

Efficient Use of Robots in the Undergraduate Curriculum

Judith Challinger
California State University, Chico
400 West First Street
Chico, CA 95929
(530) 898-6347

judyc@ecst.csuchico.edu

ABSTRACT

In this paper we describe how a single Khepera II robot was used for an assignment in a senior level course on graphical user interface implementation. The assignment required each student to implement a graphical user interface for remotely controlling a Khepera II robot. From the author's point of view, the focus of the project was how to efficiently and effectively allow the 35 students enrolled in the course to have access to the robot for the purposes of testing and debugging. Furthermore, two of the students were enrolled through the distance education program and were remotely located. The intent was to see if an assignment making use of the robot would stimulate student interest, and whether it could be done with a minimal investment in hardware, lab, and instructor resources. The students were surveyed at the conclusion of the assignment in order to assess the effectiveness of the assignment. The results of this survey are presented, along with plans for future work.

Categories and Subject Descriptors

K.3.2 [Computers and Education]: Computer and Information Science Education ---Computer Science Education

General Terms

Algorithms, Design, Human Factors, Languages.

Keywords

Java, Swing, GUI, Graphical User Interface, Robotics, Khepera.

1. INTRODUCTION

Using the enticing qualities of a robot to attract and retain student interest in computer programming is nothing new [1]. Pedagogy based on the use of robots to engage students has become fairly common. It is especially prevalent in CS1/CS2, as well as in courses that specifically address the topics of robotics and artificial intelligence. [2] The most interesting robots are still quite expensive, however. The result is that this equipment is typically reserved for the use of a select few graduate students or advanced undergraduate students, working directly under a faculty member. When our institution recently received a grant to acquire the equipment necessary to establish an Intelligent Systems Laboratory [3], the goals of the principle investigators

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Conference '04, Month 1-2, 2004, City, State, Country.
Copyright 2004 ACM 1-58113-000-0/00/0004...\$5.00.

included encouraging the use of the equipment in supporting undergraduate education, and promoting interdisciplinary and faculty collaboration. With their encouragement, the author decided to attempt to utilize one of the more interesting robots, the Khepera II, for an assignment in a senior level course on graphical user interface implementation.

In this course the students learn to implement graphical user interfaces using the Java Swing API. The assignment that was selected required each student to implement a graphical user interface (GUI) for remotely controlling a Khepera II robot. From the author's point of view, the focus of this investigation was how to efficiently and effectively allow the 35 students enrolled in the course to have access to the single robot for the purposes of testing and debugging. Furthermore, two of the students were enrolled through the distance education program and were remotely located. The intent was to see if an assignment making use of the robot would stimulate student interest, and whether it could be done with a minimal investment in hardware, lab, and instructor resources. The students were surveyed at the conclusion of the assignment in order to assess the effectiveness of the assignment. Information on the assignment, the methods used to give students access to the robot, and the results of the student survey are presented here, along with plans for future work.

2. THE KHEPERA II

The Khepera II (see Figure 1) is a moderately expensive robot. It is small, only 70mm in diameter, and is round. The robot has 8 infrared proximity detectors - three to the front and left, three to



Figure 1. The Khepera II Robot.

the front and right, and two at the back. They can be seen around the outside circumference of the robot. It has wheels that can be controlled in a variety of ways, and it also has position counters on each wheel. The Khepera II has a variety of running modes. For this assignment the robot was connected to a host computer via a serial cable and configured to process a simple command protocol [4].

3. STUDENT ASSIGNMENT

The assignment requires each student to implement a graphical user interface (GUI) for remotely controlling the Khepera II robot. The result of the student's work will be a client program that can connect to a robot server, can query the state of the Khepera II, and can send motion commands to the Khepera II. The GUI consists of graphical components to display the state information, buttons and menu elements to control the robot, and a map representing the robot's environment as discovered during navigation (see Figure 2). The student is provided with a Java archive (jar) of classes that implement a model for the robot and all the necessary client communication software. Their task is to

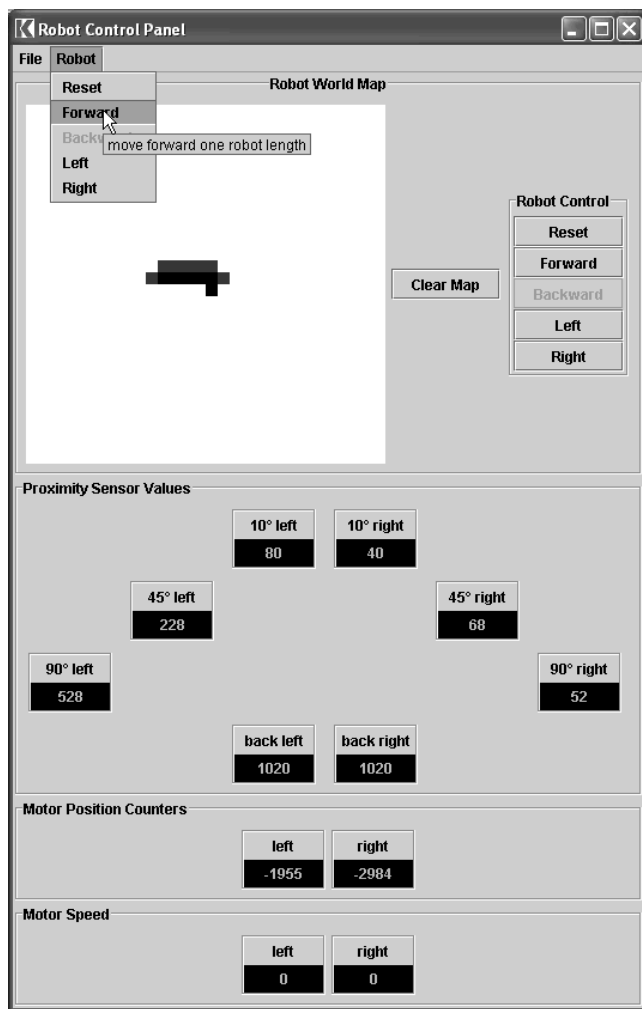


Figure 2. The student will implement this Robot Control GUI.

implement the GUI precisely as described in the assignment¹, using the classes provided in the jar file. As the purpose of the course is to learn how to use the Swing API, from the perspective of the student this assignment has the following goals in terms of the GUI implementation:

- Implement a fairly complex GUI with several different panels and files.
- Customize a JPanel by extending and then overriding paintComponent().
- Use Action objects to create both buttons and menu items.
- Use TitledBorders to group and organize components.

An additional, very important, aspect of the assignment is the continuing exposure of the student to design patterns. In fact, a main theme of the course is design patterns [5]. One in particular that is studied is Model – View – Controller. Swing provides a particularly rich demonstration of the MVC design pattern in action. The application of this design pattern is further emphasized in the separation of the Model for the robot (a class provided to the student) from the View – Controller (implemented in the classes the student will develop).

4. ROBOT CLIENT AND SERVER

Several problems needed to be considered in order to make this assignment feasible:

- How to minimize the effort needed to communicate with the Khepera II so that the student could focus on the GUI implementation.
- How to provide a connection to the Khepera II to students as they design, implement, test, and debug, their GUI.
- How to ensure fairness among the students in terms of connect time to the robot.
- How to ensure that only one student GUI is controlling the Khepera II at any given time.
- How to ensure the safety of the Khepera II.
- How to allow the student to see what the Khepera II is actually doing.

The robot physically resided in the author's office. A special table was constructed with walls around the edges to prevent the robot from leaving the table. Rounded corners helped to minimize situations where the robot would become stuck. An apparatus was placed above the table to hold both a web cam and the cable providing the connection between the Khepera II and the host computer. Students could view images from the web cam remotely, providing them with visual information about what the

¹ The complete assignment using the Khepera II may be seen at www.ecst.csuchico.edu/~judyc/0304S-csci233/program03.html

robot was doing. Communication between the Khepera II and the host computer was accomplished over a serial link. A software package was developed by the author, which supplies the classes necessary to build both a client application and the robot server application. Some of these classes are discussed in the sections that follow.

4.1 Robot Class

A class was developed to provide a model for the robot. This class is used both by the robot server, and by the robot client (for which the student is implementing the GUI), to instantiate an object that will represent the state of the robot. The Robot class is both Observable and Serializable. The student does not instantiate the Robot object directly. Communication between the client and the server is part of the functionality that is provided to the student in the jar file. When the client establishes communications with the robot server, the Robot object on the server is serialized and sent in its entirety to the client and instantiated.

The Robot class provides the following functionality in its public interface:

- A method is provided for each type of low-level command that can be sent to the robot. This is similar to the approach taken by Harlan, et al [6]. The students do not know about these methods, and are instructed to use the high-level commands discussed next. An example is:

```
public void sendWheelSpeed(int left, int right)
```

- Several methods are provided that implement high-level commands. These can be used by the student to make the robot perform certain actions (turn right 90 degrees, for example) without having to worry about calculating the parameters needed for the low-level commands to accomplish this. Examples of high-level commands include:

```
// move forward one robot length  
public void moveForwardOne()  
// turn right 90 degrees  
public void turnRightNinety()
```

- An entire set of synchronized methods to access the state of the robot (as maintained in the Robot object). This includes such things as the current proximity readings, wheel motor position and speed, and so on. For example:
- The Robot class also provides a method that will parse a command from the robot and update the instance variables accordingly. This method is also synchronized. Again, this is a method the students are not told about. The client communications will pass any commands received from the robot (via the robot server) to the Robot class to parse. Since the Robot class is Observable, the students can arrange to receive events for Robot state changes just by registering their GUI object(s) as an Observer of the Robot.

4.2 Robot Client Communications

Included in the jar file provided to the student is the RobotControl class. This class contains the main method for the client application. The main method instantiates the RobotControl object and then immediately calls a method on the object that will loop forever, reading from the server. During initialization, the RobotControl object will establish a connection with the server, receive the initial state of the Robot object and instantiate it, and instantiate an object of type RobotControlGUI, passing it a reference to the Robot object and to the RobotControl object. The student is given an empty class to fill in as the basis for their GUI:

```
// RobotControlGUI.java  
  
package robot.control;  
import robot.shared.*;  
// more import statements  
  
public class RobotControlGUI  
    extends JFrame implements Observer  
{  
  
    // use to call shutdown( )  
    private RobotControl robotControl;  
  
    // the Robot model  
    private Robot robot;  
  
    // more instance variables as needed  
  
    public RobotControlGUI(RobotControl rc, Robot r)  
    {  
    }  
  
    public void update(Observable r, Object o)  
    {  
        // schedule on the event dispatch thread!  
    }  
  
    // more private methods are OK  
}
```

4.3 Robot Server

A robot server application was developed. This application runs on the host machine to which the Khepera II is connected via a serial communications link. The server establishes

communications with the robot, and then waits for connections by client applications. The server provides the following services:

- In the running mode that was selected, the Khepera II does not have any way to generate events to the host computer. Therefore, it must be polled. The server polls the robot several times per second and updates the state maintained in the Robot object (the server uses the same class as the client for this purpose).
- The server will accept multiple client connections. Each client that establishes a connection is sent the current robot state in the form of the serialized Robot object. All messages coming in to the server from the robot (in response to polling, for example) are parsed and used to update the Robot object, and are also passed along to each client connection.
- Only one of the clients is allowed to send commands to the robot at any given time. Messages coming in from a client are passed to the Khepera II robot, assuming that client is currently selected to be controlling the actions of the robot. These messages are passed directly through the server to the robot. The Khepera II will generate a message in response to a command that changes its state. It is this confirmation message from the robot that is parsed, used to update the state of the Robot object, and passed along to all client connections. This approach ensures that the state of the Robot object will accurately represent the state of the real robot.
- One of the first problems encountered was the need for a timeout on client connections. The problem arose when the client application written by a student would have an exception, but the student would not kill all the threads. This could cause the connection to the server to exist for quite some time without the student even realizing it. Not only did this take up a connection, but also if that client was the one selected to control the robot then other clients were prevented from doing so. A timeout of 10 minutes was settled on and seemed to satisfy most people.
- Another problem was encountered in which some students' client applications would put the Khepera II into a state where it was spinning madly, or running continuously even though it was up against a wall. It is again possible that the student inadvertently left a thread running that was sending commands to the robot. The intent of the assignment was for the user to be able to control the robot remotely to explore an environment by moving forward or backward one robot length at a time, or turning left or right. Using the high-level command methods of the Robot class, this should have proceeded at a fairly sedate pace. However, things don't always get used as intended! The solution that was effective was to have the server refuse to forward a "move" command to the robot unless the current wheel speed was zero.
- The server logs the date, time, and IP address for each client connected or when a connection is closed. Any time a command is refused, this is logged as well. All error conditions are logged. The logging functionality was essential for solving problems with the server.

5. RESULTS

All students were able to complete the assignment. Most of them enjoyed the challenge, although it certainly wasn't without some level of frustration at times. The Khepera II robot survived the experience without any damage. The author was generally able to keep one step ahead of the students in solving problems such as the need for a timeout on client connections.

Students were asked to complete an anonymous survey in order to give feedback on the quality of the assignment. Ten questions for which the student could answer either Excellent, Good, OK, Not Good, or Terrible, were as follows:

1. How clear were the learning goals for the robot control assignment?
2. How clearly did the robot control assignment specify what you were to implement?
3. How helpful was the robot control assignment in giving you direction as to how to get started?
4. How appropriate was the level of difficulty of the assignment?
5. How relevant was the robot control assignment to the course objectives and to your academic goals?
6. How stimulating was the robot control assignment in terms of inspiring you to want to work on the project?
7. How stimulating was the robot control assignment with respect to the level of interest it generated in you for further investigations into graphical user interfaces, robotics, or some other aspect of the assignment?
8. How effective was the robot control assignment in helping you learn more about the implementation of graphical user interfaces?
9. What did you think of the robot control assignment overall?
10. How effective was the robot control assignment at minimizing technical difficulties that might hamper your learning?

In addition, there were two free response questions:

- Please share what you liked and/or didn't like about the robot control assignment. Be as specific as possible about why you liked or disliked any aspect of it.
- Do you think this assignment, or one similar to it, should be used in this course in the future? Please share any suggestions you have for improving the robot control assignment, or make any other comments you would care to.

There were 23 students that chose to complete the survey. The results are given in Table 1.

Table 1. Results of Student Survey.

Question Number	Excellent	Good	OK	Not Good	Terrible
1	13%	83%	0	4%	0
2	35%	52%	9%	4%	0
3	13%	43%	40%	4%	0
4	26%	40%	26%	9%	0
5	22%	57%	13%	9%	0
6	48%	26%	4%	9%	13%
7	17%	48%	17%	9%	9%
8	17%	48%	22%	9%	4%
9	9%	61%	9%	4%	17%
10	0	17%	26%	52%	4%

The results show that the majority of the students found the assignment clear, stimulating, and relevant to their learning goals. The survey also shows that technical difficulties encountered by the students were a large source of frustration. It is impressive that most students rated the assignment as well as they did in spite of the problems. The free response answers give a lot of insight into what the problems were. Most had to do with access. Students did not like having to conform to a schedule. Many like to work late at night, and the server and robot were turned off in the evening. With the current configuration of the client software, students could not test and debug when the server was not running. On the other hand, many of the students found the assignment very stimulating. Some of their comments are illuminating:

- “I really enjoyed the assignment. It was just really cool controlling a robot over the internet with a program I wrote.”
- “I liked the fact that this assignment utilized some real world technology. Too often programming assignments are trivial in their purpose. This one was fun and accomplished its purpose.”
- “I liked the robot assignment because it was different than the usual assignments.”
- “I liked writing a program that was used to control hardware.”

6. FUTURE WORK

There are many possible directions that can be explored, and a few development tasks that will improve future assignments of this type. The most important improvement to the software that has been developed would be to include a simple simulator in the client software that is provided to the student. This would not be difficult to do, and would mean that the students could do their initial testing and debugging offline. Quite a few students suggested this in their free responses to the last survey question. It is interesting that this agrees with the findings of Fagin and

Merkle [7], even though they were examining the use of robots at the introductory level.

The Khepera II is a stackable architecture. There are several different types of turrets available – circuit boards with additional capabilities that can be inserted into the connectors on the top of the Khepera II. For example, there is a turret with a video camera, one with a gripper, and one that performs radio communications so the robot does not need to be tethered to the host. It is easy to envision projects similar to this one in a variety of different courses.

One advantage to utilizing a resource such as the Khepera II in a classroom situation is that it makes the students aware of the availability of the lab and the research equipment in it. It is not quite so intimidating once they have used it in such a controlled manner. They feel more confident in proposing a senior project, master’s project, or independent study, which uses the lab and its resources. This assignment has already spun off several interesting student projects, and it is not uncommon to have students stopping by to ask if the “robot project” will be offered again.

7. ACKNOWLEDGMENTS

This classroom project was made possible by hardware, software, and funding from the National Science Foundation (NSF) Major Research Instrumentation (MRI)/Research in Undergraduate Institutions (RUI) grant EIA-0321385 for 2003-2006, a CSU Internal Grant, the College of Engineering, Computer Science, and Technology, and Pacific Media Technologies. Special thanks to the Spring 2004 students of CSCI 233 Graphical User Interface Implementation, for making this investigation fun.

8. REFERENCES

- [1] R. Pattis, *Karel the Robot: A Gentle Introduction to the Art of Programming*, John Wiley & Sons, Inc., 1981.
- [2] M. Goldweber, C. Congdon, B. Fagin, D. Hwang, and F. Klassner, The use of robots in the undergraduate curriculum: experience reports. In *Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education*, p. 404-405, ACM Press, 2001.
- [3] <http://isl.ecst.csuchico.edu/index.htm>
- [4] <http://k-team.com/robots/khepera/index.html>
- [5] O. Astrachan , G. Mitchener , G. Berry , L. Cox, Design patterns: an essential component of CS curricula. In *Proceedings of the twenty-ninth SIGCSE technical symposium on Computer Science Education*, p.153-160, ACM Press, 1998.
- [6] R. Harlan and D. Levine and S. McClarigan, The Khepera robot and the kRobot class: a platform for introducing robotics in the undergraduate curriculum. In *Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education*, p. 105-109, ACM Press, 2001
- [7] B. Fagin and L. Merkle, Quantitative analysis of the effects of robots on introductory Computer Science education. *Journal on Educational Resources in Computing (JERIC)*, Vol. 2, No. 4, 2002.