ROBOLAB Tutorial
MAE 1170, Fall 2009

(I) Starting Out

We will be using ROBOLAB 2.5, a GUI-based programming system, to program robots built using the Lego Mindstorms Kit. The brain of the robot is a microprocessor called the "RCX" which accepts programs through an infrared (IR) transmitter. This transmitter should be attached to your computer.

1) Get a yellow RCX controller and turn it on. Notice the 6 ports on the device:

Ports: 1, 2, 3 are for attaching sensors

Ports: A, B, C are for attaching motors and lamps

These attach to the RCX using the black cables in your kit.

2) Examine the IR transmitter. By placing the transmitter a few inches in front of the RCX's "infrared eye," a program can be uploaded wirelessly from the computer.

3) Using the VIEW button, you can directly see values of the sensors connected to the RCX. The PROGRAM button allows you to cycle through the 5 program slots available on the RCX. The RUN button starts/stops the program.

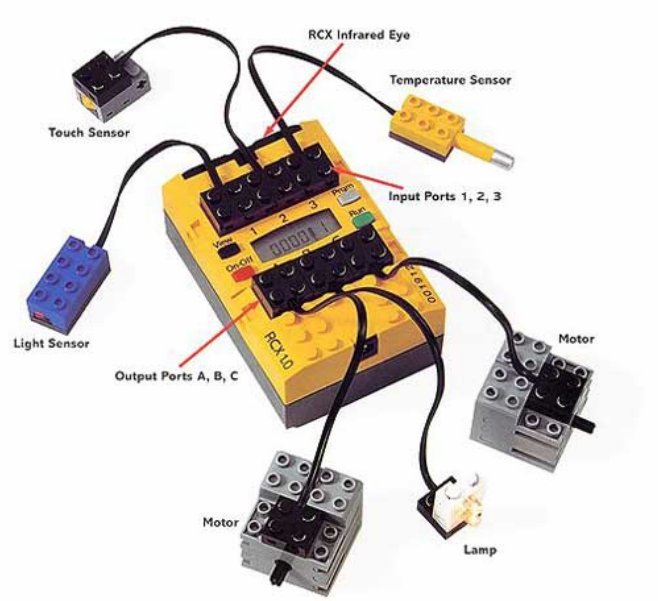


Figure 1: RCS controller with motors, lamp, and sensors

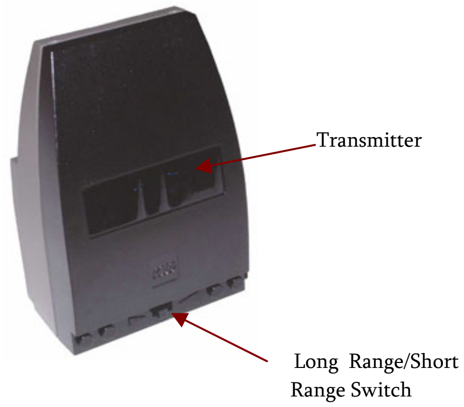


Figure 2: Infrared Transmitter

4) Now, start the ROBOLAB program:

a) Click on ***ROBOLAB 2.5*** under the PROGRAMS → ROBOLAB menu on the Start button.
b) Click on the PROGRAMMER button
c) **Double-click** on INVENTOR 4.

5) A large graphics window opens up, and you will see:

a) A ***green stoplight*** and a ***red stoplight*** -- this is the DIAGRAM window
b) A ***functions*** palette with lots of controls
c) Another window on top of the graphics window with instructions on how to return to the main menu.

The ***green light*** graphically represents the start of the program, and the ***red light*** graphically represents the end of the program.

6) Move the mouse to the DIAGRAM window.

By pushing the TAB button, you should be able to get four different mouse pointers:

- ***Arrow***: use to click on objects and highlight them – (i.e., for deleting use the arrow to highlight an object and press the DEL key to remove it). It can also be used to drag objects around the screen
- ***Hand***: double click on an object for information on the object.
- ***Spool***: use to wire things together
- ***Text***: use to add text boxes and labels
- ***Right Click*** on an object to use for other options, such as getting detailed help or replacing it directly with another object.

7) Let’s take a closer look at the ***functions*** palette:

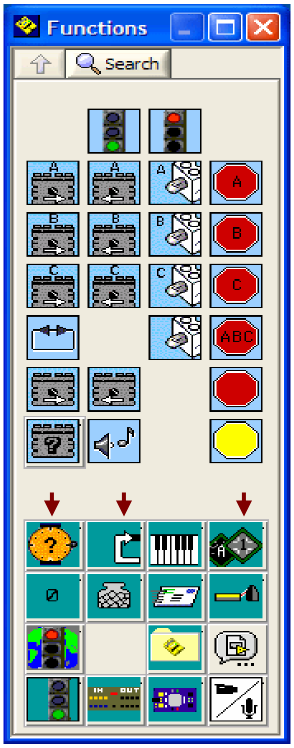


Figure 3:
Functions Palette

You can tell what a control does by hovering the mouse over the given control.

The motor controls turn motors on in the respective direction. Use STOP signs to turn motors off (note, without stop signs, the motors will continue to run after the program has terminated).

The bottom half of the *functions palette* contains links to additional menus containing additional controls such as **Wait For, Structures,** and **Modifiers.**

Wait For: Can pause execution for specified amount of time or until a certain even occurs (such as a sensor being triggered).

Structures: Allows you to split tasks, create subroutines, or generate loops within your program.

Modifiers: Modifies either the “where” or the “what” for program objects. For example, use them to specify which light sensor port to use or the motor power setting.

8) Check out the **arrow** in the top left of the DIAGRAM window. If the arrow is broken, your program cannot run and you can click the arrow to list the errors. Once the arrow is solid, you can click it to transfer the program to the **RCX.**

(II) Examples

9) Let’s start by building a simple program.

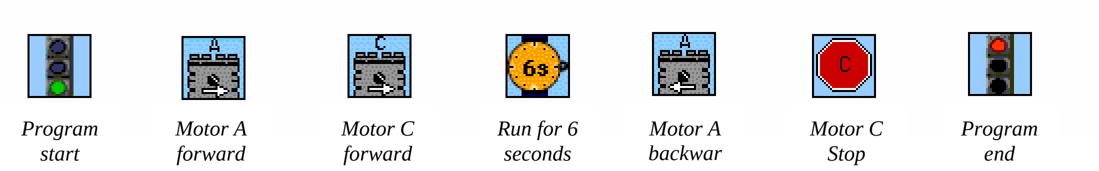
a) Press **TAB** until you get the arrow tool.

b) Drag a “motor A forward” and “motor C forward” object to the DIAGRAM window.

c) Now, go to the **Wait For** window and pull in a “***wait for 6 seconds***” object.

d) Then, drag in a “motor A reverse” and “motor C stop sign.”

Right now, your program should look something like this:

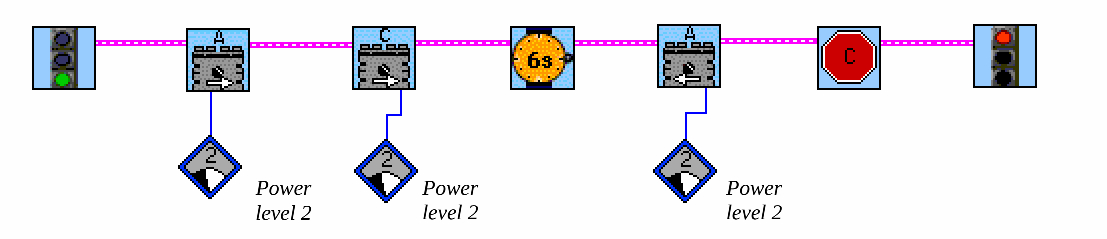


e) Use the ***spool*** tool to run a wire from the *END* of the *green light* to the BEGIN of the "motor A forward" object.

f) Similarly, wire everything in the sequence together until the ***red light.***

g) Go to the **Modifier** window and drag the "power level 2" object next to all the motor objects.

h) Wire the modifier object onto the appropriate slot on the motor using the ***spool*** tool.



The arrow in the top left of the diagram window should now be solid.

g) Set up the **RCX** and **IR** transmitter correctly and click the arrow to transmit the program.

Ask a TA for help if you experience problems with this step.

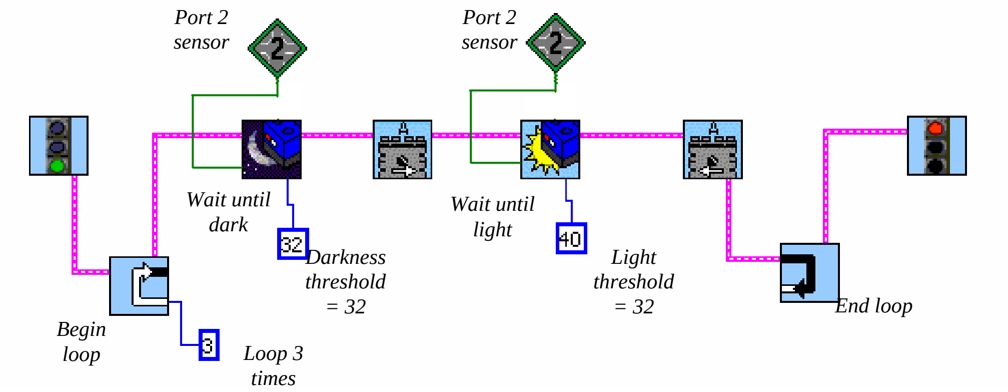
Note, you can only download programs into slots 3, 4, and 5 on the RCX.

Try out the program! Does it do what you think it should?

10) Next, we will up the complexity a bit and familiarize you with the basic tools you will need to do the lab.

a) Delete spool lines in your current program by selecting them with the ***arrow*** and hitting delete.

b) Create and transfer the program below. Objects you will need are the "loop" icons in the **Structures** window and the "wait for dark" and "wait for light" objects in the **Wait For** window. The "numerical constant modifier" can be found in the Modifier window, and can be edited using the ***text*** tool. Make sure the program works correctly and explain each element to a TA before moving to the next step.

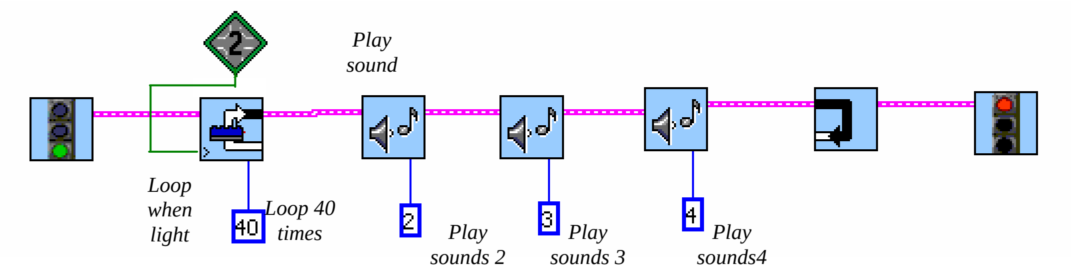


11) As a final example, build the simple looping program pictured below.

a) Find the "loop while light sensor is less than" loop in the **Structures** window.

b) Add the sound objects from the main ***functions*** palette and modify them to the sounds of your choosing.

You should find that your RCX makes noises whenever the light sensor senses a white surface, and this behavior repeats because of the loop.



(III) Programming Strategy

Using a combination of the tools, you should have no problem programming a robot to follow a black line. Instead of diving directly into programming, use this general strategy:

1) **Identify** the key actions you want your robot to perform.

2) Write "**pseudo-code"** in English on a separate sheet to specify the steps of your program. An example of pseudo-code would be the following:

- Motors A and C on (power level 3)
- Wait till touch sensor is pressed (port 2)
- Reverse Motor A
- Turn off motor C

3) **Execute** and build the program in ROBOLAB using your pseudo-code as an outline.

4) **Troubleshoot!** Almost no programs work correctly on the first try. Analyze what is happening (Is the robot overshooting? Is one motor overpowering the other?) and iteratively perfect your algorithm and/or robot.