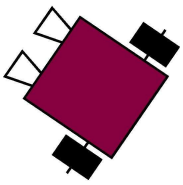


Brooklyn College The City University of New York

Exploring Robotics (CORC 3303)

UNIT D Lab, part 2 : Obstacle Avoidance

vocabulary

- touch sensor
- bumper
- fork
- branch
- jump
- lamp

materials

Make sure you have all of the following materials before you start the lab:

- lego go-bot
- lego touch sensor

instructions

1. **add a bumper and touch sensor to the go-bot**

   - Follow the **bumper sensor add-on instructions**, and work with your partner to add the bumper and touch sensor to your go-bot (as in the instructions).
   - Add the **bumper sensor** to **port 1** of the RCX using the **connecting lead** :

touch sensor

2. **add a lamp to the go-bot**

   - Add a lamp to your robot. Connect it to output **port B**. You can plug it in directly or you can use a connecting lead.

lamp

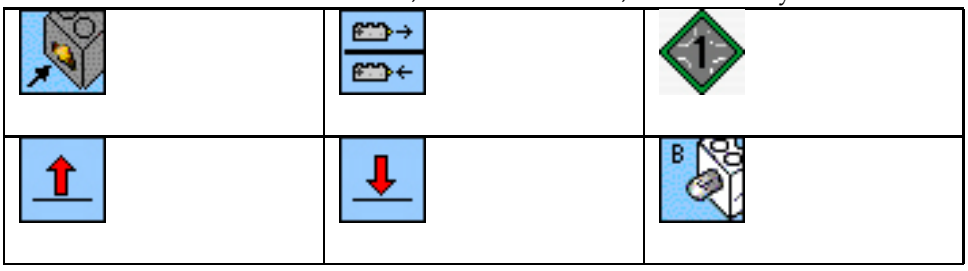
3. **start up RoboLab**

   - Find the **RoboLab** icon on your computer and double-click on it to start it up.
     Click on **PROGRAMMER**.
     Double-click on **INVENTOR 4**.
     Your screen will look like this:

   - Now you are ready to start programing in **RoboLab**!

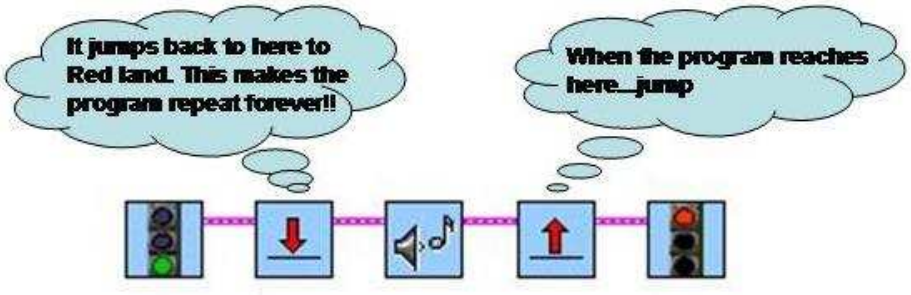
4. **find the icons**

   - Locate the **functions palette**. See if you can find each of the following **icons**. What do you think each icon does? In the boxes below, next to each icon, write down your ideas.



5. jumps

- Unlike the **loop**  , which repeats for a certain number of times based on a **counter** or a **condition**, a **jump** repeats forever. This is also called an **infinite loop**.



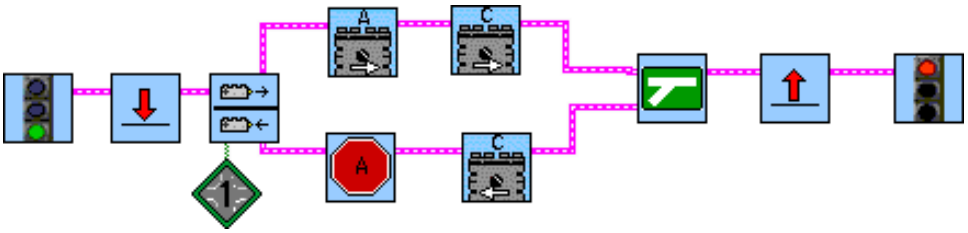
- In RoboLab, a **jump** is created by using a **land**  icon in place of a **start-of-loop** icon and a **jump**  icon in place of an **end-of-loop** icon.
- The **jump** and **land** icons come in several colors: red, blue, yellow, green and black. Always make sure that your colors match up! In other words, if you use a yellow land, you must also use a yellow jump to go with it.
- Note that with a **jump**, you do not have to include a modifier telling it how many times to repeat.
- However, it is recommended to use jumps carefully since your program will never stop! (until you turn off the RCX).
- Make sure you put the **land**  icon FIRST, BEFORE the the **jump**  icon. Otherwise, strange things may happen!

6. touch sensor fork

- A **touch sensor fork**  works similarly to a **light sensor fork**  —it helps the robot make decisions about how to behave based on input from one of its sensors.

7. create your program

- Use the mouse to select, drag and drop RoboLab **icons** onto your canvas and **wire** them together, creating the following program:



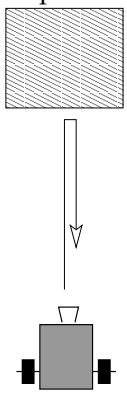
- What do you think it does? Write down your ideas below:

- Download and test your program.
  - If you have trouble, check these things:
    1. The RCX must be turned on during the download.
    2. The black part of the RCX should be turned towards your tower.
    3. If there is a lot of light in the room, try covering your RCX and the tower with a piece of paper.
    4. If your download arrow is broken, then you have an error in your program. Make sure that you copied all the icons correctly and that the wires all start and end in the right places.
    5. Make sure that the **bumper sensor** is connected to **port 1**.
- What did your robot do? Is that what you expected? Write your answer below.

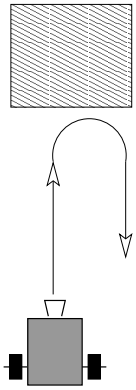
---

8. obstacle avoidance

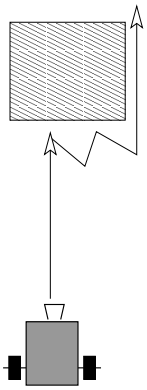
- One of the classic tasks that mobile robots perform is known as **obstacle avoidance**. Basically, this is a behavior in which robots avoid bumping into things they don't want to bump in to, in other words, obstacles.
- The way that robots do this is actually by bumping into obstacles to discover that they are there, and then backing up and/or turning around to avoid them.
- Some examples are shown below:



(a) robot backs up when it hits the obstacle



(b) robot turns around when it hits the obstacle

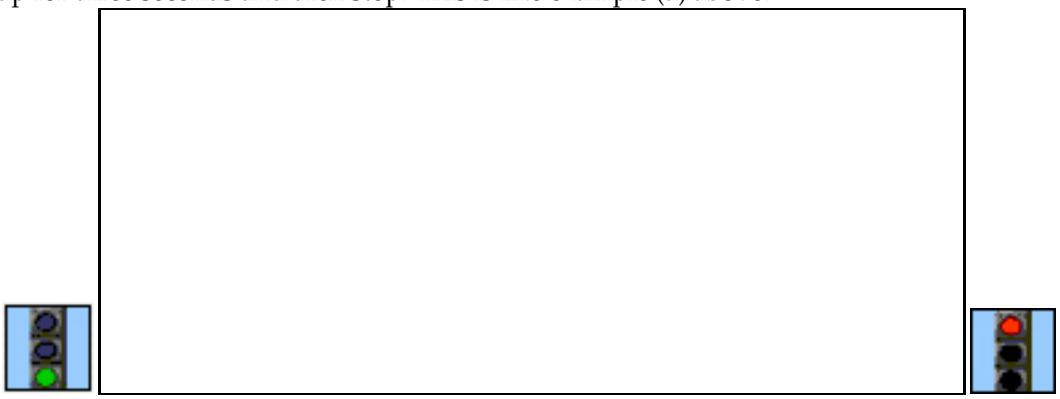


(c) robot backs up and edges its way around the obstacle

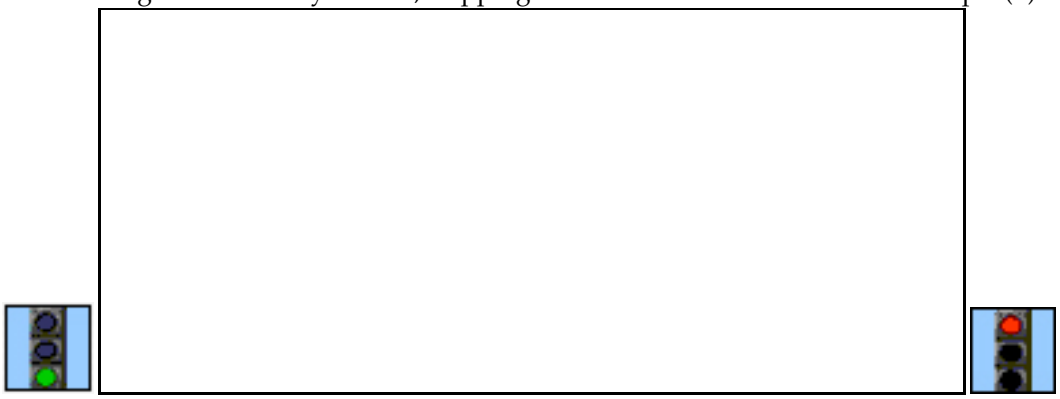
9. programming challenges

- Complete as many of the following programs as you can.
- After you get each program to work, draw the code in the boxes provided on the next page.

(a) Program the robot to go forward until it bumps into something, and then turn on the lamp, back up for three seconds and then stop. This is like example (a) above.



(b) Program the robot to go forward until it bumps into something, and then turn on the lamp, turn around and go back the way it came, stopping after three seconds. This is like example (b) above.



(c) Program the robot to go forward until it bumps into something, and then back up for one second and go forward again until it bumps into something, then back up and go forward again—repeating this behavior forever. This is like example (c) above.