

Mentor sheet

Week 6

Line-Following, 2 sensors

1. Rescuing the victim; strategy and programming
2. Navigating the shortcuts; strategy and programming
3. Example code

Equipment needed:

2 light sensors attached to the front of a robot that will straddle a black line 15 mm wide.

1 touch sensor and/or distance sensor

The lesson plan maps directly into the student worksheet. Make sure that students follow along.

1. Rescuing the victim

Strategy

As a team, have your students answer the following questions:

- How will my robot know it's time to look for the victim?
- How will the robot look for the victim?
- How will the robot notify that it has located its victim?
- How will the robot remove the victim from the chemical spill?

Each directly relates to the final program students will write.

How will the robot know it's time to look for the victim?

Have your students look at the chemical spill tile in comparison to the other tiles. How is it different? (Ans. No line, green) How can the robot tell it's on the chemical spill?

(Ans. Both light sensors read green)

REMEMBER!!! – Light sensors read a value of 0-100, where 0 is darkest and 100 is the brightest. Green will fall in between the values your light sensors read for black and white.

When both sensors read green, this will be your trigger to start looking for the victim

How will the robot look for the victim?

The answer to this question depends on the configuration of the robot. Is a touch sensor used? Is a distance sensor used? Do you use both?

If just a touch sensor is used, the robot should do some sort of back and forth movement over the spill so that the robot moves in search of the victim until the touch sensor is depressed. If your robot utilizes a distance sensor **remember to use it in Port 1!** A robot with a distance sensor should scan the spill area until it sees something closer than a certain distance away and move towards it to push it out of the chemical spill.

How will the robot notify that it has located its victim?

Your robot should either light up or play a sound when it has found its victim.

How will you rescue the victim?

For this challenge, it is sufficient to push the victim out of the green. No claws or arms are necessary.

Programming

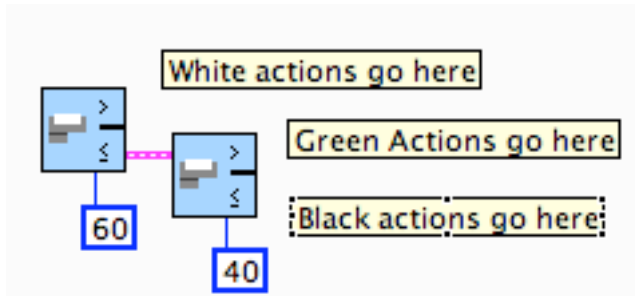
How will the robot know it's time to look for the victim?

Since you will be using a simple light sensor instead of a color sensor, you will need to differentiate between white, black, and green. This can be simply accomplished using forks or nested forks. As always, the numbers are made up.

White = 70

Green = 50

Black = 30



You will need to set up a system of forks within your existing program to figure out if both sensors are reading green. This should be relatively straightforward for teams who wrote programs using forks or events. To change an existing nested forks program, use the same methodology to account for the new colors.

If you're using events, you have several options. You can replace your "Motor A forward, Motor B Forward" with a set of forks accounting for white and green. You could also look for green within your event actions.

If you used task splits, you will have to make good use of the "stop tasks" and "start tasks" icons in addition to forks.

How will the robot look for the victim?

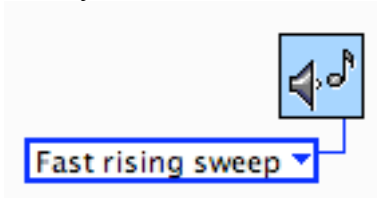
The only way to master this is through trial and error. I think the easiest/most reliable method is using a touch sensor. You may want to talk to your students about design and how a large touch sensor bar could help/hurt your chances at rescuing a victim. Again, events will be easy to add to. Your group may want to make extensive uses of jumps and lands.

How will the robot notify that it has located its victim?

Light up...



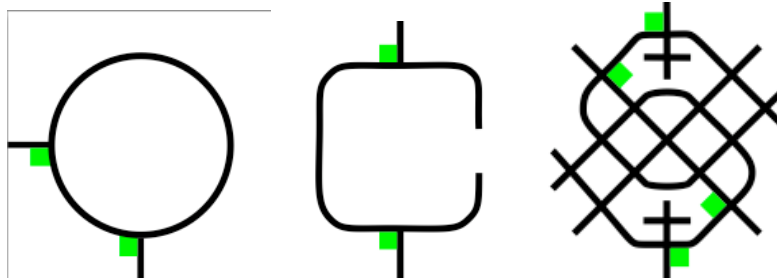
or Play a sound!



How will you rescue the victim?

Just push them! Ok, so I'm kidding a little bit. It may be helpful to make sure that your touch sensor is depressed the entire time that your robot is supposed to be pushing the victim out of the chemical spill. If you notice that it isn't, you could have an escape sequence to look for the victim again. Or you could just push them.

2. Navigating the Shortcuts



These are the 3 “shortcut tiles” that may appear in the competition.

How does the robot know when it has reached one of these tiles? This occurs when one light sensor reads green while the other does not. You set this up using forks similar to the ones outlined in the previous section.

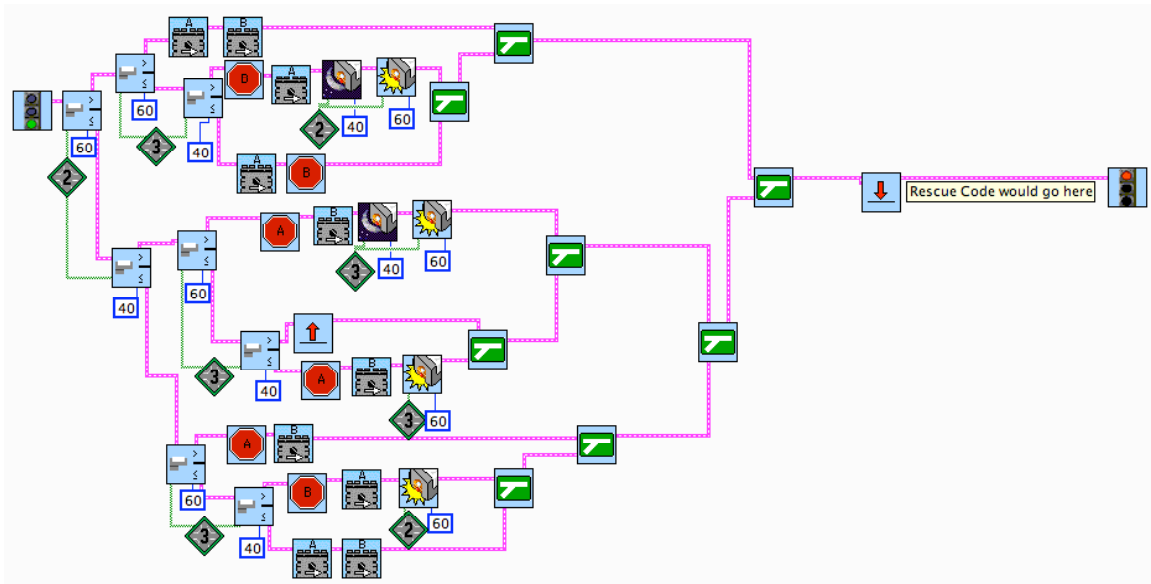
Strategies

The first two are relatively straightforward. When 1 sensor reads green and the other does not, you can either temporarily switch your robot to a single line follower or make sure the “non-shortcut side” crosses the line to resume following with 2 light sensors. I will leave the “gridlock tile” for more ambitious groups to tackle.

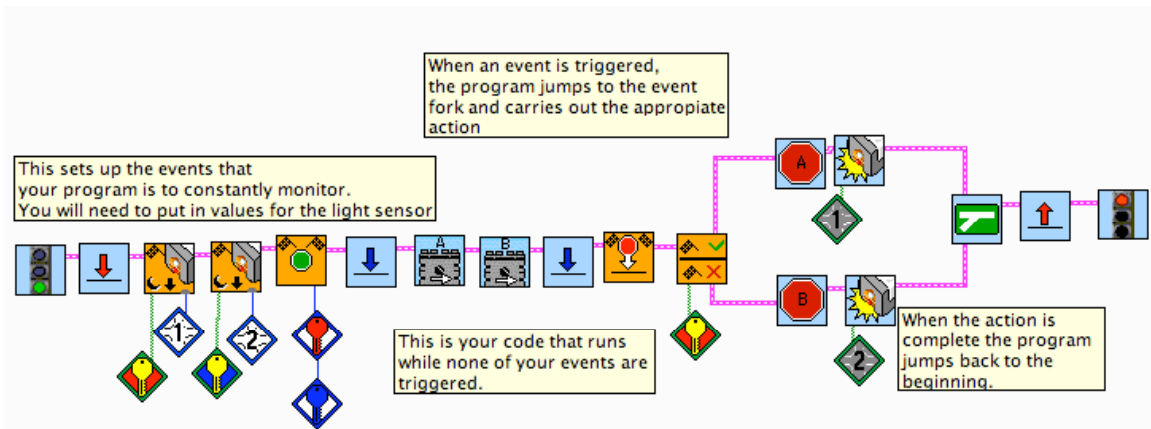
3. Sample Code

Forks

This long, complicated code isn't really long and complicated at all. It may help your students figure out when and how things need to happen.



If your students understand forks, it may be easiest to incorporate them into a program that also uses events. A basic two line-following event program follows.



The same concept is possible using task splits, but you will have to know when to turn on/off certain tasks using "start tasks" and "stop tasks."