

CMSC 143: Object-Oriented Programming with Robots

Lab 1: Personal Robots

Due February 6, 2014

This lab introduces you to your robot, Calico (our Python programming environment), and the wiki¹.

Learning Objectives

- Become familiar with the IPRE robot kit.
- Learn how to start Calico and save programs.
- Write simple functions.
- Learn how to issue robot commands.
- Learn how to navigate the myro wiki.

Deliverables

Submit an electronic copy of your lab using moodle. Your submission should be a zip file that expands to a folder with two files:

```
cmssc143_lab1_LASTNAME/  
  lab1.py  
  lab1.pdf
```

Your Python program should begin with a comment with your names and the date. Your `lab report` should address the `questions` throughout this document. Submit the report as a PDF; be sure to include your names and the date.

Getting to Know Your Robot

1. `What COM port did you use to connect to your robot (keep in mind this will change often)?`
2. `What was the name of your robot? What did you name it?`
3. `What battery voltage does your robot report?`

Self Portrait

Use the robot's camera to `take your picture and save it to a file` using the following code snippet:

```
p = takePicture()  
show(p)  
savePicture(p, "me.jpg")
```

¹<http://wiki.roboteducation.org>

Driving Your Robot

There a variety of ways of getting your robot to move; in this lab we will stick to using `motors`. The function `motors` asks for two power values in the range $[-1, 1]$ for both the left and right wheels. Experiment with the `motors()` function. Describe the behavior of the robot for the following invocations of `motors`:

| | |
|--------------------------------|--|
| <code>motors(1, 1)</code> | |
| <code>motors(0, 0)</code> | |
| <code>motors(.8, .8)</code> | |
| <code>motors(-1, -1)</code> | |
| <code>motors(1, -1)</code> | |
| <code>motors(-.75, .75)</code> | |
| <code>motors(-1, 0)</code> | |

How can you use `motors` to make the robot follow an arc? Fill in the blanks in the following functions. They act as shortcuts for making the robot stop, go forward, backward, spin left and right:

```
def stopWheels():
    motors (_____, _____)

def goForward():
    motors (_____, _____)

def goBackward():
    motors (_____, _____)

def spinLeft():
    motors (_____, _____)

def spinRight():
    motors (_____, _____)

def test():
    goForward()           # start going forward
    wait(1)              # wait for one second
    spinLeft()           # spin left
    wait(.1)             # wait for one tenth of a second
    stopWheels()         # stop the robot
```

General-Purpose Movement Functions

Write eight functions that act as more general purpose shortcuts for these various motor patterns; these will allow you drive the robot forward, backward, left, right with varying speeds and durations:

```
def goBackward(p):
    # go backward with power p that ranges from 0-1
    motors (-p, -p)

def goBackwardAndStop(p, secs):
    # complete this comment
    motors (-p, -p)
    wait(secs)
    stopWheels()

def test():
    goBackwardAndStop(1, .5)
    goForwardAndStop(1, 1.5)
    spinLeftAndStop(.9, .5)
    goForward(.5)
    goBackward(1)
    spinRight(.9)
    spinLeft(.9)
    spinRightAndStop(.9, .5)
```

Scribbling

Use your robot to draw a square, a 5-point star, or another shape using (a) the `gamepad()` and (b) the functions you wrote. Write a paragraph reflecting on the differences between these two approaches.