USE OF ROBOTS TO TEACH INFORMATION TECHNOLOGY AND PROBLEM SOLVING AT WEST POINT

Tom Morel, Rusl Flowers, Jerry Schumacher, Don Welch

Abstract
As part of an ongoing initiative to continually revise and improve its introductory computer science/information technology courses, the Department of Electrical Engineering and Computer Science at the United States Military Academy has added the use of LEGO MindStorms robots and Java as part of the active-learning environment used to teach Information Technology (IT) and problem solving with computers. The use of robots and a robot simulator adds a visual component to problem solving using computers. The Army and the Nation must ensure that its future leaders understand and are capable of taking advantage of IT; therefore, the United States Military Academy at West Point requires all students to take a course on IT and problem solving with computers. This course is an important first opportunity to expose undergraduate students to technology and concepts that will be a part of their daily lives and future careers. The LEGO MindStorms robots are used in the introductory computer science course to teach problem solving skills and fundamental computer programming concepts, and to introduce the concepts of autonomous vehicles, embedded computer systems, sensors, and computer simulation. Positive short-term impact on the students taking the course has been substantial, and while the long-term impact has yet to be measured, it also has the potential to be substantial. Members of the faculty at West Point developed a Java-based programming environment for the LEGO MindStorms robot called Jago. Jago combines the object-oriented Java language with the LEGO MindStorms robot and enables students to write programs in Java that will run in a graphic simulator that can be executed on their own machines. Jago enables the students to see their algorithmic solutions, which helps students to more easily grasp what is happening versus a text-based solution. The cadets are clearly excited to use these learning tools. We have also added lessons that require the students to construct sensors for the LEGO robot as well as program the sensors to complete a final problem-solving project. The short-term results include increased interest in the course objectives and graded assignments. Long-term results have yet to be measured but we are encouraged by both the students' and instructors' positive feedback.

Introduction

The United States Military Academy requires all incoming plebes (freshmen) to enroll in CS 105 – Introduction to Computing. This 40-lesson course provides an introduction to the principles of computing along with an overview and introduction to information technology (IT). The course has two objectives, which are accomplished using hands-on activities, group projects, and active learning within the classroom. The first objective is to teach the cadets about problem solving through the use of the Engineering Thought Process and the Java programming language. This objective is accomplished with lessons on problem solving and programming. The second objective is to introduce and familiarize the cadets with current and evolving information technology, including teaching the cadets how to teach themselves using the Internet, World

Wide Web, and the IT tools available to them. Seventeen lessons of the course are devoted to this objective.

Course Objectives

The first objective introduces the Engineering Thought Process to solve problems. This 4-step process is used throughout the course and is used by the cadets to analyze, design, build, and test their solutions to problems. This problem-solving methodology is critical to their thorough completion of the programming assignments given to them. These programming assignments cover the basic computing principles of sequence, selection, and iteration. Java is our language of choice as the vehicle for learning, understanding, and implementing these principles. In addition, the use of robotics and simulation as part of our programming instruction and assignments makes the problem-solving process more tangible by allowing the cadets to see their algorithms and solutions successfully, or unsuccessfully, implemented. Most students seem to benefit from seeing their algorithms and solutions actually played out by a physical actor and the simulation environment, Jago, provides the cadets with immediate, visual feedback, especially when there is a logic error. Though problem solving is emphasized throughout the course, programming only comprises about 50 percent of the course material.

The second objective of the course focuses on information technology. The cadets learn about computer hardware, software, information systems, security (both data and physical), and the basic concepts of networking. In addition to IT, cadets learn about sensors and participate in a 2-hour lab in which they build a light sensor to be used in conjunction with the Lego robots.

Course Overview

There are forty lessons taught during the semester. Three of the lessons teach problem solving and seventeen lessons cover programming in Java. These twenty lessons support the first objective described above. The remaining lessons support the second objective and cover hardware, software, networking, security, sensors, and emerging technologies. However, as discussed above, problem solving is emphasized throughout the course, not just during the programming lessons. The cadets are given two programming assignments that cover solution design, sequence, selection, and iteration. They are also given an assignment that requires them to design and create personal web pages using HTML. In addition to these assignments, the cadets complete two group projects: an IT project and a final project. The IT project requires a two-person team to design a networked information system that solves a problem given to them in a scenario. The team also has to research, and within a given budget, purchase the hardware and software required to implement their solution. The final project is a programming project in which the two-person teams' solutions are written using the Jago simulator and are then downloaded into a Lego robot. Both projects also require each team to conduct a formal presentation of their solution. The presentation includes a 6-8 minute briefing and a demonstration of their solutions using the Lego robot.

The Robot – Lego MindStorms

The robot used in our course is the Lego MindStorms Robotic Invention System. The MindStorms kit, which is available from major retailers for less than $200, consists of about 700 Lego pieces, a manual on building robots with the kit, and the software needed to program the robot. The central component of the kit is the RCX brick. The RCX brick, which is a Hitachi H8/300L processor, was inspired by research on programmable bricks done at Massachusetts Institute of Technology (MIT). The brick presents a constrained programming environment similar to that of embedded systems. The RCX has 16K of memory for its own firmware and 6K available for user programs. An infrared (IR) transceiver is also provided which provides the interface between the user's computer and the robot. Sensors that detect light and touch are included in the kit as well as motors, which provide for the robot's movement. The software that comes with the kit is very easy to use and also provides a tutorial. Because it isn't flexible enough for our needs, however, we created a Java-based environment, Jago, for the cadets to use for programming the robots.

The Environment – Jago

Jago (http://www.dean.usma.edu/dean/computingatwestpoint/Robots/index.htm) uses Java as the programming language, provides a graphical simulator, and allows the cadets to create and test their solutions on their own machines. This allows the cadets to design, code, and test their solutions without having to assemble a robot. When the cadets are confident their implementation solves the problem presented to them, they can come to the lab and download it into the actual robot to see if it solves the problem in a real-world environment. The cadets then demonstrate their solutions, using the robot, during their final project presentations at the end of the semester. The simulation capability in Jago also relieves us of the logistical and fiscal burden of issuing and maintaining more than 500 Lego kits.

The firmware that ships with the MindStorms robot does not support using Java as the programming language. We therefore had to find a replacement for the firmware that would allow the cadets to write their code the same way, and in the same language, as they did for assignments that did not involve robots.

We currently use LeJOS as our firmware and have had great success with it. LeJOS is available as open source and is freely downloadable at http://sourceforge.net/projects/lejos/. LeJOS allows us to load up to five programs at once and each program may be run repeatedly. Since the RCX has only 16K of space for the firmware, it can't support all of the Java libraries. However, it does contain enough to allow the cadets to write small, but complete, Java programs, including the ability to perform basic input and output.

Using Jago, a cadet could write the following Java program that has the robot go forward until it hits a wall, thereby activating the touch sensor, then turn around and head the other way until it hits a wall again. The program will continue until four walls have been hit.

```
import eecs.jago.*;
import eecs.jago.element.*;

class HitFour
{
        public static void main(String[] args)
        {
                Robot robbie = new Robot();
                SimBox sim = new SimBox();
                sim.add(robbie,135,400,0);
                robbie.setPower(10);

                for(int i = 0; i < 4; i++)
                {
                   while(robbie.getSensorValue(Sensor.TOUCH) == Sensor.OFF)
                   {
                         robo.goForward();
                    }
                    robo.pivotRight(180);
                }
        }
}
```

The Jago package also allows our instructors to customize and individualize the projects they give to their cadets. The simulator can simulate walls, obstacles, and additional “actors” as desired. The simulator also allows us to simulate the sensors attached to the robot. As illustrated above, the touch sensor can be polled and once it has been activated will return a value indicating such. We can also simulate colored light sensors for the MindStorms. Currently, we are able to sense red, green, blue, black, and white. The constraint we’ve imposed on our simulator, however, is that the physical robot must replicate those actions incorporated into the simulator. This constraint is necessary if we are to remain true to our goal of having the cadets see their implementation work in the physical world, not just in simulation.

Figure 1 below illustrates a project given to the cadets that required them to have the robot clear a route through an urban area. The robot can clear mines but is unable to maneuver through barbed wire, dead tanks, or buildings. To simulate this, we require the robot to use its light sensor to detect the colored line in front of the obstacles it cannot navigate around. The buildings are detected via the touch sensor. The robot’s objective is to find a path through the wall of obstacles, namely, the mines.

An alternate project is to navigate through a maze, as shown in figure 2. Both projects work well to reinforce the concepts of iteration and selection. Cadets need to use selection statements to make their robots take different actions based on input from the robot’s sensors. Using iteration, the cadets will have their robot repeat these actions until the obstacles are overcome and/or until the maze is completed.

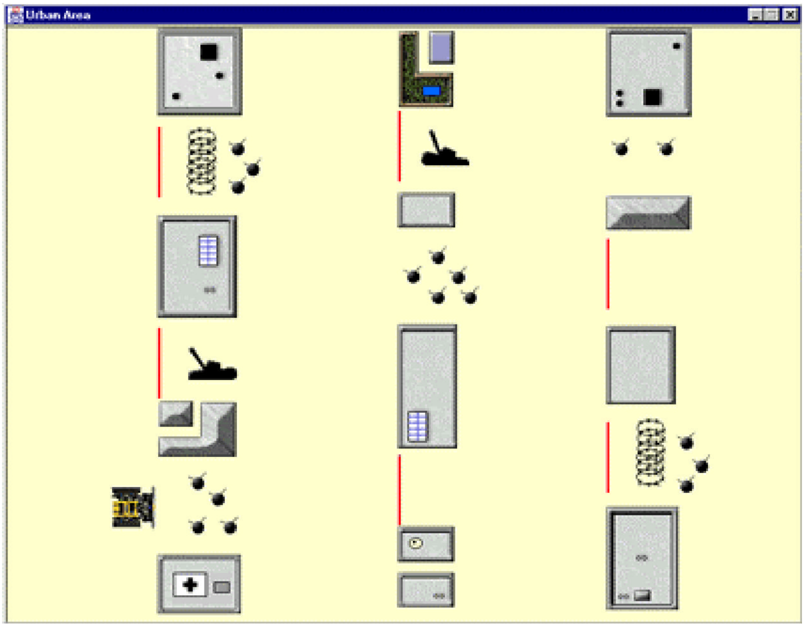


Figure 1: Jago Simulator (Mine Field)

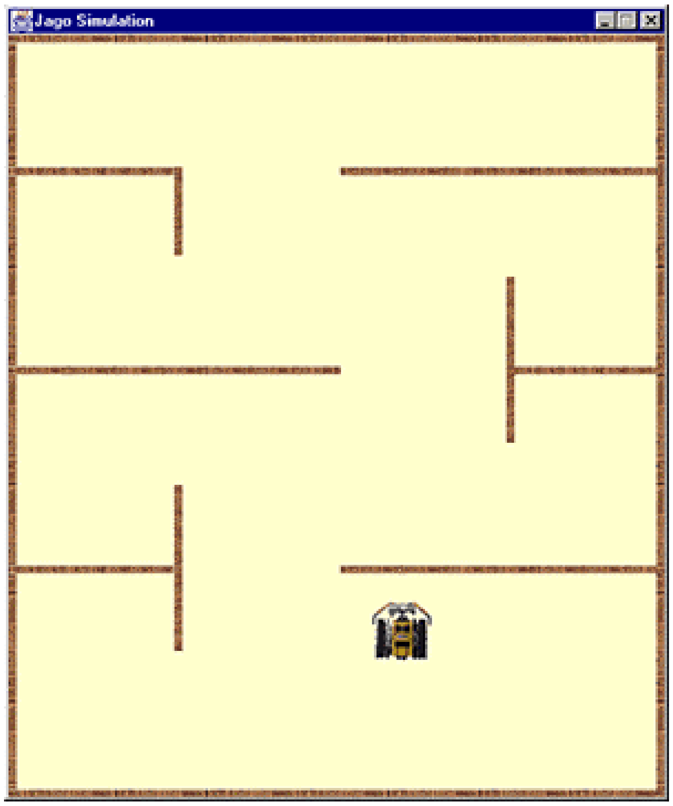


Figure 2: Jago Simulator (Maze)

In addition to usable programming environments and feedback of actual robots, carefully scoped assignments are important for effective teaching. Here is an example scenario the cadets are given and the assignment that goes with it.

a. Your Role

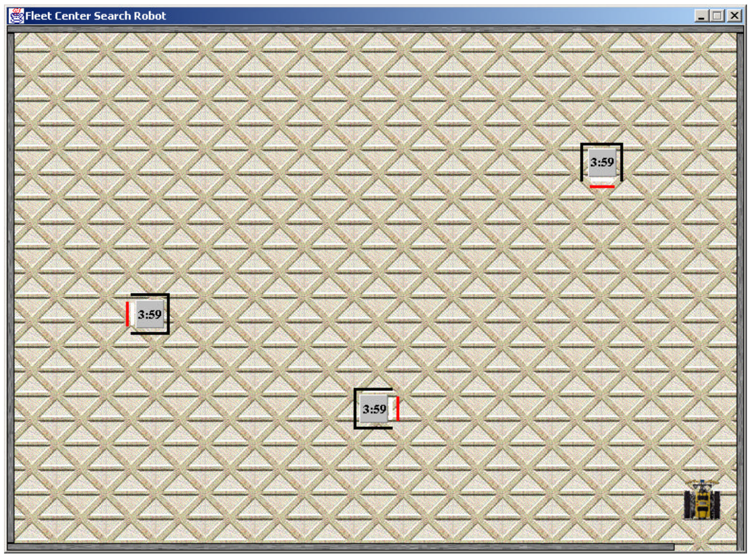
The Army has developed an inexpensive autonomous robot that can perform reconnaissance, surveillance, and minor manual tasks. You have been assigned as the mission engineer. You build the robots and write the programs that enable the robots to perform their assigned missions.

b. The Situation

A militant organization claims to have successfully placed detonator-triggered devices inside the air-conditioning ducts of the Fleet Center in Boston, with the intention of causing mass casualties during the upcoming Bruins and Flyers hockey game. Boston Police explosive experts have located the devices but don't have the technology to disarm them. The size of the crawl space is roughly 12'x 24'x 2'.

The experts believe the devices can be disarmed but to disarm them someone would need to depress a button that is located on the outside of the box. The police chief, however, doesn't feel comfortable sending a person down into the space to disarm the bombs, since he doesn't know if pushing the button will actually disarm the bomb.

The chief has asked you to work on a solution that uses a robot to disarm the bombs. To help the robot find and disarm the devices, a Special Forces reconnaissance expert actually entered the crawl space to mark each bomb with colored tape. Each device has black tape on three sides. The fourth side is marked with red tape. This is the side that has the button. The robot must depress this button in order to disarm the bomb.



c. Your Task

Your task is to design, build, test, and present a solution to the problem that demonstrates in the Jago simulator that the robot could successfully navigate its way through the crawl space under the following conditions:

(1) The robot must start in the lower right-hand corner of the crawl space, facing north (Because the robot will enter the crawl space through a small access panel on the right side of the southern wall).

(2) There are three detonators in the space that must be disarmed.

(3) The robot must disarm all three detonators in four minutes or less.

Given the assignment above, the cadets must then follow a disciplined problem solving methodology as part of the assignment. The cadets will analyze the problem to identify the objectives, the outputs, the inputs, the assumptions, the constraints, and any formulae required to solve the problem. The cadets produce a written document called a problem specification that articulates their findings during the problem analysis phase.

The cadets then design a solution that addresses the requirements identified during the problem analysis. We teach our cadets to use both written algorithms and flowcharts to design their solutions. We try not to focus on the mechanics of writing an algorithm or drawing a flowchart when evaluating their design. Instead, we focus on the quality of the design. Is it complete? Does it address the requirements identified during the problem analysis? Does it solve the problem?

The cadets then implement the design using Java and Jago. Finally, they test their solution to ensure it satisfies the requirements identified during the problem analysis. A sample cadet solution to the problem follows.

```
import javabook2.*;
import eecs.jago.*;
import eecs.jago.element.*;

class FleetProgram
{
        public static void main(String[] args)
        {
                Robot robo = new Robot();

                boolean randomLayout = true;
                FleetCenterSimulation sim = new FleetCenterSimulation(randomLayout);

                MessageBox msg = new MessageBox(sim);
```

```
sim.add(robo,710,500,0);
sim.startDetonators();

robo.setPower(10);

boolean north = true;
int numDisarmed = 0;

while(numDisarmed<3)
{

        while(robo.getSensorValue(Sensor.TOUCH) == Sensor.OFF
                && robo.getSensorValue(Sensor.LIGHT) == Sensor.OFF)
        {
                robo.goForward();
        }  //end while sensors off

if(robo.getSensorValue(Sensor.TOUCH) == Sensor.ON)
{
  if(north)
  {
   robo.goBackward(200);
   robo.pivotLeft(90);
   robo.goForward(500);
   robo.pivotLeft(90);
  }//end if north

  else
  {
   robo.goBackward(200);
   robo.pivotRight(90);
   robo.goForward(500);
   robo.pivotRight(90);
  }//end else
   north = !north;
}// end if
else if (robo.getSensorValue(Sensor.LIGHT) == Reflective.BLACK ||
        robo.getSensorValue(Sensor.LIGHT) == Reflective.RED)
 {
   robo.goForward(100);
   robo.goBackward(300);
   robo.pivotRight(90);
   robo.goForward(450);
   robo.pivotLeft(90);
   robo.goForward(700);
```

```
                robo.pivotLeft(90);

                //move to touch second side
                robo.goForward(500);
                robo.goBackward(300);
                robo.pivotRight(90);
                robo.goForward(550);
                robo.pivotLeft(90);
                robo.goForward(700);
                robo.pivotLeft(90);

                //move to touch third side
                robo.goForward(400);
                robo.goBackward(300);
                robo.pivotRight(90);
                robo.goForward(650);
                robo.pivotLeft(90);
                robo.goForward(700);
                robo.pivotLeft(90);

                //move to touch fourth side
                robo.goForward(400);
                robo.goBackward(300);
                robo.pivotLeft(90);
                robo.goForward(650);
                robo.pivotRight(90);
                robo.goForward(650);
                robo.pivotLeft(90);
                numDisarmed ++;
                }//end else if

               }//end while < 3

           if (numDisarmed == 3)
           {
             robo.pivotRight(360);
           } // end if == 3

        }  //end main
} //end FleetProgram
```

Assignment Feedback

Two-person teams were given the above assignment and the teams provided us very positive feedback. Though it was still "homework" to them, they were motivated to see if their implementation would actually work in a “real” robot. The use of real-world scenarios, and the fact that they would have to demo their project in front of their peers motivates the cadets to spend much more time in the design and problem-solving phases than they do in the code writing portion; a desired behavior. The final project instills in our cadets an understanding of how to use computers and information technology to solve problems. This is exactly what we strive for in this course. Learning the syntax and nuances of a programming language should not be obstacles to the learning experience. We want the cadets to leave our course as problem solvers, not necessarily programmers. Even those cadets that were unable to completely solve the problem enjoyed the project and learned from it. Though their solution may not have worked, during the demonstration of their solution they could clearly explain what was wrong with their algorithm and how they could correct it. Academically, we consider these projects to be successful learning experiences. The robot allowed the cadets to follow their algorithm visually as the robot executed it and enabled them to identify their errors. Additionally, if the robot solved the problem but not within the time constraints given, the cadets were able to explain how they could modify their algorithms to improve the robot’s performance.

Results

The Lego MindStorms robots and Jago have been used in our course for two years now. The results from their use have been very positive. Jago and the robots help the cadets to visualize their problem and the inherent problems associated with it. This visualization causes them to think more deeply about the problem and its solution. We have observed cadets acting as the robot and walking through their algorithms to ensure they do what they think they do. This helps them to see their errors in sequence, selection, iteration, and logic. This doesn’t happen on other non-robot assignments.

Future work

We plan to improve the simulator by making it behave more like the real world. Distances are currently not to scale, and the robot moves faster in the simulator than in real life. For now, the simulator is best for problems involving only one robot but we hope to incorporate interaction between multiple robots. Additional enhancements include the integration of such technologies as lasers, sensors, and IR communications. We are working with the Civil and Mechanical Engineering Department to develop lessons that incorporate sensor technology into the curriculum and have designed a light sensor for the robot that the robot can use to follow or avoid a light source. Students will build the sensor in class and then write the code that uses the light sensor to solve a problem like the “cockroach problem” where the robot will seek out the darkest area of a room or have the robot move toward a light source.