Project Proposal of 15-112 Term Project

Introduction

The term project is trying to implement the game called “Nidhogg” into python using the pygame. User can “technically” edit the map of the game, but it requires some works.

Goals

The project is trying to create a primarily 1v1 game. The player is on the left of a 2d screen, and the enemy is on the right. The objective of playing is using the estoc on the player’s hand to kill the enemy on the right, and move player to the right. Once the enemy is killed, the enemy would resurrect on the right of the player after a certain amount of time. Once the player reached to the end of the current map (which is very small), the player would enter another section of the maps. Once the player reached the end of all map sections, then the player wins this level. Notice that enemy also has estoc and they would be able to kill the player. The goal of the enemy is similar to the goal of the player, but they just want to walk to left. So the player loses if the enemy reaches the left-end of the map sections.

Solutions

0 - pygame

Using the pygame instead of the tkinter would definitely help solving the problem. pygame could give a smoother experience, and it provides a lot of useful built-in functions such as the function for the pixel perfect collision. Also, pygame won’t break easily, and arguable runs faster than the tkinter.

1 - setting up initial variables

Windows size, FPS, the variables for the pygame, the title for the window, all things to setup the pygame should be considered. Also the code needed to be well structured to make working on the project more easily.

2 - Object-Oriented: player, enemy, ground, playerEstoc, enemyEstoc

A lot of useful functions in pygame require objects to inherit from (pygame.sprite.Sprite). So there will be a top object named Sprite inherited from pygame.sprite.Sprite. Everything else would inherit from the sprite object. This makes changing the variables more easily.

There is also something called pygame.sprite.Group. This group is a special group which can directly use the group draw functions and collision functions in pygame. Thus, it would be easily to detect if the player collide with “enemyEstoc” group, which means player is hit by the enemy’s estoc.

3 - GUI and graphic

The GUI requires player to jump or arrive at specific location. The something will show up. It would rather be an instruction, a fun joke by me, or something interesting. All UI does not require mouse control, because I designed and coded in this way to make it clean and neat.

4 - enemy AI

The enemy AI is made using the Q-learning. It stores information at a local file (.json). And it will learns from player’s action to choose the best action he should do. He will learn and become smarter. Initially it seems like stupid, but it will be clever after you play and interact with it. Also there is a map specifically designed to train the AI in order to make it smarter.

5 - audio

Audio are made for some actions (actions that I think are useful.) Also, background music are added.

Algorithms

Enemy AI:

The enemy ai will probably use a lot of if-statement to think what is should do. Their are some of the basic requirements:

I would do some researches on how to make the AI. The currently planed method is to “study” player’s movement and try to response to each movement. The AI would “technically” become smarter and smarter by learning player.

Q learning algorithms is implemented. With some core parameters entered, the enemy will choose the best action based on the current state. It would train it self and learn from the player. It can not learn itself, and it specifically react to one player’s action. Enemy AI could be reset by running a python file.

Outside modules

Pygame-https://www.pygame.org/docs/ref/time.html

Update:

No more my own features, since it costs a lot of time and it is not complex enough
The enemy’s AI is going to be more cleverer. And the AI should learn through time.
The online feature and the better audio quality would only be done after the AI is completed.

Update:
The enemy AI uses Q-learning. It learns itself.
A massive amount of instructions, with “setting” and “credit”.
Audios are added.
PVP and PVE mode are completed.
Bugs are fixed. (collision bugs)
User “edibility” increases. If some user got my code and want to change a map, it would be much easier. (Cleaner code)