Group 2 Computer Graphics 445 Project

Project Description

Our project focused on delivering an interactive chess simulator for a user to engage in games of chess with another player or against an AI opponent. The project ultimately incorporated the use of WebGL, Three.js, and the Stockfish Chess API. Utilizing these technologies, our group was able to accomplish all of our major goals for this project.

The previously mentioned major goals include the following: Firstly and most importantly, we wanted to create a 3D Chessboard and accompanying pieces, ensuring that the pieces on the chessboard are correctly placed when moved or taken.

Next, we wanted to ensure that camera controls and a convenient menu for said controls are available for the user. Additionally, an environment has been set up for the game to take place in with accompanying illumination from our light sources (ambient, directional, and point lights) so that the components of the chess set are visible.

Finally, we wanted to implement extra functionality such as an undo button that can revert moves made by the player, the AI opponent, and several other features that would enhance the visual experience for players such as animations and physics for pieces.

Three.js

We used Three.js to focus on loading the models that were vital to making a 3D chess simulator. Three.js has libraries that provide loaders that work well with the free models which we were able to find online. We used orbital controls which allowed us to implement functions for the user to change camera views. Three.js also provided us with raycasting capabilities so that we could select the chess pieces and the coordinates to move them to.

Stockfish

Stockfish is an open source chess engine implemented in C++ that provides the user with a comprehensive evaluation of a board state. The board state contains the complete information of the chess game such that it could be restarted again at a later time, and it is written in a notation called a FEN string. While Stockfish does not natively have a REST API, there is an external interface that allows browsers, HTTP Clients, and XMLHTTP Requests access to the same comprehensive evaluations by simply providing a FEN String, difficulty, and mode through a URI, following the format: *stockfish.online/api/stockfish.php?fen=:fenstring&depth=:difficulty&mode=:mode*

Stockfish then gives a JSON response consisting of a data string. When using the optional Mode.Move, the data provided is the best move for the AI to make.

Implemented Features

We are able to implement the majority of our proposed features. The chessboard is available and the 3D chess pieces can be interacted with and moved around the board. We implemented an HTML interface for the user to interact with certain game functions such as: undo, toggle AI, camera options, and start a new game. We used several light sources to illuminate the scene such as ambient lighting, point lighting, and directional lighting. An environment is set up around the board to simulate the feeling of a real room, with some furniture scattered around. With this, the majority of our proposed goals are implemented.

Unimplemented Features

Throughout the development cycle of this term project, there were still features that we did not get to implement within the time frame. Special moves such as en passant, castling, and pawn promotion were not implemented for this chess environment. We also haven't implemented rule enforcement and checks, which leaves our simulator less usable for those not very familiar with chess. Along with these moves, features such as collisions, animations, sound effects, and other board game options were some other environmental improvements that ended up beyond the scope of our project.

Sources:

Chess Models: https://free3d.com/3d-model/low-poly-chess-black-white-48044.html

Fishstock: https://stockfish.online/

Three.js: https://threejs.org/