CAST CPET-563 Embedded Systems Design II **Dr. Kaputa**
Spring 2018
Lab 1

Student Name: _____

Introduction:

Your task is to create an arena [50 x 50] with a rover located in the center of it facing to the right. Use an appropriate rover size. I used a [4 x 2] rover with different colors to indicate the front of the rover. Extra credit for using a cool rover graphic. On each timer tick the program will read in a line of code from an encoder text file shown below and move the rover appropriately. Initially one can ignore the *l_dir* and *r_dir* columns and only focus on the *left* and *right* columns. The *l_dir* and *r_dir* columns indicate if the wheels are moving forward or backwards. The main lab is only concerned with forward wheel motion but there is extra credit for getting things to work with backwards motion.

l_dir	left	r_dir	right
1	1	1	1
1	1	1	1
1	0	1	1
1	1	1	1
1	1	1	0
1	2	1	2
1	1	1	1
1	2	1	2
1	2	1	0

Since 2 encoder ticks is defined as 90 degrees of rotation, then 1 tick is 45 degrees of rotation. When the first line is read the rover should translate 1 unit to the right which is forward since the rover is pointing to the right. Make sure to watch the lab1.mp4 video in order to understand how the above table corresponds to rover motion. You will need to keep track of rover angle and add or subtract from that angle based upon the left and right encoder values. The below snippet of code will then convert from polar angle to Cartesian coordinates in order to perform the motion. Ex. Imaging the rover is at 45 degrees and the left and right encoders both read a value of 2. The rover would have to move forward, however since it is at 45 degrees one would have to translate the rover in both X and Y.

```
# resolve forward vector into x and y components.  Invert y due to QT coordinate system
self.forwardX = self.forward * math.cos(self.angle * (math.pi/180))
self.forwardY = -1 * (self.forward * math.sin(self.angle * (math.pi/180)))
```

Requirements:

- Shall be done in pyqt4
- Shall make use of a QmainWindow
- Shall include a function called something like 'getEncoders(hwFlag)' that is passed a hardware flag
    - If hwFlag = 0: data is read from a file

    - If hwFlag = 1: data is read from the hardware
        - Just read back zeros for now
- Shall use a board size of 50 x 50
- 2 encoder ticks shall equate to 90 degrees of rotation.
- It shall be clear which end of the rover is the front.
- Timer update shall be between 2 sec and 4 sec.

Resources:

PyQt4 tutorial at http://zetcode.com/gui/pyqt4/

PySide libraries at https://srinikom.github.io/pyside-docs/index.html

Python CSV reader at https://docs.python.org/2/library/csv.html

```
# open the encoder file for reading
encoderFile = open('encoders.txt','r')
next(encoderFile) # skip headings
self.encoderReader = csv.reader(encoderFile,delimiter='\t')
```

[self.leftDirection, self.leftEncoder,self.rightDirection, self.rightEncoder] = next(self.encoderReader)

Lab Submission:

1. Print out this lab with your name on the front and attach all code.
2. Submit all code by 4:40 pm on Feb 1st at: https://www.dropbox.com/request/W7dN18XQRfAIkCQvcoJK

Grading:

	Score	Pts
Block Diagram Show parent child relationships and overall data flow. Ex. Where does the timer exist?		/2
Code No tabs please. Proper spacing and formatting. Adequate variable names. Consistency.		/2
Comments Proper header and comments throughout		/2
Demonstration To be performed within lab week. Can show after due date with time stamped code.		/4
Extra Credit [1] Use a cool rover image instead of a boring rectangle like me		/1
Extra Credit [2] Create text file with reverse direction and perform demonstration		/2
Final Grade		/10