If player is inactive for a certain amount of time or has closed the game, it should be a win for the opponent.

Pairing systems:

Add friends via code – creates a lobby where the same players are able to play together

Random matchmaking queue – server sorts players who want to play together and matches them.

Queue rankings:

Increased amount of time waited = increased priority in the queue.

|  |  |
| --- | --- |
| Objective: | Deadline: |
| Monte Carlo Tree Search in python working with simple tic tac toe with a 95% win rate (if human player performing at best) | 6/6/19 |
| Monte Carlo Tree Search in python working with Ultimate Tic Tac Toe with a 95% win rate. | 24/6/19 |
| Set up networking between server/client so two computers can exchange information. | 2/7/19 |
| Set up the networking system with Ultimate Tic Tac Toe. | 9/7/19 |
| Set up systems which allow the two computers to be paired via different means:  Sending friendly codes to join the same lobby,  Or joining random games. | 19/7/19 |
| Set up the full networking system allowing the two players to play together:  Queue will be prioritized based on amount of time waited, and number of times the player has left an unfinished game while the other player has not left. | 30/7/19 |
| 3 different difficulties implemented for the Ultimate Tic Tac Toe, with variable win rates. | 6/8/19 |
| Set up a co-op mode where 2 players can play on the same computer. | 10/8/19 |
| Set up clean menus and easy to understand input/output. | 14/8/19 |
| Clean up the code and make sure it is as efficient as possible + testing | 25/8/19 |

Increased number of unfinished games left = decreased priority in the queue.

Final = 11/9/19

Finish by September – 1/9/19

22/5/19 – 1/9/19

14 weeks, 3 days

Ultimate tic tac toe AI – 8.5

Networking – 6

Co-op, clean up, menus, UI - 2