"Choo Choos", Robots and Computer-Based Instrumentation for Students [1]

R. Stephen Dannelly, Carl W. Steidley
Texas A&M University – Corpus Christi

Abstract

This paper describes a versatile and innovative active learning laboratory currently under development that will be used to teach a wide variety of computer science and mathematics subjects. The laboratory, equipped with a model railroad system, autonomous mobile robots and a network of control computers, offers students hands-on experiences with hardware interfacing, data acquisition, real-time programming, client/server programming, and observing actual physical results of numerically computed solutions to problems. The goal of this work is to produce students with experience in designing solutions to challenging problems that have many dimensions and also the ability to effectively implement and test those solutions. This laboratory will enable us to transform a wide range of computer science algorithms and abstract concepts into physical realities. This paper briefly describes the laboratory hardware and overviews an incrementally complex series of assignments supported by the laboratory.

Background

Our Computer Science program has an enrollment of approximately 250 undergraduates. Although the undergraduate program produces graduates who are able to attend graduate school, many choose to join industry. Our graduates are highly proficient at developing software to solve real problems. The approach we use to achieve this is to emphasize the design and writing of software in a range of areas.

Computer Science 4348 - Systems Programming is a three hour senior level course which addresses the design and implementation of system software such as application support libraries and interprocess communication facilities. This course has a particularly strong emphasis on software writing and addresses low-level software development. Prior to funding for this project, students enrolled in this course developed software using a network of DEC workstations. Assignments included logging user activity over a network, simple interprocess communication between Unix machines, use of semaphores to control shared resources, and an extensive client-server development project.

We believe the effectiveness of this type of course can be greatly improved by placing it in a setting where students can actively extend the concepts being taught into tangible realities [1, 2]. Therefore, we have begun building a new laboratory equipped with a model railroad system

[1] This work was sponsored in part by a National Science Foundation Course Curriculum and Laboratory Improvement – Adaptation and Implementation grant, grant number DUE-9950839.

and five networked linux-based control computers, and two ActivMedia Pioneer class autonomous mobile robots. The Real-Time Lab will initially be used to support only the Systems Programming course (first use of the lab will be during Spring2002). But, once sufficient interface applications and libraries have been developed by our students, we also plan to use the Real-Time Lab in our Artificial Intelligence courses, our senior capstone design project course, Operating Systems courses, and the graduate-level Mathematical Modeling course. We also have plans to use the Real-Time Lab as an important part of tours given to prospective students.

A model railroad component was selected for five reasons. First, both discrete-state (modeled by state machines or Petri nets) and continuous-state (described by transfer functions) control problems can be modeled. Second, a sensor-instrumented model railroad is versatile and will enable students to work on a large variety of potential projects ranging from simple to complex. Third, it is not difficult to obtain the necessary equipment; model railroads are easily available in our community and many other locations. Fourth, it is virtually impossible to think of a more inexpensive option with the same capabilities. Finally, we believe that the model railroad will stimulate interest throughout the campus and the local community. Students will be enthusiastic about a course that will teach them to use the computer to control a model railroad [5, 6, 7].

We selected the ActivMedia Pioneer mobile robots as the robotic platform for the laboratory for several reasons. First, it is a sturdy, proven platform. Second, the robot control software runs a reactive planning system with a fuzzy controller, behavior sequencer, and deliberative planner all of, which are easily interfaced with the C language. Third, it has seven sonar transducers to provide object detection, range information, and feature recognition. Fourth, it has an optional versatile, high-performance vision system. Fifth, each of its drive motors includes a high-resolution encoder for precise position sensing.

Railroad Setup

The Real-Time Lab's model railroad system has two main parts: the track bed and the control/development computer network. The track bed is composed of approximately 100 linear feet of HO scale track with 30 electronically controlled track switches called turnouts. Sensors have been placed every three inches along the tracks. These sensors only detect the presence of an object, not the type of train car, nor its speed. The layout will also utilize an electronically controlled model crane that is able to lift, move, and lower objects with very fine precision. These components rest on half inch paperboard that is fixed to a 30-inch high, six foot by eight foot table constructed of plywood. All track, turnouts, and sensors were purchased off-the-self. We believe this low cost setup will provide our students with an enormous number of projects that they will find very interesting, challenging, and useful.

Control and Development Computer Network

Five Pentium III PCs will be used for software development and testing. These linux-based computers are attached to the campus LAN and hence the Internet. This will allow our students to use the system from anywhere. One computer is equipped with an electronic signaling device, known as a "throttle", that controls all components of the track system. We have configured the throttle to control the movement of five separate locomotives, each turnout, and the model crane. This main control computer is equipped with the A/D card that supplies location sensor data.

In addition to the computer controlled track system, we are also installing four low-cost cameras. Each camera will be fixed in place to provide different views of the system via the Internet. One camera will overview the entire trackbed, while two others will provide close up views of key track portions. The fourth camera will focus on the model crane.

Robots

Pioneer Mobile Robots were chosen as a component of the Real-time Lab because of their ease of use and ability to support complex and varied programming. The robots will be used in support of our artificial intelligence course for the purposes of offering a unifying theme that draws together the disparate topics of artificial intelligence, focusing the course syllabus on the role artificial intelligence plays in the core computer science curriculum, and to motivate the students to learn by using concrete, hands-on laboratory exercises.

In addition to the many standard problems that can be assigned in advanced courses such as artificial intelligence, we have begun to use our robots in lower-division courses. For example, we have used these robots in our sophomore-level data structures course. [1] With the appropriate introduction we believe that the use of such manipulables promotes retention and turns abstract concepts into real-world problems and solutions. [3, 4]

Sample Railroad Assignments

The very first assignments that use the model railroad system will ask students of the Systems Programming course to create libraries to allow easy control of train speed and direction, control of turnouts (track switches), and access to location sensor data. This will require the students to create several device drivers. These libraries will be used by the Systems Programming students to create a server process that communicates requests to both the throttle and sensor A/D card.

A student project to be assigned soon after those listed above will be in a summer short course covering the X Window System. Each student will create a graphical user interface to display the current status of the system, control train speed and direction, and control each turnout. Simple display status information will include the location of each train. This will require the use of the C library routines and device drivers written by earlier Systems Programming students to determine both the number of trains on the track and their lengths (trains may have varying numbers of attached cars). Other information to be displayed includes the actual and scaled speed of each train, direction of each turnout, crane status, and pre-collision warning messages. Effectively displaying this wide array of information in real-time will present an exciting challenge to the students. Effective interfaces will become modules to be used by later projects.

A very interesting problem to be tackled by either the Systems Programming or an Operating Systems course will be to create an application that can run on each of the five computers so that each computer shares the turnouts and simultaneously controls a different train. This distributed resource control problem is a classic operating systems theory problem that will be presented, literally, right in front of the students' eyes. Students will be expected to derive a centralized control solution where one computer serves as a gatekeeper and each application makes requests for turnout changes to that one gatekeeper. Students will also be expected to derive a distributed control solution where each application shares equally in the decision making as to which application may control the shared resources of track space and turnout direction. Of course, this classic problem has many algorithmic solutions which students can implement and view the tangible results.

A problem requiring students to draw heavily on their mathematics background to derive a solution is to move a train in proportional speed. Of course, a model train can start and stop very quickly, whereas real trains start and stop slowly. The objective of such an assignment would be to create mathematical formulas to determine scaled acceleration and deceleration.

A challenging problem for an Artificial Intelligence course will be to have an application automatically move a single train from one point on the track to another point with the minimum number of changes to turnouts. In other words, implement a version of the travelling salesman problem. This also will force the students to draw on their data structures background to determine an appropriate abstract data type to represent the track.

Building on the previous problem, we plan to have students design, implement and test algorithms to have one application automatically move a train from point A to point B, while a separate application simultaneously moves a different train from point C to point D. Two possible versions of this problem are apparent. First, both applications may be running on a single machine allowing the shared resources to be controlled via semaphores. Second, each application may be running on separate machines, requiring one of the distributed control methods described above.

Continuing this natural progression of theoretical problems, an assignment to be solved by students enrolled in either the Operating Systems or the Systems Programming courses would be to detect deadlock. Detecting when one train occupies track portion 15 and is requesting track portion 20, while another train occupies track portion 20 and is requesting 15 will be an interesting problem. Once again, the Real-Time Lab will facilitate the detection of deadlock in both a distributed and centralized environment.

The Real-Time Lab also supports wide ranges of other student projects. Beginning in Spring2002, students in the undergraduate Software Engineering course will create design specifications for the system. Beginning in Fall2002, students in the Object-Oriented Design and Programming course will begin creating classes for reuse in other Train Lab projects.

Sample Robot Assignments

The first law for great many robot projects can be stated as "move about and don't get stuck." Jones, et al [9] describe a series of introductory robotic projects the first of which is described as in the previous sentence. They call this project, "The Lewis and Clark" project. We use this and variations of several other projects described by Jones in which we require students to incrementally incorporate the sensor suite of the robots. These projects have names such as: Mouse, where students are required to program the robots to follow walls; Magellan, students are required to program the robots to navigate a closed path; Apollo 13, where students are required to program the robots to home in on a single bright light source; Fire! robots are programmed to traverse a maze and extinguish a lit candle placed somewhere in the maze. (See www.trincoll.edu/~robot/ for a description of the Fire Fighting Home Robot Contest.)

Moreover, this laboratory is being utilized by master's degree students to complete more advanced projects. As an example, a student is developing a "path finder" module for the navigation of one of our mobile autonomous robots. This project is utilizing a "Best First Search" algorithm to determine a path for a robot between two fixed points and a series of intermediate points in a predefined external world. The Best First Search Algorithm is a "goal directed" and "knowledge based" heuristic algorithm. That is, the points are well defined at the outset and all points of the pre-defined world are taken into consideration during the search. The value of a heuristic function is calculated at each node or location along the navigation path. This value is an estimate of the cost to move to the next location.

Another student has created an API to allow one of our robots (or robot simulator) to be controlled by a CLIPS expert system. This will enable AI Student toeasily create an intelligent control system without having to know details of interfacing with sonars or drive motors.

Summary

We see the Real-Time Lab as a highly effective means of turning abstract ideas into firm realities while exciting the interests of students. This inexpensive system supports the teaching of a vast array of computer science concepts. We expect that involving science students in "real-life" challenging problems in a setting that they find interesting will promote retention and aid in recruitment [6, 8]. While this paper has introduced the hardware setup and overviewed numerous potential projects, more information concerning the Real-Time Lab can be found at www.sci.tamucc.edu/trainlab.

References

[1.] Dannelly, R. Stephen, "Use of a Mobile Robot in A Data Structures Course", The Journal of Computing in Small Colleges, Vol. 15, No. 3, pp. 85-90, March 2000.

[2.] Dannelly, R. Stephen and Carl Steidley, AA Hardware Laboratory for Incorporating Computer Science Theory with Real-World Interests and Difficulties, Proceedings of the 2000 ASEE-GSW Annual Conference, CD-ROM Session 61, Paper C1, March, 2000

[3.] Kumar, Depak and Lisa Meeden, "Robots in the Undergraduate Curriculum" The Journal of Computing in Small Colleges, Vol. 13, No. 5, pp. 105-112, May 1998.

[4.] Kumar, Depak and Lisa Meeden, "A Robot Laboratory for Teaching Artificial Intelligence" SIGCSE Bulletin, Vol. 30, No. 1, pp.341-344, March 1998.

[5.] Steidley, Carl, "A Laboratory for Real-Time Software Systems Development", Computers in Education Journal, Vol. VI, No. 2, pp. 10-14, April-June 1996.

[6.] Steidley, Carl and M.M. Asoodeh,"Developing An Interdisciplinary Hardware Laboratory With CIM Capabilities", Proceedings of the 1996 Annual Conference of the American Society for Engineering Education inaugural CDROM edition, Session 1620.

[7.] Steidley, Carl, "An Undergraduate Laboratory for Real-Time Software Systems Development", Journal of Computing in Small Colleges, Vol. 11, No. 7, pp. 1-5, April, 1996.

[8.] Steidley, Carl and M.M. Asoodeh "Developing an Interdisciplinary Computer Science/Industrial Technology Laboratory", Proceedings of the 1996 Annual Conference of the Gulf-Southwest Section Conference of the American Society for Engineering Education, pp. 366-370, March 1996.

[9.] Jones, Joseph L., Anita M. Flynn, Bruce A. Seiger, Mobile Robots: Inspiration to Implementation, A.K. Peters, Ltd., Natick, MA, 1999.

R. Stephen Dannelly is an Associate Professor of Computer Science. He earned his Ph.D. at Auburn University. His research interests are in software engineering, applications of artificial intelligence, and scientific visualization.

Carl W. Steidley is a Professor of Computer Science. He earned his Ph.D. at the University of Oregon. His research interests are in robotics and the applications of artificial intelligence. Steidley has had research appointments at NASA Ames Research Center, Oak Ridge National Laboratories, and with Electro Scientific Industries , Portland, OR.