Teaching Artificial Intelligence and Robotics via Games [Abstract] *

Daniel Wong and **Ryan Zink** and **Sven Koenig**
Computer Science Department
University of Southern California
Los Angeles, CA 90089-0781
wongdani@usc.edu, ryanzink@gmail.com, skoenig@usc.edu

Abstract

The Department of Computer Science at the University of Southern California recently created two new degree programs, namely a Bachelor's Program in Computer Science (Games) and a Master's Program in Computer Science (Game Development). In this paper, we discuss two projects that use games as motivator. First, the Computer Games in the Classroom Project develops stand-alone projects on standard artificial intelligence topics that use video-game technology to motivate the students but do not require the students to use game engines. Second, the Pinball Project develops the necessary hardware and software to enable students to learn concepts from robotics by developing games on actual pinball machines.

Introduction

The Department of Computer Science at the University of Southern California recently created two new degree programs, namely a Bachelor's Program in Computer Science (Games) and a Master's Program in Computer Science (Game Development) (Zyda and Koenig 2008). In addition, we explore whether games can be used to teach traditional concepts from computer science in our regular computer science classes because games motivate students, which we believe increases enrollment, motivation and retention and thus helps us to educate more and better computer scientists. In general, games can be used to teach almost every area of computer science. For example, computer architecture is important for understanding how game consoles work, networking is important for building networked games, human-computer interaction is important for designing user-friendly games, and algorithms are important for implementing efficient games. Furthermore, games can be used to teach a variety of important job skills for computer scientists in both academia and industry, including technical skills (such as computational thinking, software engineering and programming skills), creativity, design skills, problem-solving skills and teamwork skills (such as collaboration skills with non-computer scientists). In this paper, we discuss two projects that explore how to teach traditional concepts from artificial intelligence and robotics using games as motivator. It is future work to evaluate the effects of our efforts.

Computer Games in the Classroom Project

The undergraduate and graduate versions of the artificial intelligence class at the University of Southern California are taken by both game students and regular computer science students. We explore how to coach their projects in terms of video-game technology. Video games can be used to teach artificial intelligence because artificial intelligence is important for creating more realistic or more fun games. Most video games use search algorithms for path planning. For example, Dragon Age: Origins uses sophisticated search algorithms with abstraction hierarchies to satisfy the runtime and memory requirements imposed by BioWare. Some video games also use planning or machine learning algorithms. Those students who are familiar with game development already understand game engines well while the regular computer science students might not want to learn about them, at least not at the same time as learning artificial intelligence. We therefore decided to create several stand-alone projects on standard artificial intelligence algorithms that use games as motivator but do not make use of game engines (Zyda and Koenig 2008). We now describe the three projects that we have developed so far, some of which have already been used successfully at other universities, namely the University of Nevada at Reno, the University of Central Florida and Massachusetts Institute of Technology. Each project text is about 15 pages long and motivates the project, introduces an algorithm, gives examples of its use and then provides a variety of possible questions, including easy and difficult ones. Teachers need to select among them since each project text lists too many of them for a project of a reasonable size and some of them are difficult research questions. More information can be found on our web pages at

*The following students made the Computer Games in the Classroom Project possible: Kenny Daniel, Alex Nash, William Yeoh and Xiaoming Zheng. The following students made the Pinball Project possible: Hrudesh Doke, Darren Earl, Clark Kromenaker, Allen Pan, Selby Shlosberg, Jaspreet Singh, Daniel Wong, Ryan Zink and Fred Zyda. Our initiatives were partly supported by a grant from the Fund for Innovative Undergraduate Teaching at the University of Southern California, the Rose Hills Foundation and the National Science Foundation under grants 0350584, 0413196 and 0113881. This abstract reuses ideas, photos and a small amount of text from our longer papers cited in the bibliography.

The Nintendo Wii is a big success because its motion-sensitive paddle lets users innovatively control objects on the screen with hand gestures. Still, some people say that all gaming instruments that we use today will look ridiculously old-fashioned and redundant in ten years ... In the near future, for example, users will likely be able to control objects on the screen with bare hands ... We limit our ambition to a static variant of the gesture recognition problem, where the computer has to classify hand gestures in single images. ... [C]omputers can easily get confused by ... different hands, angles, backgrounds, lighting conditions and other differences. In this project, we use (artificial) neural networks to recognize hand gestures in single images. Neural networks are among the most important machine learning techniques. They are too general to reliably classify a large number of hand gestures without any preprocessing of the images but are able to classify a small number of hand gestures out of the box. There exist more specialized gesture recognition techniques that scale much better but the advantage of neural networks is their generality. They are able to solve a large number of classification problems ..., and are thus often ideal solutions for simple classification problems and for prototyping solutions of more complex classification problems.

Figure 1: Start of Project Text (Zheng and Koenig 2010)

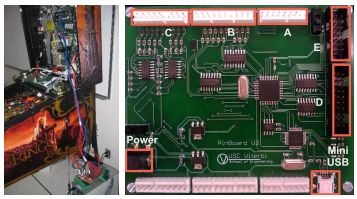

Figure 2: Pinball Interface (1st and 2nd Generation)

idm-lab.org/gameai.

Fast Trajectory Replanning with Variants of A* The students need to code A* and extend it to the recently developed incremental heuristic search algorithm Adaptive A*, which requires them to develop a deep understanding of A* and heuristics and gives them an introduction to incremental heuristic search, a new area of search not yet covered in standard textbooks. The students then use Adaptive A* to move game characters in initially unknown gridworlds to a given target location (Koenig and Yeoh 2008).

Any-Angle Path Planning with Variants of A* The students need to code A* and extend it to the recently developed any-angle search algorithm Theta*, which requires them to develop a deep understanding of A* and heuristics and gives them an introduction to any-angle search, a new area of search not yet covered in standard textbooks. The students then use Theta* to plan any-angle paths for game characters from a given start location to a given target location (Koenig, Daniel, and Nash 2008).

Gesture Recognition with Neural Networks The students need to use neural networks to recognize user gestures for video games, which requires them to develop an understanding of the back-propagation algorithm (Zheng and Koenig 2010), see Figure 1. This project extends a project from Tom Mitchell's Machine Learning book (Mitchell 1997).

Pinball Project

The standard computer science education tends to teach students only about software but not about interfacing it to mechanical systems. It thus does not prepare students well for robotics, which requires at least basic knowledge of electronics, signal generation, embedded systems, communication protocols, interface programming or real-time programming. Designing pinball games can be used for this purpose since pinball machines are simple (although rather unusual) robots. They contain actuators (such as solenoids), sensors (such as switches) and visual outputs (such as lights). We therefore developed a hardware and software interface between a PC and a solid-state pinball machine, see Figure 2. The students of the small pilot CS499 class "Designing and Implementing Games on Pinball Machines" at the University of Southern California then designed and implemented Pinhorse, a simple pinball game that features a true multi-player mode where each player directly influences the game of the other player. Pinhorse is a proof of concept game that demonstrates what can be accomplished with such an interface (Wong et al. 2010). To the best of our knowledge, this is the first time that anyone has managed to control an existing pinball machine completely, although others have tried before (Clark 1997; Bork 2005). We have recently made our interface available to the University of Alberta (Canada) for research and teaching purposes and are now thinking about integrating artificial intelligence techniques into pinball games, for example, to adapt the difficulty of the game to the abilities of the players. More information can be found on our web pages at **idm-lab.org/pinball** together with an 11-minute YouTube video that demonstrates the features of Pinhorse.

References

Bork, J. 2005. Controlling a pinball machine using Linux. *Linux Journal* 139.

Clark, D. 1997. Progress toward an inexpensive real-time testbed: The pinball player project. In *Proceedings of the Real-Time Educational: Second Workshop*, 72–79.

Koenig, S., and Yeoh, W. 2008. A project on fast trajectory replanning for computer games for 'introduction to artificial intelligence' classes. Technical report, Department of Computer Science, University of Southern California, Los Angeles (California).

Koenig, S.; Daniel, K.; and Nash, A. 2008. A project on any-angle path planning for computer games for 'introduction to artificial intelligence' classes. Technical report, Department of Computer Science, University of Southern California, Los Angeles (California).

Mitchell, T. 1997. *Machine Learning*. McGraw Hill.

Wong, D.; Earl, D.; Zyda, F.; and Koenig, S. 2010. Teaching robotics and computer science with pinball machines. In *Proceedings of the AAAI Spring Symposium on Educational Robotics and Beyond: Design and Evaluation*.

Zheng, X., and Koenig, S. 2010. A project on gesture recognition with neural networks for 'introduction to artificial intelligence' classes. Technical report, Department of Computer Science, University of Southern California, Los Angeles (California).

Zyda, M., and Koenig, S. 2008. Teaching artificial intelligence playfully. In *Proceedings of the AAAI-08 Education Colloquium*.