Initial Plan

Co-operative Turn Based Strategy Game with AI and Networking

Josh Hornsby
Supervisor: Dr Frank Langbein

One Semester Individual Project (CM3203)
40 Credits

1 Description

The project will be a turn based strategy game with two player cooperative gameplay. The game will include a user interface for each player, allowing them to command their units one after another, and networking to connect the two players' games together. At the end of each of their turns, the AI will command it's units, until all of one teams units are dead. The interface for commanding units will be the same for each player type (local, remote and AI) in order to abstract this behavior.

Since the game functionality is not the focus of the project, there will only be a few simple unit functionalities, namely: moving, shooting nearby units and throwing grenades. It should however be relatively simple to implement new functionalities into the client and have them be supported by local, remote and AI players. I.e. the final project should be a base game that is easy to expand and add more gameplay features.

The networking will be peer to peer, where each client sends its action to each remote client. The project does not attempt to solve problems such as cheating as it is presumed that the two cooperative players know eachother and are not randomly paired players. Since the game is turn based, it removes any chance of contention as each player acts one after another and not in parallel.

The AI will be run on the host (the person who sets up the game) and commands from the AI forwarded to remote players. During the project time will be spent researching what AI technique to use, but the current plan is to implement something more sophisticated than minimax. The two directions currently being researched are a more advanced search model than regular minimax (Monte Carlo tree search, beam minimax, and narrowing beam minimax), but the project would a reflex model using intelligent agents that pick the best move based on their evaluation function. Depending on the amount of time left in the project, it is possible to create the weights of the evaluation function by hand, or by using some learning algorithm such as temporal difference learning in order to procedurally generate weights.

The game will be implemented in C/C++ using the data oriented development approach. Low-level libraries to simplify tasks like image loading and operating system specific details like opening a window. A low level graphics API will be used in order to support multiple platforms and graphics APIs (like OpenGL, Vulkan, DirectX). The engine will be created on top of these simple low level libraries to allow the best performance and control over the game.

2 Aims and Objectives

- C/C++ Game Engine
  - Cross-platform (Windows, Mac OS, Linux)
  - Basic diffuse shader graphics
  - Simple user interface
  - Fast nearby entity lookup using quad trees
  - A* path-finding around cover to location
- Turn based strategy gameplay
  - Unit stats
  - Unit actions (move, shoot, throw, etc)
  - Each player (AI/remote/local) gets a turn in round-robin fashion
- Networked co-op
  - Join other players game
  - Keep game state consistent between multiple clients
  - See the other player take actions
  - See the AI player take actions
- AI player
  - Take the same actions as a local or remote player
  - "Challenging" gameplay without a large unit advantage (i.e. the AI should be smart enough to win a game vs a regular player without having many more units than the player)
  - Use cover effectively in the game

3 Work Plan

The work plan includes 12 weeks of work, which leaves 3 weeks of float time to account for any unexpected delays. These 3 weeks will also include time to complete the final report. The AI tasks are a bit more abstract than the other weeks due to the lack of a decision on the exact AI method to be used. The review meetings are in week 6 and week 11 of the project.

Week Number	Tasks	Deliverables	Meetings
1	**Initial Report and GUI**	Initial Report	Meeting with Frank (31/01/19, 3pm)
	Plan out project week-by-week		
	Complete Initial Report		
	Create GUI shader		
	Draw GUI images		
	Invoke callback function on click		
2	**Basic Gameplay Actions**	Single-player turn based game	Meeting with Frank (07/02/19, 3pm)
	Add unit health		
	Allow units to move through GUI		
	Units shooting other units		
	Units throwing projectiles		
3	**Networking Implementation**		Meeting with Frank (14/02/19, 3pm)
	Join remote game through IP		
	Communicate unit movement		
	Communicate unit shooting		
	Communicate unit throwing		
4	**Networking Implementation**	Co-operative turn based game	Meeting with Frank (21/02/19, 3pm)
	Round robin turns		
	Communicate enemy spawning		
	Communicate game end		

Week Number	Tasks	Deliverables	Meetings
5	**Quad Tree Entity Locations**	Performance analysis	**Review Meeting** (28/02/19, 3pm)
	Create quad tree implementation		
	Change entity list from a regular linked list to a quad tree		
	Compare quad tree performance		
6	**AI Player**		Meeting with Frank (07/03/19, 3pm)
	Decide AI algorithm		
	Start AI implementation		
7	**AI Player**		Meeting with Frank (14/03/19, 3pm)
	General AI implementation		
	AI moving actions		
8	**AI Player**	Game with AI	Meeting with Frank (21/03/19, 3pm)
	AI shooting actions		
	AI throwing actions		
9	**Cover System**		Meeting with Frank (04/04/19, 3pm)
	Create cover on map		
	Determine if unit is in cover		
	Low chance of hit if in cover		
10	**Cover System**	Game with player cover system	**Review Meeting** (11/04/19, 3pm)
	Check shot for line of sight		
	Remove cover if flanked		
11	**AI Player**		
	Implement cover into AI		
12	**Final Polishing**	Finished co-op AI game with cover system	
	Finish AI using cover		
	Test AI in various scenarios		
	A* path to move around cover		