Chapter 1

Evaluation

Overall, I believe that my final program has effectively achieved the requirements set out by my client. It is a step up to current alternatives that I have found online, and seems to be a solid choice for anyone wanting to play laser chess digitally.

1.1 Objectives

1.1.1 Objective 1

All laser chess game logic should be properly implemented. Refer to Figure **??**.

Both the play and review screens display a 10x8 laser chess board. The laser fires automatically after every move, and follows the trajectory as directed by the pieces, destroying any that are facing the wrong way (Objectives **??**, **??**). All the pieces reflect and are destroyed by the laser correcty, with the colour to move alternating between both players (Objectives **??**-**??**). The game ends when a player either draws or resigns, has their pharoah destroyed, or reaches three-fold repetition (Objectives **??**, **??**).

The game logic runs successfully, Objective 1 is fulfilled.

1.1.2 Objective 2

Game should process user input correctly. Refer to Figure **??**.

Players can select, move and rotate pieces on their respective turns, through clicking squares, rotate buttons or by dragging the pieces (Objectives **??**, **??**, **??**, **??**). The game cancels any invalid moves (Objective **??**). When a player holds down on a piece, the piece is moved through the drag-and-drop mechanism, and is released on the hovered square (Objective **??**).

Objective 2 is fulfilled.

1.1.3 Objective 3

Save or load game options should be implemented. Refer to Figures **??** and **??**.

In the browser screen, users can scroll through all previous games, with the option to delete or review them, or copy their FEN string (Objectives **??**, **??**, **??**). Games can be sorted, and pages of games can be scrolled through (Objective **??**). The game positions are encoded using a custom FEN string format, and games are saved as rows in a local SQLite database table (Objectives **??**, **??**). In the review screen, users can scroll through all the moves of a previous game (Objective

??). The right side bar displays information such as the winner and move number, and the timer, pieces and move list updates accordingly (Objective **??**).

Objective 3 is fulfilled. A possible improvement would be to add the option of loading up a position to resume playing straight from the review screen. This would be a convenient feature, but is not a priority, as users can currently copy the FEN string from the browser screen to the config screen and load the position there as a current alternative.

1.1.4   Objective 4

Other board game requirements should be implemented. Refer to Figure **??**.

In addition to the core game mechanics, other ancilliary aspects are added as well. Timers can be enabled and disabled, decrement correctly, and end the game when they run out (Objectives **??, ??**). Buttons also exist for drawing and resigning the game (Objectives **??, ??**).

Objective 4 is fulfilled.

1.1.5   Objective 5

Game settings and config should be customisable. Refer to Figures **??** and **??** and **??**.

In the settings screen, the user can change settings to toggle the volume, fullscreen, board colours, particles and shaders (Objectives **??, ??**). All settings update correctly, and are saved to a local JSON file when the settings screen is exited. The game configurations are changed in the config screen. Here, users can change the game's starting colour, configure timer settings, change to the CPU player and specify the difficulty, select a board preset, or create a custom board layout through the editor screen (Objectives **??-??**). The editor screen comes with all the basic operations of placing pieces, rotating pieces, and erasing and dragging them, but also helpful buttons to erase or reset the board (Objectives **??-??**). Another method to change the board layout is through inputting a FEN string, and the board either updates or displays an error message depending on the validity of the string (Objectives **??, ??**).

Objective 5 is fulfilled.

1.1.6   Objective 6

Game UI should improve player experience. Refer to Figure **??**.

Indicators are present to highlight the selected square and any adjacent squares that it can be moved to (Objectives **??, ??**). Audio cues also improve the immersiveness of the game, as well as the laser visuals and particle effects (Objective **??**). THe mouse cursor was also modified to help user-friendliness, by changing its icon depending on the current action being performed, for example, a closed hand icon for grabbing a piece (Objective **??**). The move list and status message widgets are positioned on the right of the game screen, update accordingly and provide information on the game (Objectives **??, ??**). The piece display also correctly updates to show all destroyed pieces (Objective **??**).

Objective 6 is fulfilled.

1.1.7   Objective 7

GUI design should be functional and display concise information. Refer to Figure **??**.

I have created most of the custom pixel art and icon graphics to improve the game's visual and attempt to make it more engaging (Objective **??**) There are currently 7 screens in the game: menu, settings, config, editor, game, browser and review, each containing their own eponymous function (Objective **??**). Most screens also contain both the main menu button and help button.

Clicking the menu button switches to the main menu screen, and clicking the help button shows an overlay displaying the functionality of drawn widgets (Objective **??**). Shaders are also used to improve the game's visuals, such as by giving the laser a light-emitting effect, but is also used to draw all the backgrounds (Objective **??**). The program runs well, with moments involving heavy calculation being helped by multithreading, and also resizes seamlessly (Objective **??**, **??**).

Objective 7 is fulfilled. Although the program runs smoothly on my device, some fragment shaders are computationally expensive, and may therefore affect the performance. I have attempted to resolve this by adding the option to disable shaders, however, a better solution would be to optimise the expensive shader code, for example, by manually calculating masks using widget dimensions instead of the in-built Pygame mask function. Additionally, working on Windows, I did not bundle the program to MacOS or Linux. Although this was not a priority, having access to other OS testing environments would've widened access to the game.

1.2 Client Feedback

The following interview was conducted after Mr Myslov received and played through the program.

Q: Did the final product meet your expectations?
A: Yes, it certainly did. The game plays perfectly, I could move, rotate all the pieces as expected, and the laser works well. All the features I requested for were there, and the visuals were also more than I expected and welcome.

Q: What improvements could be made to the program?
A: I found that it was difficult for my laptop to run the program without sacrificing some of the visuals. I also found your FEN string implementation confusing, as there was no guidance on what the characters meant, or what the string represented. Figuring out how to play the game was also initially difficult, as I didn't know what the pieces represented through their icons, or how they would deal with the laser.

Q: What other aspects would you like to have been added?
A: Being able to somehow easily obtain a digital file for a game would be useful, so I could share it with my students easily. Having a multi-player option instead of people surrounding one laptop would also be a great addition.

Discussion

My client was satisfied with my final program. His perspective revealed some aspects of the program that could've better considered and improved upon.

User-friendliness was a point raised by Mr Myslov. He highlighted the fact that it would be hard for beginners to understand the rules of laser chess, which I agree on, considering that the rules and pieces are very different from regular chess. To amend this, I could add more tutorials or help dialogs, for instance, by creating a new tutorial screen containing a miniaturised board, where the user can learn how the pieces interact with the laser; although a help screen is already implemented, an interactive one would perhaps serve the purpose more efficiently. A similar approach could be applied to my custom FEN string and algebraic move notation.

Sharing games   was a suggestion made by Mr Myslov. Although the FEN string system allows for sharing of board positions, I realise that there is value in also sharing the entire progression of the game. This could be implemented by exporting the game to the equivalent of a chess 'PGN' file, by writing all the game information (e.g. CPU depth) to a .txt file, and then the contents of the move list.

Multiplayer   was another improvement suggested by Mr Myslov. Admittedly, this is a big feature for a digital board game, however, I did not implement this as it was not a requested objective, and would've required restructuring my code for handling moves and the game logic. This could be achieved within Python through a peer-to-peer system, sending and parsing formatted moves either through pickling or JSON objects via HTTP methods using the `requests` module, although it would also require an overhaul of the GUI and screens to allow the option for multiplayer.

Regardless, given the limited time frame, we were both pleased with the final result.

1.3   Further improvements

A point not raised by Mr Myslov was the ease-of-use of the widgets. The one that I found concerning was the text input, as constantly deleting the entire FEN string, and having to manually press arrow keys to edit specific characters was tedious. I have tried to improve this by adding shortcuts such as `Ctrl+Backspace` to delete the entire contents of the text Input. To improve the text input to make it feel more familiar to standard ones, I would have to use two pointers to track the indexes of the start and end characters, for the ability to highlight and delete chunks of text. The text displayed can also be hard to read currently, especially if there are a lot of characters. This could be improved by either fixing the font width, detecting if the text width is greater the widget width and implementing text wrapping, or allowing text scrolling.

1.4   Conclusion

The final product fulfilled all proposed objectives and more, and has been approved by my client.

Throughout 6 months of development, I felt that I improved on many aspects needed to be a proficient developer. Creating bespoke software allowed for conducting research, interviews, and also obtaining continuous feedback. In terms of coding, working on a large project helped in improving file management, documentation, and keeping robust backups through Git. It also gave me the oppurtunity to learn GLSL for my shaders, LibreSprite for pixel art, and LaTeX for my documentation. Having a greater familiarity with Pygame, I also plan to rework my widget system as a library, to be shared for the benefit of all future Pygame users.

Overall, this project was a success.