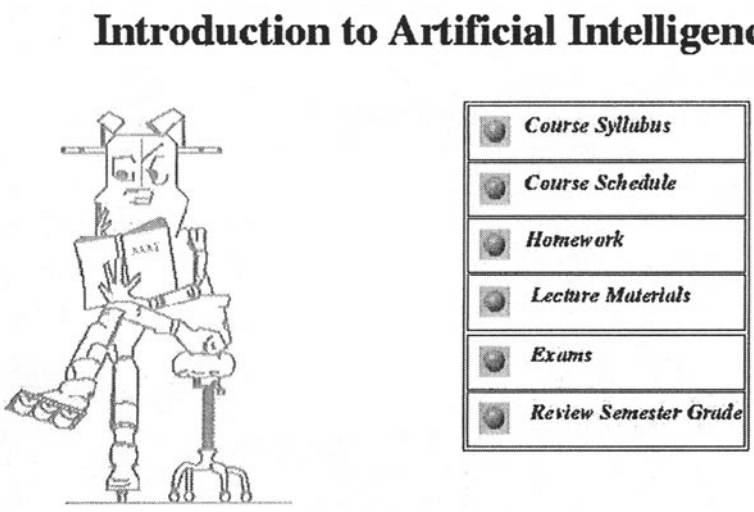

Figure 2. AI class home page

Coordination Experiment (MICE) testbed (Montgomery, 1990) is yet another simulator which focuses on multiagent interaction in a timed environment.

Our tool presents a number of new features beyond these other environments. Our system uses a Java-based server design, which facilitates visualization of the environment, incorporation of additional AI classes written in various programming languages, and multiplatform use. The teaching environment we are developing provides not only the simulated agent environment but a set of AI algorithms that operate within the environment, including tools for knowledge representation, search, planning, learning, vision, uncertainty reasoning, multiagent coordination, and natural language processing.

THE WUMPUS WORLD GAME

THE BASIS FOR THE GAME is an agent who explores an NxN grid world while avoiding a creature known as the Wumpus. Other elements of the world consist of obstacles, bottomless pits (which do not affect the Wumpus), and bars of gold. The agent starts in the lower left cell of the world with a single arrow in his arsenal. The objective of the game is to collect the gold bars, return to the initial grid location and exit the cave. The Wumpus is a fierce creature, capable of killing the agent instantly. If the agent steps into a square containing a Wumpus, he dies immediately. If an agent walks into a pit, he dies instantly. An example Wumpus World game is shown in Figure 3. This particular implementation of the Wumpus World follows the description given by Russell and Norvig (1990).

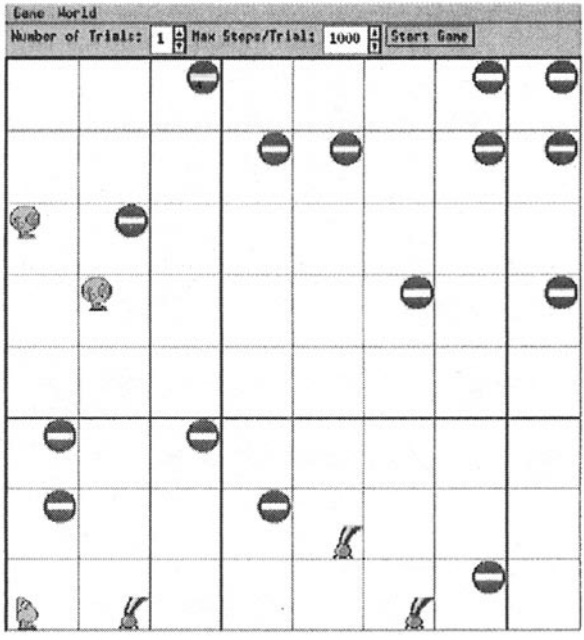

Figure 3. Sample Wumpus World game

Figure 4. Symbols used by the Wumpus World simulator

At the start of each turn, the agent can perceive his surroundings. He has the potential of sensing any of five basic percepts and two extended percepts. A *stench* percept indicates that a Wumpus, which emits a foul odor, is in one or more of the four squares adjacent to the player's location. A *breeze* indicates that a pit is in one or more of the four adjacent squares, and a *glitter* indicates that a gold bar is at the agent's current location. A *bump* is sensed whenever the agent attempts to move into a wall or an obstacle. If an arrow hits a Wumpus, the player will sense a woeful *scream*, indicating the Wumpus' demise.

The player may perform any of the following actions: turn left, turn right, go forward, grab some gold, shoot an arrow straight ahead, and climb out of the cave. The player can only climb out of the cave from the start location. The player receives a score of -1 for each executed move. Gold bars are worth 1000 points, but only if the player successfully carries the gold out of the cave. The agent loses 1000 points upon dying, and any gold that is carried at the time is lost.

Since there may be more than one agent in the world, agents are given one minute to respond with an action. If the agent does not come back with an action, a NO_OP (no operation) is assumed, the

agent loses 1 point and is sent a new percept. Actions are collected from the agents, and the action of the agent with the highest priority is executed first. The priorities of agents are established by the order in which the agents establish a connection with the simulator server.

One extended percept is a Natural Language Hint, namely, an English sentence that gives some kind of information about the Wumpus World. Statements might indicate that a particular percept would be sensed at a certain grid location. On the other hand, hints might be more informative, indicating the exact position of some object or some other player in the grid. Grid locations mentioned in the hint can either be absolute references, or relative to the player's current location and orientation.

The second extended percept is a color image, showing the contents of the cell one step away from the agent along his or her current orientation. The overall image is a square image in the ASCII PPM format and provides visual representation of obstacles, Wumpi, agents, pits, and gold bars for a given cell location in the Wumpus World.

The simulation may be set up for multiple trials. Each trial is over when all the agents in the grid are either dead or victorious, or have been in the game for more than 1,000 steps, where a new step starts each time the agents' percepts are sent to them.

SIMULATOR DESIGN

OUR WUMPUS WORLD SIMULATOR is designed to provide a simple interface for intelligent agent design. The simulator supports multiple independent agents in the Wumpus World, each with its own separate logic and possibly separate hardware platform. The simulation employs a client-server architecture so as to separate completely the simulation code from the agent design. The simulator uses an object-oriented design and can be executed on a variety of platforms.

To support platform independence and an object-oriented design, a Java-based environment was selected, and the simulator was implemented using jdk version 1.2. This language also supports transfer of objects such as command selections and sensed percepts between the client (agent) and server (simulator) using Object Serialization.

The simulator consists of five Java Packages. The Action package provides interpretation of valid agent actions; The Client package provides the basic agent capabilities and mechanisms for communicating between the agent and the server. The Input package contains classes that are useful for the most frequent I/O operations; the Percept class contains functionality to pass percepts between the server and client; and the World package forms the heart of the simulation.

The Wumpus World simulator behaves in the following manner:

1. The server is initialized.
2. The server waits for a specified number of clients to connect to it.
3. The clients connect to the server.
4. The server assigns priority to each agent depending on time of connection.
5. The server starts the first trial.
6. The server sends out an *Initial Percept* to each client. This percept is displayed for each agent, as shown in Figure 5.
7. The server waits for the clients to respond within a certain time limit.
8. The server carries out the actions specified by each client.
9. If a client specifies no action, the server carries out a default action (SIT) for that client.
10. If the agent dies or victoriously comes out of the cave the game has ended and the server sends a *final percept* for that client. If the trials have not ended, it starts the next trial, and repeats from step 6. If the trials have ended, then it calculates the statistics for each agent. If the game has not ended, the server sends the next percept to the client and repeats from step 7.

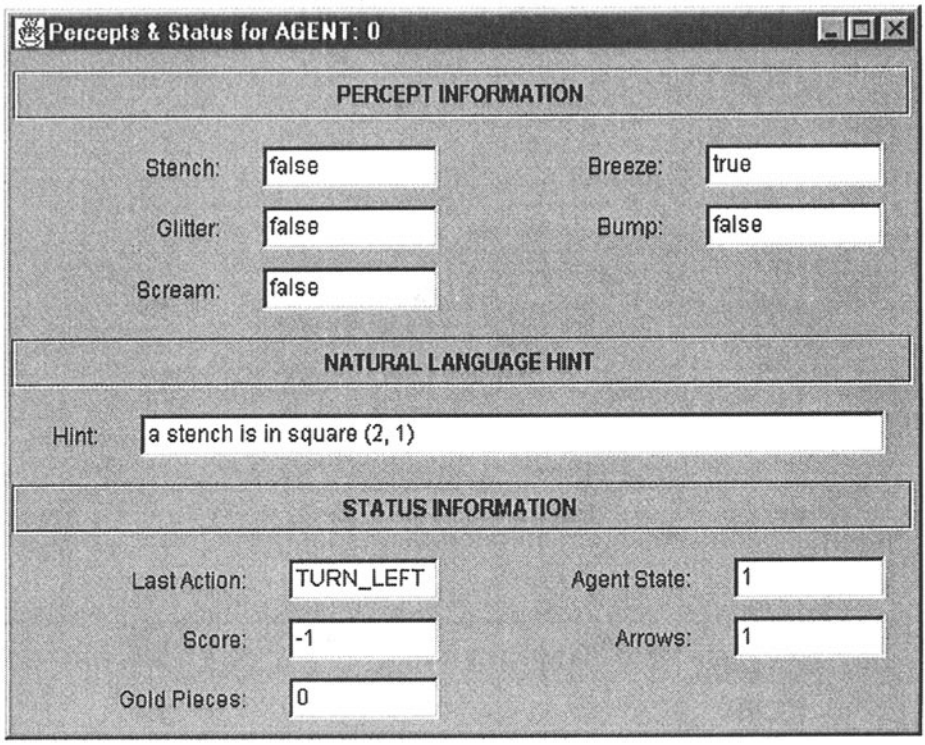

Figure 5. Percept information displayed for Agent 0

Command-line options can be used to specify the world size, the number of game trials to simulate, the number of agents that will be entering the simulator together, the time to wait for each agent command, and a port number for the simulator server. A file can be specified that contains world information. In this file, the first line defines the world size. Each subsequent line defines an object (Wumpus, gold, or obstacle), its location and details about the object.

In our simulator, we provide the Wumpus with a number of strategies. The agent will have to rely on its reasoning capabilities to outmaneuver the Wumpus. Possible Wumpus movement strategies that can be specified include sit, spin, random, walk, move-to-gold, and move-to-pit. Using the *stay* strategy, the Wumpus does not move from its initial position. In the *spin* mode the Wumpus executes a clock-

wise circling movement, and in *random* mode the Wumpus executes a random walk. In the *move-to-gold* and *move-to-pit* strategies, the Wumpus selects the nearest gold (pit) location, moves to that location and waits for the agent from that location.

ASSIGNMENTS AND PROJECTS USING WUMPUS WORLD SIMULATOR

THE WUMPUS WORLD SIMULATOR provides an excellent media for demonstrating AI techniques in the classroom. The simulator also provides a method for students to implement and test their own versions of these algorithms. We have developed a package of assignments with solutions that allow students to learn about a variety of AI techniques in the context of the Wumpus World simulator. Each project builds on the foundation of the tool and enhances the student's previous addition. As a result, much of the overhead involved in designing AI programs is eliminated, and students can see how performance of a rational agent can improve with the introduction of each new technology.

Student submissions are tested and graded based on their performance using different Wumpus world configurations. Each agent is given multiple trials for each world, with the knowledge gained about the world cumulating from one trial to the next. The agent's performance is reported as the maximum, minimum, and average performance over multiple trials for each world. Here are the basic assignments that are given.

Agent 1

Students are initially asked to create a simple reflex-driven agent that reads the sensor input and makes a decision of which action to select and execute. Because this agent is not looking beyond the current move, performance will be low. However, this provides a base for comparison with more sophisticated agents.

Agent 2

Students augment the capabilities of Agent 1 by adding state information. The state structure contains all information the agent has gathered about the world to this point in the game and can be used in selecting the next action.

Agent 3

In order to learn properties of various search and problem-solving techniques, students are then asked to implement a search engine for the Wumpus World agent. A variety of search techniques can be explored and their properties compared. Because this agent has the ability to look ahead at the result of sequences of moves, Agent 3 will in general outperform Agents 1 and 2 but will require more processing time.

Agent 4

Planning a sequence of moves for the agent can be accomplished using a search engine, but many search algorithms exhaustively consider all combinations of moves and thus require substantial processing time and computer memory. AI planning techniques are taught in this class that use principles of logic and constraint satisfaction to more efficiently generate a sequence of actions to achieve a goal. In this case, the goal is to grab the gold without being killed.

Agent 5

This agent is called DT_Agent, because unlike the other agents, this one has the capability to reason about uncertainties in the environment using techniques from decision theory. Pieces of evidence, including sensing breezes (meaning a pit is nearby), stenches (meaning a Wumpus is nearby) and glitter (meaning gold is nearby) contribute to the probability that a pit, a Wumpus, or gold is in a given location. These probabilities are combined with associated rewards (or costs) to allow the agent to make a decision that will yield the greatest *utility*. In this assignment, students use a structure called a *belief network* to represent all of the features of the environment and their associated probabilities, to update the probabilities each time evidence is collected, and to decide upon a corresponding action.

Agent 6

This agent makes use of machine learning techniques to improve decision making with experience. There are many ways that machine learning can be used to improve the performance of the agent. In this assignment, agents use a technique known as reinforcement learning to learn the value of each action from a given location in the world. Values of the actions are learned from past experiences with similar situations, and the values get increasingly accurate as the number of completed games increases.

Agent 7

Until this point, agents use only the five basic percepts (stench, breeze, glitter, bump, and scream) to decide upon the best action. Now we create an oracle in the Wumpus World. The oracle provides natural language statements that act as hints, if the agent is able to parse and understand the statement. Examples of such hints include:

"the Wumpus is in square 1 2"
"there is a pit in square 2 3"
"the gold is in square 5 2"
"the Wumpus is behind the agent"
"the agent is facing the Wumpus"
"the size of the world is 8x8"

The agent must parse the sentence (recognizing the parts of speech) and extract the information necessary to aid the agent in deciding upon an action.

Agent 8

This agent uses additional AI capabilities to acquire information. In this case, agents use their vision system to determine what objects are in the grid square they are currently facing. The agent "looks" in a specific square and receives a PPM image such as the one shown in Figure 6. To accurately reflect the inaccuracies vision systems handle due to shadows, noise in the digitization process, and sensor weaknesses, a few random symbols are placed in pixels throughout the image.

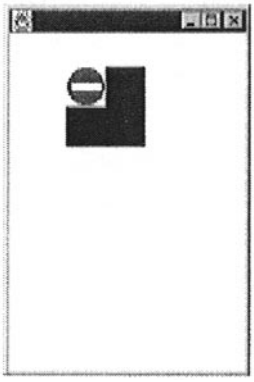

Figure 6. Sample PPM image showing a cell location containing a pit

PROJECTS

The first AI class teaches the foundation techniques and is centered on homework assignments and examinations. The advanced AI class includes a project that the students must complete by the end of the semester. Student projects use the Wumpus World simulator as a testbed. Sample projects implemented over the last two years include the following:

- The ability to make use of multiple agents playing the game as a team. Agents communicate information that is learned about the environment and divide the work between them.
- The ability to navigate in an environment with moving Wumpi. The Wumpi move according to one of a set of possible strategies. Agents use uncertainty reasoning and machine learning to recognize the movement pattern and select an action based on predicted Wumpi locations.

Because these projects employ the object-oriented Wumpus World tool, projects are incorporated into the simulator to provide more functionality each semester the classes are taught.

Table 1.
Student Evaluation of AI Curriculum

	Mean	
	SimMaterial	Non-SimMaterial
Confidence in material presented in this section of the course (0=none .. 5=high)	4.27	4.17
Effect of Wumpus World simulator on learning (0=none .. 5=large)	3.59	N/A
Effect of tool on student interest in material (0=none .. 5=large)	3.70	N/A

EVALUATION

To assess the potential benefits of teaching with a visual integrated simulation environment, we tested students' use of the tool over the past three years. Student level of interest and confidence level were tested on material using the simulator and material that did not use the simulator. Additional questions regarding the value and use of the simulator were also posed and results tabulated.

We expected that students would spend more time on simulator-related projects than on assignments, which did not require integration with the simulator. This is because knowledge of Java was required in early versions of the system to work with the simulator, and many students were learning Java during this course. However, we also predict that the confidence level with a particular subject would increase if hands-on experience using the simulator was obtained.

The results tabulated thus far, shown in Table 1, support our hypotheses. Student confidence in the material employing the simulator was greater than for assignments which did not use the simulator, despite the fact that these homeworks were frequently more difficult. The effect of the tool on interest in the material is also quite

positive. Student comments elicited during evaluation of the class and of the Wumpus World simulator indicate that the tool is accomplishing its intended purpose. A few of these comments are listed below.

- "The agent shell and code is very useful for the homework assignments."
- "This graphical project has the advantage of making it easier to see the results of our work."
- "This hands-on approach is a good way to learn material, [the approach] answers a lot of practical questions."
- "Something very good was that we did not have to implement the whole environment, but just insert our agent in it. Everything is already there, ready to use."

Another indicator of the success of this project is the number of other programs that are integrating the tool into their classes. The Wumpus World simulator has become so popular that the authors of a popular AI textbook, written by Russell and Norvig (1995), have included a link to our site for our previous C++ implementation of the Wumpus World simulator (they are currently updating the link to include our Java-based simulator, see http://www.cs.berkeley.edu/~russell/aima.html). This software is accessed an average of 13.22 times each day (statistics collected since April 25, 1999), and we have received many e-mail messages from professors using the software in their classes.

CONCLUSIONS

IN THIS PAPER WE DESCRIBED a multimedia environment for teaching AI classes. This environment is built upon an integrated tool that simulates agent-based technologies including search, planning, learning, vision, and language processing. Preliminary results indicate that students benefit from using this tool in terms of subject interest, confidence in the material, and ability to understand and utilize the presented techniques.

The simulator can be enhanced in a number of ways. First, we would like to add an option to choose between two modes of operation. In *TimeOut* mode, the server would give each client one-minute to respond before collecting actions in a round-robin fashion, executing actions in order of agent priorities. To avoid unnecessarily biasing agents, the *Continuous* mode would execute agent actions as the server receives them. We would also like to add features such as the ability to rotate and scale images for more realistic image processing and audio generation of hints to support implementation of speech recognition algorithms.

ACKNOWLEDGMENT

This work was supported by NSF grant IRI-9502260.

REFERENCES

Clark, W.M. (2000). Learning chemical reaction equilibria on CD-ROM, (pp. 30-33) In D.G. Brown (Ed.). Vignettes from America's Most Wired Campuses. Bolton, MA: Ankar Publishing.

Graham, C.R., & Trick, T.N. (1998). Java applets enhance learning in a freshman ECE course. *Journal of Engineering Education*, 87(4), 391-197.

Hanks, S., Nguyen, D., & Thomas, C. (1993). A beginner's guide to the truckworld simulator. *Technical Report* 93-06-09, University of Washington, Seattle, WA.

Durfee, E.H., & Montgomery, T.A. (1989). MICE: A flexible testbed for intelligent coordination experiments. *Proceedings of the 1989 Distributed AI Workshop (pp. 25-40). Orcas Island, WA.*

Pollack, M.E., & Ringuette, M. (1990). Introducing the tileworld: Experimentally evaluating agent architectures. *Proceedings of the National Conference on Artificial Intelligence* (pp. 183-189). Boston, MA.

Reinhardt, A. (1995). New Ways to learn. *Byte*, 20(3), pp. 50-72.

Russell, S., & Norvig, P. (1995). *Artificial Intelligence: A modern approach.* Upper Sadle River, NJ: Prentice Hall.

ABOUT THE AUTHORS

Lawrence Holder is an Associate Professor and Associate Chairman in the Department of Computer Science and Engineering at the University of Texas at Arlington. His research interests include artificial intelligence and machine learning. Dr. Holder received his M.S. and Ph.D. degrees in Computer Science from the University of Illinois at Urbana-Champaign in 1988 and 1991. He received his B.S. degree in Computer Engineering also from the University of Illinois at Urbana-Champaign in 1986. His current home page address is http://www-cse.uta.edu/~holder. Author's present address: Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, TX 76019-0015 E-mail: holder@cse.uta.edu

Diane Cook is an Associate Professor in the Computer Science and Engineering Department at the University of Texas at Arlington. Her research interests include artificial intelligence, machine planning, machine learning, robotics, and parallel algorithms for artificial intelligence. Dr. Cook received her B.S. from Wheaton College in 1985, and her M.S. and Ph.D. from the University of Illinois in 1987 and 1990, respectively. Descriptions of her research projects and publications can be found at http://www-cse.uta.edu/~cook. Author's present address: Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, TX 76019-0015 E-mail: cook@cse.uta.edu