CSC631-01 Spring 2021
Multiplayer Game Development
Homework #2

Team A

Member : Miguel Mellado mmellado@mail.sfsu.edu
Member : Christian Achacoso cachacoso@mail.sfsu.edu

Instructor: Dr. Ilmi Yoon

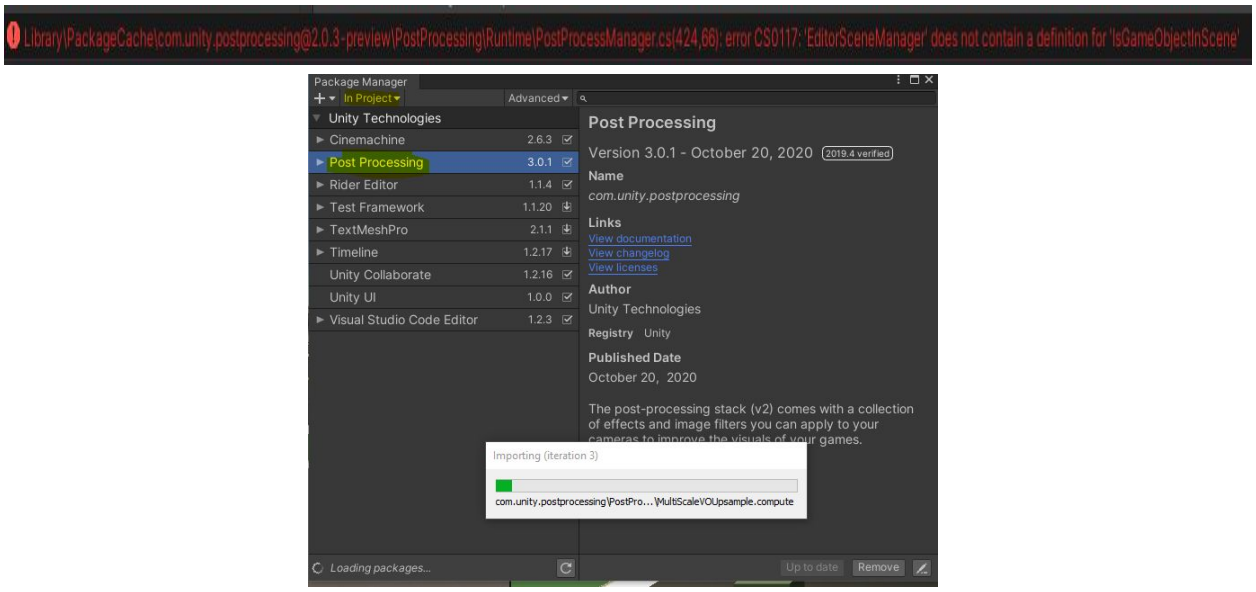
Version 1
02/17/21

Tank Game

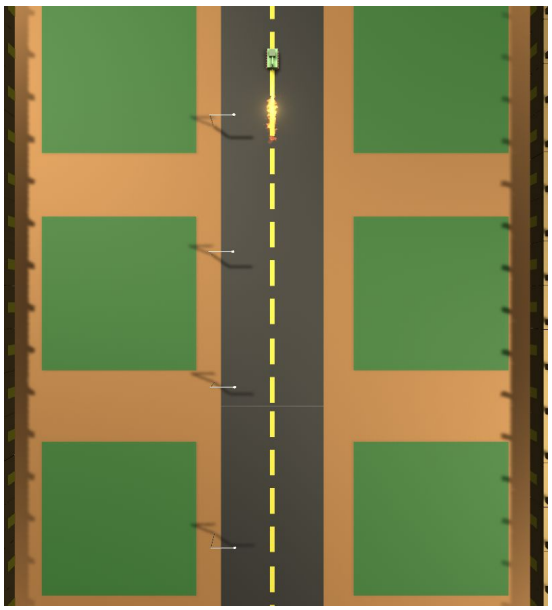
https://github.com/Team-Odyssey/HW2

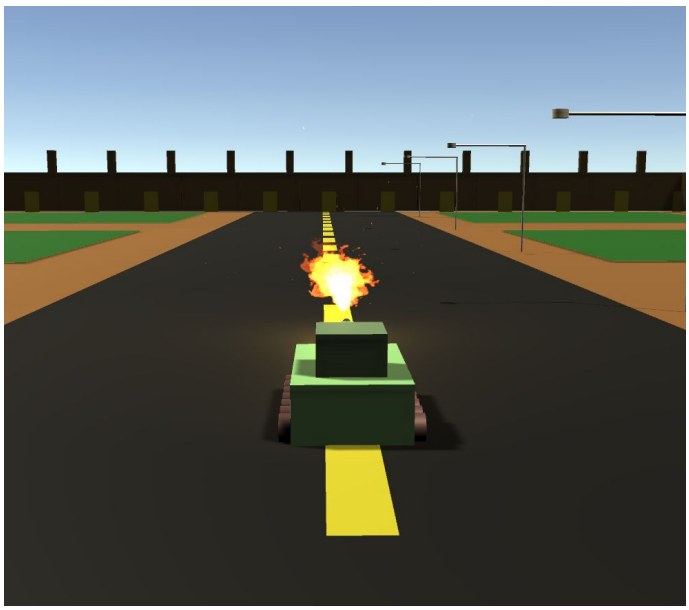
(a) Toggle Camera

- I worked on the “Miguels” Scene. Make sure you update the post processor if it is throwing errors



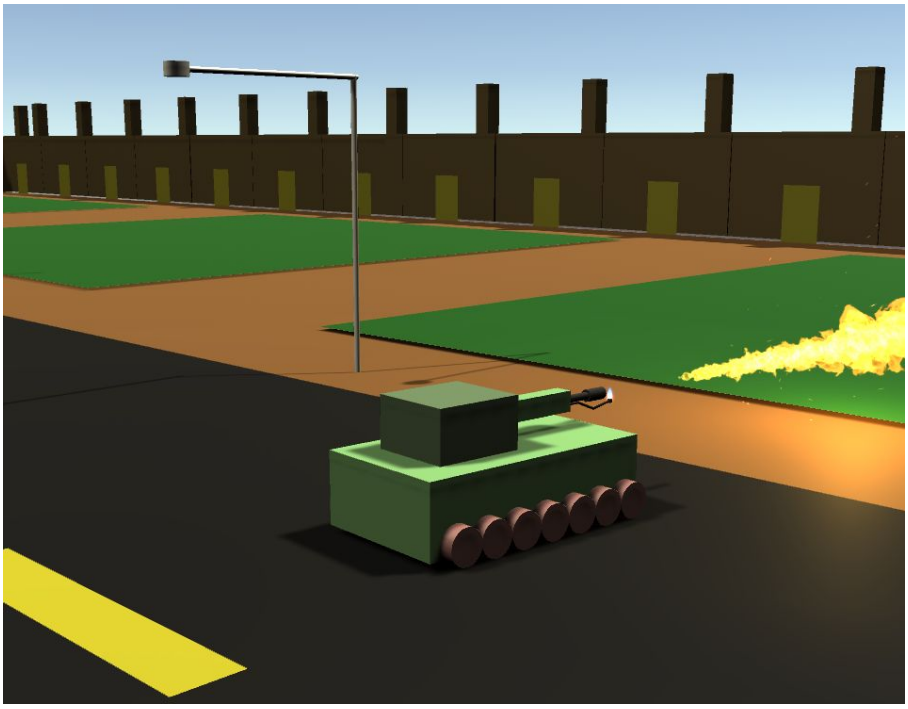
- Third person and Skyview camera can be toggled with C

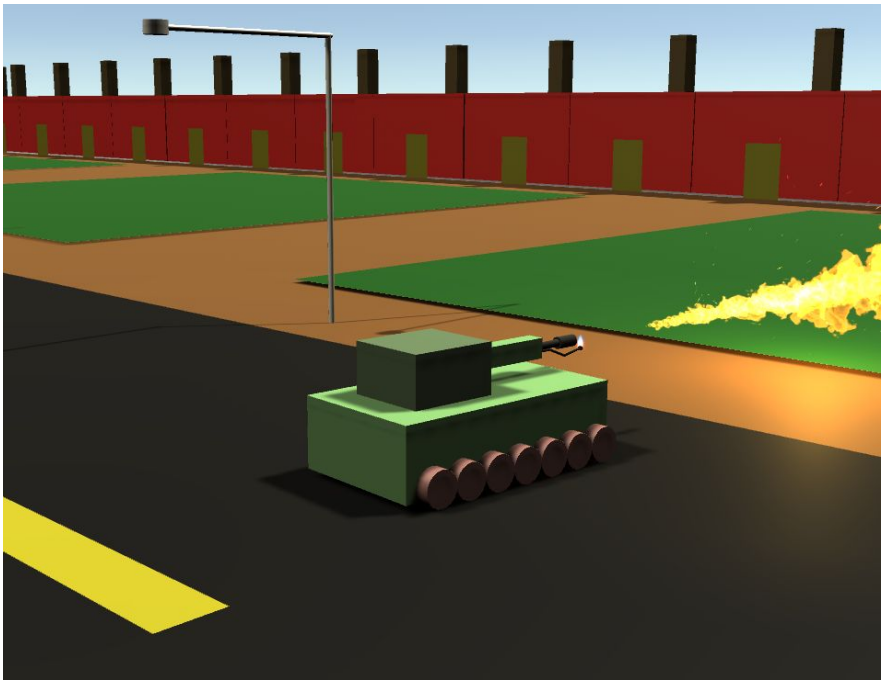




(b)Toggle Textures

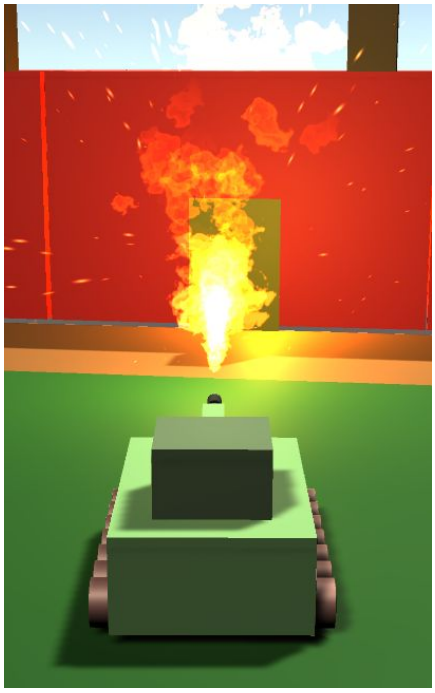
- Textures of the houses can be toggled with X

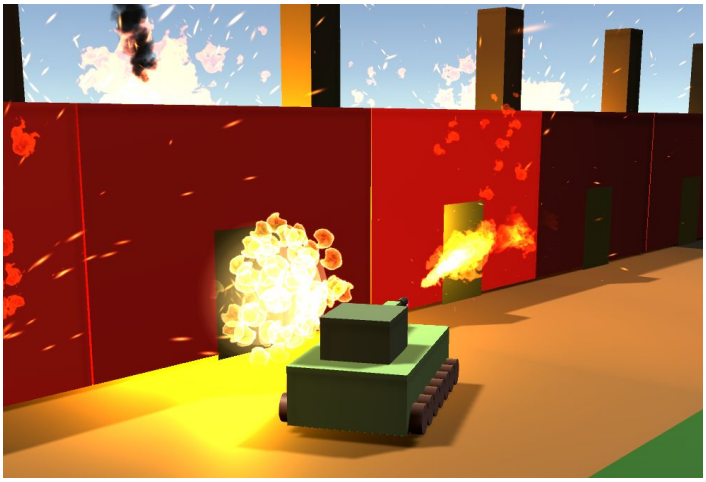




(c) Collision Handling

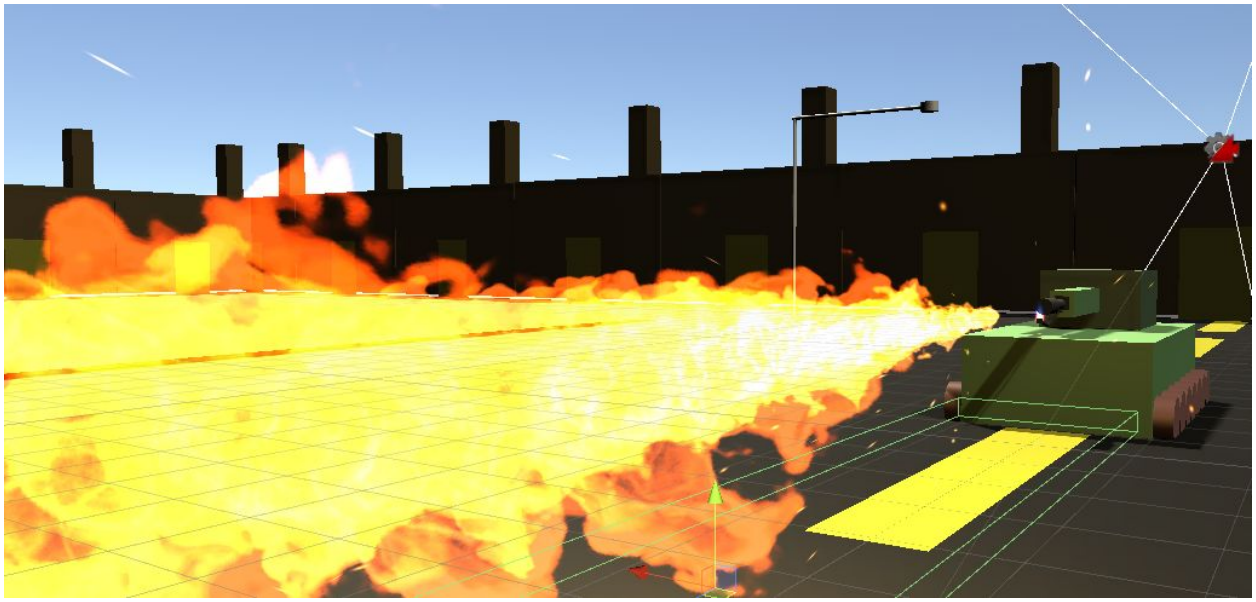
- Shooting a house with the flamethrower triggers an explosion.
    - Specifically shooting the door
    - Explosions will loop after they are triggered





(d) Particle Effect

- Added the flamethrower particle effect as well as the explosions



Reflection:

I was not sure how to start this assignment. Every task seemed daunting and I was very inexperienced. Luckily George, the Teacher Assistant, recommended some unity tutorials on humble bundles that were on sale. I had previously used these tutorials during the first homework to learn how to use GitHub along with Unity. The instructors in these tutorials are very descriptive with their explanations and you can follow along very easily on your computer.
I started with making a car in the tutorial but we were challenged to make it our own. I ended up making a tank and from there we learned how to make our car a prefab. Since I had already made a tank and felt inspired from CSC 413's tank game, I decided it would be fun to base our mini game for this homework off that concept. I created a street prefab as well as a prefab for the houses that way I can have a nice little game world for my partner to work with. Working on a Unity project together using GitHub isn't so difficult once a base world/scene has been created. I found that the easiest way to prevent merge conflicts was to duplicate the same scene and have both of us work on different scenes, then bringing it together in the end.

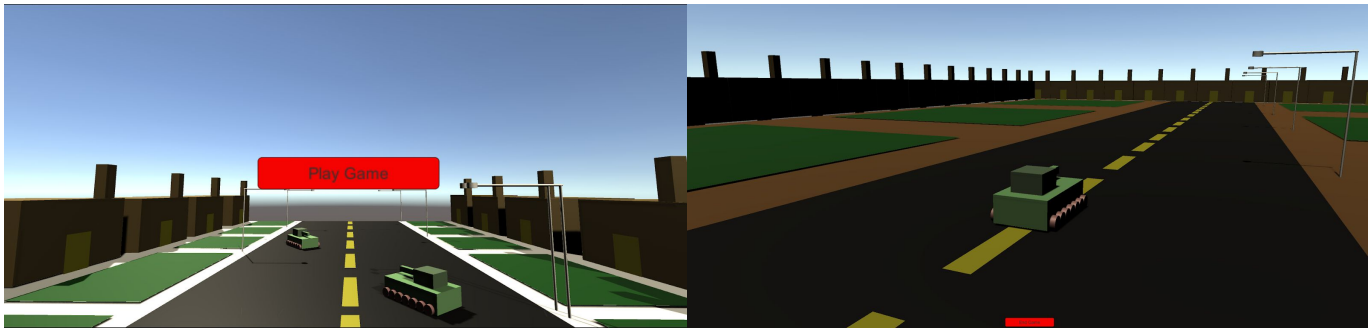
After creating the game world and the prefabs I was on to working on particle effects. I spent around an hour or so trying to learn how to use blender and create my own particle effect before one of my friends mentioned there were some really good free ones on the Unity asset store. While it was easy to download and install them, working with their properties was very foreign to me. The tutorial I was following was handling collisions using a different component in the inspector and I wanted to detect the collision from the particle effect so I had to do a lot of googling before I gave up and just started trying things for myself. Eventually, with enough trial and error, I was able to get the expositions to trigger by the a collider I attached to the flame on the flamethrower.

The Camera toggle was very easy to implement, however, the third person camera was a lot of fun to get working correctly. I wanted to make sure the camera was in a comfortable location for the user to be able to see everything in front of him. My partner also sent me a video that enabled me to change the direction the tank was moving in respect to the direction the user is facing.

Changing the materials with a button click was no challenge at all. A single google search suggested I store the materials we want to change to in an array and change the value that way. Since I wanted to alternate between only 2 colors, I used modulo to alternate back and forth

using even and odd numbers. I think the main thing I learned from all those google searches and YouTube videos is that there are many ways of implementing a solution to your problem. When searching for a solution, don't focus so much on the implantation as much as what they are achieving. Don't worry so much about if they are using a script or a player controller to move the player, just focus on what they are achieving and how.

(e) Scene Transition

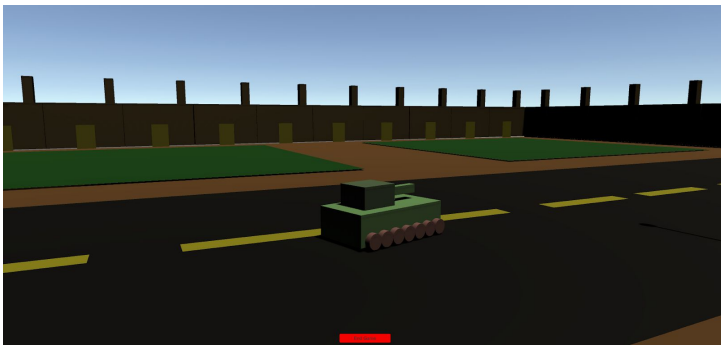


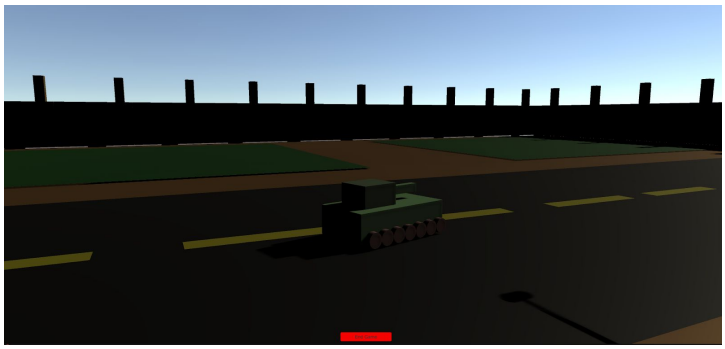
- The program begins with a screen that gives the user the option to click on “Play Game”
- Once clicked, the user is brought into the game and they are able to end the game by clicking “End Game” located at the bottom of the screen.

(f) Simple Character Animation

- The user is able to make their tank go faster by holding down the Left Shift key, once released, their tank will go at normal speed

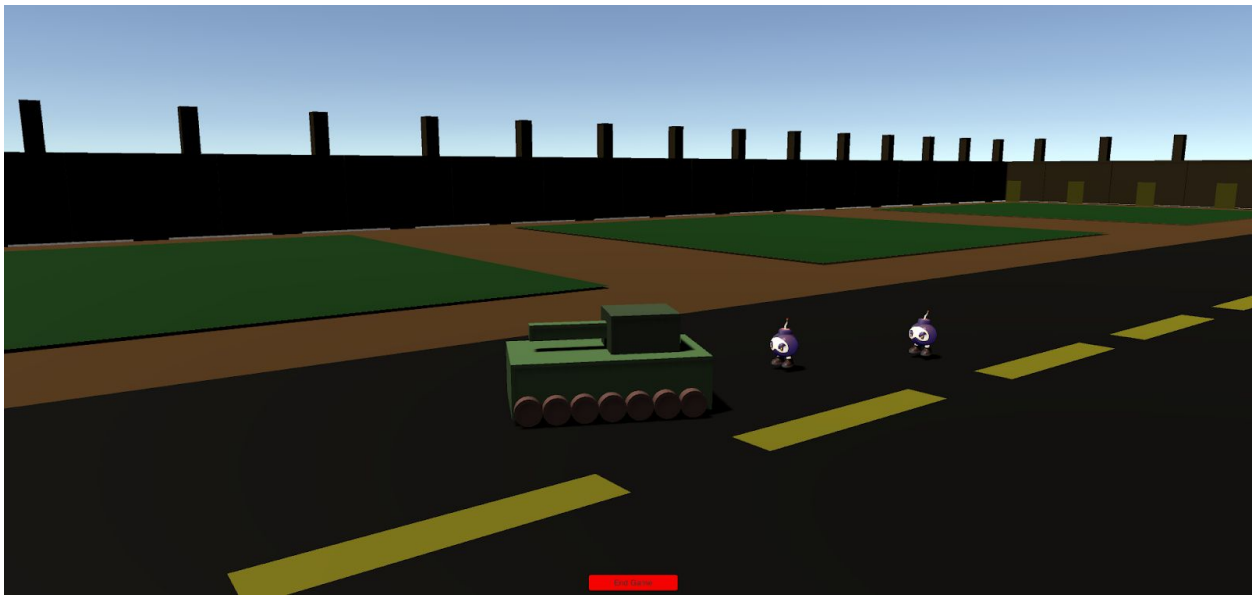
(g) Change Shading Effect





- To change the time of day, the user can press the “Z” key that will rotate the directional light and make it look like day or night.

(h) Mouse Click Creates Object



- When the user clicks while moving their tank around, they will drop a bomb where they are located. Unfortunately, it will not explode yet, which will be implemented in the future.

Reflection:

At first, I thought the given tasks would be easy and simple to implement. However, with little experience with Unity and Visual Studio using C#, and collaborating the two, I struggled a lot. I searched and "Googled" many tutorials and solutions to my compiling errors, but none of them worked. Until one of my friends, who somewhat knows and has some experience on how to work Unity and C#, helped me realize that I was not writing the script in the solution editor of Visual Studio, in which I couldn't build the code and have it compiled into Unity. After the fix, I was glad that it started to work and then I began developing the necessary features. What I needed to work on for the assignment was to make buttons that would transition scenes within our game, change the shade of light as if it were a change in the time of day, create a simple character animation, and have our character create an object with a simple mouse click. While I had an idea of what I needed to do and how to implement my thoughts, I still needed to get familiar with the Unity API and the various functions and classes that I would need. While I knew how to theoretically and conceptually map out what was needed for the features, I could not perform it practically. I was able to find the necessary functions and classes that I needed to implement the features for the game through the Unity API. Although I had to go through many trials and errors, I successfully developed all the features for the game and was able to have the game compile and run. Once the game and features were developed, committing and pushing my part of the assignment into our repository was very challenging, as there would be many potential conflicts and errors the more I tried. With the help of my teammate, Miguel, we were able to solve the problem, then commit and push into the repository. Although I had to make another repository for our team to push my code, it all worked out in the end, in which Miguel was also able to commit and push his code and part of the assignment into the same repository. This resulted in somewhat of a clean slate, in which we avoided potential conflicts and errors. If I had not reached out to my teammate, I probably would have been stuck as I had no experience with committing and pushing into a repository that was shared with other people. Overall, there were struggles and success with this assignment for me, but I was glad that I was able to experience the struggles, so that I know what to do in the future for the upcoming assignments or when I develop my own game with Unity. Reflecting on my knowledge, I will need to have more practice with Unity, C#, as well as committing and pushing code into a shared repository.