CS 1301

Individual Homework 4 – Getting to Know Your Robot – Dancing Menu & Avoiding Walls

Due: Friday September, 30th, before 11:55 PM
Out of 100 points

Files to submit: **danceMenu.py**
avoidWalls.py
(2 files total)

For Help:
- TA Helpdesk – Schedule posted on class website.
- Email TAs

Notes:
- **Don't forget to include the required comments and collaboration statement (as outlined on the course syllabus).**
- **Do not wait until the last minute** to do this assignment in case you run into problems.
- If you find a significant error in the homework assignment, please let a TA know immediately.

Part 1a – Dance, Robot, Dance! (20 points)

Hopefully by now, you've gotten your robot out of the box and made friends with it (or at least acquaintances); given it a name and a long, intricate history. Well, how about now you take it out to a dance?

Of course, you'll have to teach your robot to dance. Using the movement functions:

http://wiki.roboteducation.org/Myro_Reference_Manual#Movement_Functions

have your robot do a little dance. The dance should last for at least 30 seconds, and contain at least 3 distinct dance moves. Don't just go back and forth for 30 seconds; vary the dance a bit. Pretend your robot is well-versed in rhythm and soul, or is at least a little spastic. In addition to the movement, your robot should also make some noise! The **beep()** function is very helpful – it allows the robot produce various tones. You are allowed (and encouraged) to make your own helper functions that contain individual dance moves.

Write your dance as a function called **dance()**, and save it into a file called

danceMenu.py. As always, please name your file exactly as requested.

Part 1b – Dance (Menu) Remix (25 Pts)

Now that you've taught your robot how to dance, you demonstrate it to your friends and family. They are very impressed by its performance, but each wants the robot to do his or her own version of the dance. (Plus you and your partner can't agree on which dance moves you like the best...) You can't simply alter dance() to please everybody-- what can you do? Luckily, with a little Python, you can let anyone customize your robot's dance routine without having to teach him or her how to program.

For this part of the assignment, you'll use conditionals ("if" statements) and a while loop to create a menu that allows the user to select a dance step. Name the function that contains this menu **menu()**, and save it in a file called **danceMenu.py**. If you used helper functions in your dance() from the first part of the assignment, you can re-use those. If not, you'll need to create a few short functions that cause the robot to do a particular dance move for a few seconds. Your menu must implement **at least 3 distinct dance choices**, plus **an option to exit the program**. If the user types something invalid, you should let them know and print the menu again. Once the robot does it's dance, print the menu again until the user chooses to exit.

Here's an example of how menu() might work. The user's input is shown in blue.

```
1. The Charleston
2. The Tango
3. The Foxtrot
0. Exit
Which dance step would you like? 1
```

(The robot does a dance move called the Charleston. The program doesn't show the menu again until this move finishes a few seconds later.)

```
1. The Charleston
2. The Tango
3. The Foxtrot
0. Exit
Which dance step would you like? 5
I'm sorry, I don't know that one.
1. The Charleston
2. The Tango
3. The Foxtrot
0. Exit
Which dance step would you like? 0
Have a good day!
```
(The menu() function terminates.)

Part 2 – Avoid Walls (55 Pts)

Avoid Walls Code

There are five kinds of sensors on the robot: light sensor (detect the how bright the light is), proximity sensor (see whether there is anything around the robot), stall sensor, the camera, and the battery voltage. We are going to use the robot's proximity sensors for this homework. Be sure to read all of the information below. It explains many functions that are vital to the successful completion of this assignment.

Mission:

Your robot will be randomly placed in an arena of size 4' x 4'. Your robot will be at least 3 inches away from a wall, but it could be near a corner or right in the middle. You need to write a function called **avoidWalls()** to move your robot around for one minute (+/- 5 seconds) without hitting walls. The robot needs to be moving at a minimum of 1/3 of it's maximum speed. (Your robot may drive “backwards” if you want to use the getIR() sensors instead of the getObstacle() sensors.) Your robot must translate, not just rotate! The robot should also celebrate after finishing the mission successfully. How it is going to celebrate is up to you, although it must be recognizable. We suggest moving around and beeping at a minimum.

For more information on the robot arena, see the posted file, or visit the TA helpdesk to try your code out in our pre-constructed arena.

--- What's on the robot? ---

Proximity Sensors:

Proximity sensors are used to detect objects that are close to the robot. The robot has two sets of proximity sensors: one set is on the robot and the other set is on the fluke.

The IR sensors on the robot:

There are two infrared (IR) sensors on the back of the robot (Assuming the fluke is facing forwards). You use the sensors by calling the **getIR(<position>)** function.

Examples:

```
>>> getIR()
[1, 0]
>>> getIR('left')
1
>>> getIR(0)
1
```

```
>>> getIR('right')
0
>>> getIR(1)
0
```

getIR(<POSITION>) Returns a integer value in the <POSITION> IR sensor. <POSITION> can either be 'left' or 'right' or one of the numbers 0, 1, which correspond to “left”, and “right”.

IR sensors return either a 1 or a 0. A value of 1 implies that there is nothing in close proximity of the front of the sensor and a 0 implies that there is something right in front of it.

The proximity sensors on the Fluke:

There are three proximity sensors on the fluke: one front sensor and two side sensors. They give different values than the ones on the robot. To use this set of sensors, you need to call **getObstacle(<position>)**.

getObstacle() return a list that contains the values from all three sensors. To get a value from a specific sensor, you can call **getObstacle(<position>)**. <position> can be “left”, “right” or “center”. <position> can also be number 0, 1, or 2, which correspond to “left”, “center”, and “right”.

Examples:

```
>>> getObstacle()
[1703, 1128, 142]
>>> getObstacle('left')
1703
>>> getObstacle(0)
1703
>>> getObstacle('center')
1128
>>> getObstacle(1)
1128
>>> getObstacle('right')
142
>>> getObstacle(2)
142
```

getObstacle(<position>) returns a integer value between 0 and 7000. A zero (0) indicates there is nothing in close proximity of the sensor. Higher value implies the presence of objects in front of the sensor(s). Note that the center sensor is the most accurate. The side sensors will detect objects located at about a 45 degree angle, but not those directly to the sides of the robot.

More details about the sensors can be found:
<http://wiki.roboteducation.org/Learning_Computing_With_Robots – Chapter 5 – Sensing the World>

Reminder: Robot needs to be moving at a minimum of 1/3 speed.

Internal Clock function:

To keep track of time, you can use the myro internal clock function. To get the current time, call currentTime() function. The function returns the number of seconds past since sometime in the past. [Epoch or Unix time if you're interested.]

Example:

```
>>> currentTime()
1222374008.360949
```

To keep track of time, you need to call currentTime() and save the time to a variable (e.g. time). You can get the time that has passed since you last called currentTime() by subtracting time from currentTime().

Example:

```
time = currentTime()

doing something.....

doing something.....

timePast = currentTime() - time
```

More detail about the internal clock function:

http://wiki.roboteducation.org/Learning_Computing_With_Robots – Chapter 4 – Sensing From Within

If you want, you may use the timeRemaining() function and a while loop instead of the currentTime() function to time your robots motion.

If you need help with the move functions, go to
http://wiki.roboteducation.org/Learning_Computing_With_Robots – Chapter 2 – Personal Robots

In the file:
Be sure to put the lines from myro import * and initialize() or init() at the beginning of the file (after the required comments). Be sure not to specify the port parameter in your initialize command, such as initialize("com4"). This makes it very time consuming to grade if we have to go into your code and change the com port to the one that works on

our specific system. The TA will type avoidWalls() to start your robot moving, so you don't need to include a call to that function in your homework file.

Part 3 – Turning it in

Once you're done, please submit **danceMenu.py** and **avoidWalls.py** to T-Square.

Remember – if there are serious errors in a part of the homework, you will lose credit for that particular part. If your file fails to run at all, you will lose 50% credit for the entire homework. **Please test your code thoroughly before turning it in!**

Grading Rubric

Part 1a – Dance – 25 points

Function named correctly (dance) –	**5 points**
Dance lasts for at least 30 seconds –	**5 points**
Contains at least 3 distinct moves –	**5 points**
Robot beeps –	**5 points**
Creativity –	**5 points**

Part 1b – Menu – 25 points

Function and file are named correctly (menu) -	**5 points**
Displays a menu & accepts input from the user -	**6 points**
Loops correctly -	**3 points**
Handles three cases with conditionals -	**3 points**
Shows off 3 distinct moves -	**3 points**
Exits correctly –	**5 points**

Part 2 – Avoid Walls - 50 points

File/Function named correctly	**5 pt**
Uses iteration to move for 1 minute	**10pt**
Uses IR obstacle sensors to detect Walls	**10pt**
Avoids walls!	**10pts**
Moves at 1/3 speed or higher	**10pts**
Celebration in the end	**5pt**

For a grand total of 100 possible points.
You can earn up to 5 points bonus [discretion of the TAs] for extra creativity/general awesomeness, for a possible total of 105/100.

Written By: Melody Nailor, Spring 2009; Kevin Jones, Summer 2009.