Project Plan - Navi

I. Introduction

Team Navi is made up of Nicolas Sim, Christian Roccanova, and Charles Jadin. Our capstone project idea is a 3D video game made in Unity; specifically, the game is a first-person shooter set in a zero-gravity environment, with a format similar to the training battles from the novel *Ender's Game*.

II. Program Description

The user will play as a member of one of two armies that battle one another in a non-deadly clash. If we choose to be true to the book, each army will have 41 soldiers. However, simplicity will likely require smaller numbers (perhaps about half that). User-defined army sizes may be an option as well.

The members of each army float around in zero-G in an arena with large objects placed around it for cover and tactical movements. Each soldier has a stun pistol and attempts to use it to incapacitate members of the other army. The user will have the ability to move; shoot; "jump," or propel the character off of surfaces; and cling to objects for cover or to change direction. Once in motion, a player will stay in motion unless a collision occurs.

The game ends when all members of one team are frozen/stunned. If the human player is frozen, the user can assume command of a different member of the army if there are any left. Other options to be considered for game completion are time limits and objectives (such as entering a gate on the other side of the battle zone). These are not part of the basic vision but can be added if time permits.

III. Structure

This project is going to be an aggregation of various visual, audio, and scripting elements. It will consist of several objects which will require some or all of those types of elements.

The first of these is going to be the game manager, a script-based object which dictates the flow of the game. The game will be started through this object, and it will develop, finish, and produce a result based on the conditions set in these scripts.

Next is the map object. Its scale will have to be determined through trial and error to find an optimal feel, but it is known that the play arena will be an enclosed cube with minimal surface detail. The map will include lighting, textures, and any child objects that are placed within, as well as the combatant character objects.

There will be one human-controlled player, and the rest will be NPCs. Both types will have a weapon object with similar sounds and animations, with different scaling for human-held and NPC-held weapons. Both will also be governed by the same rules of physics. The human player will have controls, a first-person camera view with UI and crosshair, and a script for behavior upon getting shot. Non-human characters will have scripts that dictate their movements, cause them to seek out and shoot characters from the opposing team, and immobilize them when they are shot.

IV. Tools Required

This game is to be developed using Unity for its game engine and C# for the programming language. Pre-built and publicly available 3D assets and/or sounds will be relied upon if accessible. 3D modeling and animation software like Blender will be required for objects and their movements.

V. Task Breakdown

- Map creation - Christian
    - Backgrounds
    - Lighting
    - Objects (cover, terrain etc.)
    - Spawn points
    - Contextual audio

- Character models - Nicolas
    - Textures
    - Meshes
    - Controls

- Character Animations - Nicolas
    - Firing weapon
    - Grasping/clinging
    - Propelling/jumping
    - Getting shot

- Weapon Animations - Nicolas
    - Stun beam/ball
    - Rate of fire/travel

- AI - Charles
    - Battle movements
    - Find targets
    - Shoot targets
    - Behavior in the open
    - Behavior while behind cover

- Physics - Christian
    - Propulsion
    - Anchoring to surfaces
    - Collisions

- Game Management - Nicolas

    - Beginning a game
    - Effects of hitting shots and getting shot
    - Game result

- UI/Menu - Charles
    - Ability to play new game
    - Game Settings
        - User-defined team sizes
        - Map choice

VI. Time Breakdown

A. Charles Jadin

Week 3
- Build basic menu - 5
- Plan movement/lint-of-sight AI scripts and research effective methods - 8

Week 4
- Build basic gameplay UI - 5
- Begin drafting movement/line-of-sight AI scripts - 8

Week 5
- Plan/research combat AI scripts - 7
- Continue work on movement AI - 6

Week 6
- Begin drafting combat AI - 8
- Revisit menu and UI - 4

Week 7
- Continue work on combat AI - 7
- Experiment with group formations - 7

Week 8
- Continue combat AI with focus on environment/object interactivity - 7
- Test and tweak AI to fit team size specification/variations - 6

Week 9
- Refine battle awareness AI and its effects on facing/movements - 5
- General tweaks/debugging in AI - 5
- Add any potential additional features - 4

Week 10

- Continue tweaking and debugging AI - 3
- Test and assist with other areas - 4
- Final product testing - 6

Total estimated hours: 105

B. Christian Roccanova

Week 3

- Research map editing tools - 4
- Build structure of basic arena style map - 6

Week 4

- Research/design collisions - 7
- Apply collision detection to map and map objects - 4
- Apply collision to place-holder character model - 2
- Test/debug collision - 4

Week 5

- Research/design propulsion - 7
- Apply ability to propel from objects - 2
- Apply propulsion to place-holder character model - 2
- Test/debug propulsion - 4

Week 6

- Research/design anchoring to surfaces - 7
- Apply anchoring to map and map objects - 4
- Apply anchoring to place-holder character model - 2
- Test/debug anchoring - 4

Week 7

- Integrate collisions, propulsion and anchoring into finalized character model. - 3
- Test/debug collision, propulsion and anchoring in conjunction with each other. - 7

Week 8

- Apply textures and background to map - 2
- Apply lighting to map - 2
- Research audio integration - 3
- Apply contextual audio to map objects - 2

Week 9

- Construct tutorial level map - 5
- Apply physics - 3
- Apply textures and background - 2
- Apply lighting - 2
- Apply audio - 2

Week 10

- Construct additional standard gameplay map - 6
- Apply physics - 3
- Apply textures and background - 2
- Apply lighting - 2
- Apply audio - 2

Total estimated hours: 107

C. Nicolas Sim

Week 3

- Research character models - 3
- Research character sprite movements - 3
- Research Character object creation in Unity with C# - 4

Week 4

- Test out simple character object creation/addition - 4
- Test out simple character movement - 4

Week 5

- Test out simple character movement - 4
- Test out simple character projectile features - 4
- Test out projectile velocity/acceleration and functionality (longer you hold down button, longer it fires - different visualization distinction) - 4

Week 6

- Create the base character model with working movements - 10
- Create ability to shoot beam out of characters gun - 8

Week 7

- Implement aiming of gun - 5
- Create character animations for jumping, firing, clinging, and getting shot - 5

Week 8

- Perfect character movement flow - 5
- Start detailing character textures more finely - 6
- Detail gun textures - 4

Week 9

- Test out integration with other members functionality (can get started on this step as early as Week 7) - 12
- Fix any problems with integration - 5
- Fix any problems with bugs - 5
- Talk to team members on the final week schedule - 1

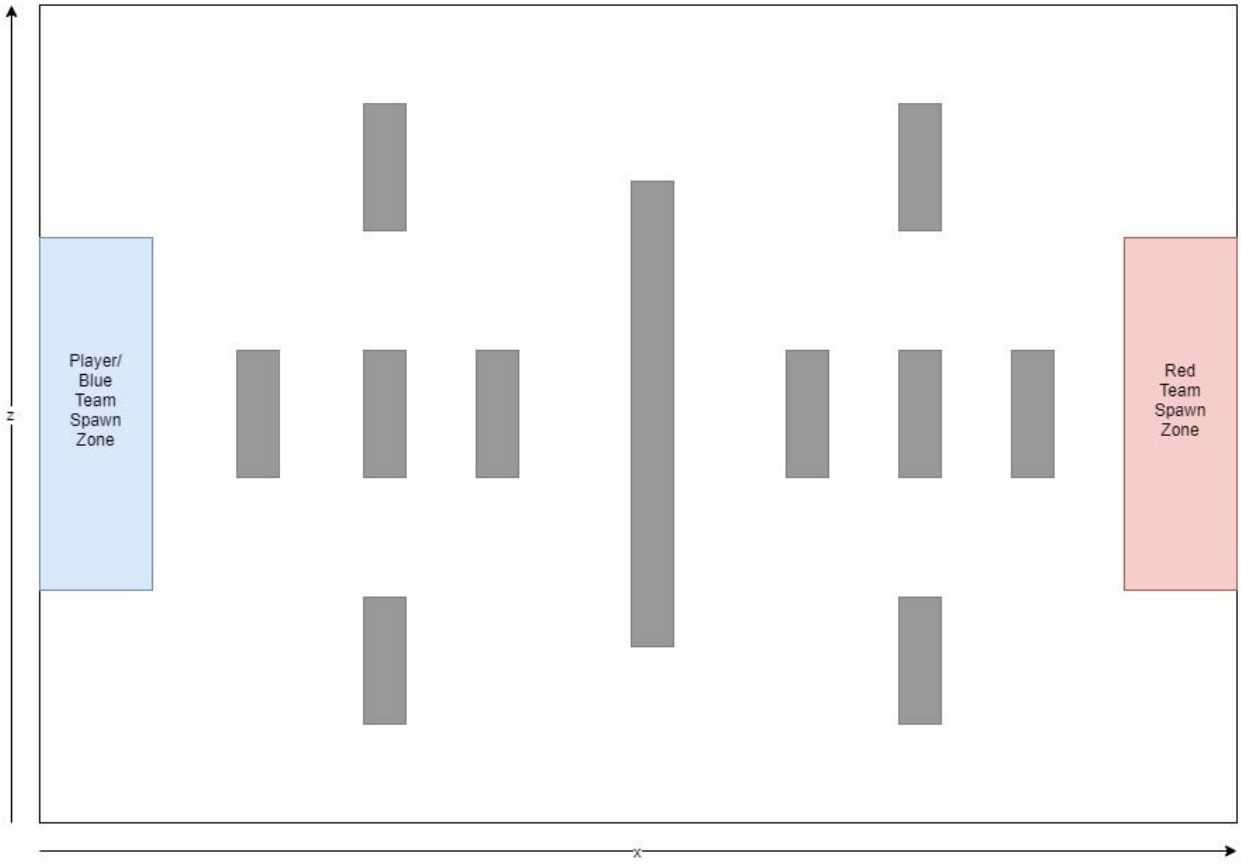
Week 10

- Help with final touches - 5
- Final integration steps - 4
- Add anything simple that can add value to the game if time persists

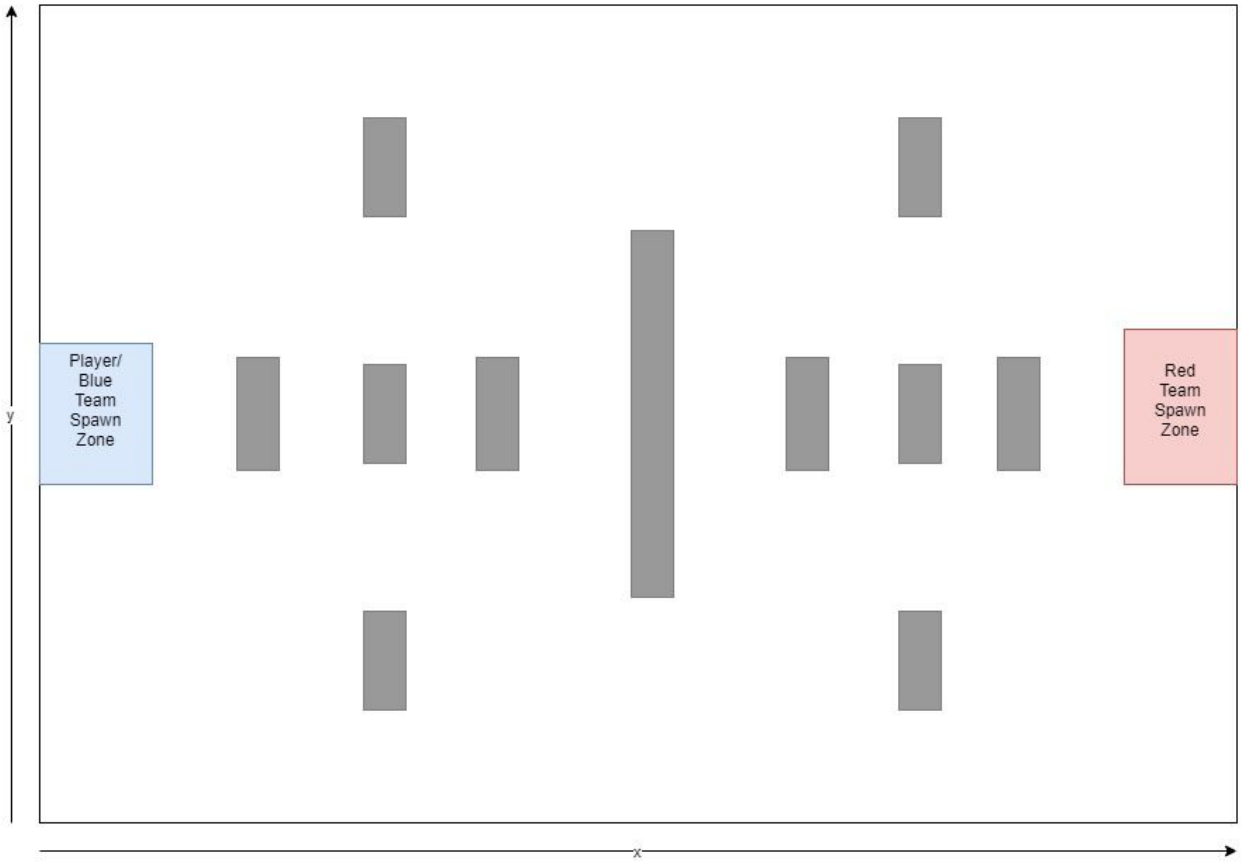
Total estimated hours: 105

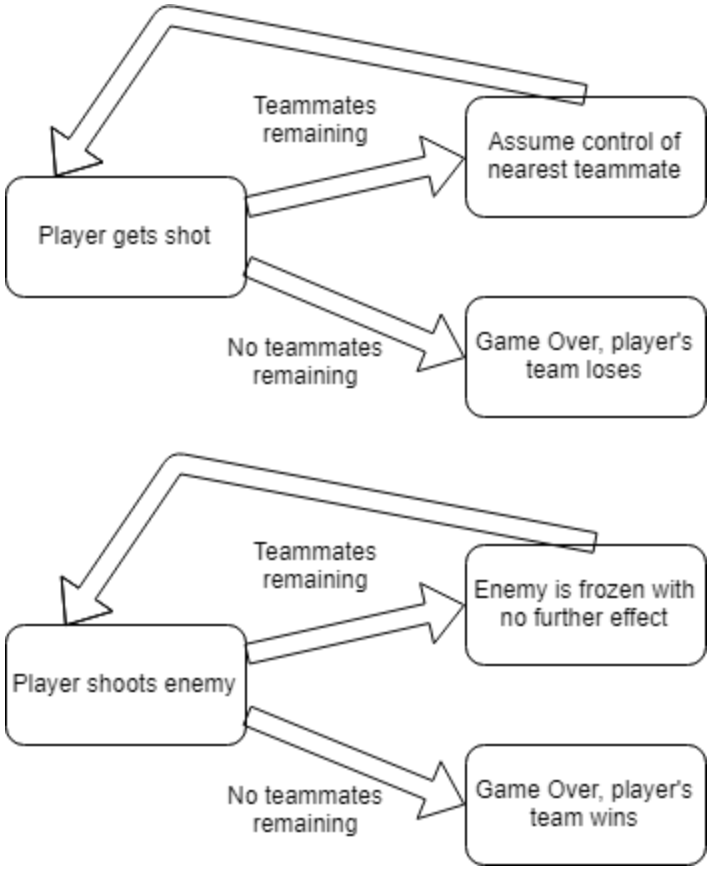
VII. Early Sketches

Map Concept (Top down view)



Map Concept (Side view)

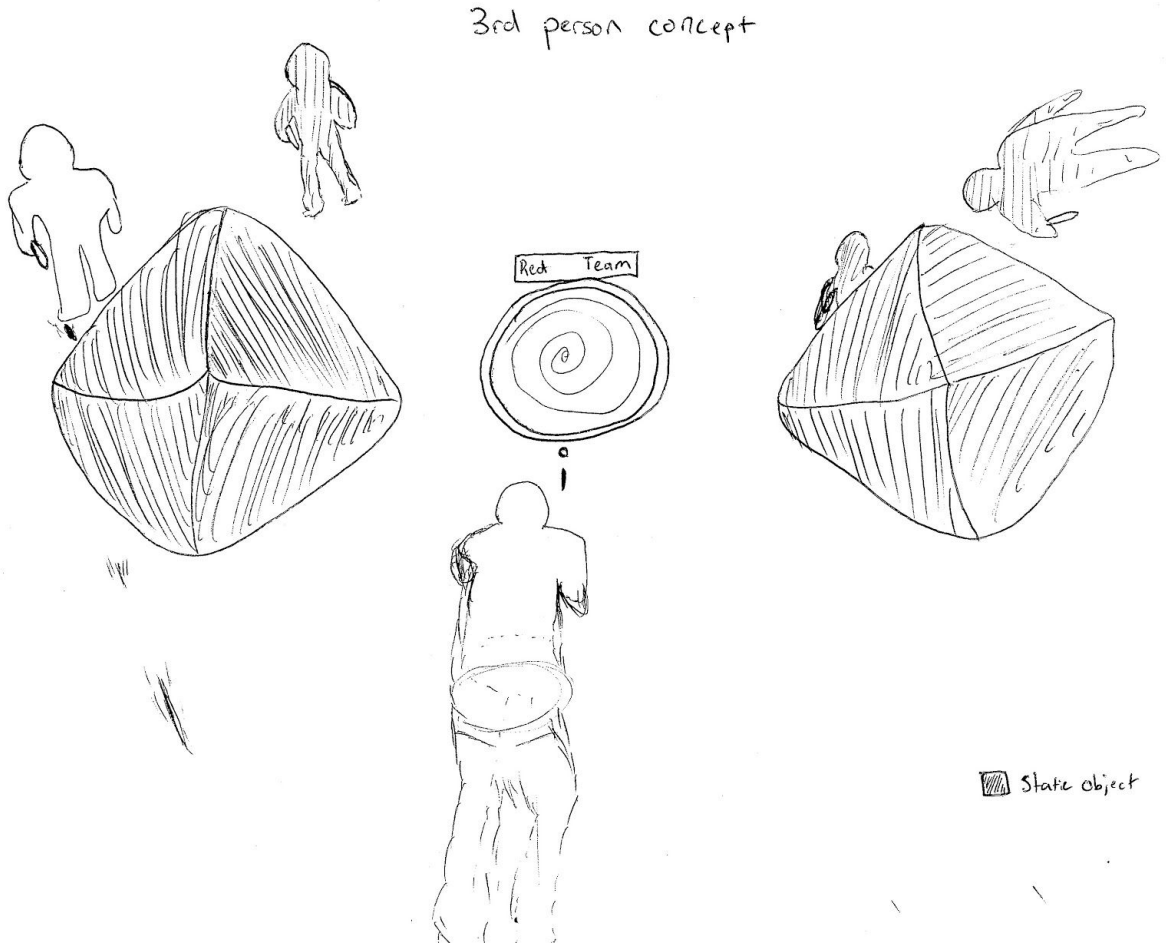




Flowchart for result of player or CPU opponent getting shot

Third Person View concept includes

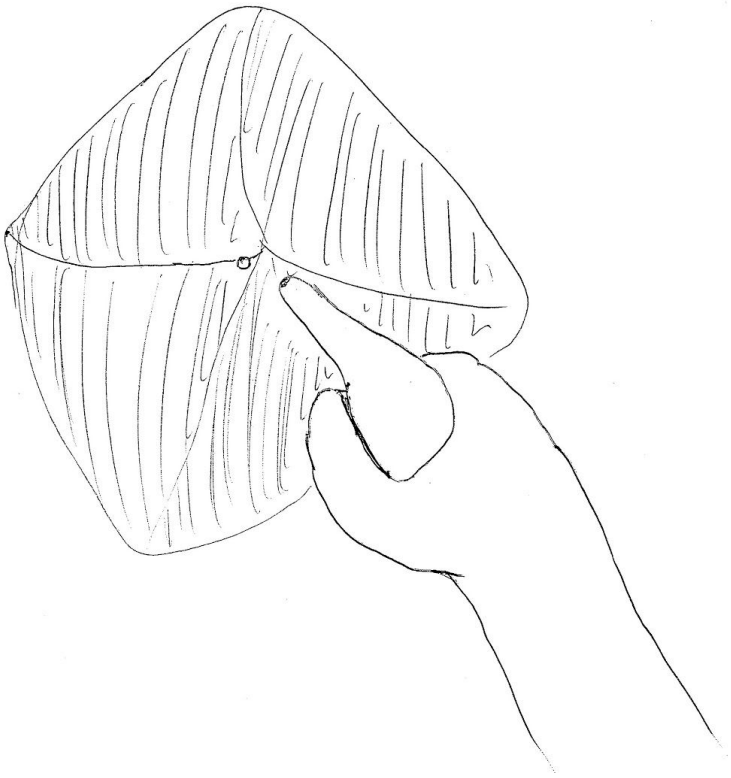
- 3rd person view of your player
- Reticle in middle of screen
- Opposing teamgate
- Static objects to interact with
- Other players (static or alive)
- Gunshots



First Person View concept includes

- 1st person view of your player's arm and gun
- Reticle in middle of screen
- Static Object

(First person and third person only differ in what the player views, all outside features will be the same in either view)



VIII. Task Delegation

REPORT	DUE DATE	FORMAT	SUBMITTER
Project Plan	**Jan 20th**	**Written Report**	**[ALL]**
Week 4 Update	**Feb 1st**	**VIDEO**	**Nicolas**
Week 5 Update	**Feb 8th**	**VIDEO**	**Christian**
Mid-Point Check	**Feb 15th**	**Files + Report**	**Charles**

Week 7 Update	**Feb 22nd**	**VIDEO**	**Christian**
Week 8 Update	**Mar 1st**	**VIDEO**	**Charles**
Week 9 Update	**Mar 8th**	**VIDEO**	**Nicolas**
Poster	**Mar 17th**	**PDF files**	**Nicolas**
Final Report	**Mar 17th**	**Written Report**	**Christian**
Demonstration	**Mar 17th**	**Project .zip file**	**Charles**

IX. Conclusion

The Navi team strives to create a 3-D game that emulates the shooter game, Halo, using the Unity game engine and the C# scripting language. We want to mimic the gameplay and rules according to the shooting game in the novel and movie, Ender's game, in order to create a fluid shooter style game that allows players to fly in zero gravity, shoot enemy players, cling to static objects, and propel off objects to a desired direction.

The project's work will be divided evenly among three members and should take around 100 hours each to complete.