Creating Significant Learning Experiences in Introductory Artificial Intelligence

Amy McGovern
School of Computer Science
University of Oklahoma
Norman, OK 73019
amcgovern@ou.edu

Jason Fager
School of Computer Science
University of Oklahoma
Norman, OK 73019
jfager@ou.edu

ABSTRACT

We introduced an arcade-style gaming environment for use in a mixed undergraduate and graduate introductory artificial intelligence (AI) course. Our primary goal in this course was to provide students with a "significant learning experience" [3]. We achieved this goal by creating projects based in the game environment that illustrate several major AI topic areas. These projects were designed to be challenging, enjoyable, and to demonstrate AI programming in a realistic environment. Each of the projects was designed to be feasible for all the students yet flexible enough to allow the stronger students to explore alternative solutions. We evaluated our success in achieving these goals through student evaluations, comments, and exam grades.

Categories and Subject Descriptors

I.2.m [**Artificial Intelligence**]: Misc.; K.3.2 [**Computers and Education**]: Computer and Information Science Education—*Computer Science Education*

General Terms

Experimentation

Keywords

Significant Learning Experiences, Games, Semester Projects

1. INTRODUCTION

Student involvement is one of the primary distinguishing characteristics of a "significant learning experience" [3]. In engineering and science courses, efforts to foster student involvement often take the form of semester long projects. Projects that enable an introductory AI class to succeed at the goal of creating a significant learning experience should meet the following criteria.

1. Be enjoyable enough to stimulate and maintain student interest throughout the semester.
2. Be flexible enough to illustrate many of the possible topics that can be covered in an AI class. The project should focus on the AI aspects of each task and not require a significant amount of external knowledge.
3. Be extensible so that stronger students can explore alternatives to the main solution approach while weaker students can still complete the project.
4. Be sufficiently challenging to invest students in creating a good AI solution but not allow trivial answers.
5. Be realistic enough that students gain an appreciation for working on real-world applications of AI.
6. Be feasible[7]. This is particularly important for maintaining student interest.

The introductory AI class at the University of Oklahoma is open to seniors and graduate students with some advanced juniors also entering the class. In the spring 2006 class, we had 33 students. Five of those were graduate students and the remaining 28 were undergraduate students.

In an effort to create significant learning experiences in our AI class, we introduced an arcade-style gaming environment. Our game, Spacewar, was designed and programmed primarily by a small team of students enrolled in a graduate machine learning class in the fall of 2005. Spacewar was inspired by the classic game of the same name written by Stephen Russell at MIT in the early 1960s [4] and the arcade game Asteroids. Two or more teams, each with at least one ship, battle each other in an environment that contains moving obstacles, bullets, and energy beacons. A complete description of Spacewar is given in the next section.

This gaming environment satisfied all six of the criteria for a successful project specified above. By their nature, game-based projects tend to satisfy criterion 1. Providing a graphical setup helps to maintain student interest in the game. Spacewar allows the students to play alongside the intelligent agents that they have created, which also helps to maintain student interest. Many of the students are motivated to create an agent that can play better than themselves. This becomes a fun challenge for the students and an introduction to AI programming that many AI researchers are familiar with. To help stimulate and maintain interest, we also created a class-wide competition for each project. The competition ladder ran daily and students who rose to the top of the ladder received extra credit on the project. Once the top player received the maximum extra credit, the extra credit went to the next player on the ladder.

SIGCSE'07, March 7–10, 2007, Covington, Kentucky, USA.

During the past semester, we used the spacewar simulator for three major projects. The first project involved navigation, the second project focused on learning, and the third project used traditional planning techniques. Our game environment is flexible enough to adapt it for each of the projects and to allow the students to use the results of the earlier projects to complete the subsequent and more difficult projects. This satisfies criterion 2.

The stronger students demonstrated that Spacewar was extensible by exploring alternative solutions that still made heavy use of AI techniques. This satisfies criterion 3. These students quickly rose to the top of the daily competition ladder. If the problem was too trivial, these extensions would not have been useful. This satisfies criterion 4.

Given that the Spacewar environment is partially observable, stochastic, sequential, dynamic, continuous, multiagent, cooperative and competitive (see [6] for definitions of these terms), criterion 5 is satisfied. Satisfying criterion 6 requires a tradeoff in how realistic the tasks are. We evaluated this tradeoff using student feedback at the end of the semester and the students felt that the project was realistic.

2. SPACEWAR

Figure 1 shows a screenshot of the Spacewar simulator in the capture the flag configuration used for the planning project. Students can program individual ships or teams of ships. Each ship carries an energy cell that fuels its thrusters, cannons, and life support systems. If the energy cell is depleted, the ship self-destructs. A ship can recharge its energy cell by collecting energy beacons from the environment, or by returning to its home base if it is available. Ships can also be damaged or destroyed by collisions with asteroids floating in space or from the cannon fire of other ships. Ships can compete individually or cooperate in teams.

The problem domain is difficult for an AI agent because it has a large and continuous state space, is a competitive and cooperative multi-agent environment, and taxes the agent with several possibly conflicting goals at any given time. The agent must make low level control and navigation decisions while simultaneously dealing with the higher-level cognitive tasks of choosing between attacking an enemy, avoiding an obstacle, defending a teammate, capturing a flag, or moving to collect an energy beacon.

Spacewar is written in Java so that it can run on a variety of machines and is simple enough that changes are fairly straightforward to make. For instance, the capture the flag functionality was added by student request. The physics simulation follows traditional asteroids-style physics in a toroidal world. Game parameters such as number of ships per team, number and size of obstacles, and game type are controlled with an XML configuration file. The source code and compiled releases are available online under an MIT-style open-source license at `www.cs.ou.edu/~amy/spacewar`.

The focus of the environment is on allowing individuals or teams to program intelligent agents that can be plugged into the simulator. An interface is provided for controlling ships either using a software agent or via direct control by a human. At each timestep, an agent chooses between rotating to the left or to the right, firing its thrusters, firing its cannon, doing nothing, or some reasonable combination of the above. For example, an agent can turn left, thrust, and shoot at the same time. This interface is sufficiently simple that a human player needs only 4 keys to fully control a ship. Higher-level actions can be programmed as well. It is also possible to define a central command agent that can communicate with individual ships.

While the graphical environment is useful for providing human interactivity and immediate visual feedback, a real-time simulator is not ideal in all situations. Some learning algorithms may require many individual games to acquire sufficient experience for improved performance. Additionally, the requirements of a nightly round-robin competition ladder make real time simulation infeasible. For these reasons, Spacewar also provides a non-graphical mode that allows the simulator to run at a much faster rate. In this mode, we can collect detailed statistics for student analysis or to track performance for the ladder.

3. PROJECTS

The course focused on three fundamental areas of AI: search, learning, and planning. Each of the projects addressed one of three core areas. The projects are described in more detail below.

3.1 Search

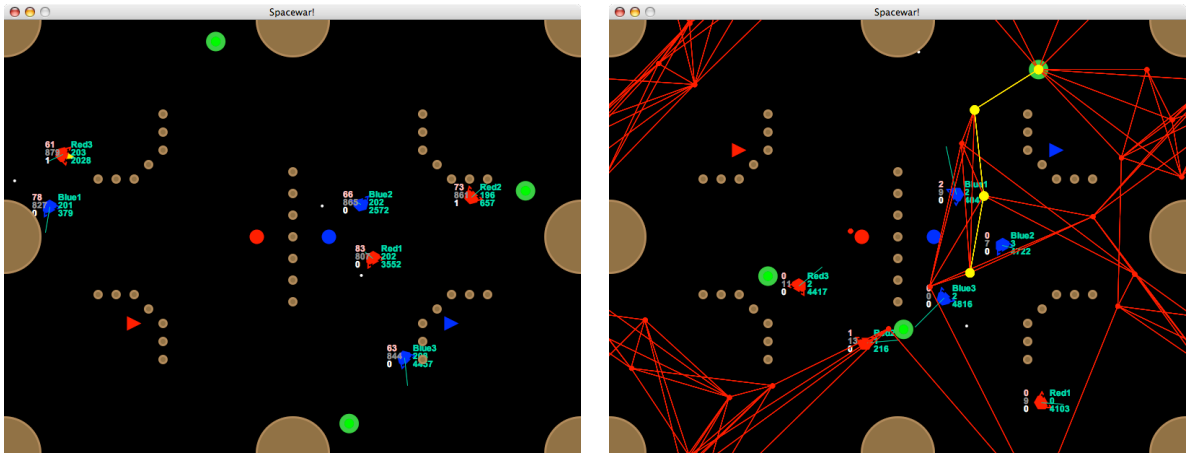
A*[6] is one of the most fundamental of AI search techniques and it is well suited for navigation. A* is an optimal informed search technique that requires an admissible heuristic which estimates the remaining cost to the goal from any state. The environment for the first project had a single ship, moving asteroids, and an energy beacon that was placed randomly in a clear space in the environment. Beacons did not move unless they were hit by an asteroid or by cannon fire. In those cases, they reappeared somewhere else in the environment. Students programmed their ship to collect as many beacons as possible using A* search.

Because the environment was continuous and dynamic, plain A* search could not be used. This helps to satisfy the criterion of making the project more realistic because very little of the real world is static, discrete, or deterministic. To address the continuous nature of the domain, students could choose to implement A* in three ways. The first was gridded A*, where the environment was broken in $n \times m$ grid squares. Each square was a vertex in the graph for A* and edges were created between empty adjacent squares. The second approach was a hierarchical gridded approach where the grid would only be refined in areas where the path was promising. The last approach was roadmap A*, where the students created a random set of vertices in clear spaces in the environment and connected nearby vertices as long as the path between them did not intersect an obstacle. Figure 1 shows a graphical example of using roadmap A* in the capture the flag environment.

These approaches enabled the students to implement A* in a continuous environment but none of them addressed the dynamics of the moving ships, obstacles, and cannon fire. This was tricky and we had the students deal with the dynamics by replanning frequently. Some of the students chose to replan based on trigger events (such as running into an obstacle) and some replanned on a fixed time interval. They were prohibited from replanning at each step.

3.2 Learning

The second segment of class focused on learning with a particular emphasis on reinforcement learning [8]. Reinforcement learning is a semi-supervised machine learning



Capture the flag A* Navigation

Figure 1: Left: A game of capture the flag in the Spacewar simulation environment. Energy beacons are the green circles, flags are triangles, and the brown circles (both small and large) are asteroids. Each ship is labeled with its name, kills, hits, beacons collected, and current energy levels. Right: Showing the A* path is optional but we include it here to demonstrate navigation using roadmap A*. The red circles and lines show the A* graph and the best path is highlighted in yellow.

technique where agents learn through interaction with the environment. Unlike supervised learning where an external teacher tells the agent the correct answer for each stimulus, a reinforcement learning agent must learn through trial and error. The agent interacts with the environment by executing actions and receiving a scalar reinforcement signal, usually called reward. The reinforcement learning agent learns a mapping from states to actions that will maximize the cumulative reward that it receives over time.

The second project had the students create a reinforcement learning agent to intelligently choose from a set of high-level behaviors. These behaviors included fleeing from asteroids and ships, moving toward a ship, firing on a ship, and moving to a beacon. The high-level behaviors were provided to the students but they could also choose to implement their own. The students had to design their own state spaces and reward functions and implement one of two reinforcement learning techniques. The overall goal for the learning agents was to survive as long as possible. To accomplish this, agents needed to navigate safely around the environment, collect energy beacons, and shoot enemy ships.

This project built on the previous project in that students could make use of their A* agent for navigation. We also provided A* code. Reinforcement learning is well suited to handle the dynamic nature of this environment but the students had to deal with the continuous state space. We suggested that the students create a tabular state space by identifying a set of discrete features that represented the most important features of the environment and we provided a list of potential features.

3.3 Planning

The final segment of the course focused on planning and knowledge representation. Although we originally intended to make each of the projects individual, by student request, we modified the final project to use teams. As part of the course, we had created 6 discussion groups. The final project was assigned near the end of the semester and the students had already formed cohesive groups from the discussions and group quizzes. For this project, each of the six groups created a team of agents to play the game of capture the flag. Teams consisted of 3 ships and a flag. Agents cooperated within a team and teams competitively tried to retrieve as many flags as possible within a fixed time period. Each game was played with two teams.

The project required each group to use STRIPS-style [6] planning to implement a central controller. A pure STRIPS planner would use a restrictive subset of first-order logic as a knowledge representation. However the language restrictions imposed by STRIPS made the project too difficult. We allowed the students to extend STRIPS by specifying negative pre-conditions on their actions. We also allowed them to specify functions in the pre-conditions, such as *hasMoreFlags(team1, team2)*, so long as the functions could be grounded at planning time.

Each team created a set of high-level behaviors for their ships. They also specified STRIPS-style action schemas for each of these behaviors, complete with pre and post conditions. This project required the students to understand and express a logical model of the environment and actions. To make the planning problem more feasible, we made the asteroids stationary and allowed ships to respawn. STRIPS is designed for static deterministic environments which meant that the students needed to replan frequently to handle the dynamics. As with the previous projects, students could replan based on trigger events or could replan on a fixed time interval. Students could also use the previous two projects to create their high-level behaviors.

3.4 Results

Navigation with A* was the most straightforward of the three projects but several students had very creative solutions. Students could earn extra credit for both creativity and ladder placement and 10 of the 33 students earned some

extra credit. Creative solutions included approaches to generating the map points dynamically, an admissible heuristic that was not straight line distance (32 of the 33 students used straight line distance), and dynamically replanning based on trigger events such as running into obstacles, running low on fuel, or noticing that the beacon had moved. The project specified a minimum planning interval but the students who chose to replan on events performed better.

The learning project piqued the creative interest of nearly half of the students with 16 of the 33 students receiving some amount of extra credit for creativity. Additionally, 7 students received extra credit for placement on the competitive ladder. Although many of these students overlapped, several students were able to rank well on the ladder with no additional creativity. For creativity, one student used knowledge from a previous class to implement neural nets [5] for function approximation. Many students focused their creativity on improvements to the high-level behaviors, particularly on creating a more accurate shooting function. One student implemented Kahlman filtering to calculate a better cannon-firing algorithm and several other students implemented discrete approximations to this approach. As with the student who implemented neural nets, these students brought knowledge from other classes to this project. One student implemented such an impressive set of high-level behaviors that his learning agent was able to use a very tiny state space and still be quite successful.

Student ingenuity shone for the planning project as well. Four of the six groups received extra credit for creativity. Most of the originality focused on creating interesting high-level team behaviors. Several teams focused on creating complex behaviors for individual ships with relatively loose coordination (the planner took care of the coordination). One team created a complex set of modes that the team could enter into and the planner focused on which mode was best to be in at any one time. Once the mode was chosen, the individual ship assignments was straightforward. Additionally, several groups added concurrent conditions to the action schemas which simplified their planning problems.

4. EVALUATION

Although it is clear from discussions with the students that they enjoyed the projects, we evaluated the hypothesis that this class was a significant learning experience using the student evaluation forms at the end of the semester as approved by the University of Oklahoma's Institutional Review Board (IRB). The IRB required that the surveys be anonymous, which meant we could not observe gender or race because of the small number of women and minorities. These numbers are low across the entire major and not specifically low for this class. If we can enroll enough women and minorities to study the effects of this project separately for them, we will seek approval to ask for gender and race on our surveys. Twenty one students answered the supplemental questions. Each question was scored on a Likert scale where students could strongly agree, agree, be neutral, disagree, or strongly disagree.

If the project was a successful significant learning experience, then the students should be interested in taking another class in AI (many of them are graduating seniors so we cannot simply measure whether they do take another AI class). In addition, they would highly recommend the class to other students. The results of these two questions are shown in lines 1 and 2 of Table 1. In both cases, the overwhelming majority of the answers are quite positive, which gives us evidence that the class was a successful significant learning experience. Selected written student comments are included below.

> "Literally one of my favorite courses at OU. Even better than topology! Very intense class but the payoff is exponential. Would definitely look at AI for grad school as a result of this class."
>
> "You did a really great job! I feel like I really learned a lot, and reinforced the stuff I already knew. I want an RL [Reinforcement Learning] class!"
>
> "I really enjoyed this course. And I'm taking machine learning because of it."
>
> "This class has been the most interesting and enjoyable CS class to date. ..."
>
> "The best CS class I have taken at OU."

The next evaluation question examines whether spacewar was flexible enough to be an interesting example for the broad set of AI techniques that we asked the students to implement. This addresses criterion 2 above. The results of this question are given in line 3 of Table 1. The overwhelming majority of the students strongly agreed or agreed with the question and no one disagreed.

Criterion 5 specified that a successful project would give the students an understanding of what it would be like to work on a real-world AI application. The results of this evaluation were slightly mixed and are shown in line 4 of Table 1. Although a majority of the students agreed with this comment, more of them simply "agreed" rather than just "strongly agreed." A comment from one of the students sheds some light on this. This student wrote: "Spacewar project is beneficial in learning AI but it has too many parameters to justify (and take into account) which is not in real life case [sic]". While it is true that you can't tweak the parameters of the real world, AI and machine learning systems that function in the real world generally require a tunable parameter set. We will address this issue next year and we expect it will help the students to understand how spacewar is more realistic.

While group projects are not directly addressing the criteria that we listed for successful projects, they indirectly address criteria 1, 3 and 6. The results of the student evaluation of the groups are shown in line 5 of Table 1. The majority of the students benefited from the group project. As one of the students wrote, "My primary problem with the spacewar projects was understanding the games environment and how to turn it into a searchable state space. Working in the group project and seeing how my other group members handled it helped a lot in that respect."

Student evaluations give us one approach to quantifying the success of the spacewar project for teaching AI. We also examine average grades as a second measure of success. If the projects were successful in helping students learn each of the main concepts, we would expect students to receive higher grades on the questions related to the projects on the final exam than on the questions that did not relate to the projects. This hypothesis was upheld with students receiving an average of 78.2% on the project based questions and an average of 72.0% on the non-project questions. We

Line	Question	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
1	This class and the instructor stimulated my interest in taking another class in artificial intelligence.	57.14	19.05	14.29	4.76	4.76
2	I would recommend this course to other students.	42.86	38.10	19.05	0.00	0.00
3	The Spacewar projects helped me to apply multiple principles and techniques to a dynamic working example of artificial intelligence.	52.38	42.86	4.76	0.00	0.00
4	The Spacewar projects gave me an idea of what it might be like to work on real life artificial intelligence applications.	33.33	47.62	14.29	4.76	0.00
5	My participation in the group project provided me with experiences that enabled me to learn more than I would have by doing the project independently.	38.1	28.57	19.05	9.52	4.76

Table 1: Student evaluation questions and responses. N = 21 for each question. The numbers shown are the percentage of students who answered each category.

used a paired t-test to check the hypothesis that the project based questions had a statistically higher mean than the non-project questions. The difference was significant with a p-value of 0.006. Other hypotheses, such as easier exam questions for the project based questions, could explain this difference as well. However, we feel that the difference in scores combined with the student evaluations demonstrates the success of our project.

5. DISCUSSION AND CONCLUSIONS

The variety of projects and the student involvement in the projects helped to make the AI class a significant learning experience. The students enjoyed the project and it fulfilled all of the criteria of a successful project outlined at the beginning. We plan to use the project in the AI class for the spring of 2007 and beyond. Since this was the first year of using the Spacewar system in an AI class, we did encounter some difficulties that we will address in future classes. The primary issues revolved around the second half of criterion 2 in that the projects required programming beyond that focused on the current topic. For example, although students could compute their A* graphs, they did not understand how to create a ship to follow lines. Once we provided them with line following code (implemented using pd-control [2]), the difficulties smoothed out. For any future projects, we will always ensure that we have a solution implemented before we hand out the project. This will enable us to hand out code to help them focus only on the AI aspect of the project.

For the group project, we had several class sessions devoted to the groups themselves. The students really felt that they learned a lot from these discussions and we plan to extend this to discussions on individual projects in future years. We hypothesize that these discussions will help the weaker students. We will evaluate the impact of these discussions through a control group and through project outcomes (grades).

While the current Spacewar software met the needs of the projects this past semester, we would like to be able to leverage more advanced software packages for future classes. One example is the MUPPETS system which is used as a platform for TankBrains, an interactive Java simulator used for introductory programming [1]. This may be able to be used to recreate the Spacewar game in an even more compelling environment for AI programming.

6. ACKNOWLEDGMENTS

We would like to thank Josh Beitelspacher for his key role in developing the Spacewar environment and the anonymous reviewers for their helpful comments.

7. REFERENCES

[1] K. Bierre, P. Ventura, A. Phelps, and C. Egert. Motivating OOP by blowing things up: an exercise in cooperation and competition in an introductory java programming course. In *SIGCSE '06: Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*, pages 354–358, New York, NY, USA, 2006. ACM Press.

[2] J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Prentice Hall, 3rd edition edition, 2003.

[3] L. D. Fink. *Creating Significant Learning Experiences: An Integrated Approach to Designing College Courses*. Jossey-Bass, 2003.

[4] J. M. Graetz. The origin of spacewar. *Creative Computing*, pages 56–67, August 1981.

[5] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[6] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, second edition, 2003.

[7] M. R. Scheessele and T. Schriefer. Poker as a group project for artificial intelligence. In *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*, pages 548 – 552, 2006.

[8] R. S. Sutton and A. G. Barto. *Reinforcement Learning. An Introduction*. MIT Press, Cambridge, MA, 1998.