Activity 2: STOP!

1 Behavior Introduction

A behavior-based program is formed by the interactions of several simple reflex-like responses, called *behaviors*. Each behavior is triggered by a sensor, and the behavior becomes active when the sensor becomes active. Groups of behaviors are arranged in order of importance from top to bottom, with the most important behaviors at the top. The highest priority behavior that is active is the one that controls the robot. Take, for example, the following program:

1. `avoidLoudNoise()` ← *highest priority behavior*
2. `moveToLight()`
3. `doNothing()` ← *lowest priority behavior*

This program would run the `doNothing()` behavior unless the `moveToLight()` or `avoidLoudNoise()` behavior is active. If a light is present, then the `moveToLight()` behavior will override the `doNothing()` behavior and the robot will move towards the light. If a light and loud noise are both present, then the `avoidLoudNoise()` behavior will override both the `moveToLight()` and `doNothing()` behaviors and cause the robot to avoid the noise. What do you think would happen if the noise source and light source were in the same location?

2 Overview

The goal of this activity is to make the robot stop. You will write three behaviors: `wallStop()`, `lightStop()`,and `goalStop()`.

You have new functions to use today:

1. `behControl.obsFront()`: Determines if there is an obstacle in front of the robot. Input: None, Output: `True` if an obstacle is in front of the robot, `False` if otherwise.
2. `behControl.lightSense(sensor)`: Determines the amount of light on sensor given in the argument. The `sensor` argument can be one of 'fl', 'fr', 'r'. Input: sensor - which light sensor to read
   'fl' = front left sensor
   'fr' = front right sensor
   'r' = rear/back sensor
   Output: value between 0 and 1023 indicating the amount of light present at the input sensor.
3. `behControl.nbrGoal()`: If a goal robot is a neighbor of the robot, return `True`, else return `False`. Input: None, Output: True if a goal robot is detected, False if otherwise.

2013-07-23

3 Tasks

3.1 Wall Stop

Write a function `wallStop()`. It should take no arguments and return a tuple of `(active, tv, rv)`. The function will use a single if statement. Use the function `behControl.obsFront()` to check for obstacles. If there is an obstacle in front, the behavior need to command the robot to stop. To do this, it becomes active, and sets the `active` variable to `True`.

3.2 Light Stop

Next, write the function `lightStop()`. It should take no arguments and return a tuple of `(active, tv, rv)`. The user needs to call `behControl.lightSense(sensor)` to check a sensor. The function will use three if statements to see if any sensor is above a threshold value. The user needs to set a threshold variable between 0-1023. If `behControl.lightSense(sensor)` for any sensor returns a value greater than the threshold, the robot needs to stop. To do this, the behavior becomes active, and sets the `active` variable to `True`. Experiment with different threshold values until you find one you like!

3.3 Goal Stop

Write the function `goalStop()`. It should take no arguments and return a tuple of `(active, tv, rv)`. The function needs to call `behControl.nbrGoal()`. If the robot sees the goal robot it needs to stop.