School of Computing
National University of Singapore
CS1101 Programming Methodology
Semester 2, AY 2003/04

Lab Assignment #8
Programming the Lego Robot

Issued: Mar. 24, 2004
Due: in PC Lab 2 at 1pm on Apr. 5, 2004 (Judgment Day)[1]

1. Introduction

Robots have always fascinated people. It immediately evokes feelings of fear and marvel. Robotics has always been perceived as complicated and expensive. Not any more. We can now design, build and program simple robots and have fun with them. This lab assignment is about programming the Lego Mindstorm™ robot.

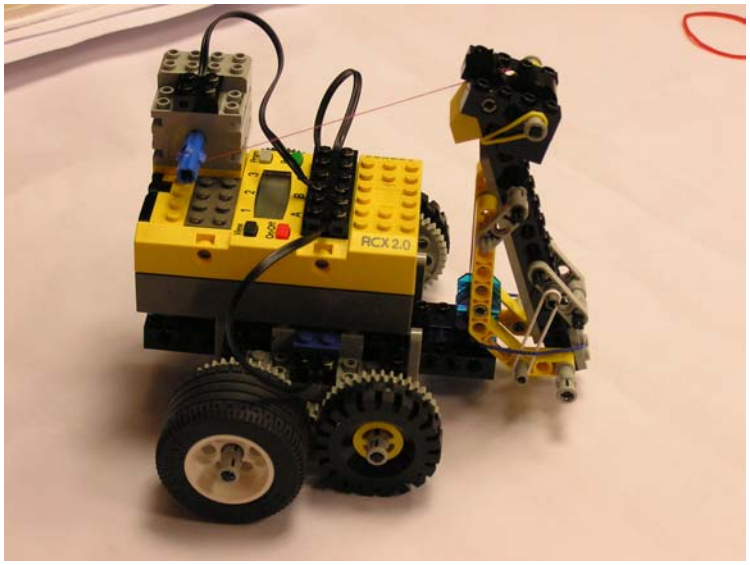


Figure 1: The Lego Robot

The main emphasis of this lab assignment is on the Java program that controls the robot, rather than the hardware. Therefore, you are encouraged to assemble the robot following the instructions given here. You do not have to design your own robot. However, if you do create a robot of novel design, you can get up to 10 bonus marks.

[1] Borrowing the terminology from the robot movie Terminator 2.

The goal of this assignment is to control the robot to write words on the floor. More specifically, you will program your robot (with a pen attached) to draw a sequence of characters. This is accomplished by combining the primitive operations that the robot can perform: move forwards/backwards, turn, raise and lower the pen. You will be assessed on how well your robot accomplishes the Three Tasks, rather than on your code *per se*. Nevertheless, you still need to submit a hardcopy your code.

Please read this assignment completely before starting. The deadline for this lab is as indicated above. **No late submission will be allowed**, because everything will be judged on the same day.

2. What to do

1. Work in groups of 2. Find a partner and inform the lecturer by email of your group members. Give your group a name.

2. Check out a robot kit from Mar. 25 onwards, from Mr. Chong (chongpk@comp.nus.edu.sg) at the Graphics Lab (S16 #09-01). Be sure that all the important parts are accounted for. You will be given the following:

   (a) 1 Lego Mindstorm box plus a plastic bag containing the important parts.
   (b) 1 set of 6 AA batteries
   (c) 5 sheets of mahjong paper
   (d) 1 marker pen

3. Work on this assignment.

4. On Judgment Day, bring your robot to PC Lab 2 for it to be assessed. You will be asked to demonstrate the capabilities of your robot to accomplish the Three Tasks. Once you turn on your robot and let it run, there should be no manual intervention until the robot finishes, or aborts.

5. Also submit a hardcopy of your code, together with a brief 1-page write-up of what you did. You can mention any difficulties you encountered, or any assumptions or simplifications you made.

6. After Judgment Day, disassemble your robot and return it to Mr. Chong. Be sure to account for all the important parts. **If things go missing, you may be asked to pay for them.**

Have Fun!

3. How to Assemble the Robot

This is the suggested design for the robot. You are encouraged to use this design because the goal of this lab assignment is on the software control, rather than on hardware. Nevertheless, feel free to create your own design. You can earn up to 10 bonus marks if your own design works and is creative.

The robot is made up of 3 portions: the **driving base**, the **pen holding arm** and the **arm lifting mechanism**.

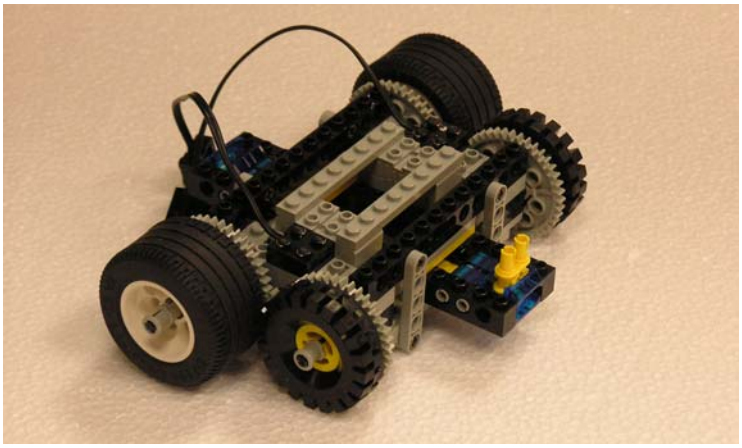


Figure 2: The driving base

The instructions for the construction of the driving base can be found in the Lego Mindstorm manual. Please refer to the building instructions for the construction from pages 12 to 17. Note that you will need 2 motors to control the wheels: one for the left, the other for the right.

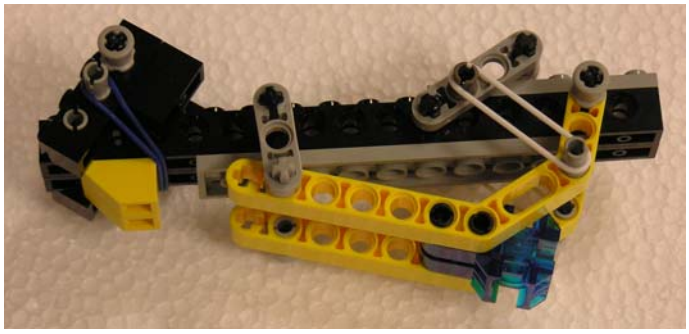


Figure 3: The pen holding arm

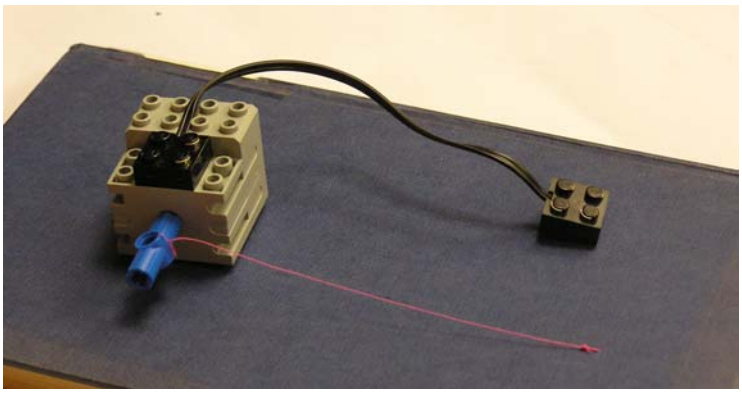


Figure 4: Arm lifting mechanism

To build the pen holding arm and the lifting mechanism, refer to the pictures provided to guide you along. You will need a third motor to raise and lower the arm. Actually, the arm is constantly in a "down" position, because a rubber band holds it there. The motor is used to raise the arm to lift the pen off the ground. Note that it is not necessary to follow exactly the same design. If you can devise a better arm, feel free to do so.

Figure 5: Attaching the pen holding arm to the driving base

For a close up view of the various pictures in this instruction, please visit http://www.comp.nus.edu.sg/~chiangim/lego.htm

Alternatively, you can approach the TAs to view the model they have constructed for reference.

4. How to Program the Robot

4.1. Initial Setup

You control the robot using a Java program. That is, you write your Java code on a PC, compile it, and then transfer the compiled code (also called bytecode) to the robot. More precisely, you compile your program using Lejos (instead of javac) and transfer the bytecode using the infrared (IR) tower to the **RCX**. This is the yellow brick that houses a Hitachi H8/300 16Mhz CPU and 28Kb of RAM.

For this to happen, you need to perform a one-time setup for the following: (a) the IR tower on your PC, (b) Lejos installation on your PC and the RCX. For detailed instructions on how to do this, please see this web page:

http://www.comp.nus.edu.sg/~kanmy/courses/3243_2004/lejosSetup.html.

Alternatively, you may use any of four PCs in the lab that have already been configured. These are: Soctf-pc2-037, Soctf-pc2-038, Soctf-pc2-039 and Soctf-pc2-040 in PC Lab 2, SOC1 Level 8.

4.2. The Lejos API

To program the robot, you need to use the classes and methods provided by Lejos. An introduction to Lejos is available here: http://lejos.sourceforge.net/apidocs/index.html.

Basically, the Lejos API is a smaller version of the Java API with some specialized classes. The **`Integer`** class for example has only a **`toString()`** method. Hence when you try to use some of the classes provided by Java and Lejos such as **`Integer`**, please check that the methods are indeed supported in Lejos.

There are many classes provided by Lejos, but we will focus only on 2 of them: **`TimingNavigator`** and **`Motor`**.

The RCX can give 3 different commands to 3 motors, as labeled on the RCX: Motor A, Motor B and Motor C. The 4 commands that are useful in the control of the motors are **`forward()`**, **`backward()`**, **`stop()`** and **`setPower(int aPower)`**.

The **`TimingNavigator`** helps you to navigate your robot. So instead of controlling each motor individually, you can assign a **`TimingNavigator`** to control the 2 motors that move your robot around. The **`TimingNavigator`** offers a variety of methods in which you can do that. Please refer to the Lejos API for more details.

The following is an example program "**write.java**".

```
import josx.platform.rcx.Motor;
import josx.robotics.TimingNavigator;

public class write {
      static TimingNavigator TM;
      public static void main(String[] aArg) throws Exception {
          Motor.A.setPower(1);
          Motor.C.setPower(1);
          TM = new TimingNavigator(Motor.A, Motor.C, 120, 30);

          TM.travel(2);
          TM.rotate(7);
          TM.travel(-2);
      }
}
```

This program moves the robot 2 units distance forward, rotates 7 units in the clockwise direction and 2 units distance backwards. How long is 1 unit? How many degrees are 7 units? This you have to test for yourself. As the name **TimingNavigator** suggests, it calculates the distance or angle it moves by the length of time it takes. A heavier robot may take a longer time to turn 10 degrees as compared to a lighter robot. Similarly for the distance it travels. You can adjust the turning and the travel distance in 2 ways, either you modify the **timeOneMeter** parameter and the **timeRotate** parameter when you construct a **TimingNavigator**, or you can adjust the power of each motor using **setPower** method.

A hint on how you can go about adjusting these parameters: Get your robot to draw one full circle and draw a straight line. Because no two motors or wheels are exactly the same, each will spin at a different rate. This will cause your robot to veer off a straight line. You will need some trial and error to get your robot to draw straight.

Note that the raising and lowering of the pen/arm is done by a motor. Which means your Java program should activate the motor accordingly. After you have written your code, compile it using the Lejos compiler (instead of javac), type this command in the Windows command prompt:

```
lejosc  write.java
```

This creates a **write.class**. To transfer the program to the RCX, make sure your RCX is turned on, and the RCX receiver faces the IR tower, type this command:

```
lejos write
```

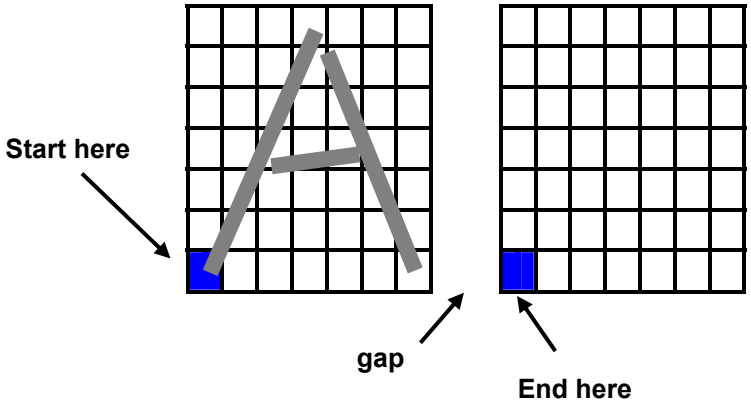
When the transfer is complete, the command prompt shows `100%` and the RCX will sound a beep. For more detailed information on how compile and transfer program using one of the terminals in PC Lab2, please read the "Step-by-Step.pdf". You can now turn on your RCX and press the "Run" button to execute your program.

5. The Three Tasks

5.1. Task 1: [10 marks for each letter]

The Java code given to you already contains code to write two letters: "E" and "A". You are required to write code to make the robot write three more letters: "M", "C", and "T". These should be in capital letters (uppercase). Your "C" may be squarish instead of curved, that is: [

One helpful approach is the following: Imagine that each letter is enclosed in a grid. Your robot should begin at the bottom left corner of the grid, and end at the bottom left corner of the next grid. There should be a gap between the two grids, as shown below. This is the **drawing protocol**. You should observe this protocol to make your task easier. You are free to determine the size of the grid. There is no need to be exact, as long as all letters drawn are about the same size. There is even no need for a grid, as long as your robot writes the letters from left to right, leaving a space in between.



5.2. Task 2: [10 marks]

Your robot now as a **Vocabulary** of 5 characters (letters): "E", "A", "M", "C", "T".

In this task, you will program your robot to write a string of your own choosing, subject to the following constraints: (a) Your string should be at least 3 characters long, where each character is chosen from the Vocabulary. (b) Each character should be disjoint from other characters, that is, your robot should lift the pen between characters. (c) At least 2 of the 3 characters must be different. Your string need not be a valid English word. Any sequence of 3 characters will do.

5.3. Task 3: [20 marks]

This is an on-the-spot task: on Judgment Day, we will ask your robot to write a new character, expanding its Vocabulary. For example, your robot may be asked to write the letter "K". You will be given 10 minutes to re-program your robot, and you must complete it within this time limit, failing which you will be disqualified. Please plan ahead for this task. *Hint*: what primitives are needed? Write helper methods that may be useful for this task.

6. Additional Tips

1. Start early!

2. The communication between your PC and the robot is via the Infrared (IR) tower. This may be sensitive to surrounding light. Do not work near a window, and try to shield your robot from fluorescent lights.

3. You are **strongly advised** to test your robot in PC Lab 2, where it will be judged. Robots are known to be finicky, and although it works at your home, it may not work in the lab.

4. Your robot will not be judged on aesthetics, so do not waste time getting your robot to write in beautiful cursive font. However, what is written must be legible, and not some random sequence of lines and dots.