Robots Can Wear Multiple Hats in the Computer Science Curriculum at Liberal Arts Colleges

Christine Shannon

Centre College
600 W. Walnut
Danville, KY 40422
shannon@centre.edu

Abstract

Faculty at liberal arts colleges are often challenged to offer a quality education to their students, complete with opportunities for undergraduate research. To guard against a curriculum that is too theoretical, students want to see applications of their course work and tangible results of their efforts. Like all computer science educators, we want to attract students to our discipline. The use of robotics can often be part of the answer in each of these realms.

Undergraduate Research Projects

Centre College, like many good liberal arts institutions, is anxious to provide its students with opportunities to engage in undergraduate research. Robotics provides a venue where students can make interesting contributions and where an ongoing project can be maintained.

In 2004, enabling funds from a professorship were used to purchase a Sony AIBO and a grant from the Kentucky Space Grant Consortium provided stipends for students involved in the project. These resources were available because machine learning and robotics were of interest to the funding agency. A sophomore and two junior level students were involved in the project over the course of two summers and one academic year.

The project was centered around the topic of machine learning, specifically reinforcement learning, and the software package *pyro* (http://pyrorobotics.org/) was selected as the means of communication between the computer and the AIBO. Because Centre computer science students learn Python as their first language and then Java, use of pyro enabled them to get started on the project more quickly than if they had had to learn C++ first.

While this robot has a large number of sensors the work was almost entirely involved with the vision system which depends on a camera mounted in its nose. Vision processing software allowed us to set filters to detect the presence of the bright pink of a ball in the image produced by the camera. The earliest projects involved the use of reinforcement learning to teach AIBO to move his head so as to locate the pink ball. The results were noticeable. In one experiment, both the ball and the robot always began in the same position. In this case the average number of steps the robot took to locate the ball decreased with training from 29.9 to 5.58. There were similar results for random initial positions of the robot head. With a small data set, initial path averages were as high 33.14 steps but decreased to less than 8.0 with training.

The complexity of the problem increased substantially when locomotion was added to the mix. Now the goal was not only for the robot to locate the ball visually but also to move toward it. When the size of the blob exceeded a threshold value of 750 pixels, the experiment declared he was close enough and the number of steps was recorded. AIBO wagged his tail to announce success.

In performing these experiments, one has to make decisions about the sizes of various parameters: how much should the head be moved or tilted, how fast should the robot move forward or turn. Based on ideas contained in the literature (Kohl and Stone 2004) an experiment was devised to find optimal values for six of the parameters. The policy gradient reinforcement learning described there was quite successful in the present setting. Beginning with a base policy which, on average, required 196 steps to locate the ball, the process of generating base policies, testing, and adjusting the policies in the direction of maximum improvement yielded approximately a fifty per cent improvement in performance. (Soenneker, 2006).

Many of the experiments dealt with temporal difference learning and the problem of estimating the action value function, $Q^{\pi}(s,a)$, which is the value of taking action a in state s under policy π. The students only worked with action value functions which were implemented as a table. However, they experimented with several representations of the state, trying to capture as much information as possible while maintaining a reasonable table size.

Inspired by a another paper in the literature (Smart and Kaelbling, 1998) they worked on ways to provide initial

settings for this array so that the robot did not start with a *tabula rasa* but instead had some guidance in making early decisions. The result was a program which could be run either in a "teacher" mode or a "learning" mode. In the teaching mode, the choice of action was dictated by the program but based on the rewards or punishments received, the action value function was updated as though the robot had made the decision. Once the program was switched to learning mode, the robot chose its actions based on the contents of the $Q^{\pi}(s,a)$ table, slowly improving the estimates depending on the rewards and punishments resulting from its choices. Of course, some of the time the action was chosen randomly to encourage exploration for even better choices than had been tried in the past.

Results from these experiments show that the average number of steps per episode fell from over 350 to less that 250 as training progressed.

Observations

The kinds of experiments that were carried out by these students gave them a lot of freedom to pose and test hypotheses. In the course of our work, we tried a variety of state descriptions, different types of reward and punishment schemes, and variations in the types of questions asked. Since the students were not required to understand all the mathematical details, the work was accessible to them. One student submitted an abstract on the results of this work to an undergraduate research contest and presented a poster session at a regional meeting.

There are a great many more things that can be explored – including the possibility of implementing the entire thing without the use of the pyro software which does add a layer between the programmer and the hardware. The introduction of eligibility traces will probably be the next stage in this process.

The intention is to keep this as an ongoing project that students can join primarily in the summer but also during the academic year as time permits. Centre maintains an honors program named in honor of John C. Young in which students can propose research projects to be conducted during their senior year. The research papers are presented in a Spring symposium open to the entire community and published by the college. The existence of this continuing project makes it possible for students to propose an honors thesis which is substantive while at the same time having a reasonable horizon. Robotics offers a very suitable platform for interesting and useful experimentation.

Robots as Projects in CS Courses

Before the inclusion of robotics projects in the artificial intelligence course, it was a struggle to find assignments that enabled students to implement many of the ideas that were presented. Most projects were simply too big to implement in a semester. However, whether it is using Lejos to program Lego robots, or the use of pyro with simulators, there are now ways to make interesting but reasonable assignments in small worlds where students can exercise the techniques that are introduced in lecture. They can use Lego robots to implement a simple reactive agent early in the course and later an agent which makes use of a more complex decision structure and state to make its decisions. The use of *pyro* allowed us to implement the logical agent for the "wumpus world" problem found in the text by Russell and Norvig (Russell and Norvig, 2003).

Robots are also useful in introductory courses. There is about a two week segment on robotics at the very end of the CS I course. Having learned Python in the course, the students have not had any experience with complex syntax or the need to declare variables and types. The use of NQC to program Lego robots gives them an opportunity to try a second language with a more complicated syntax as well as to engage some other ideas like asynchronous execution.

One of the easiest tasks to program is simple random motion in which the robot moves forward for a random amount of time and then turns either left or right for a random amount of time. Once the students learn to use the light sensor to watch out for a black boundary line, a second task can be added to back up whenever the boundary is detected. Almost every student writes at least one incorrect solution to this problem. Unless they include code to stop the random walk behavior while the boundary avoidance task is running, they will occasionally see their robot plunge recklessly over the edge when the random walk behavior interrupts. It is a very good lesson on the difference between synchronous and asynchronous behavior.

Other useful assignments include designing wall or line followers, agents which measure lengths or count stripes, and proximity sensors.

Observations

Students like the idea of building robots but they also experience a significant level of frustration when they realize that effectors are not as exact as they would like them to be. Performance is affected by the strength of the batteries and the lighting in the room. What worked flawlessly the night before, fails miserably in the morning. They find that building and testing a robot is more complicated than trying a line of code to see if it will work.

For the students in AI, it is useful to write code which must fit in a small amount of memory. In one project where students had to build a scanner to identify block characters constructed from green and yellow Lego plates, one team attempted a brute force approach in which they

planned to store a two dimensional (5 by 8) array description of each possible letter. They ran out of memory. Another team studied the collection of letters and discovered that they could distinguish the letters on the basis of just three of the columns. Their decision tree solution took very little memory and since they had to make fewer passes over the block, they had less difficulty with the torque issues which tended to misalign the scanner. This was a clear case in which a careful design was much superior to a brute force solution.

In all of these cases, the use of robots provides students with experiences they would not have had otherwise. And of course, the enthusiasm and cheering which accompanies project demonstrations mitigates the frustrations that are experienced along the way.

Robots and Outreach

All of us are concerned about attracting more students to our discipline. In recent years, robots have been an attraction to some students. Presentations with the AIBO at the local middle schools have been very warmly received and indeed, they would have been quite satisfied just to watch the robot! Similarly, while the numbers are not large, talks about robots and artificial intelligence to prospective students do garner an interested audience.

In the past year we have attracted a few students who have participated in some sort of robotics competition during their high school years and want to continue that involvement during college. We have only begun the process of developing something in this area. We have acquired a VEX radio controlled robot but significant progress awaits more time to devote to this activity.

Observation

Time is probably the resource in shortest supply around a liberal arts college. Developing significant outreach can be quite prohibitive in view of the time commitments involved. When the content of research projects can be used to effectively garner student interest, the extra time involved can be marginal. Projects in robotics can often be used to demonstrate the kind of excitement that often arises in the study of computer science.

Conclusion

Robotics can be used as a source of undergraduate research projects and to enhance the learning of topics in computer science as well as to attract additional students to disciplines like computer science and engineering. In a liberal arts college it offers the opportunity to provide students with experiences which enrich their education and provide applications of theory. They can be a good investment of faculty and student time.

Acknowledgements

Research described in this paper was done with students Peter Burns, Jackie Soenneker and Will Larson. It was supported in part by a grant from the Kentucky Space Grant Consortium and by funds in the Margaret V. Haggin professorship.

References

Kohl, Nate and Peter Stone. 2004 Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2619-2624. New Orleans, LA.: IEEE International Conference on Robotics and Automation.

Russell, Stuart J. and Peter Norvig. 2003 *Artificial Intelligence: A Modern Approach*, 2nd Ed. Upper Saddle River, NJ.: Pearson Education, Inc.

Soenneker, Jackie. 2006 Machine Learning on an AIBO Robot. In Student Research Contest Extended Abstracts. Nashville, TN.: CCSC:SE

Smart, William D., and Leslie Pack Kaelbling. 1998. A Framework for Reinforcement Learning on Real Robots. January 19, 2007. Available at <http://www.cse.wustl.edu/~wds/content/papers/aaai1998/aaai1998.pdf>.

Sutton, Richard S., and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. London: The MIT Press.