Zachary Mason (zymason)
15-112 Project Proposal

Description:

My project is a game of ultimate tic tac toe, in which the larger board is played on a regular 3x3, and each sub-game for the 3x3 square is a 3x3x3 game of tic tac toe. This is a three-dimensional version of tic tac toe, in which players can win not just in a horizontal plane, but also vertically and any combination that yields the traditional 3-in-a-row. There will be an option to play single player or two-player, in which the former will use a minimax AI to place pieces in place of a second human player. I will also employ a unique method to display the 3x3 game board in both an array of 3x3 grids (2D) and a rotatable 3D-like model of the game board.

Analysis:

I have seen various versions of tic tac toe online, including 3D versions of the game similar to what I will attempt to do. Additionally, I have seen games of ultimate tic tac toe as well, in which there are sub-games within a larger game, but these have only been in a traditional 2D format, rather than a 3D I will be using. The most notable feature of my version is the method in which I plan to display the 3D game board. Nearly all games currently online utilize an array of 2D grids to both visualize and place the game pieces, so the user has to work out the spatial arrangement themselves. I will retain this method to place pieces because it makes it easier for the player(s), but I will employ the 3D model as a primary method of visualization.

Structural Plan:

At the highest level, I will be organizing the game into several game modes: start, rules, 2D game, and 3D sub-games to simplify and select which functions are running at any given moment. The start and rule pages are simple, only involving text and a couple of clickable regions. For the 2D game, there will be a function that handles the drawing of the grid/pieces, one to handle the AI, one to check win conditions, and some that can call the 3D sub-game model of a given position, if desired by the player. The game state for each cell in the 2D grid will be stored in a new class, containing the winner and winning 3D board.

In the 3D sub-games, the duties needed to be carried out are largely similar to the 2D page, only that they involve an additional layer of everything needing to be in 3D. This includes the AI, win-checking, and piece placement functions.

Algorithmic Plan:

By far the most complex elements of this project will be the 3D model draw functions and the 3D AI. For the 3D model, I plan to use an isometric model for display, which is the simpler part, but the complexity comes in the form of displaying the pieces properly. The locations can be easily determined with Cartesian-isometric conversions, but I will have to devise an algorithm that decides what pieces are behind the wireframe grid and what pieces are in front, so it appears correct for the viewer. This is complicated by the fact that the view is constantly rotating using the rotation matrix, so this functionality will have to work for all viewing angles around the cube.

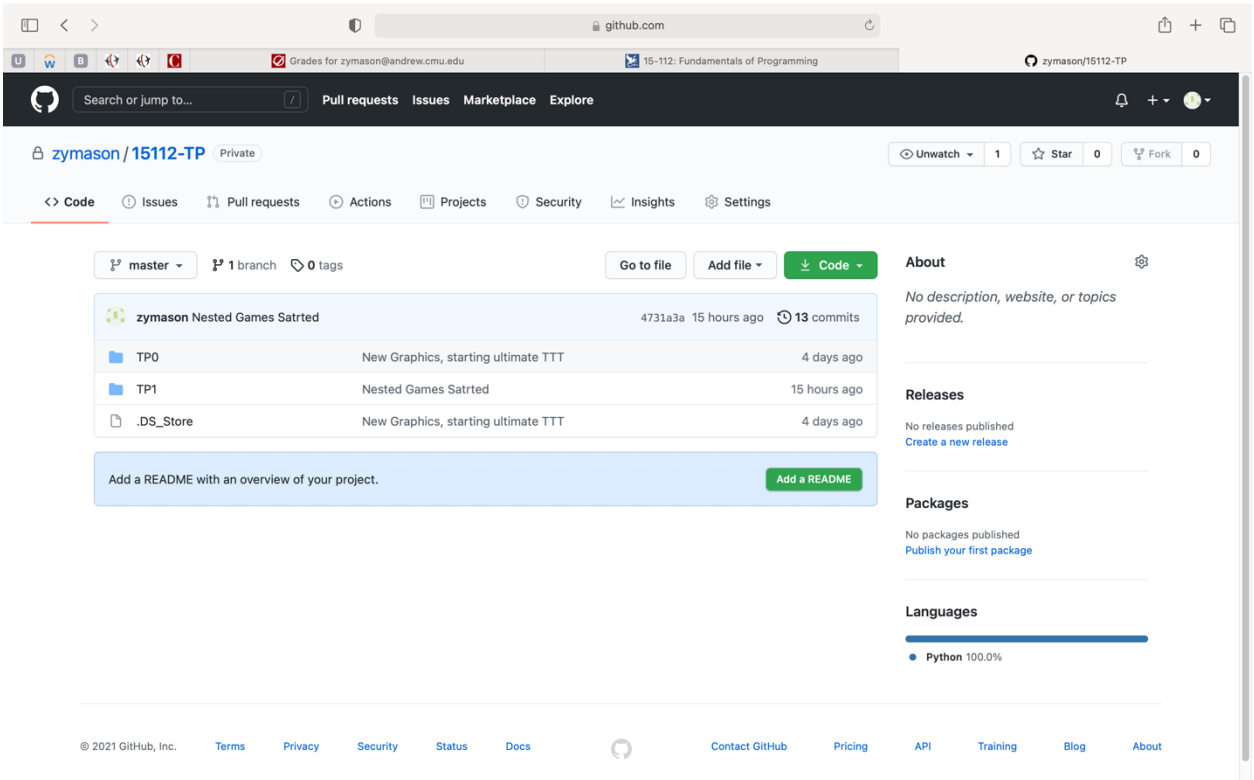
For the AI in both 2D and 3D, I plan to use a minimax function to play for a second player, which is a recursive function that decides where pieces should be played. In the case that the AI goes first, there are 2^27 different possible game states in which no space is empty at the end, so it will have to be greatly simplified to have any hope of running in a timely fashion. Because of the 3D grid, I can utilize symmetries to quickly narrow down the number of potential moves into a more manageable size when performing recursion.

Timeline Plan:

At TP1, my goal is to have completed the 2-player functionality and have started the minimax algorithms and 3D piece viewing. By TP2, my goal is to have the minimax algorithms complete, and have the 3D viewer functioning as well. In the time between TP2 and TP3, my focus will be on finalizing the deliverables for TP3, in addition to adding elements that make the game more visually appealing. This is also where I will add the "game rules" page, as it is a far simpler feature to encode.

Version Control Plan:

For version control, I am using the GitHub extension for VSCode, where I have a repository that I push all changes to.



Module List:

I am not using any external modules for this project.