CMSC 143: Introduction to Object-Oriented Programming with Robots

Lab 3: Personalized `gamepad()`

Due September 21, 2009

Submit a copy of your python program (cmsc143-lab3-LASTNAME.py) on moodle. Your program should have your name(s), email(s), and the date at the top of the file as a comment.

The `gamepad()` function (from the lab #1) makes all the gamepad buttons do something interesting (e.g., move the robot, beep, speak, or take a picture). In this lab, you will create your own `myGamepad()`. You can make the buttons do whatever you wish, for instance, report the battery voltage, save a picture to a file. For moving the robot with the gamepad the `move(translate, rotate)` function could be handy.

Be creative.

Learning Objectives

- Create a personalized gamepad.
- Use if statements.
- Use while loops.
- Share programs.

Some Gamepad Details

For the first part of the assignment become familiar with `getGamepad()` figure out how the different buttons work and the order they are reported. What are the differences between the directional pad on the left and the buttons on the right?

The `getGamepad()` function waits for a button to be pressed[1] and then returns the status of the controler (represented as something called a dictionary). If you ask for the `'button'` entry you get a list of eight items: `[0, 0, 1, 0, 0, 0, 0, 0]` each item corresponding to one of the buttons. In this case, the third button is pressed. Similarly, the `'axis'` entry returns a list of two items, the status of the directional pad: `[-0.9999694824, 0.0]`. In this case, the x-axis of the directional pad is pressed left. To retrieve an item from a list you can use the index operator. For example, `getGamepad()['button'][0]` grabs the value of the first button, `getGamepad()['axis'][1]` grabs the value of the second axis.

```
status = getGamepad()
print "The entire status of the gamepad looks like", status

buttons = status['button']
print "The buttons:", buttons
print "The first button reads", buttons[0]

axes = status['axis']
print "The x-axis reads", axes[0]
print "The y-axis reads", axes[1]
```

[1] A function named `getGamepadNow()` immediately returns the status of the gamepad without waiting for a button press.

myGamepad()

Once you are comfortable with how `getGamepad()` works you should implement your own `myGamepad()` function. You can make the buttons do whatever you wish, for instance, increase or decrease the beeping frequency, or report the light sensor values. Or maybe make pairs of buttons do some action.

There are two requirements: 1) one of the buttons must quit your function and 2) use all the buttons.

Your program should start something like:

```
def myGamepad():
    ''' HOW TO USE MY GAMEPAD
         up button:  EXPLANATION
       down button:  EXPLANATION
                  ...
    '''

    keepGoing = True

    while keepGoing:

       status = getGamepad()
       buttons = status['button']
       axes = status['axis']
```

ourGamepad()

For the second part of the lab, you should work with someone else in the class to combine the best features of both your gamepad programs.

1. Make sure each of your programs are working.
2. Email each other your programs.
3. Each person should try each other's gamepad and understand how it works.
4. Decide on the best features of each gamepad.
5. Create a new **super gamepad** `ourGamepad`.

```
def ourGamepad():
    '''Author1 <author1@bard.edu>
       Author2 <author2@bard.edu>

       HOW TO USE OUR GAMEPAD
         up button:  EXPLANATION
       down button:  EXPLANATION
                  ...
    '''
```