Space Wars

Project Starfighter – Final

Developer(s): Jason Hoang

Class: CSCE 4220 – Game Programming II

Instructor: Curtis Chambers

Date: 5/12/2016

Table of Contents

[Personal Accomplishments 4](#_Toc450787748)

[Requirements 6](#_Toc450787749)

[Project Schedule 11](#_Toc450787750)

[Milestone I: 2/24/2016 11](#_Toc450787751)

[Milestone II: 3/24/2016 11](#_Toc450787752)

[Items marked in red and bolded indicated requirements that did not make it to the Final. 12](#_Toc450787753)

[Final: 5/12/2016 12](#_Toc450787754)

[Resources 13](#_Toc450787755)

## Personal Accomplishments

Here is a complete list of all the work and research that I put in myself towards the final product this semester.

* I did extensive research into actual Starfighter movement and controls within the unreal engine. This was followed by actually implementing such movements in 3D space by hand. I implemented specific directional movements such as the forward and backward movements, yaw, pitch and roll movements, along with a simple speed boost mechanic.
* The boost mechanic was also implemented with a dynamic FOV mechanic that adjusts with the speed of the character during boost. The change in FOV is very basic, but still noticeable.
* I implemented the use of 2 in-game cameras for 3rd person and 1st person. The 3rd person camera was also built with a spring arm to keep the camera from going inside other physical objects in the world and losing sight of the player character.
* I migrated several materials from Unreal’s educational projects to create my own engine thrusters and laser projectiles. I was also able to change some of these materials’ colors to be better identified in the game.
* I did research on all sorts of AI for the Unreal engine, but most of what I found was for ground based games. I basically had to use an attractor node and set movements for the AI in 3D space randomly enough to make the AI seem sophisticated. Then I used an AI turret tutorial that I found to give my Starfighter AI some basic targeting systems towards the player. The idea of using the attractor node came from another Unreal learning example that implemented AI butterfly movements.
* I carried the Starfighter AI movements over to the bigger starship movements so that the battlefield is more dynamic.
* I was able to figure out how to spawn the AI ships in random intervals across the map so that they all do not spawn in one place at the same time.
* I implemented projectile firing functions for the player and all of the capable AI. I also built a basic damage system with the implementation of projectile firing so that the player and the AI can be destroyed after taking damage.
* Then, I copied parts of the damage system over to static game actors so that I could build other game objectives for the starships(i.e. the command bridge, engines, shield generator, etc.).
* I also built very simple collision detection functions with physical bodies in the game to deal damage to the ships when they crash into them or each other.
* I imported a simple picture of a outer space, and then turned it into a texture to use within Unreal. I wrapped the texture around a sphere mesh and turned it into a skybox using handmade materials.
* I was able to build the entire starship and all of its destructible components using example static meshes, materials, and textures. I connected everything to the starship by making the components children to the base of the starship. This way I could move the entire ship wherever I want with all of the components staying together.
* I did some research on widget menus and built a main menu screen, a victory screen, and a defeat screen. Thus, I also implemented specific win conditions for both teams in the game.
* I implemented my own very basic cross hair from literal scratch. It is drawn as a direct part of the character.
* I imported plenty of “open sourced” sounds off of the internet and from my own music library to be used and played throughout the game. Please note that these sounds do not belong to me and I do not intend to use them for financial gain; only educational purposes.
* I also imported a few “open sourced” 3D objects and static meshes to use for my game players and AI all for educational purposes only.
* I implemented the entire scoring system and called the system into the player’s UI and the game’s instance to keep track of game progress. The player’s health is also tracked and appears on the player’s UI.
* I also implemented UI for the player to pause and unpause the game.
* I implemented game controller controls for use with an Xbox 360 controller alongside keyboard and mouse controls.
* I figured out how to make an executable version of the game.
* Finally, I spent the last couple of weeks researching and trying to implement two different ways of player targeting to utilize projectile homing. Neither works as the cast to the AI actor fails every time it is called. Some parts of each algorithm due end up locating a target to track for the player, but without being able to cast to them, the homing projectile will not track the targeted AI. This issue does not exactly break the game, as you can still fire the projectile, however it will only go straight instead of home in on the target. It is just frustrating being unable to solve the issue. The problem is either related to the Unreal Engine that we used this semester being outdated or I missed something about actors not being able to cast to static mesh actors.

## Requirements

This section provides the original requirements that have evolved over the course of the semester.

### Items marked in red and bolded indicated requirements that did not make it to the Final.

Notation:

* **Overarch [OA]** - These are the Overall main aspects of the game that each requirement and function can fall under.
* **Requirement [Req]** - These are all of the requirements of the overarch requirements that need to be in the game for completeness.
* **Function [Func]** - These are all of the functions associated with each requirement to further breakdown the aspects of the game.
* **Bonus [B]** - These are all of the bonus additions to the game that could be worked on after the above requirements and functions are implemented.

**General:** *These are the general requirements of the game that help to define what type of game is being made and how it will be played overall.*

* 3D Starfighter shooter [Req]

*Definition: Basically the game will be a third person shooter set in space.*

* Real Time [Req]

*Definition: The game will have no stops until the match is over and a team wins.*

* Play is divided into two teams [Req]

*Definition: There will be two sides to the battlefield for the player to either join or attempt to defeat throughout the game. Each side will have their own visual definitions but will be balanced in terms of functionality and gameplay.*

* Battlefield will consist of at least **2** main star cruisers, enemy and **allied** AI Starfighters, and a Starfighter that is playable by the player. [Req]
  + AI will spawn at random intervals throughout the game. [Func]

*Definition: Throughout the game, as AI are destroyed throughout the battle, they will be respawned at random to repopulate the battlefield.*

* + Star cruisers and corvettes will be operated by the AI. [Func]

*Definition: this type of AI will roam the battlefield in fixed areas and paths to bring a more dynamic feel of a changing environment in space.*

* Play will be reserved to Single player to start. [Req]

*Definition: There will only be one player exploring the game at the beginning of development to test everything out before implementing multiplayer.*

* The game is over when either team scores enough attrition points to ultimately defeat the enemy, and there will be no time limit. [Req]

*Definition: The gameplay will be based on a points system where accomplishing, or in this case destroying, objectives will reward the appropriate team points towards winning the game.*

* + The player dying rewards points to the enemy team, but doesn’t end the game. [Func]
  + There will be destroyable objectives on each side to score additional points. [Func]

**The World:**

* The sky box will be set in space. [OA]
  + XYZ Plane [Req]

*Definition: Using the XYZ plane will help to define the game as being 3D.*

* + - There will be boundaries for the playing field so that the game is focused on the battle. [Func]
  + The Starfighters and Starships will not be affected by gravity. [Req]

*Definition: There will be no need to use gravity in space because it doesn’t exactly exist.*

**Player Character [OA]:**

* Able to fly one ship [Req]
  + Standard Starfighter [Func]

*Definition: The standard Starfighter is fast with high powered lasers and can also fire torpedoes. The Starfighter will also be able to boost and have full pitch, yaw, and roll movement systems.*

* Takes damage from collisions [Req]
  + Can be shot down by any enemy lasers and torpedoes.[Func]
  + The player is dead when both the **shield** and hp bars are at 0%. The player will then have an explosion animation for death. [Func]

*Definition: HP bars and* ***shield bars*** *will be represented in UI.*

* + The player will have a short cooldown timer for them to change ships before re-entering the battle. [Func]

*Definition: This function will allow the player to have some sort of breathing time if needed to take a look at the battlefield and make some decisions based on the gameplay.*

* **Shields will have a regeneration rate while flying around no matter if in combat or out. [Req]**

***Definition: This will remove the need for multiple different functions that clutter the system when calculating the shield functions.***

* + **The regeneration will be stunned for about 2 seconds if fired upon during regen. [Func]**

***Definition: This will give some balance to players trying to finish off a skilled pilot, but short enough for the unskilled to learn and attempt to get away.***

* + **Regeneration rate will be consistent throughout the match. [Func]**
* The player will also have a virtually unlimited number of lives until either side wins the battle. [Req]

*Definition: There is a background tracker for the number of lives involved with the game due to the score counting down the end of the game and the function of the player death rewarding the enemy team points.*

**Camera[OA]:**

* 3rd Person and 1st Person modes [Req]
  + Third person mode will be placed behind and above the ship. [Req]
    - You will be able to see most of the exterior of the ship. [Func]
    - The 3rd person camera will avoid obstructions by other starships and star cruisers whenever the player flies near them. [Func]
* FOV changes when boosting[Req].
  + Camera seems farther away when boosting to give off the immersive effect[Func].
  + Returns to normal when not in boost[Func].
* Camera lag implemented for a more immersive Starfighter flying feel[Func].

**Non-Player Characters[OA]:**

* The NPCs will control all of the other Starfighters fighting within the battle along with controlling the corvettes and star cruisers weapons. [Req]
  + They will actively be searching for any enemy team Starfighters and starship objectives. [Func]
  + Turrets will stay stationary on the starships. [Func]

*Definition: The AI will be split into different roaming groups to spread out the space combat. Some ships will fight in the center area of the map, while others will fly over to attack the enemy cruiser or defend their own.*

* Each NPC controlled ship/turret will have a death animation. [Req]

**Furniture[OA]:**

* **Space debris from previous space battles. [Req]**

***Definition: Space debris consists of ship wreckage, asteroids of all sizes, and possibly some meteor showers or planets. The space debris will also be dynamic in movement if they collide or interact with the PC/NPCs.***

* + **These can be collided with and cause the player to self-destruct. [Func]**
  + **These pieces of furniture can move across the battlefield based on player or NPC interaction (crashing). [Func]**
  + **Everything will spawn in a fixed location for each game, and will move based on dynamic interaction. [Func]**

**User Interface[OA]:**

* Controller will control the Player. [Req]

*Definition: The goal is to hook up a game controller such as the XBOX 360 controller to play the game instead of using the mouse and keyboard.*

* + Unrealistic space flight will be simulated. [Func]

Definition:Flying the Starfighter will feel like flying jet fighters in Earth’s atmosphere.

* The screen will display[Req]:

*Definition: This section defines what will be displayed on the player’s UI in real-time.*

* + The health bar, **shield bar** [Func]
  + Attrition points for each team [Func]
  + **Space map [Func]**
  + **Targeted enemy health and shield bars [Func]**

**Lasers[OA]:**

*Definition: Starfighters will each be given a primary fire function to actually deal damage in the field.*

* High-Powered [Req]

*Definition: These lasers will be used by the standard Starfighters.*

* Low-Powered [Req]

*Definition: These lasers will be used by the bomber class Starfighter and some turrets on the star cruisers.*

* Medium-Powered [Req]

*Definition: These lasers will be used by the turrets on the star cruisers.*

* + Can be beams and short burst lasers [Func]
* Enemy lasers are Red. [Req]
* Allied player lasers are Blue. [Req]

*Definition: The color definitions are to help the player distinguish their own laser fire among the enemy’s.*

**Torpedoes[OA]:**

* Carried by Both Starfighters. [Req]

*Definition: Starfighters will also be given a secondary fire to bring more dynamic gameplay into the mix. These torpedoes will have significantly different attributes compared to the laser to also be visually appealing, alongside having lethal effects in the battlefield. To balance this addition to the gameplay, a cooldown timer will be implemented.*

* + Slower than lasers in terms of speed. [Func]
  + **Can lock on to the target and track them until they collide with the target or any other crash-able obstacle in the way. [Func]**
  + Explosion animation on collision. [Func]

**Star Cruisers[OA]:**

*Definition: Star cruisers serve as the main bases for each faction within the game. Each cruiser comes with several objectives that the other team can attack and destroy to earn points for their team. Each of the following requirements are destructible in the game.*

* Shield Generator [Req]
* Engines [Req]
* Multiple turrets [Req]
* The Command Bridge [Req]
* Life Support [Req]
* Hanger Bay [Req]
  + Ships will spawn from outside this area [Func]

*Definition: The hangar bay will serve to give each team an area to spawn that feels immersive to the gameplay. Of course this aspect will need to be adjusted as the dynamics of the battlefield change to avoid negative results like spawn camping.*

Bonus: *The list of bonus requirements serve as optional additional content that could be added later once the above Overarching requirements are completed. They only bring depth to the completed game alongside other options to explore.*

* Multiplayer [B]
* Additional Starfighters [B]
  + Bomber – Slow with low powered lasers, but powerful bombs and shields
  + Bombs:
    - Launched from the bottom the bomber.
    - Falls as if it is affected by gravity until it collides with something.
    - Deals high damage.
    - Explosion animation on collision.
* Additional turret types [B]
* Maybe a ground game like Battlefront 2 [B]
* Star Corvettes [B]
  + Turrets
  + Engines
  + Shield Generator
  + Main Hull
* Make the star cruiser turrets controllable [B]
* Add distant planets and moons. [B]

## Project Schedule

This is a list of the accomplished and unaccomplished requirements according to the proposed schedule of development throughout the semester.

### Milestone I: 2/24/2016

* Skybox [OA]
  + There is room for easy scale expansion.
  + The resolution could be adjusted for a better looking star field.
* Camera [OA]
  + The 1st and 3rd person cameras are built and are functioning appropriately.
  + A spring camera is working for the 3rd person view that will not let environmental objects interfere with vision.
* Player Character [OA]
  + Most of the Player Character functions are built and setup.
  + Needs the ability to take damage, including shield damage.
  + Also needs to reward points on death.
* Laser [OA]
  + The basic High-Powered laser blueprint has been built with their respective colors.
  + Only need to duplicate it and adjust the material textures and damage output for the other types of lasers.
* Bonus: Implemented some basic NPC functions, but it is very buggy right now.

### Milestone II: 3/24/2016

* Finished Implementing basic Enemy AI features: AI Starfighter and AI Turret.
  + Locating and tracking the enemy when in range.
  + Shooting and approaching the enemy when in range.
  + Ai Starfighters will roam randomly in a specific area of space when not tracking the enemy.
  + Ai turrets will remain idle when the player is not in range.
* Placeholder for the star cruiser base put in place to hold the turrets.
* Xbox controller inputs implemented.
* Basic Player UI implemented.
  + Health Bar – green depletes right to left
  + Score Board – player score vs. enemy score
* Basic Score System implemented. (Buggy right now)
  + Player deaths account for 250 points to the enemy team.
  + Enemy Starfighter deaths account for 100 points to the player.
  + Enemy Turret deaths account for 50 points to the player.
* Player Boost implemented.
  + Boost with Left Shift.
  + Small particle effects coming from the engines.
  + FOV changes to farther away when boosting, then reverts back to normal.
* Damage system implemented.
  + Lasers from the enemy AI deal damage to the player and currently have an unintentional friendly fire attribute.
  + Lasers from the player deal damage to all enemy AI.
  + Lasers are uniform, at the moment, to deal the same damage to each actor(25 points of damage).
  + The player currently has 200 points of health.
  + The enemies have 100 points of health, for now.
  + The actors can also apply collision damage whenever the player and the AI crash into each other. Each collision does 100 points of damage to each collision participant.

### Items marked in red and bolded indicated requirements that did not make it to the Final.

### Final: 5/12/2016

* **Implement AI tracking for Allied AI vs. Enemy AI.**
* Implement Full assets for the Star Cruisers.
  + Including making it move across the battlefield.
* Balance the Damage system.
  + Include different powered lasers into the system.
* Finish UI [OA]
  + Fix scoring system
  + **Add shield bar**
* **Furniture [OA] (Computer was not strong enough to process more items in the game world)**
  + **Asteroids**
  + **Other starships**
* Game sounds
  + Ship sound effects
  + Blaster fire for ships and turrets
  + Explosion sounds for ships, turrets, and starship components being destroyed.
  + In game music.
* Menu Screens
  + Main Menu – The beginning menu that the player sees whenever starting the game.
  + Victory Screen - Appears after the player has scored 10000 points or more.
  + Defeat Screen – Appears after the enemy has scored 10000 points or more.
* Game Paused
  + Gave the player the ability to pause the game while playing and unpause.
* Player Crosshairs
  + A basic red X surrounded by a small spinning animation for flavor. It is a component of the player character blueprint.
* **Homing Torpedoes**
  + **Code implemented but it does not work.**

## Resources

This is a list of all the various sites and Unreal developers that I have researched and a learned from to produce some of the basic concepts in my final product.

**Youtube**

* [Sam Jones](https://www.youtube.com/channel/UCJekoQFYfhHEjQbfvWf_E4g) – menu screens
* [Unain](https://www.youtube.com/user/mrunanimous) – homing projectiles
* [UnrealTek](https://www.youtube.com/channel/UCX44PoFn-gUJ2_8VYAaVrbw) – most basic tutorials such as damage, collision, and space flight simulation
* [4music2junkie0](https://www.youtube.com/channel/UCHyMmw-p89oCp_VoPEa1Xog) – turret AI

**Google**

* <http://shootertutorial.com/> - basic Unreal 4 tutorials along with some AI ideas
* <https://www.docdroid.net/wsq5/how-to-create-a-homing-missile-using-blueprints-in-unreal-engine-4.pdf.html> - homing projectile research
* <https://answers.unrealengine.com> – various unreal 4 questions and issues answered or explored here
* <https://www.yobi3d.com> – obtained some static meshes to use in game.

**Unreal 4’s Learn tab** – Here I utilized Unreal’s educational community tab to explore example game implementations to use for my own game and build working alterations that support my game.

* Content Examples
* Blueprints
* Inventory UI with UMG