CIS 434 Software Engineering

Group 1

Unity Chess

April 28, 2023

Nadia Cannon, Angelo Figueroa, Fabio Hinojosa Jimenez,

Lukas Miciunas, and Shakeeb Rahman

Individual Contributions

Lukas Miciunas

- Determined and wrote up the plan for development as outlined in the objectives and methodology sections of the proposal
- Slides in proposal and final presentations about objectives and methodology
- Programing:
    - Functionality of piece classes including finding valid moves and related helper methods
    - Move validation in the game controller logic
    - Player class
    - Game over screen with ability to quickly start a new match
    - Involved with turn passing and enforcement
    - Fine tuning and bug fixing many aspects of gameplay including piece capture and winning the game
- Final report: abstract, objectives, project description

Angelo Figueroa

- Slide in Proposal
- Programming
    - Get prefabs of chessboard and pieces from unity store
    - Have a timer for each player
    - Change turns after each move
    - Piece movement
    - Generate chessboard

    - Generate chess pieces and set to each team color
    - Set chess pieces in correct position

Shakeeb Rahman

- Defined the scope of the project.
- Contributed to the development of key slides for the project presentation such as challenges faced, key takeaways, and team introduction.
- Created and managed the group's GitHub repository.
- Conducted extensive research on Unity game engine and chess to understand the technical aspects of the project.

Nadia Cannon

- Researched the project
- Did some slides
- Programming:
    - Win state function
    - Capturing function
    - Involved with user input

Abstract

The goal of this project was to develop a multiplayer chess game that is both engaging and functional using the Unity game development environment. The game is designed to implement the traditional rules of chess in a familiar and player friendly way that is accessible to all levels of skill. There were multiple stages of development including a planning and research phase, early development of the base game, and the final stages of development where complex game mechanics were implemented and tested. The 3D graphics present in the game were a pivot from the planned 2D graphics with the purpose of creating a more realistic experience akin to sitting down to play chess with a friend. The hope for this project is that it provides an entertaining and educational experience, while also fostering community through multiplayer gaming.

Objectives:

1. Configure Unity project for use with Github
2. Acquire pre-built 3D models for the game pieces
3. Write classes:
    + Player
        - Team affiliation
        - Which pieces belong to the player
    + Board
        - 8 by 8 array of tiles as the playing field
    + Piece
        - Team affiliation
        - Board position coordinates
        - Subclass for each piece type (pawn, king, rook, etc.). Describes movement ability with an array of valid coordinates
    + Time Control
        - Time remaining for each player
4. Move validation - A player cannot put themselves into a check position. The player will be forced to re-take their turn, or if no legal move is available the game ends in a draw.
5. Graphical movement - Pieces move to the correct space on the board, and disappear if captured.

6. Implement capturing pieces - A piece is captured when an opponent's piece ends its move on the same occupied space. Pawns have a special case of being able to move diagonally in order to capture a piece.
7. Checking for check - If a King piece is able to be captured on the opponent's next turn a state of check is declared. After every move all possible moves by all pieces must be reviewed in order to determine if there is a check.
8. Checkmate - Once a state of check is declared, it must be determined if there is some move that can be made by the player in check that will save their King from capture. If there is no such move available then checkmate is declared and the game ends.
9. Enforcing turns - A player must be disabled from moving their pieces when it is their opponents turn
10. Implement a computer controlled opponent - Using a min-max algorithm, design a set of functions that will simulate a real opponent.

Project Description:

Unity Chess is a facsimile of real world chess games using 3D models to bring a feeling of realism to this computer program. The main feature of the project is a simulated chess board with pieces that can be picked up and moved in order to facilitate a game of chess. To that end, the application enforces the rules of the game. With the white team playing first, players take alternating turns moving one piece according to the movement rules defined in the piece classes. Turn order is enforced; a players' pieces are unable to be moved when it is their opponents' turn. To make a move the player must click-and-hold a piece, then drag it to a space on the board. If this move is valid for the specific piece then it will be placed on the board in the desired location. If the attempted move is not valid, the piece will be placed back on the space that it was previously on.

A players' own pieces will block their other pieces from moving through that position with the exception of the knight. The knight piece is special in that it jumps from its initial position to its destination effectively bypassing pieces that may be in the way. The pawn also has a unique move set in that it can move diagonally only if it is to capture an enemy piece. A player can capture an opponents' piece by moving their own piece onto the same square occupied by an enemy, as long as it is a valid move. When a player captures their opponent's king piece they are declared the winner of the game, and a game over screen appears over the game board. From there a new game can be started immediately, or you can return to the main menu. The main menu of the game contains two buttons that allows the user to start a game of chess, or exit the application entirely.

At the outset of the project it was immediately decided that Unity would be the best option for creating a visually appealing video game given the time and resource constraints. It can be called into question whether or not it was a good choice to go with Unity, as the group had practically zero experience with it going in. The first two weeks of the project consisted almost entirely of getting acquainted with the Unity editor, and learning about its different game object and component classes. While this did delay the start of work on the actual end product, becoming proficient with Unity proved to make parts of the project easier than expected.

The work was split into components by functionality: game generation/initialization, classes for the main game elements (board, pieces, player, time control, and menus), piece movement, piece capture, and the end game check. Game generation was the priority issue, and solving that first allowed for easier testing of the other components. This was closely tied to the creation of the board class as well. The piece classes were the next objective. The difficult aspect of the piece classes was determining the moves available to a piece from any given space. Given a starting position in X,Y coordinates, the method calculates new coordinates for possible moves based on the piece type. Helper methods were created to check potential paths for obstruction by friendly or enemy pieces. The piece in question would then contain a list of coordinates that represented valid moves. When piece movement was first coded, a piece could be moved anywhere on the board at any time. Incorporating a pieces' valid moves list allows the game to validate a potential move before letting the piece be placed. The mouse controls for moving pieces uses a raycast system where a ray is

cast from the position of the mouse into the game screen. It will then return any game object that the ray intersects with. This is how a piece is selected in the 3D environment. Turn order was enforced using a boolean flag that declared whether or not it was the white teams' turn.

Project Timeline:

Project Proposal

Cleveland State University

Project Start: Wed, 3/22/2023

TASK	ASSIGNED TO	PROGRESS	START	END	Week 1		Week 2	Week 3	Week4	Week 5
						Mar 22-28	Mar 29-Apr 4	Apr 5-11	Apr 12-18	Apr 19-25
Planning and Research			3/22/2023	3/28/2023						
Team task		100%								
Design			3/29/2023	4/5/2023						
Fabio Hinojosa Jimenez		70%								
Nadia Cannon										
Shakeeb Rahman										
Lukas Miciunas										
Programming			4/2/2023	4/15/2023						
Angelo Figueroa		20%%								
Lukas Miciunas										
Art Assets			4/5/2023	4/20/2023						
Shakeeb Rahman		10%								
Test			4/19/2023	4/26/2023						
Team task		0%								
Deployement			4/27/2023	5/1/2023						
Team task		0%								

Week 1 (Research and Design):

- Goal: Conduct research on Unity game engine, chess game mechanics, and AI algorithms to inform the design of the game.
- Conducted research on Unity game engine and chess game mechanics, and identified key features to include in the game such as game modes, user interface, and multiplayer option.
- Explored various AI algorithms for chess and discussed their implementation with the team.
- Designed the game mechanics and user interface, and created a project plan.

Week 2-3 (Programming and Implementation):

- Goal: Develop the chess game using Unity game engine, implement the designed game mechanics and user interface, and add AI algorithms for single-player mode.
- Collaborated with team members to develop the game mechanics and implement the user interface using Unity game engine.
- Worked on implementing AI algorithms for the single-player mode, but faced challenges and had to revise the approach.
- Created a functional two-player mode.

Week 4 (Deployment):

- Goal: Presentation of the chess game including live demonstration where we showcased game features.
- Finalized the project report

At the end of the project, the team was able to create a functional chess game with a user-friendly interface and basic sound effects. However, the team was unable to deploy AI for single-player mode and didn't include music in the art assets. Despite these challenges, the team was able to complete the project on time and deliver a satisfactory product.

Conclusion:

In conclusion, the chess game project was a challenging but rewarding experience for the team. Despite facing technical difficulties and time constraints, the team was able to successfully develop a functional chess game using Unity game engine. The project required extensive research, collaboration, and testing, which helped the team gain valuable insights and skills. Although the team was unable to deploy AI for single-player mode and didn't include music in the art assets, the final product was satisfactory and met most of the project goals. Overall, the project provided an opportunity for the team to apply their knowledge and skills in game development and to work as a team to achieve a common goal.

Future Work:

- Expanding the game to include online multiplayer options
- Implementing AI algorithms for intelligent and engaging gameplay that uses some sort of Algorithm (minmax)
- Adding special moves (e.g castling, en-passant)
- Adding animations to piece movement
- Option to surrender
- Finishing check validation
    - As it stands a player must actually capture the opposing King, or run out of time on their turn clock in order to end the game.
- Add camera control so you can look around the board.

References:

1. “FIDE Laws of Chess.” *FIDE*, International Chess Federation, https://www.fide.com/FIDE/handbook/LawsOfChess.pdf.
2. Hartikka, Lauri. “A Step-by-Step Guide to Building A Simple Chess AI.” *FreeCodeCamp.org*, FreeCodeCamp.org, 15 Mar. 2017, https://www.freecodecamp.org/news/simple-chess-ai-step-by-step-1d55a9266977/.
3. Jurtz, Marcel. “Building a Chess Game in Unity.” *Medium*, Medium, 12 Sept. 2018, https://jurtzmarcel.medium.com/building-a-chess-game-in-unity-85408f86b749.
4. “Unity Documentation.” *Unity Documentation*, 24 Mar. 2023, https://docs.unity.com/.