Arvind Seshan
15-122 Term Project

Project Plan

Project Description

The name of my project is **Magic Axolotl Academy**. It is a spell casting game where users use different hand movements to cast various spells. Enemies can be defeated by performing the correct order of spells. The main character is Kimchee the Axolotl. The main enemies are smog monsters because the biggest threat to axolotls are urbanization and pollution, threatening their environment. The idea for the game is based on the 2016 and 2020 Google Doodle called Magic Cat Academy (https://www.google.com/doodles/halloween-2020).

Structural Plan

In order to collect input from the user, I will write a function to retrieve the x and y coordinates of each hand. A second function will be used to store coordinates over time and save to a text file. A separate program will use the text files to train a machine learning model to recognize several hand gestures. Separate files for the ml model and data scaler will also need to be stored. A function in the game program will then use the trained model to detect gestures in the game and update the game accordingly (e.g. enemies will take damage). The enemies and axolotl player will both be structured as a class with several important properties. Enemies will have a health, speed, size, color, gesture list, and x, y location. The player will have a health, size, color, and x,y location. Both will need methods to take damage and move. The enemy gesture list dictates what gestures are needed to defeat that enemy. An enemy moving function will run in appStarted to update the location of the enemies as they try to reach the player. I also plan to have multiple levels in the game with increasing difficulty.

Algorithmic Plan

Hand Detection: To detect the player's hand, I will use OpenCV. To make things easier, I will wear blue gloves that stand out from the background. I will filter the image for the proper color and use the contour area to filter out background noise. I will ignore nested contours and use contour segment compression to simplify the tracking and make sure everything runs smoothly.

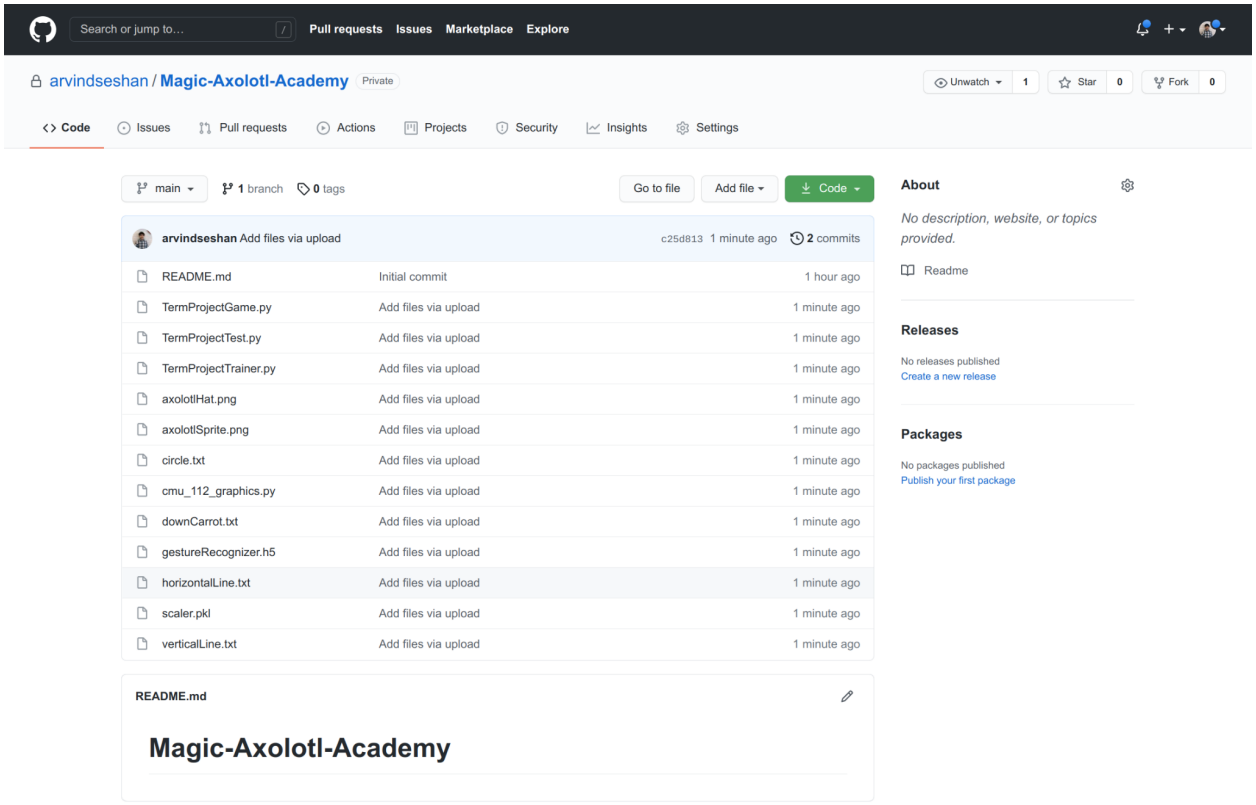
Gesture Recognition: A difficult part of the project is recognizing the gestures accurately. For this, I plan to collect a set of x and y coordinates from the center of the hand. I will then take pairs of two points and compute the slope of a line between them. The list of slopes alongside the gesture they represent will then be used to train a neural network that can recognize these

gestures. I will normalize the input (list of slopes) using a scaler to prevent improper weighting of variables.

Map Generation and Path Planning (After MVP): For the map generation, I will randomize the number of walls, the length of the walls, the direction of the walls, and the location of the walls. This will place random walls in the game that the enemies cannot cross through. To find the shortest path to the player, I will employ a Breadth First Search (BFS) algorithm. It will work by keeping track of previously visited nodes. It will create a queue of unvisited neighbors and select the current node as the first item in the queue. If it is the target node (the player), then stop the search. Else, it will oop through each of the neighboring nodes and add them to the queue if they have not been visited. This is repeated until the queue is empty.

Version Control Plan

I am using GitHub for version control. Below is a picture of the repository.



TP2 Update

For the map generation, I use a cellular automata model to create natural looking walls. This works by having a 2D list of cells in the game and starting by randomly setting a certain percentage of the cells to walls. It then runs through several iterations of set steps to alter the

cells. The steps/rules I am using are as follows: If there are 3 or less neighboring cells with walls, the wall in the cell goes away. If there are more than 5 neighboring cells with walls, the cell becomes a wall. Else, the cell remains the same. This process is repeated many times to produce walls.

TP3 Update

I added several new features. The first is line of sight. Now the axolotl can only attack the Smog Monsters when she can see them directly (except for lightning bolt attacks).

I also added a new auto mode: Pressing “a" allows the human player to have the axolotl automatically move to the best position away from a Smog Monster while still maintaining a line of sight to attack them. Scores are calculated for each cell in the game based on the total distance from smog monsters and multiplied by 2 for each monster in line of sight cells close to a monster are given a large negative value since the axolotl should avoid those cells if possible.

I also added new testing features: Pressing “t" enables testing mode. It displays the best path that enemies and the player are taking. It also displays scores for each cell in the game which tell the best spots to travel to to avoid monsters and maintain line of sight. It additionally shows red or green lines from the player to each monster to tell whether the monster is in a direct line of sight and a red circle to tell where the line of sight is blocked. Enemies and the player are displayed as circles, indicating their intersection area for collision detection between enemy and player. Pressing “w” enables and disables wall manipulation and clicks with the mouse adds or removes walls. There is also pausing and stepping with “p” and “s”.