Concurrency Based Board Game

Introduction

The aim of the project is to develop a game-playing environment that provides a nice graphical interface and the ability to play multiple games at the same time i.e. play multiple of a games concurrently.

Aims

- A game environment with an intuitive user interface that the user can interact with to play a board game ✓
- Allow concurrent behaviour by letting the user play multiple games at once ✓
- Allow the user to play different types of board games through the same program ✗
- Have multiplayer support so that multiple users can play against each other from separate devices ✓
- Have a chat feature so that users playing against each other can communicate ✓

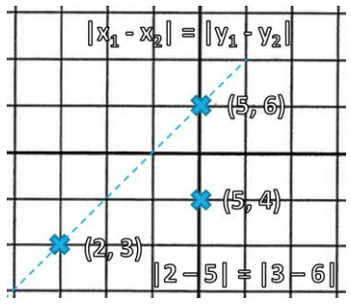


Figure A

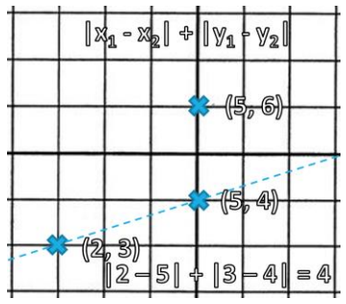
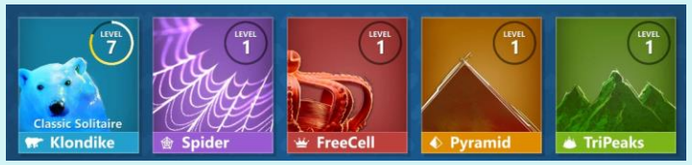


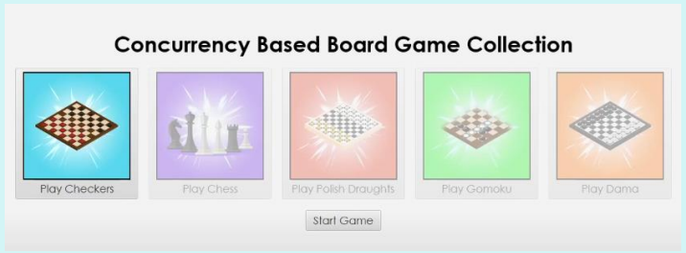
Figure B

Background

Microsoft Solitaire Collection provides an environment for playing multiple different types of card games, which each share similar rules and functionalities.



I used this as inspiration for my project, aiming to create a similar environment, except for board games, allowing the user to not only play multiple of the same game at the same time but to play multiple different games through the same environment.



Each of these games share similar features such as using the same board or same type of board.

Due to multiple issues I was only able to implement the checkers board game in this project. These issues were mainly to do with poor scheduling, related to underestimating how long some parts would take, and not accounting for the time spent on other work.

Interesting Parts

When researching I found more efficient ways to check the legality of moves. In checkers a move must be diagonal to its original position.

If $|x_1 - x_2| = |y_1 - y_2|$ then the two points are diagonally aligned. (Fig A)

It is also sometimes useful to know the exact distance between two points as the distance of a move can affect the legality of that move.

$|x_1 - x_2| + |y_1 - y_2|$ gives the Manhattan distance between two points. (Fig B)

In order to achieve multiplayer support I had to create a new thread that constantly checks for messages being received, when a move is received the server thread has to communicate with the interface thread to update the board. (Fig C)

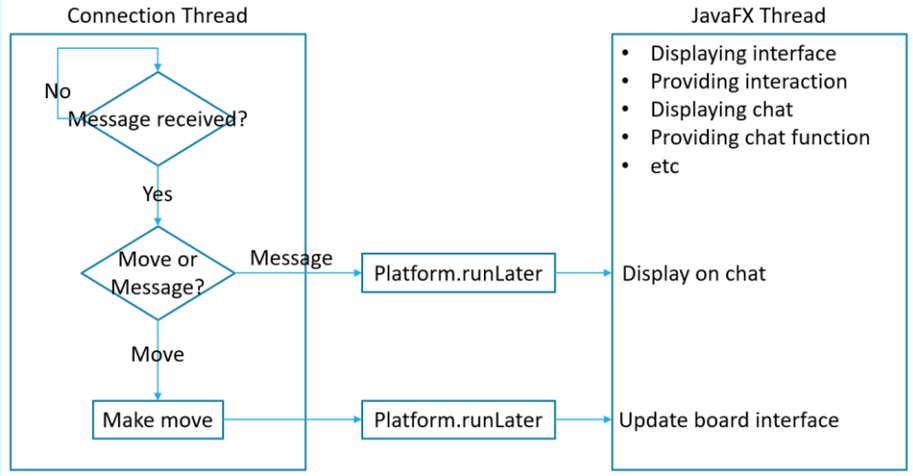
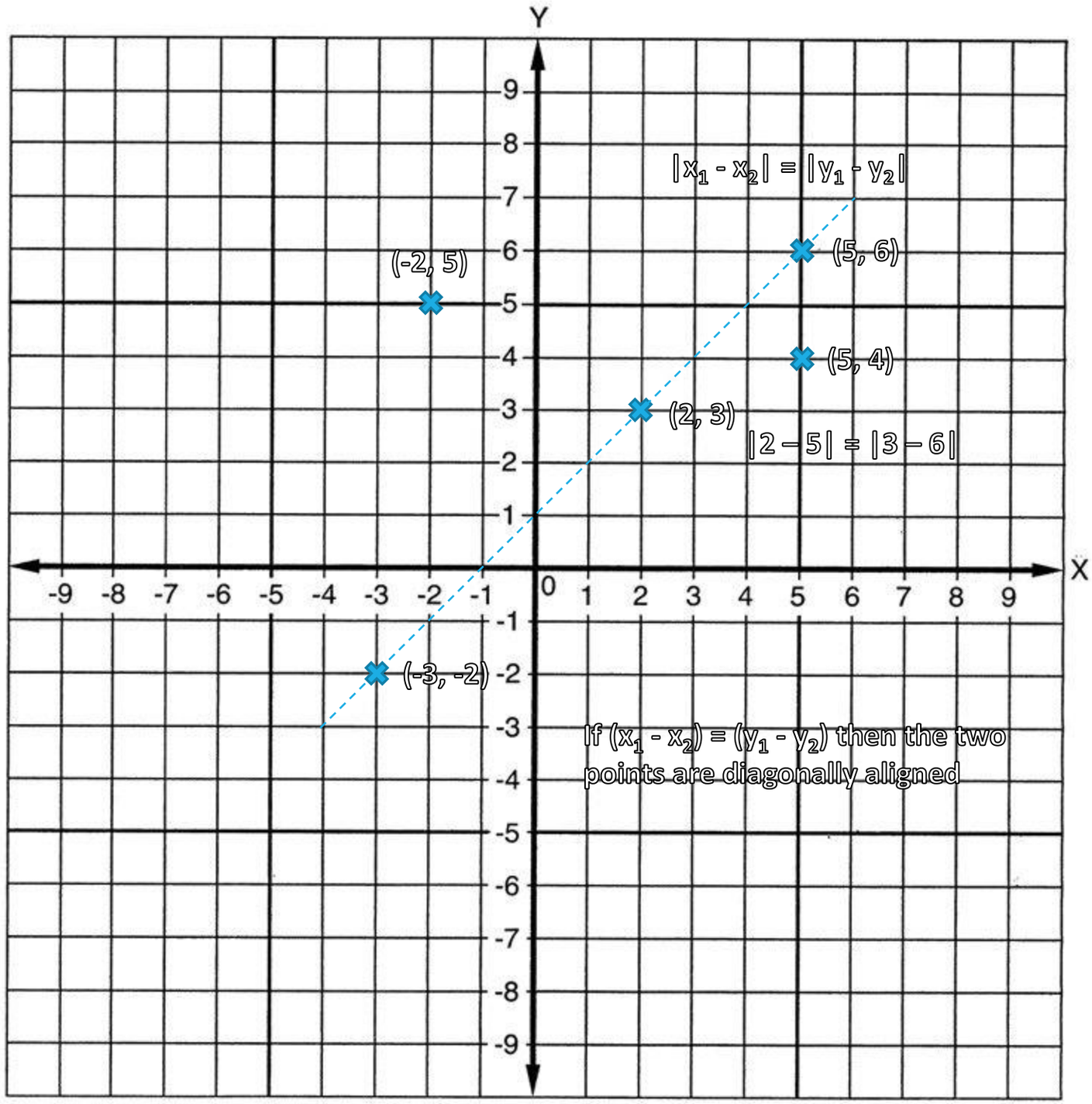
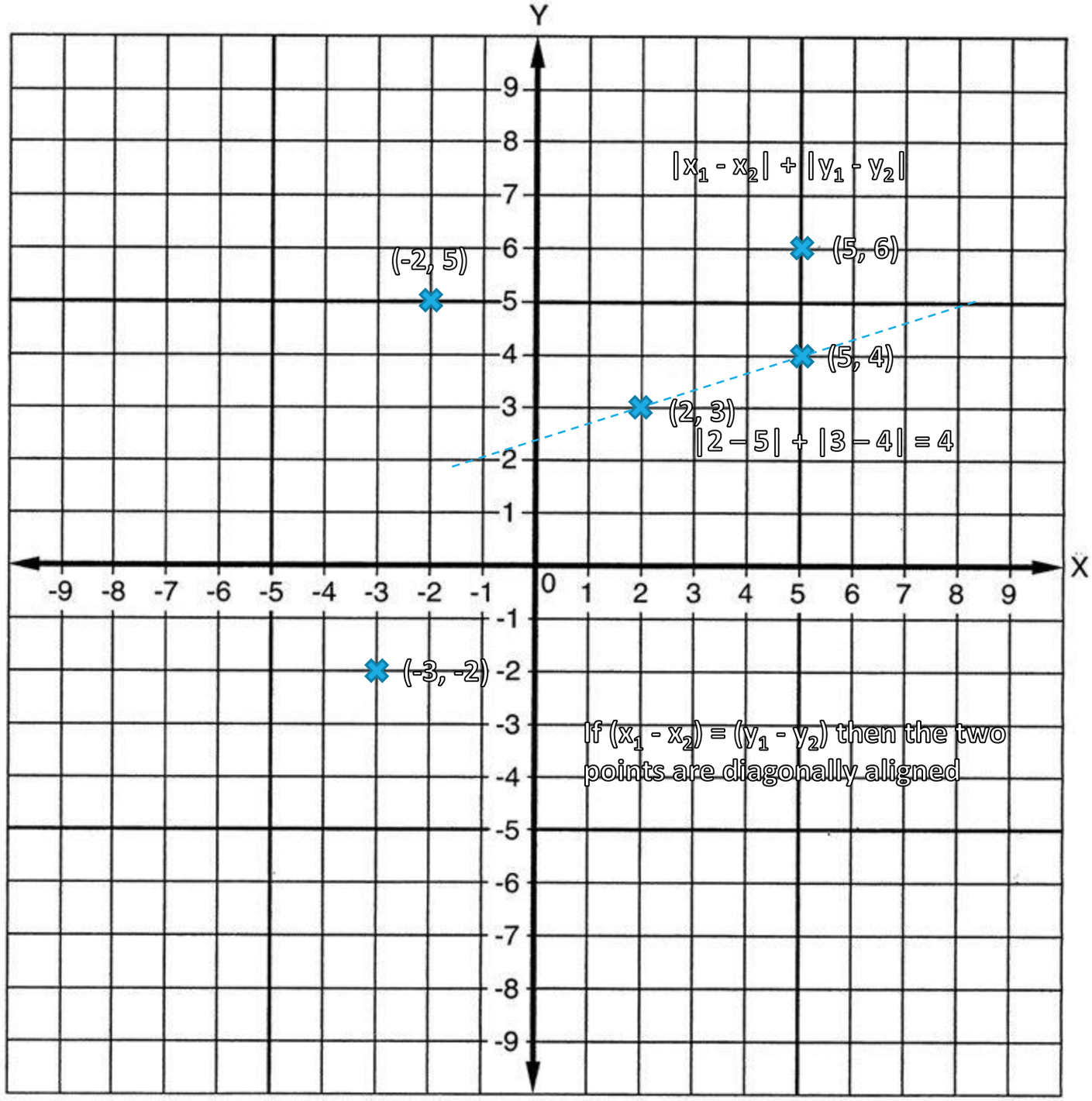
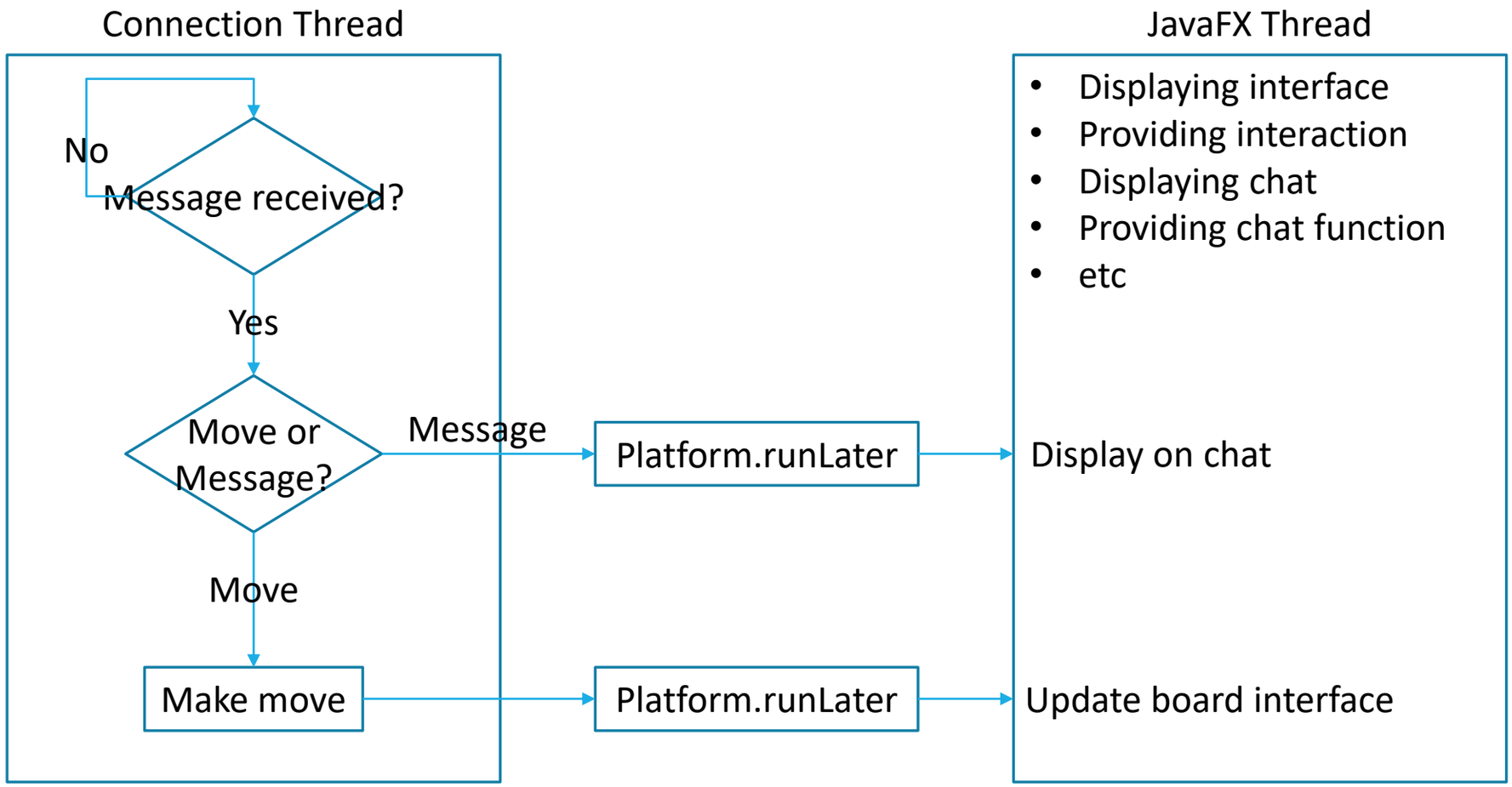


Figure C


Y
X
$|x_1 - x_2| = |y_1 - y_2|$
(-2, 5)
(5, 6)
(5, 4)
(2, 3)
$|2 - 5| = |3 - 6|$
(-3, -2)
If $(x_1 - x_2) = (y_1 - y_2)$ then the two points are diagonally aligned


Y
X
$|x_1 - x_2| + |y_1 - y_2|$
(-2, 5)
(5, 6)
(5, 4)
(2, 3)
$|2 - 5| + |3 - 4| = 4$
(-3, -2)
If $(x_1 - x_2) = (y_1 - y_2)$ then the two points are diagonally aligned


Connection Thread
No
Message received?
Yes
Move or Message?
Message
Move
Make move
Platform.runLater
Platform.runLater
JavaFX Thread
Displaying interface
Providing interaction
Displaying chat
Providing chat function
etc
Display on chat
Update board interface