EE 267 Final Project
Interactive Virtual Reality Application Created in Unity: Forest Fire Fighter

Marsie Trego
Stanford University
Stanford, California
mtrego@stanford.edu

Abstract

This report details the design process of an interactive virtual reality application developed in Unity and shares lessons learned. As part of the design process, three user interviews were conducted with an experienced gamer (almost two decades of gaming). These user interviews and user testing informed the design process and lead to important design changes. The application developed is a game where the user fights forest fires. This application is designed for the Oculus Go, which allows the user to both see the application in three dimensions, via the head mounted display, and interact with the application in three dimensions, via the controller.

1. Introduction

I decided to make an interactive virtual reality application in Unity for my final project for EE 267. I have had an Oculus Go for a while, and I decided to develop my application for that virtual reality headset. Creating an interactive virtual reality experience seemed like a fun way to expand on what I had learned about virtual reality over the course of EE 267.

As suggested in class, I used Unity to develop my application, and I used the Unity Asset store to find many of the 3D models I used.

While I was developing this application, I used user testing as a design tool throughout the design and development process. I found an experienced gamer (nearly two decades of video game experience) to test the application. I interviewed this user when selecting the final idea, after creating the initial environment, and after creating the first working version of the game.

2. Project Selection

In order to select an idea to develop my interactive virtual reality experience around, I used a brainstorming and project select process I learned in ME 271E earlier this year. This process includes brainstorming several ideas, determining the most important project criteria, and ranking each idea based on the criteria to objectively select the optimal idea. I used this ideation and selection process because this process results in a sound idea that is chosen based on what will be optimal for the project, and this process prevents me from choosing an idea based on my personal bias. I further tried to limit my personal bias in the final idea selection by discussing the criteria and ranking of each idea with an experienced gamer as part of a user study

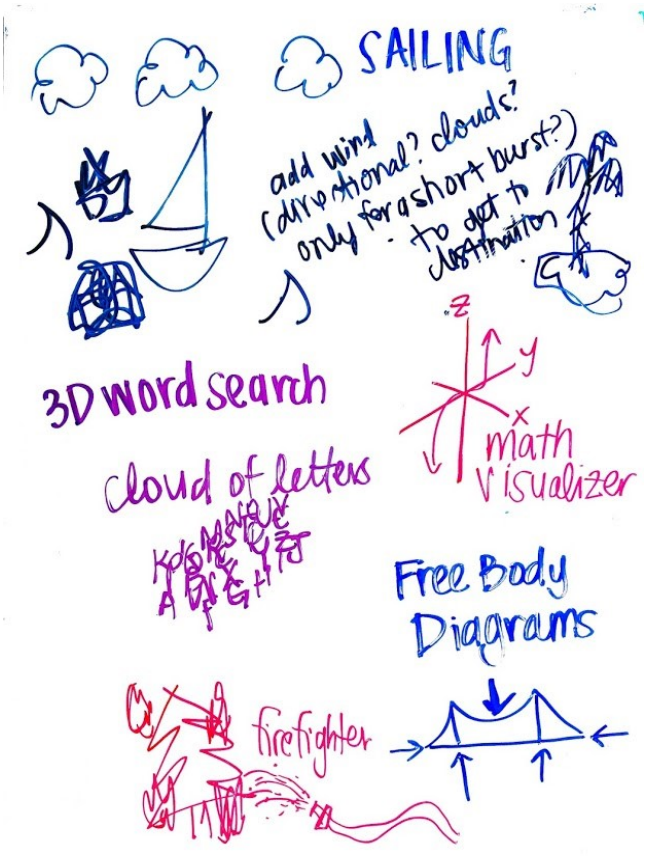

Figure 1: Initial list of ideas created during brainstorming session.

interview.

2.1. Initial Ideas

At the end of my brainstorming session, I had several initial ideas for an interactive virtual reality experience. These are shown in Figure 1, and listed below.

1. Sailing: Guiding a sailboat through a shark-filled ocean to an island, using the controller to generate "wind."

2. Word Search: A 3-dimensional word search that mimicked traditional 2-dimensional word searches.

3. Fire Fighting: Fighting a fire using the controller as a "hose" to point and shoot water at flames.

4. 3D Graph: An educational application that allowed users to visualized 3-dimensional math functions.

5. Free Body Diagrams: An educational application for introductory college-level physics and engineering students where users create free body diagrams of objects in different physical situations.

2.2. Idea Selection Criteria

In order to select my final idea for the application, I used the following criteria:

1. Uses 3D effectively

2. Potential number of users

3. Build feasibility

4. Interactivity of experience

5. Fun

After determining what the most important aspects of this project were, I ranked each idea on a scale of one to three, with one being the worst and three being the best, after discussing each application idea with an experienced gamer. During this initial interview about application ideas, the gamer stressed to me the importance of having a fun application if I wanted to appeal to a wide audience. Based on this feedback, I added the fifth criteria "Fun" to my list of criteria.

The rankings for each application idea are shown in Table 1. The fire fighting application idea was the clear winner, scoring higher than the other four ideas, and I went forward with this idea.

	1. Sailing	2. Word Search	3. Fire Fighting	4. 3D Graph	5. Free Body Diagrams
1. Uses 3D effectively	1	2	3	2	3
2. Potential number of users	3	3	3	2	1
3. Build feasibility	2	3	2	1	1
4. Interactivity of experience	3	2	3	1	2
5. Fun	3	2	3	1	1
Total Score	12	12	14	7	8

Table 1. Idea Selection Table

2.3. Headset Capabilities and Limitations

I built my application for the Oculus Go headset. This is a low-cost headset starting at $169.00 that features 3 degree of freedom (DOF) tracking for the head mounted display and 3DOF tracking for the controller (https://www.oculus.com/go/). This headset is a standalone device, allowing users to experience VR tether free. This headset has built in speakers. This VR experience with this headset is similar to that of the headset we created in EE 267 or a Google Cardboard, but with the important addition of a controller with 3DOF tracking.

While this headset is cost effective and the standalone aspect of the headset is nice, the 3DOF tracking is somewhat limiting. At times, the controller can feel unintuitive to use, because users simply expect the controller to track all 6DOF and are confused when the controller does not respond to translational movements.

2.4. Final Idea

My final idea was to create a fire fighting application. My initial idea for this application was for the user to be in an environment where there were several small fires breaking out. I decided to have the user fight a forest fire. Having the user fight a house or building fire seemed potentially traumatic. Fighting a forest fire seemed urgent and important, without traumatizing the user. The controller would be modeled as a hose and the user would extinguish the fires by pointing to them and clicking the controller. This experience would be both intuitive—only requiring the

user to point and click—and fun—a challenge for the user in a fast-paced environment.

3. Application Design

I designed this application iteratively, adding a new piece of functionality and then testing it. I followed the prototyping process suggested by Scolastici and Nolte in *Mobile Game Design Essentials*:

1. Defining it: What is your good idea supposed to do?
2. Building it: Pick a prototype type and get it done.
3. Testing it: Is it doing what it's supposed to do?
4. Fixing it: How can it better match the intended design? [1]

I created a scene in unity dedicated to testing. When adding new functionality, for example adding the background, I worked in the testing scene I had created. Once I figured out something that worked in the testing scene. I would apply it to each of the game scenes.

3.1. Preliminary User Testing

After creating the initial environment for my game, I conducted an interview with an experienced gamer. This interview provided me with important feedback that guided my application development process.

While the experienced gamer liked the cartoon feel of the trees and forest tiles, they suggested I add more detail to the background and surrounding environment. This feedback made me take a step back and focus on the overall experience of users. The iterations I went through in creating the game environment are shown in Figure 2.

During this user interview I also asked the experienced gamer about game goals and strategies. They thought that making the fires propagate randomly, instead of in an orderly fashion, would make the game more exciting and more intense.

3.2. Changes

Many of the changes I made were because I realized the difficultly of creating an application that behaved in the way that I expected, and I decided to find a simpler way to accomplish a similar task while still preserving the integrity of the game.

One of the most significant changes I made when designing this application was the goal of the game. When I had initially thought about the game, I had planned on allowing the user to extinguish all of the fires and save the forest. However, as I began to implement functionality, I realized that to have the fires continue to propagate until the user could no longer keep up with putting them out would be very similar to a popular arcade game: Whac-A-Mole. Since this is a game that has obviously stood the test of time, as it has been around since 1976 [2], I decided to model my game strategy after this classic game. Whac-A-Mole type games are described as “a quick visceral experience” by Trefry in *Casual Game Design* [3]. Virtual reality is an excellent medium to create visceral experiences in, and I

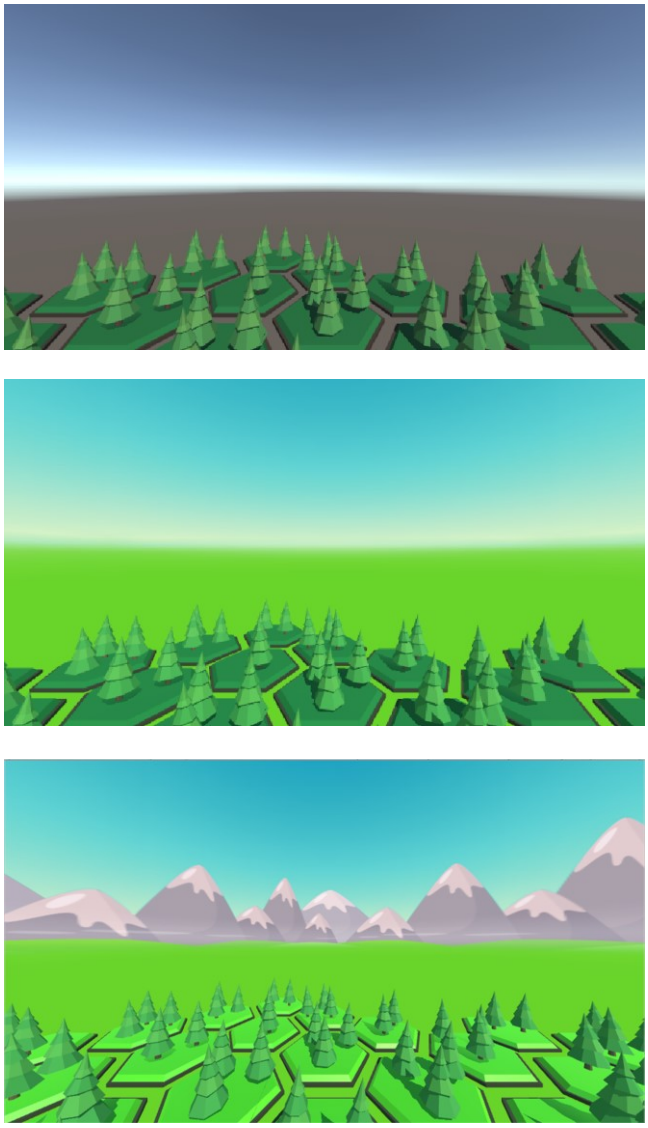

Figure 2: Evolution of game environment design; starting with basic background and progressing to final environment. Creating background scene with details was an important part of improving the virtual reality experience.

think this fire fighting game creates just that type of experience. Furthermore, the act of "hitting" (or in this case, pointing and pulling the trigger) is "a primal expression of power and aggression. It taps directly into our play instinct." [3].

Another change I made was how and where I create the fires. While initially I had envisioned creating fires on the trees themselves (see Figure 1), I soon realized that creating fires at random locations on each tile would be more reasonable (see Figure 3).

When it came to the method of extinguishing the fires, I had to adjust my plans. While ideally the fires would extinguish when they were individually selected by the user, I instead made it so that the user merely had to select the tile the fire was on. This prevented the user from becoming frustrated with needed very accurate aim to extinguish the

Figure 3: Forest tile with two fires burning.

fires. Since I had the user select a tile to extinguish a fire, I had to decide which fire on the tile to extinguish each time the user selected a tile. For simplicity, each time a tile was selected by the user, the most recently created fire was extinguished.

3.3. Further User Testing

I had a third and final interview with the experienced gamer after I had created a functional version of the game. During this interview, I had the experienced gamer try out the application, and then I asked them questions about their experience. Several important changes resulted from this interview.

In the first working version of this application, the water get coming out of the fire hose was simply modeled as a light blue semi-transparent cylinder. It looked a little like a giant, light blue laser beam. I had done this for simplicity, and it did not seem too important to me, since the rest of the graphics in the game were in the cartoon style. However, it was one of the first things the experienced gamer noticed. They suggested implementing some sort of active, animated effect that would better mimic water. They said the current visual reminded them of a light saber.

After this interview I immediately went to work learning about and implement particle effects to the best of my ability in order to make the water look a little like water and little less like a lightsaber.

A second key take-away from this user interview was that the game was too easy. While I had intended the game to be easy to learn and play, this user thought that it was a little too easy, because they could easily keep the forest fires at bay for almost a minute. They felt this was much too long for this type of game and suggested increasing the difficulty of the game by increasing the rate at which the fires propagated.

At the suggestion of this experienced gamer, I designed the application so that the user could last no more than about 30 seconds in the forest. I achieved this by making the rate of fire propagation a function of the total time the user had spent in the forest scene.

4. Final Product

The final product was a short game built for the Oculus Go. This application allowed user to feel like they were

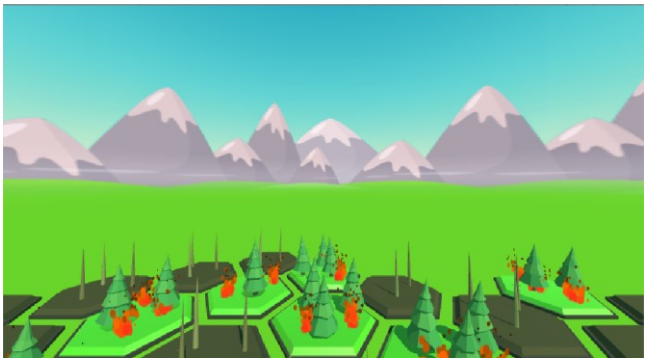

Figure 4: Forest with some burned and some alive tiles.

fighting a forest fire.

4.1. Description of Game

After loading the application, the user is in an empty field surrounded by mountains. There is floating text instructing the user to "Spray water to save the forest" and below that, a white button that says "go!"

Once the user selects the button (using the trigger button on the Oculus Go controller), the main game scene loads. The background stays the same, but the user is now looking at a forest from a bird's eye view. The forest consists of many hexagon shaped tiles with two or three pine trees on each tile.

After a few seconds, fires randomly start to pop up on the tiles. The user can extinguish these fires by pointing the Oculus Go controller at a tile, which is now represented by a giant fire hose, and clicking the trigger with their pointer finger. When the user has selected a tile (pointed the controller at it and clicked the trigger), a fire on that tile was extinguished, a particle effect made it look like water shot out of the firehose, and a sound effect made it sound like water shot of the firehouse.

After three fires were burning on tile, it became "burned" and was replaced by a dead tile (see Figure 4 for an example of a partially burned forest). The fires continued to pop up at a faster and faster rate as the user spent longer in the main game scene. Eventually, the user can not keep up with the fires and all of the tiles become "burned."

Once all of the tiles are "burned," the scene switches to the end scene. The end scene features the same background and empty field as the start scene, with text congratulating the user on having lasted XX seconds in the forest fire, and two buttons. One button, "retry," takes the user back to the main game scene, while the other "exit," closes the application.

4.2. Features

This application features a start screen, a game, and an end scene. This game is made in cartoon style and features background and props from several different assets found for free on the Unity Asset store.

This application also incorporates sound effects that make the user feel like they actually are surrounded by crackling fire and that when they turn on the fire hose, water actually gushes out.

I chose to make the controls in this game as straight forward as possible in order to make the game intuitive and accessible. Furthermore, when I think about the times I have sprayed a garden hose in my yard (my closest experience to spraying a fire hose), that was as simple as pointing the hose and pulling down on the lever that turn the nozzle on. In order to mimic the experience with an actual hose, I made the controller operate on the same principle: point the controller in the direction the water should go and pull down on the trigger.

4.3. Learning to Use Unity

I learned a great deal about using Unity while working on this project. As a mechanical engineer, I do have some experience with modeling in 3-dimensions. However, this was in the context of CAD modeling, which is similar but not quite the same as modeling in Unity. There were a few key Unity tools I learned how to use that made this project possible.

Prefabs were one of the most useful tools I used while creating this application. A prefab is a saved version of a game object that can have scripts and functionality attached. When a prefab is added to a scene, the instance of that prefab retains the link to the actual prefab, so any changes that are later made to the prefab are automatically made to all instances of that prefab. Prefabs make it easy to quickly populate a scene and make changes to all instances of the same object. I created prefabs for all of the assets I used, and prefabs were particularly useful as I iteratively designed the forest tiles. I created a "forest" of forest tiles and then fine tuned the appearance and functionality of the tiles. Instead of having to change every single forest tile individually when I created a script to control the tiles, I only had to add the script to the prefab and all of the instances of that tile automatically worked.

Another Unity tool I learned about were audio sources. Once a game object was created, I could attach an audio source to that object, and Unity would make any sound played by that audio source sound like it was coming from the position and direction of the game object. There are many adjustable settings that audio sources. The most important setting for me was the priority of the audio source. Initially, the water spray sound effect was not always audible, especially when there were a lot of fires "burning" and emitting the fire crackling sound effect. In order for the water spray sound effect to always be audible above the sound of the crackling fires, I had to set the priority of the water audio source to the maximum and leave the priority of the cracking fire at medium.

Unity destroys all game objects in the old scene whenever a new scene loads. This limits the amount of information that has to be stored but can be frustrating when trying to keep game objects or information from scene to scene. To deal with this issue, the DontDestroyOnLoad() function was very helpful. This function keeps the same game object from scene to scene, preserving the object itself as well as any data associated with it. In order to report to the user how long they had lasted in the forest fire in the end scene, I created a timer object in the main game scene and the end scene. I set this object to not destroy on load and saved the total time in the game scene. This way, I was able to access the total time in the game scene even after the game scene was destroyed.

Coroutines were a very useful tool when creating the fires. This allowed me to create a function that only started a fire at certain intervals and was not running between these intervals. Using a coroutine made the fire propagation a background process that I didn't need to put in the update loop with some sort of timer and reset.

Particle effects are a quick way to create more complex animations. Unity has particle effects that generate particles in a designated space. The size, color, speed, randomness, and many other aspects of the particles can be set by the developer. For this project, I used particle effects to create the water jet coming from the fire hose. I modified the default particle effect by changing the color and size of the particles to look more like water and by changing the shape in which the particles moved to look more like the spray of water that would exit a fire hose. Modifying the particle effects settings took a great deal of time.

4.4. Lessons Learned

One of the most valuable resources during this project were the interviews with the experienced gamer. These interviews provided valuable insight and pointed me to problems I had not realized needed fixing and improvements I had not thought about making. User testing and user interviews should always be an integral part of the design experience.

Before starting this project, I had only limited experience programming in C#. One of the things I learned the most about was object oriented programming. I created several classes with functions that objects in that class could perform, and then I attached these scripts to each game object that was an instance of that class. For example, I created the tile class (TileMoniter.cs) which had functions to start and extinguish fires and an update function to check if the tile was burned.

I also learned the importance of camera placement in a scene. Initially, I had the camera located near the "ground," so the user would feel like they were actually standing in a forest. However, since the point of the game was for the user to fight a forest fire, having them in the middle of the forest where they could only see a couple of trees did not work very well. I instead opted to give the user a more bird's eye view, and I raised the camera up. This allowed the user to see the entire forest as they looked around.

4.5. Future Work

As I spent time learning how to use Unity and developing the different aspects of this application, I had a plethora of ideas that I simply did not have time to implement for this project.

The main goal of this application could be modified. The user currently just tries to put of fires and save the forest for as long as they can. This could be modified to let the user eventually extinguish all of the fires and have them save the forest. The fires could also propagate from a single point, rather than occurring randomly. This would allow the user to be systematic in their approach to extinguishing all of the fires.

The method to extinguish the fire could be refined and require greater precision. For example, the fire would not be extinguished until the user had sprayed it with water for a certain amount of time proportional to the fire's size. Currently, to extinguish a fire the user merely has to click on the tile the fire is burning on. A future iteration could require the user to select the fire itself in order to extinguish it.

This application could be expanded to included multiple levels. These different levels could consist of different environments, such as the desert or plains, and would be increasingly difficult.

Future levels could also include different methods of extinguishing the fire. This could be as simple as providing a "higher pressure" fire hose that has farther reach and extinguishes fires quicker, or something more complex like chemical fire retardant that has prevents new fires in an area.

5. Conclusions

I learned a great deal while creating this application, from how to code in C# to the importance of camera placement in a scene. This project gave me the opportunity to expand on what I had learned about virtual reality in EE 267 to create

an interactive virtual reality application for the Oculus Go headset. User interviews were the most useful and important part of the design process, and the interviews allowed me to create the best application I could for this project.

6. References

[1] C. Scolastici and D. Nolte, Mobile Game Design Essentials. UK: Packt Publishing Ltd., 2013.

[2] P. Rugg, "The Man Who Made the Whac-a-Mole Has One More Chance," *Popular Mechanics*, 03-Feb-2016. [Online]. Available: https://www.popularmechanics.com/technology/gadgets/a18927/the-man-who-made-the-whac-a-mole-has-one-more-chance/. [Accessed: 07-Jun-2019].

[3] G. Trefry, Casual Game Design. Elsevier, 2010.