Today’s topics

Robots
Myro
Loops

Notes from the Institute for Personal Robots in Education

Upcoming
  ➢ More Python

Reading
  *Learning Computing with Robots, Chapter 1-5,9*

What is a robot?

● Definitions
  ➢ Webster: *a machine that looks like a human being and performs various acts (as walking and talking) of a human being*
  ➢ Robotics Institute of America: *a robot is a reprogrammable multifunctional manipulator designed to move material, parts, tools, or specialized devices through variable programmed motions for the performance of a variety of tasks*
  ➢ What’s our definition?

● Components of a robot system?

Uses of robots

● Where and when to use robots?
  ➢ Tasks that are dirty, dull, or dangerous
  ➢ Where there is significant academic and industrial interest

● Ethical and liability issues

● What industries?

● What applications?

Agents and Environments
**IR Obstacle Sensors**

- The Scribbler has two IR obstacle sensors (under the light sensors) that return a binary value of 0 or 1.
- The robot actually has 1 IR receiver located in the center, and two emitters located on the left and right side of the emitter.

**Return value of:**

- **0** – IR light is bouncing back to receiver
- **1** means that infrared light is not bouncing back to the receiver, so nothing is in front of the emitter/detector.
**IR Obstacle Sensors**

- `getIR()` returns a list of two items [1,1]. You can also call `getIR("left")` to get just the left sensor, and similarly with `getIR("right")`. The function also accepts 0 and 1 as the parameter to select which sensor value to return.

**Light Sensors (3)**

- The scribbler has 3 light sensors
  - Left, Center, and Right
  - On the same side as getIR sensors.
- `getLight()` returns a list of all 3 values.
- `getLight("left" / "center" / "right")` or `getLight(0/1/2)` selects one value.

---

**Light Sensors (3)**

- Light sensor values range from 0 to 5000. Zero is very bright, 5000 is full dark.
- `getLight()` returns a list of all 3 values.
- `getLight("left" / "center" / "right")` or `getLight(0/1/2)` selects one value.

**Internal Scribbler Sensors**

- `getBattery()` - returns battery voltage
- `getStall()` - returns stall condition (0 or 1)
  - This value changes to a 1 when the motors are overworked
  - Note that it takes a half second to re-set the stall sensor
- `getName()` - returns the robot’s name
  - can be changed with `setName("newName")`
Fluke Board

- The Fluke add-on board has its own IR obstacle sensors and a camera.

Fluke Board – IR Obstacle sensors

- Myro uses the `getObstacle()` function to differentiate the IR sensors on the Fluke from the IR sensors on the Scribbler.
- The fluke sensors are more sensitive than the Scribbler sensors. Instead of just returning a zero or one, they return an integer value between zero and 7000.

Fluke Board – IR Obstacle sensors

- The fluke has 3 IR emitters, one pointing forward...
- And two pointing to the left and right.
Fluke Board – IR Obstacle sensors

- The fluke has 3 IR emitters, one pointing forward...
- And two pointing to the left and right.
- They all bounce light back to a center mounted receiver.

Zero indicates no IR light is bouncing back from an obstacle.

[0,0,0] = getObstacle()

Larger numbers indicate that more IR light is bouncing back.

[0, 1842, 0] = getObstacle()

Robot Actuators

- Beep
  - beep() - issues a random beep
  - beep(1) - random beep for 1 second
  - beep(1, 800) - Beep at 800Hz for 1 second
  - beep(1, 440, 880) - two tone beep at 440 & 880Hz for 1 second.

- Motors
- LED Lights
Motor commands – Synchronous vs Asynchronous

- What is the difference between these pieces of code?
  
  forward(0.85)
  wait(1.5)
  stop()

- And...
  
  forward(0.85, 1.5)

Motor commands – Synchronous vs Asynchronous

- What is the difference between these pieces of code?
  
  forward(0.85)
  beep(1.5, 880)
  stop()

- And...
  
  forward(0.85, 1.5)
  beep(1.5, 880)

Motor commands – Translate & Rotate

- Other functions exist:
  
  - translate(speed)
  - rotate(speed)
  - stop() is equivalent to translate(0); rotate(0)

Fluke Board – Camera

- You can take a picture using the camera on the Fluke board.

  p = takePicture()

  show(p)
Fluke – Camera as a brightness sensor

- Similar to the getLight() function on the Scribbler, the fluke allows you to use the camera as a “brightness” sensor with the getBright() function call.
- getBright() returns a list of three values
- getBright(0/1/2) or getBright(“right”/“center”/“left”) return a single value.
- The numbers returned represent the sum of the luminance values of the pixels in the right-center/left of the camera, so they are quite large!
- The lower the number, the darker that area of the camera is.

Taking / Saving / Loading a Picture

```python
p = takePicture()
savePicture(p, “class.gif”)
p2 = loadPicture(“class.gif”)
```

```python
print getWidth(p)
print getHeight(p)
```

Robot Pictures

- Robot pictures are:
  - 256 pixels wide by 192 pixels high
- Each pixel is made up of 3 colors:
  - Red, Green, and Blue
  - Colors range in value from 0 – 255
  - Plus an “Alpha” value we won't use (for now).
- When you print a picture or a pixel, Python gives you some textual information about the item, but does not give a visual representation.
- The show(picture) method will display the picture on the screen.

Accessing a specific pixel

```python
print p
pix = getPixel(p, 50,50)
print pix
setRed(pix,0)
setGreen(pix,0)
setBlue(pix,0)
```

```python
print pix
```

```python
show(p)
```
Zoomed In View: One Black Pixel

Looping through all pixels

```python
print p

for pix in getPixels(p):
    setRed(pix, 0)
    setBlue(pix, 255)
    setGreen(pix, 255)

show(p)
```

Pixel Functions

- Getting a specific pixel:
  - `getPixel(picture, x, y)`
- Setting color info:
  - `setRed(pixel, value)`
  - `setGreen(pixel, value)`
  - `setBlue(pixel, value)`

- Getting all pixels:
  - `getPixels(picture)`
- Getting pixel color info:
  - `getRed(pixel)`
  - `getGreen(pixel)`
  - `getBlue(pixel)`
- Getting pixel location:
  - `getX(pixel)`
  - `getY(pixel)`

Looping through all pixels: `getPixels(picture)`

```python
p = loadPicture("class.gif")

for pix in getPixels(p):
    setRed(pix, 255)

But what if you only change the red part of the pixels?
```