Meeps

Kevin Kim (kkim068)

Steven Truong (struo006)

Joseph Reimbold (jreim001)

# 

# **Abstract**

Our project, Meeps, is a stationary horde style game. We utilized VR to place you right in the middle of the action as this creates a sense of urgency where you can quickly be overrun by a swarm of enemies. In order to accomplish this, we used the AvatarSDK to manipulate your VR hands and give a true tactile feel of wielding weapons in order to defend yourself.

# **Introduction**

Our project is a story of survival. The player is dropped into a harsh, grey world which is littered with peculiar blue rectangular prisms. After looking around, the player realizes that they cannot move and must quickly learn to wield weapons of great power in order to defend themselves from the invading onslaught of red and yellow “enemies.”

Slay enemies and aim for the high score is the name of the game here. Every certain number of enemies slain will provide you with a very powerful weapon power up which will further increase your destructive power. Once you are attacked enough, the game ends.

Our project includes the ability to reach out, grasp and wield weapons in order to dispose of the incoming enemies instead of just pointing a cursor over the enemy models. This tactic really lends to the player feeling like they are within the game world as you have to actually interact with the objects within the world. Once you are wielding the weapons, you must physically aim them at the incoming invaders lest you be vanquished yourself. Having to do this requires a bit of dexterity and skill from the player which ups the ante and creates a fun and engaging experience.

This sort of project really shines in a VR environment. We have all played similar games to this where you just spin around with a joystick or a mouse and line up your cursor over enemies and click the mouse button. It’s gotten pretty stale and boring over the years. Having the ability to now place yourself in the world brings you into it and creates a strong sense of immersion not possible otherwise.

Not only is the player able to feel more like they are within the game world and now a part of it, but VR will often also allow the player to directly manipulate the world with controllers which our project does as well. Being able to interact with the world also increases the challenge for the player as they need to learn how to properly utilize the tools that they are provided with in a new 3d world where they can physically navigate as well. A much as increased dexterity is required in order to properly “aim” weapons or grab objects in the world, and this is something that players will build up as they experience VR projects more and more.

# **Related Works**

We initially set out to create a sort of dueling type game where you would have to draw faster than the opponent AI and see who could shoot faster. As we began, this seemed somewhat boring and we didn’t see any other things like that so we thought we were right. We then changed our design to a horde style shooter as we thought it would be a fun and exciting type of game to both play and make. We did a bit of research on this and found a few good examples of such games with slightly different flavors (also to our own design).

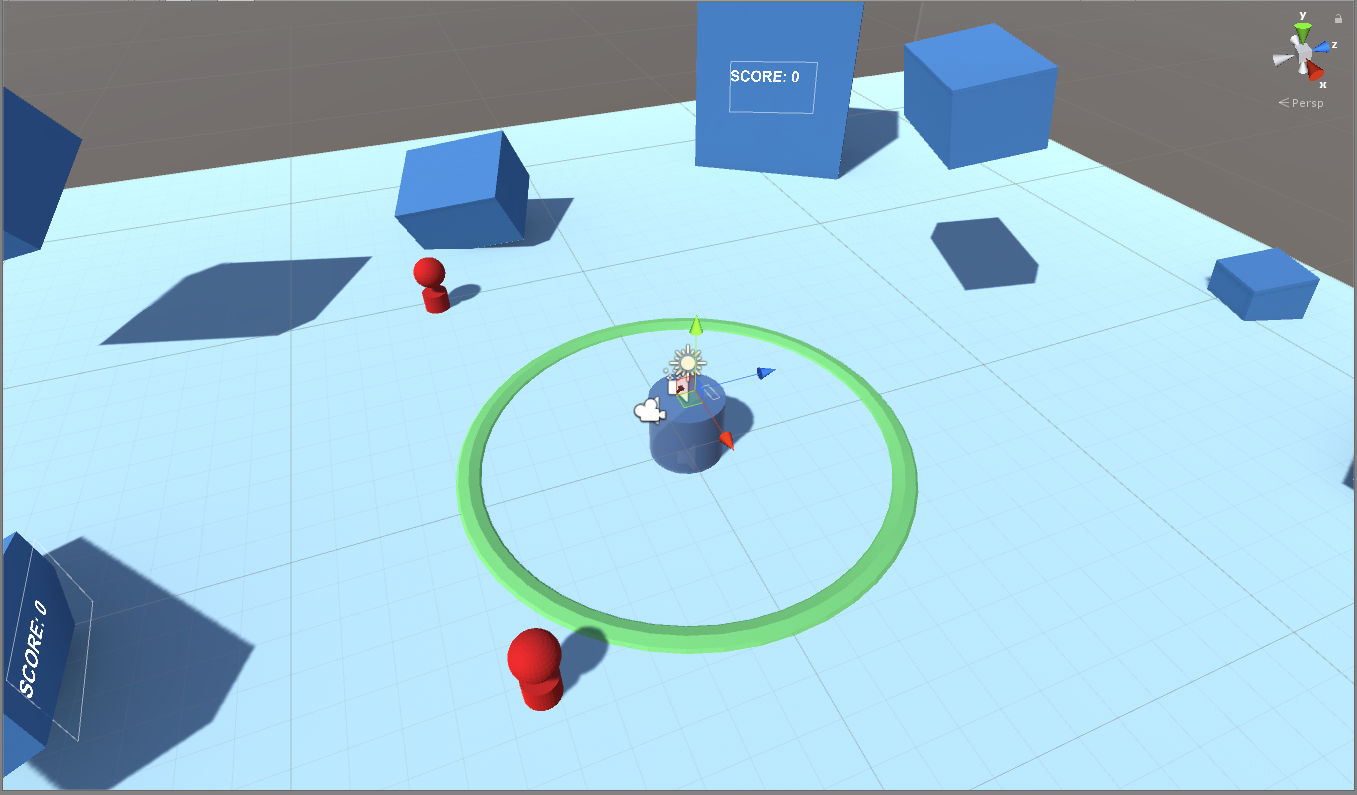
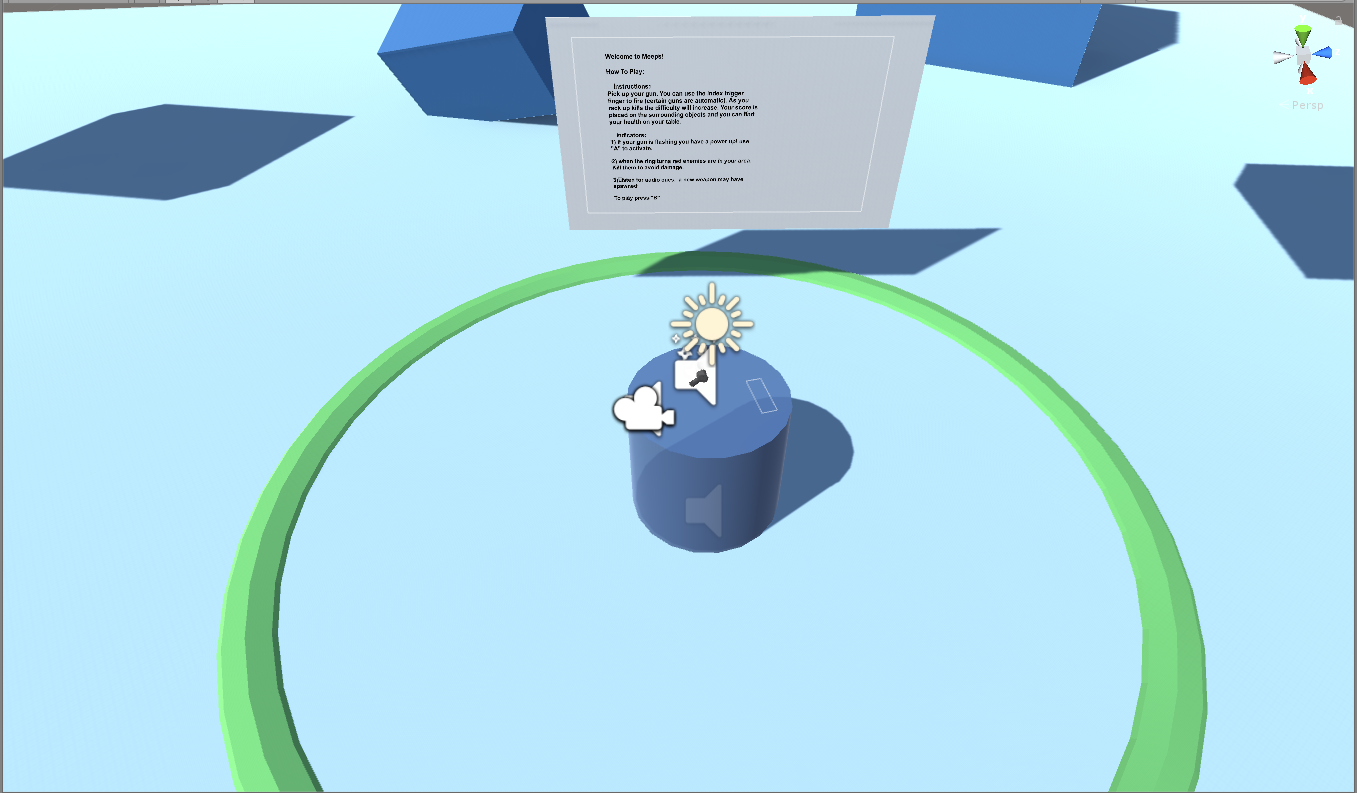
There are three notable entries in this genre and they are Arizona Sunshine (story driven zombie survival), Dead and Buried (stage based shootout) and Superhot VR (time stopping puzzle shooter).

I would say that our project would require a bit of depth added to it and further refinement to truly stand against these games. Our game definitely lacks content when put up against these other titles seeing as how we had only about a week and a half to work on this and throw it together. We think it would be fun to add more stages in that you would progress through and could provide you with new ways to eliminate the enemies you encounter keeping each stage somewhat new and fresh.

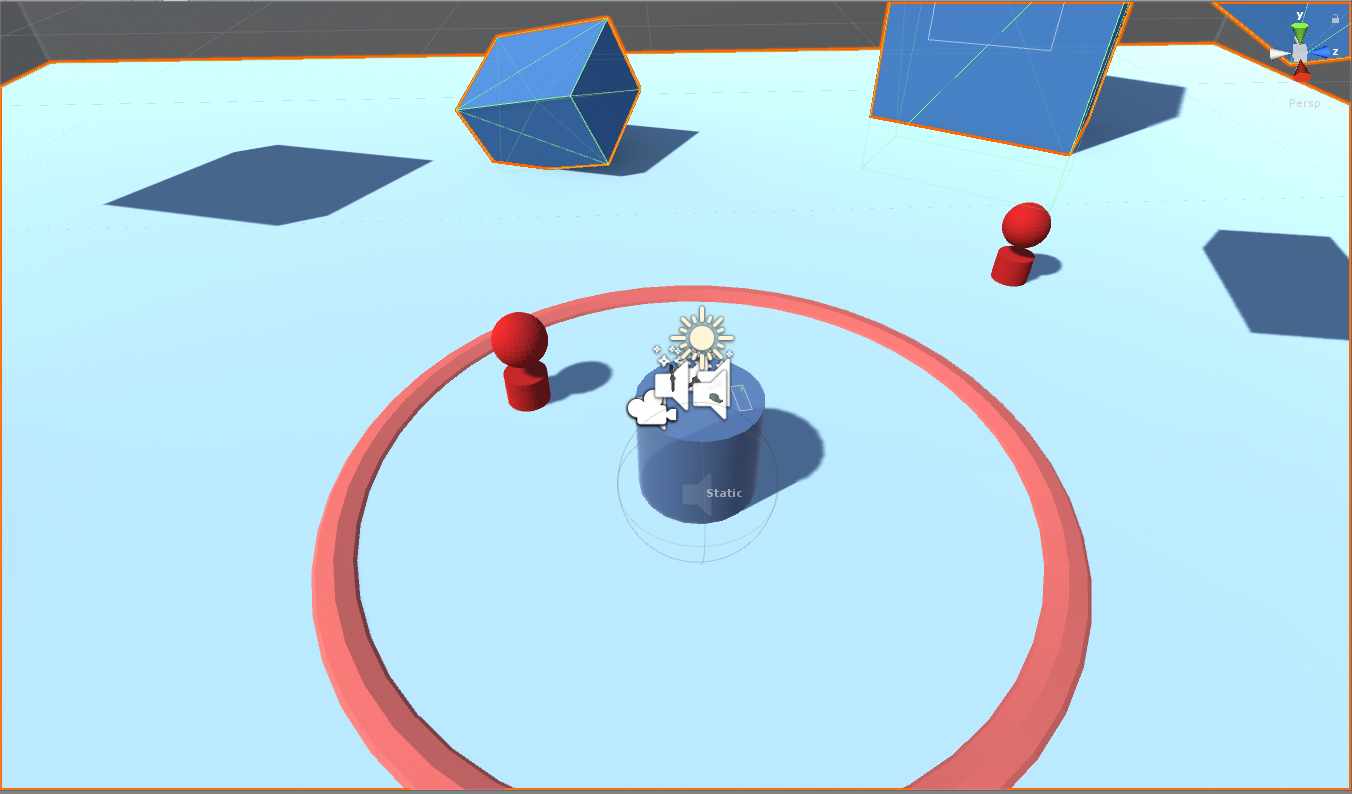
However, where I think we excel is in the simplicity of our design. We had to ensure that the gameplay was solid and it felt good to control as this was going to be integral to a fun project. Some of these other games are very complex in their systems and can make them pretty difficult to play. For example, both Arizona Sunshine and Dead And Buried have reloading animations where you have to interact with the weapons in extra ways to get them to fully function. While this is obviously very cool, it can lead to confusion and in the middle of a firefight, can be detrimental. Our game focuses on the shooting and the player’s reaction time to continue pushing further and further as they aim to get that high score.

# **Design**

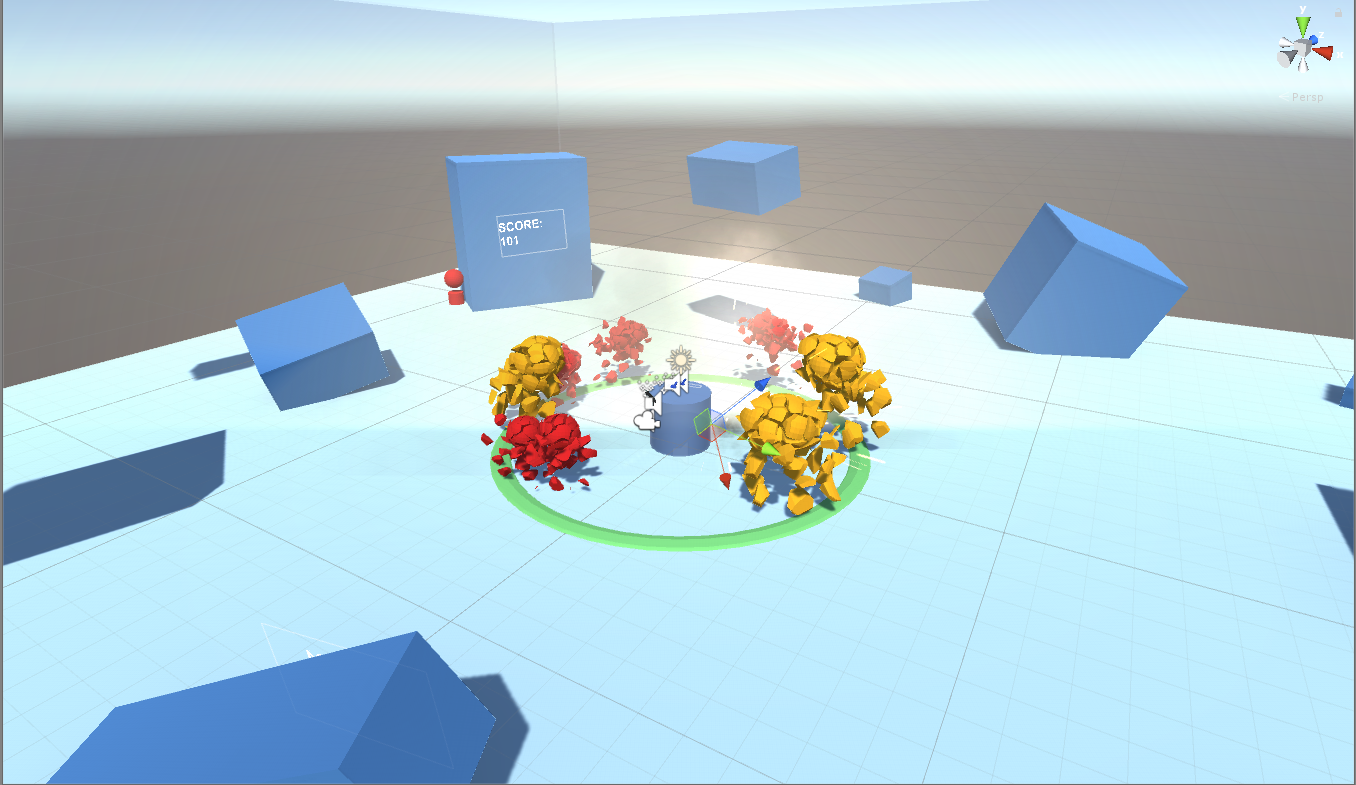
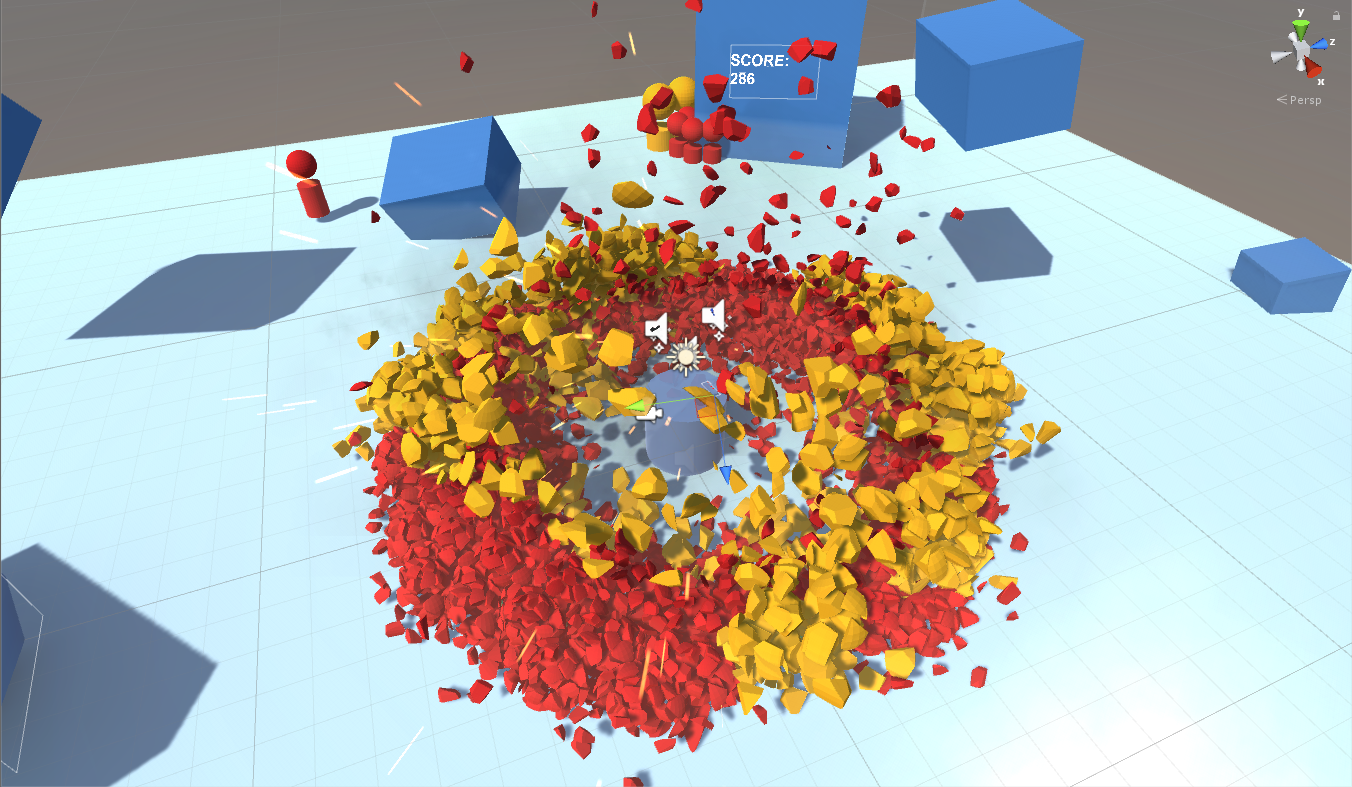
Our world design is very simplistic. We wanted to go for a very abstract and low poly aesthetic where the physics effects and object interactions could shine. The blue cubes around the world serve as obstacles for the meeps to move around as well as something to block our attacks. We have also plastered the score on some of the cubes so that the user can see their score in any direction they may be orientated. The cylinder in the middle serves as a table for the weapons to spawn on so that the user can easily reach them. The cylinder also displays the user’s health and rotates to keep the health marker in front of the user at all times. The ring around the cylinder is an indicator that alerts the user when they are taking damage from a meep. We applied matte materials to everything, including weapons and grenades, in order to keep our game aesthetic consistent. We have two different meeps which are distinguished by their colors, where red takes one shot to destroy and yellow requires two. We also had the weapons snap into position when you grab them to avoid confusion and inconsistency in object/user interaction.



1) 2)



3) 4)



5) 6)

Screenshots [ all screenshots are from the scene view ]: (1) tutorial at the start of the game, (2) safe green circle, (3) red circle alerts you when enemies are near, (4) when all of the weapons have been unlocked, (5) physics and destruction effect on the meeps, (6) mass destruction, this is what it takes for the game to crash.

We chose the Oculus touch as our user interface so that you can interact with the objects. The oculus touch makes things more immersive because you are feeling as though you are grabbing the weapon rather than just holding the index click of a mouse. We mapped our controls to the oculus triggers for the weapons. For grabbing weapons the hand trigger needs to be held, and for shooting guns the index trigger of the hand that is holding the gun needs to be pressed. For specials and starting the game, the right oculus touch “a” and “b” buttons are pressed respectively.

Our design conforms to the Oculus Best Practices because we do not change the position of the user. This lack of translational movement helps to prevent nausea. We also tried to place interactable objects as close to the user as possible so they didn’t have to reach too far out of position and put themselves or anyone else in harm’s way. However we did not conform to the best rotation practices because the user needs to look around quite frequently to search for enemies. During some of the testing we experienced a tiny bit of dizziness due to having to yaw so frequently, however, we think that the dropping of frames in later levels may have also contributed to this.

The best conditions for our designs would be with really good computers that can render the explosions and handle the spawning of a lot of meeps. This is necessary because we have AOE explosives that can break multiple meeps at once and later levels tend to instantiate game objects more frequently which causes performance to stutter. For the user experience, the best conditions would be an open area where the user can stand, reach out, and spin around without their arms hitting anyone or anything. Consequently, our design would perform the worst in a confined and restricted space.

One design decision we made for our project was to make the meeps eye level so that the user wouldn't have to pitch their head too often between looking at meeps and the weapons. We also decided, that for the grenade to explode it has to leave your hand and it could not be currently grabbed. This was to prevent the grenade from exploding in the user’s hand.

# **Implementation**

For our game implementation we basically created a wave spawner where the user is in a fixed position and must defend themselves from oncoming enemy attacks.

Our game idea is simple however, it required a lot of complexity and interaction between components and scripts. Simply put, we relied heavily on scripting where a global script managed variables and game objects that were attached or required by object specific scripts.

The main script is our gameManager which handles global variables that are used to maintain continuity across the entire game. It also facilitates interaction between game objects. Each weapon was also given a script that manages user/weapon interaction such as picking up and firing, fire rate, or in the case of our pistols, a special attack. We also had a script assigned to the environment which was responsible for spawning enemies including their speed as well as their spawn frequency. The meeps also had scripts that managed their AI functionality such as obstacle avoidance and converging to the user’s position. Meeps also had a script that handled their destruction, including the logic and physics for the crumbling effect when they were killed. We also had a script that handled the logic for the area ring which would change colors to alert you when enemies are near.

We used the AvatarSDK and LocalAvatar for our virtual hands. We also got our pistol, assault rifle, and grenade models from the asset store. To enable Oculus we used the OVR files/assets that were provided to us in lab. We relied heavily on OVR Grabbable and OVR Grabber scripts for a lot of our functionality. However we did have to make some minor changes to the scripts to make them work for us. For some of the particle effects such as the explosions and some of the sounds we also used the packages that were provided in the standard assets.

One implementation issue we had was figuring out how to use any weapon with any hand. Initially we had a gun for the right hand, and a gun for the left hand, but we felt that this led to a lack of user/weapon interaction dynamics. To fix this we ended up figuring out when an object was grabbed, dynamically checked to see whether or not it was grabbed by the left hand or right hand, and then ensuring that you could use the weapon with the respective controller. We also had trouble implementing our power up bullets. We previously had each gun manage its own power up bullets but this led to in game confusion as well as buggy edge cases. To fix this we used our gameManager script to handle everything globally which made the pistol scripts cleaner and the game smoother.

# **Lessons Learned**

We learned a lot about the AvatarSDK and OVRGrabbable and OVRGrabber. These were the most interesting things we learned for this project because it helped us make our hands visible in the game as well as enabled us to pick up our weapons. One obstacle we faced for this project was when we imported an asset that had multiple variables of the same name but different types. This caused a lot of errors which we did not understand at the time. We overcame these errors by deleting the assets that we imported and found new ones to use instead. Another obstacle we came across was how to snap the weapons into the correct position and orientation so it looks more like how you would hold a gun in real life. We overcome this obstacle by testing with OVRGrabbable snap offset transform and eventually we were able to understand the offset so the guns looked more natural in your hand. It was also hard to develop this game in such little time and with limited access to the Oculus. In the end we ended up having to do a lot of development at home and dedicate all of our time with the Oculus to debugging.

During our demo session we felt like the overall response to our project was positive. A lot of people complimented us on our shooting mechanics as well as the physics effects that we implemented. The parts people had the most trouble with were grasping the controls and the constant turning. For the controls, we could have done a better job of making them more intuitive and less complicated. For example, only the pistols had powerups and it made some people confused when they tried to use a powerup while holding the assault rifle. For the constant turning we had debated spawning enemies in one direction but it was very late into development and so we decided to leave it as it was. However, for the future we could only spawn enemies in front of the user to reduce the need for turning all the way around. This would also likely solve our issues with the sensors not being able to track hand movements well when the user is completely turned around.

To complete this project everyone worked individually on several sub components and together on more general components. Overall all three of us worked on debugging/testing and adding the game logic as we progressed through this project.

Kevin was responsible for designing the meeps in blender, including the cell fracture and destructive physics effects. He also found and modified all of the weapons (pistols, rifle, grenade) including their aesthetic and their scripts which handled raycasting, aiming, positioning, and their ability to interact with the AvatarSDK virtual hands. Kevin also handled all of the explosions and their aoe damage for the grenade and the special rounds.

Steven was responsible for creating and applying the logic for the area ring to alert users when meeps are near He also made the script to facilitate the spawning of meeps including their spawn location and speed. Along with spawning the meeps Steven made the AI script for the meeps such that they avoid obstacles around the map and converge to the user. Steven was also responsible for power up notifications which involved the flashing and color changing of the guns.

Joe was responsible for making the game environment with the game obstacles. He was also responsible for the UI including the score, and the rotating cylinder with the health attached to it. He also added the particle effects for wherever the raycast hit and at the muzzle flashes for the guns. He also implemented the background sound of the game as well as the sound guns make when they shoot.