

Coding Requirements

Game Description:

The game being designed is a turn based board game. The game is played on hexagonal grid system of either size 5 or 7 sides of the hexagon depending on the number of teams at play. In the game there are either 2, 3, or 6 teams. Each team has three robots with statistics according to fig.1. These statistics determine the attack damage, health, movement distance, and viewing range. Each robot can only see as many spaces as it's viewing range and can only move as far as it's movement range. Players take turns moving each of their robots one at a time and shooting the other teams robots to try to eliminate the other team(s) and win the game.




Scout	Sniper	Tank	
			
A:1	A:2	A:3	Attack
H:1	H:2	H:3	Health
M:3	M:2	M:1	Movement
R:2	R:3	R:1	Range

figure 1

Main Menu Actions:

The main menu of our system will consist of a number of options that the user can select to play different game options.

Person selects two player

System

- Load game board of size 5 on each size
- Initialize player's stats
- Assign starting locations to each robot of each team
- Assign direction to each robot of each team

UI

- Clear screen
- Blit image of 5x5 game board
- Blit images of players

Person selects one player vs comp

System

- Load game board of size 5 on each size
- Initialize player's stats
- Assign starting locations to each robot of each team
- Assign direction to each robot of each team

UI

- Clear screen
- Blit image of 5x5 game board
- Blit images of players

In Game Actions:

The game consists of rounds and turns.

Each round consists of 0-3 turns per player based on how many robots each player has left in the game. In each round every player takes as many turns as the number of robots they have left alive.

A turn consists of a player being able to move and shoot with one robot of highest movement stat that they haven't used yet in the current round. The player can move this robot up to a certain number of spaces based on the current robot's movement statistic. The player can shoot with this robot once up to a distance specified by the robot's range statistic.

When the player is in turn with a certain robot the player can see the board based on a field of view (FOV) that the robot has. Each robot has a certain range that they can see. This FOV determines not just the visual representation of what the robot can see but also how far the robot can shoot. If the robot can see another robot it can also shoot it.

It is a player's turn and they select a position to move to.

- *Check if that position is in the game board*

- if that position is a valid position to move to:

 - player's robot is now sent to that location

 - player's robot's movement left variable is subtracted the amount the player

moved

 - update system variables of position and stats of that robot

- else

 - error message shows explaining why the player cannot move their robot to that location(not in range, out of the board, etc.)

It is a player's turn and they select a position to shoot.

- if that position is a valid position to shoot:
 - all robot on that position experience damage equal to the damage stat on the shooter robot
 - set variable stating that the current robot has shot
 - start shooting animation(if one has been implemented)

 - check the number of robots of each team
 - if the number of robots on one team reaches zero
 - display a YOU LOSE message to that team's player if it is a human
 - if the number of teams alive is 1
 - display a YOU WIN message to that player of the winning team if it is human

- else
 - error message shows explaining why the player cannot shoot in that location(not in range, out of the board, etc...)

It is a CPU's turn.

- System calls the robot's turn who it is function with all required stats that the function would need to make a decision
- The system would use the return of the function and do what the function wanted to do on that turn.
- if the function called a move for the robot:
 - move the robot to the specified location
 - update system stats of what robot is where
- if the function called a shoot for the robot:
 - all robot on that position experience damage equal to the damage stat on the shooter robot
 - set variable stating that the current robot has shot
 - start shooting animation(if one has been implemented)

 - check the number of robots of each team
 - if the number of robots on one team reaches zero
 - display a YOU LOSE message to that team's player if it is a human
 - if the number of teams alive is 1
 - display a YOU WIN message to that player of the winning team if it is human

Player forfeits the game

There will be an option in the game for a player to forfeit.

- display a YOU LOSE message to the player that pressed forfeit
- if the number of players left alive is 1
 - display a YOU WIN message to that player of the winning team if it is human
- else
 - remove the forfeited player's robots from play
 - continue with the next players turn