A Client-Server Interactive Tool for Integrated Artificial Intelligence Curriculum

Diane J. Cook and Lawrence B. Holder
Department of Computer Science and Engineering
Box 19015
University of Texas at Arlington
Arlington, TX 76019-0015
{cook,holder}@cse.uta.edu

Abstract

The goal of this project is to increase students' interest in Artificial Intelligence, 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, and outline plans for continued development of the tool. 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.

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 Artificial Intelligence. Our class organization centers around 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. We will also outline plans for continued development of this interactive tool.

Background

The foundational classes in Artificial Intelligence at UTA are divided into AI 1 and AI 2, offered each year to upperclass undergraduate students as well as graduate students. Material in these classes is introduced by describing a rational agent and presenting techniques for creating such agents (Russell 1995). A rational agent is one that uses intelligence to interact with the world in order to maximize expected performance. 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 rational agent must employ reasoning techniques to understand the environment and formulate an appropriate action, and can change the environment by executing a selected action.

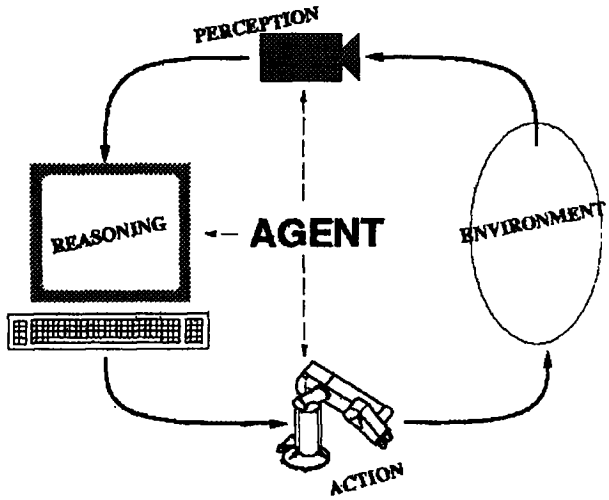

Figure 1. Rational agent design.

Students in the AI classes learn about the goals of rational agents, and learn current techniques for creating these agents. In particular, students are taught 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 to achieve desired goals. Natural language processing, uncertainty reasoning, computer vision, and speech processing are all essential components of rational agent design, and machine learning techniques are introduced so that the agent can improve performance on tasks with experience.

Many studies have been conducted that indicate the value of computer-based multimedia and simulation tools

to learning (Clark 1997; Graham 1998). This is perhaps no more important than in Computer Science curricula, where students perform a majority of their work in front of a computer screen. In response, we are developing a tool which simulates and tests ideas presented in lectures in a dynamic, competitive, and complex simulation environment.

The focal point of the class material is an object-oriented Wumpus World Simulator. 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. Because the environment is object-oriented in design, each student project actually adds to the capabilities of the tool and thus the features that can be learned by students in other classes.

Other simulation environments exist which also support artificial intelligence research and teaching. These systems include TileWorld (Pollack 1990) and Truckworld (Hanks 1993). Our tool presents a number of features not found in these environments. Our system uses a Java-based server design, which facilitates visualization of the environment, incorporation of additional objects written in various programming languages, and multi-platform 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 Wumpus World is based on an early computer game. The basis for the game is an agent who explores an NxN grid world while avoiding obstacles, pits, and a creature known as the Wumpus. The objective of the game is to collect all gold bars, return to the initial grid location and exit the cave. If the agent steps into a square containing a Wumpus or a pit, he dies immediately. An example world is shown in Figure 2. Our implementation of the Wumpus World follows the description given by Russell and Norvig (Russell 1990), and a pointer to an earlier version of our simulator is available from the oinline site for this textbook at http://www.cs.berkeley.edu/~russell/aima.html.

At the start of each turn, the agent can perceive his surroundings. 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 a Wumpus is hit by an arrow, the player will sense a woeful *scream*, indicating the Wumpus' demise.

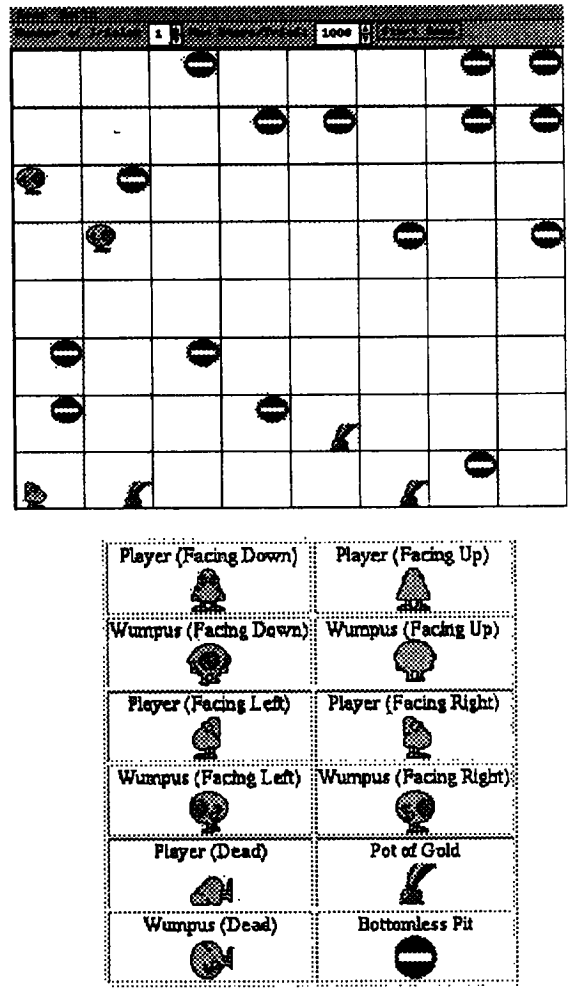

Figure 2. Sample Wumpus Word game and key

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: 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/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 1000 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 game, each with its own separate logic and possibly separate hardware platform. The simulation employs a client-server architecture to separate the simulation code from the agent design.

To support platform independence and an object-oriented design, the simulator was implemented using jdk version 1.2. This language supports transfer of objects such as command selections and sensed percepts between the client and server 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 in order of connection.
5. The server starts the first trial.
6. The server sends out an *Initial Percept* to each client, as shown in Figure 4.
7. The server waits for clients to respond.
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).
10. If the agent dies or climbs 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 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.

Command-line options can be used to specify the world size, the number of agents and the number of trials to simulate, the required agent response time, and a port number for the server. A file can be specified that contains world information.

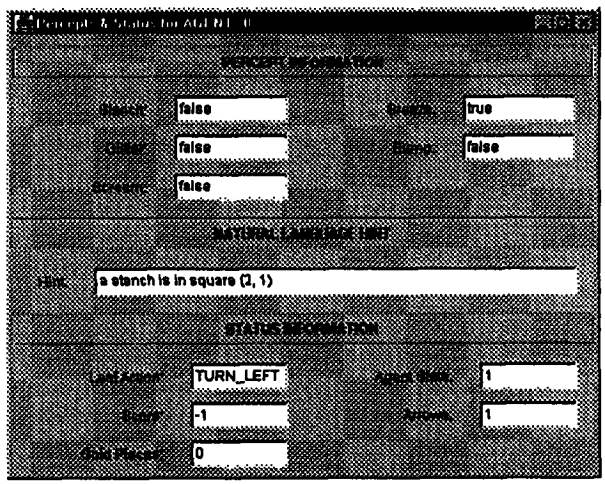

Figure 3. Percept information displayed for Agent 0.

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, circle the gold, random walk, and follow agent.

Class Projects

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 set of exercises that allow students to learn about a variety of artificial intelligence techniques in the context of the Wumpus World simulator. Each project enhances the student's previous addition. As a result, much of the overhead involved in designing Artificial Intelligence programs is eliminated, and students can see how performance of a rational agent improves with the introduction of each technology. Here are the assigned exercises.

Agent 1: Students initially 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.

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 that use principles of logic and constraint satisfaction to more efficiently generate a sequence of actions to achieve the goal (grab gold without being killed).

Agent 5: This agent is called DT_Agent, because this agent has the capability to reason about uncertainties in the environment using techniques from decision theory. Pieces of evidence, including sensing breezes, stenches and glitter contribute to the probability that a pit, a wumpus, or gold is in a given location. These probabilities are combined with associated rewards to allow the agent to make a decision that will yield the greatest *utility*. In this exercise, students use a *belief network* to represent all of the features of the environment and their associated probabilities and to select an action.

Agent 6: This agent makes use of machine learning techniques to improve decision making with experience. In this assignment, agents use reinforcement learning to learn the value of each action from a given location in the world. Action values 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, 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 its 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 of the cell as shown in Figure 2. 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.

In addition to class exercises, research projects have been designed using the Wumpus World simulator. These projects include designing multi-agent teams and predicting the next move of the Wumpus in order to learn an effective counter-strategy. Because these projects are object-oriented, they are incorporated into the simulator to provide more functionality each time the classes are taught.

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.

	Mean (0=low..5=high)	
	Sim	Other
Confidence in material	4.27	4.17
Effect of tool on learning	3.59	N/A
Effect of tool on interest	3.70	N/A

Table 1. Evaluation results.

We predicted 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.

WISE

The current simulator provides a useful environment for teaching AI techniques. However, many real-world tasks involve a greater set of issues including interacting with

humans and robots, in a physical as well as simulated environment.

To address the need for a more realistic development environment, we are extending the Wumpus World simulator into WISE, a Wireless Intelligent Simulation Environment. Instead of modeling the wumpus world as a two-dimensional grid, WISE will model an actual physical environment (in our case, the second floor of Nedderman Hall, shown in Figure 4). Agents will play the Wumpus World game within this new environment.

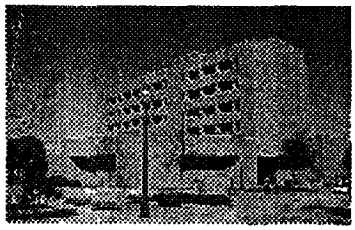

Figure 4. Nedderman Hall.

In addition, we have incorporated new types of agents into the environment. As well as software agents, human agents and robot agents can now play the Wumpus World game. Physical agents communicate with the simulator over a wireless network, and are equipped with wearable computers, similar to the one shown in Figure 5, and locator devices to send and receive information about the environment. Pioneer robots, such as the ones shown in Figure 6, will be used for this environment.

Figure 5. Wearable computer.

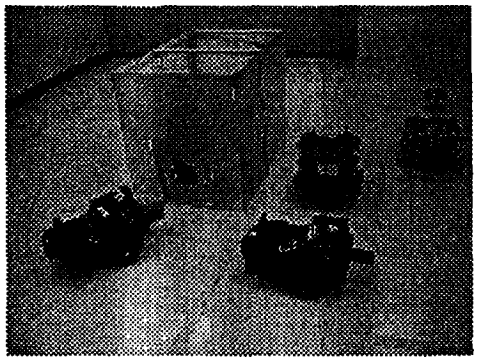

Figure 6. Pioneer robots.

The WISE system builds upon the current Java-based simulator, and also utilizes a client-server architecture. Unlike the current simulator, client movements do not need to be synchronized – the server will process commands as it receives them from various agents. This extended environment is currently under development.

Conclusions

In this paper we described a multimedia environment for teaching Artificial Intelligence 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.

Acknowledgements

This work was supported by NSF grant EIA-0086260.

References

Clark, W. M. 1997. Using multimedia and cooperative learning in and out of class. In *Proceedings of the 1997 Frontiers in Education Conference.*

Graham, C. R., and 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., and Thomas, C. 1993. A beginner's guide to the truckworld simulator. Technical Report 93-06-09, University of Washington.

Pollack, M. E., and Ringuette, M. 1990. Introducing the tileworld: experimentally evaluating agent architectures. In *Proceedings of the National Conference on Artificial Intelligence*, 183-189.

Russell, S. and Norvig, P. 1995. *Artificial Intelligence: A Modern Approach.* Prentice Hall.