Sebastian Bernal: TP1

**Project Proposal**

**\*Note:** Had a lot of trouble setting up the IDE and setting up appropriate connections between Arduino and PC; decreased the time I had to do actual programming. Implemented main menu and hand position/state tracking. Implemented a transitioning framework for different modes. Some code was experimenting with Arduino.

Project Description:

Robo Mirror Dance Club

Through this project I’ll be interfacing an Arduino robot over Bluetooth with my laptop; I’ll be attempting to implement a sort of “game” that involves the control of the robot’s movements; the user/player should be able to control the robot through the interaction of the Kinect sensor. The game aspect of this project should come to fruition in the mimic mode where the user must execute the same movements the robot is making.

Competitive Analysis:

Robots have been implemented plenty of times in 15-112 but I have yet to see one that the user interacts with the way I envision it; the omni-directional path finding robot that was created before is the most similar project to mine. The omnidirectional robot is given a path to follow and it finds the most efficient way to get to the specified endpoint. While my robot will not be capable of avoiding obstacles, it should also be able to follow some sort of linear path drawn by a user on the screen. Motion controlled robotic arms were the most common project I found when doing some research online; they utilized servos and other actuators to control the angle and movement of different joints in the robot. If all goes well, I hope to implement some sort of joint control as well.

The practical uses of the robots I found online vary greatly as well; there was a video in which a woman was controlling a robotic arm with a Kinect that was supposedly used in manufacturing. My term project in general seeks to provide some entertainment for a user; perhaps get him or her interested in robotics. The party mode feature of this whole project should be enough to captivate even the most anti-party people.

Structural Plan:

The sceneManager file is going to contain a class that almost every other “mode” inherits from; it will make transitions in the game a lot smoother in general. There will be one file that contains the classes and methods I’ll use for the logic behind the Kinect; the functions in this file should be usable in any other “screen”/”mode”. I’ll have to create about one of two other files containing class and function definitions that will specify what kind of signals to send to the Arduino; while Arduino code itself is usually written in C, I’ve found a way to appropriately translate most of the commands so that I can write them in Python.

Algorithmic Plan:

What seems to be the most difficult aspect of the project to implement is the line/path following; I am no even sure if one can save what is “drawn” on the screen but I will do my best to do something similar by following the approach used in the Turtle Graphics homework problem. I’ll keep track where exactly it is that a user closes their hand relative to the origin of the pygame window; when the user releases the hand, the difference in the coordinates after should specify how “long” and at what angle this path was drawn. Another thing that will depend on the difference in these coordinates is the algorithm I’ll implement to appropriately set the motor speeds of the robot; this algorithm is going to set a delay and specify how long the robot should go in a specific direction as well.

Timeline Plan:

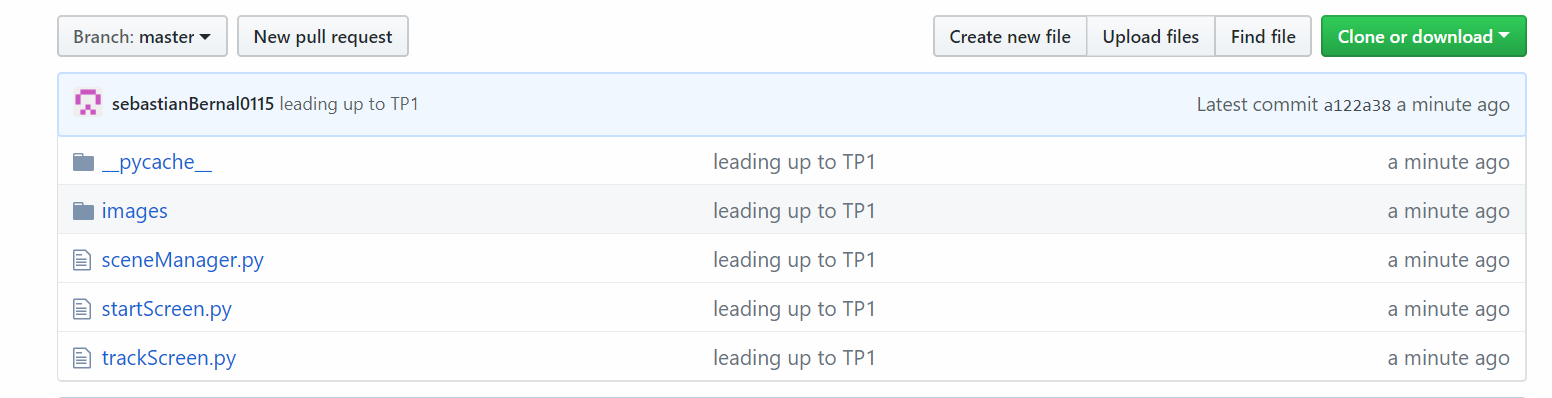
By the time break is over, I’ll have a working menu implemented; the user will be able to select among the different modes and also trace a path with their hand in path mode. The robot should also be able to interface wirelessly with my laptop and move forward, backward, right and left depending on the user’s movements (mirror mode).

By TP2 I’ll have most if not all of the logic and functionality for the actual path following finished; I’ll start looking for bugs and start cleaning up the code a bit if it seems too cluttered. The weekend after TP2 I’ll finish integrating some of the game aspect with mimic mode; the robots is going to move randomly based on a preset path that the user will have to follow for points. By the time this is done every screen should have a back button.

The week of TP3 I’ll be working on the party mode; creating random preset “dances” for the robot and make music play once the user selects a genre of music for the party mode.

Module List: Kinect, Arduino (Bluetooth), Pygame

Version Control Plan: Using Git and Github, going to make backups in Google Drive



Sources so far

#used for the detection of the z distances aka the depth / distance from the sensor

<https://pterneas.com/2016/08/11/measuring-distances-kinect/>

#Robot image for start screen

<https://openclipart.org/detail/288756/cartoon-robot-remix>

#used for robot hand

<https://pngtree.com/free-png-vectors/robot-hand>

#used for scene change

<https://nerdparadise.com/programming/pygame/part7>

#general API and SDK info

<https://docs.microsoft.com/en-us/previous-versions/windows/kinect/dn785530(v%3dieb.10)>

#transparency for main menu

https://stackoverflow.com/questions/328061/how-to-make-a-surface-with-a-transparent-background-in-pygame