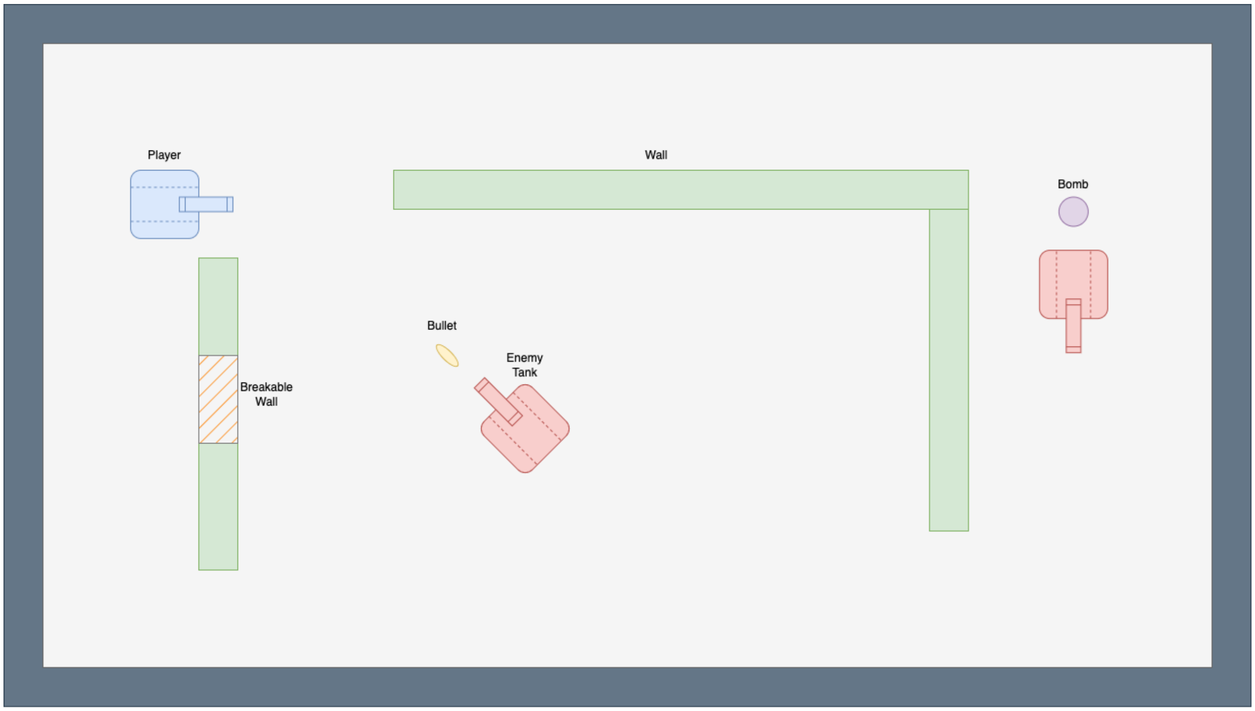
Jeremy Marks

Tank Wars

Game Design

This game will be an action based tank survival game. It is inspired by the WiiPlay tank game, but will be top down with much more simple graphics. The goal of the game will be to survive as many levels as possible. Procedural generation will be used to make incrementally harder levels the longer the player survives. There will be a few different types of enemy tanks depending on the level. Some of the types of enemy tanks will be stationary, random movement between waypoints, and chase the player. The player will be able to drop bombs that explode after a delay and shoot bullets that travel linearly and bounce once off walls. Enemy tanks will have the same weapons as the player. There will be randomized walls to navigate. Some of the walls will be able to be destroyed with a player or enemy bomb. Movement will be from awsd or arrow keys keyboard input. The bullets will be aimed with mouse position and shot with a left mouse click. Bombs will be triggered from the spacebar. Enemy tanks will be controlled by a simple AI. The level will be built using a grid system. The enemy AI will have access to the player position and wall positions. The enemy AI can use these inputs along with a path finding algorithm to find the optimal path toward the player. If there is extra time after the MVP, enemy tanks will get access to which walls are breakable. They will use this info to make new quicker paths toward the player. Player and enemy tanks will both die after a single hit from a bullet or bomb. The goal of each wave/level is for the player to kill all the enemy tanks to move on. As the player survives longer each level will have more enemy tanks as well as enemy tanks with increased stats.



Development Design

The game engine will use the model-view-controller architecture. The model will have classes for all the game objects. Some examples of game objects are player tank, enemy tank, breakable wall, bullet, wall, and bombs. The model will also include a grid that contains walls and tanks. The grid structure will be used for location for procedural generation to put walls. Enemy tanks will also use the grid to find an efficient path to the player. The tanks and bullets can move freely within the grid cells and will not have to be centered in a cell. The cells will be used for a scaled out grouping object locations for more efficient path finding. The model will also contain all game information like levels survived and procedural generation seed. The enemy tank AI will focus on moving toward the player and shooting when there is not a wall between.

The view will load the sprites and update the screen. The controller will manage user input and control all the game objects. User's will use the mouse and keyboard to interact with the game. The keyboard will be used for movement and triggering bombs. The mouse will aim and shoot the bullets. The user will also use the mouse or keyboard to control the start menu screen. The controller will keep track of game time to ensure the game plays as close to the same as possible on all devices. The controller will use the model to have access to all the game objects and then pass all needed information to the view to be shown on the screen.

The biggest technical challenge will likely be the enemy tank AI. The AI will have to be able to perform efficient path finding as well as aim and shoot toward the player. It will be difficult to get multiple tanks performing path finding and aiming at the player if it is not a fast algorithm. Procedural generation may also be a technical challenge because of my lack of experience with using it. I heard we will be covering both of these topics in class, so I will be learning from that as well as looking at what other similar games have done to implement these.

Division of Labor

I will be working by myself. I estimate I will have to spend 4-6 hours per week to get a decent MVP working by the project due date.

Timeline

Milestone 1: March 30

- Implemented rendering and game time - March 27
- Created tank sprite by photoshopping a texture I found online- March 29
- Implemented basic player tank movement to work with wasd and collide with walls- March 29
- Created wall and bullet sprite by editing textures I found online- March 30
- Implemented tank aiming toward mouse and bullets that bounce and move at a consistent speed. - March 30
- Implement procedural generation MVP of walls - March 30

Milestone 2: April 12

- Implement game menu - March 31
- Create bomb sprite - March 31
- Add debounce to tank bullets and limit to 2 shots at a time - March 31
- Test bullets kill enemy tanks and add enemy tanks procedurally- March 31
- Animate tank and bullet textures to rotate- April 3
- Split monolith code into model view controller - April 3
- Implement bombs - April 5
- Implement procedural generation that intelligently connects walls - April 12
- Implement enemy tank AI (A* and shoot toward player when visible) - April 12

Final Game Submission: April 26

Final Exam Presentation & Submission: May 4