Technical Report

BOEING SCHOLARS PROGRAM | ENTRP 496
TEAM: Boeing Virtual Reality Development
MEMBERS: Taylor Winslow, Conor Devitt, Brett Johnson, Luke Holbert
PROJECT: In-Cabin Virtual Reality Development
DATE OF PUBLICATION: May 1, 2015

Table of Contents

Introduction

Basic Technology

Our product, as of now, uses the Oculus Rift DK2 headset. It is a light, virtual reality device that straps on over the user's head and eyes. To start, we plan on developing some basic environments that a passenger could interact with. These basic environments would involve offering the passenger an escape from the reality of sitting in a sweaty, pressurized tube 30,000 feet in the air. They will also be used to create an effective flight attendant notification system, complete with both a menu and flight information panel. To complete this initial level of programming we will utilize Unreal Engine 4, a game development program that provides the tools necessary for both basic, indie-level development and high-performance games.

Background on Oculus Rift

According to TechCrunch, Palmer Luckey founded the Oculus VR Corporation in 2012. Long before he launched the company that released the Oculus Rift DK1 headset, Luckey developed his initial concept of virtual reality. At 16 he was self-sourcing old VR headsets and building his own prototypes. While developing his own concept, Luckey became acquainted with John Carmack, the co-founder of id Software, which revolutionized the gaming industry. Carmack helped publicize one of Palmer's VR prototypes, launching a new era of VR headset technology. Earlier this year, Oculus VR was acquired by Facebook for $2 billion. The Oculus Rift headset has primarily been used for gaming purposes so far. This year Oculus VR released their second generation model, which included these improvements:[1]

- Reduction in potential for motion sickness by decreasing latency (the time it takes for the device to respond to your motion)
- Better screen resolution (960x1080) to decrease the "screen door effect" (low resolution coupled with magnification around the image)
- Decreasing screen blurring by changing the display from LCD to OLED

Future of the Technology

The first commercial Oculus Rift product is slated for release next year. It will not be a perfect product, but it will be essentially unprecedented. Luckey compares it to Ford's release of the Model T, and expects great improvements in the future. So great, he said, that he believes virtual

reality will mimic real life within his lifetime.[2]

---

[1]Greg Kumparak, "A Brief History of Oculus", TechCrunch, last modified March 26, 2014, Accessed December 5, 2014. http://techcrunch.com/2014/03/26/a-brief-history-of-oculus/

[2] Taylor Clark, "How Palmer Luckey Created Oculus Rift," *History, Travel, Arts, Science, People, Places* | *Smithsonian*, November 1, 2014, Accessed November 18, 2014.

Currently, technology is in development that could greatly serve our project. External cameras, which use 3-D and depth to map an environment around them, are in development for the Oculus Rift. A feature we would like to eventually address is using these external cameras to feed a 360-degree canvass of the sky around the plane to sitting passengers.[3]

Minimum Viable Product

Our minimum viable product is a virtual reality environment for the Oculus Rift that includes as much input from our voice of customer as possible. This MVP focuses on the aspects of our product that are attainable in our timeframe, which are almost entirely software. We have ideas and concepts for the physical implementation of the headset with the plane, but for the meantime, we have decided that our focus should be on software. Discussion of hardware and implementation can be found later in this section and in the appendices.

The input from the voice of our customer that we are focusing on for our minimum viable product is contacting the flight attendant through our environment, viewing the in-flight menu, and viewing a flight tracker. Our MVP will include three interchangeable environments: a sunny environment, a rainy environment, and a snowy environment, complete with a day-night cycle.These environments keep the user in one spot while head tracking on the Oculus Rift allows the user to look around. The three aforementioned options can be selected by simply looking at the corresponding in-environment object. The level can also be switched by looking at that option as well. The performance specifications are as follows:

- All options are activated after a .5 second delay from the initial look-at trigger
- Only one option can be active at once
- All environments should run at an average of over 40 frames per second on our demoing computer. This should be 60 frames per second easily when in standard mode, however stereo mode for using the Rift renders everything twice, once for each eye, so performance will suffer.
- All levels load and render fully in under 2 seconds

Our MVP is the product that we agree we can make during our time frame. However, we do have many ideas for the development of this project that we do not think are feasible within the time frame that we are working with. We will detail those here as much as possible. On the hardware side of things, we know that we will need a method for the physical implementation of the device on the airplane. There are three parts of the hardware setup that we must consider: the headset itself, the computer or server that runs the environment, and the cords that connect the two. There is also a position tracking camera, but it is not essential for what we are trying to do. The main idea is to build the system into the seat in a way that is unobtrusive to a passenger that does not wish to use the device. To start with, the Oculus Rift needs a computer or server to run the virtual environment. We have two basic ideas for this. One is to run all Oculus Rifts using a server placed somewhere underneath the seats and run the cables to the oculus up through the floor and into the seats. The other idea is to run each individual headset from small computers placed within the seat itself. For each of these methods, we would have the cord to the

[3] Ben Lang, "The Tantalizing Possibilities of an Oculus Rift Mounted Camera," *Road to Virtual Reality*, May 14, 2013, Accessed November 18, 2014.

headset run through the seat and end at a plugin somewhere at the top of the seat. This would allow someone who wants to use the headset to simply grab it and plug it in.

Along with the hardware plans for our system, we have a few software ideas for our environment that we do not plan on working on in our time given, but would still like to detail here. After our MVP, we would like to add more functionality to our options, including allowing users to order items from the menu in the same way they select other items, adding a realistic and fully functional flight tracker instead of the mock-up that we currently have, etc. Our most ambitious idea is to create a virtual tour that allows the user to view a real time feed from cameras mounted on the exterior of the plane. A company called Jaunt VR is currently in the process of developing a camera that can be used in conjunction with a virtual reality headset.[4] This would allow the headset wearer to see everything the camera sees. This technology is going to be used to make virtual reality movies, but it would work perfectly for our purposes as well. Once this becomes possible, it will be a huge selling point for our product. Other aspects of our environment that we have discussed including in our product are an augmented reality safety video, language lessons for international destinations, and a “trip advisor”-like mode for exploring your destination.

Throughout the spring semester, we have made a lot of progress towards our MVP. At the present stage in early March, both of our environments have been created and some of our functionality has been implemented. We are on track to have our MVP completed by the end of March. After that we will focus on refining the program and possible looking into laying the groundwork for future features. Our full prototyping schedule and progress is detailed in the prototype development selection.

A full preliminary FMEA summary can be found in the appendix but will briefly be summarized here. Because our product is purely software for the time being, the results of different failures will be substantially less severe than those of a physical product. We will not have to worry about our product injuring users except in some very specific, unlikely cases. The cases regarding failure of our software mainly involve the safety aspects of our environment. If our turbulence warnings or in-cabin announcement notifications fail, the user may be placed in danger without knowing. As such it is critical we design our environment to notify the user when these failures are occurring and cannot be fixed. We are looking into potential hardware accompaniments that could offer a simple, flight attendant-triggered physical notification (like a buzz or vibration) that would notify passengers when to remove the helmets even if the software notifications malfunctioned.

We also must deal with motion sickness.[5] This is a common side-effect of VR use and can vary greatly from person to person. Some users may experience no motion sickness, while for others it can be quite severe. This is a problem we can work to minimize with our environment, but in the end there are some instances where this will be unavoidable. The vast majority of problems with this environment however would be from errors in the code, and the result of these failures would be the software crashing. This is obviously unfortunate and unfavorable, but the main side-effect will be user dissatisfaction and, in general, not bodily harm.

---

[4] Ben Kuchera, “Jaunt VR makes history repeatable, and every spectator becomes part of the show,” *Polygon*, October 2, 2014, Accessed November 14, 2014.

[5] Noah Nelson, “Virtual Reality’s Next Hurdle: Overcoming ‘Sim Sickness’,” *NPR*, August 10, 2014, Accessed November 15, 2014.

Prototype Development

Development progressed ahead of schedule throughout this semester and we are currently completed with our MVP. Our past prototypes have acted as checkpoints in our progress towards our MVP and each one allowed us to evaluate and quantify our progress. Our first prototype was completed in December of 2014. It was created using Unity because we did not yet have access to Unreal Engine 4. The performance specifications for our first prototype are as follows:

- The program runs at at least 60 frames per second on the Oculus Rift
- Level loads in under 1 second

This prototype, while being simpler to create than the subsequent ones, was much less visually appealing as we were using very basic textures in Unity. We started by creating a default sized landscape and applying the default grass texture to the entire square. After that we painted the large-scale landscape materials on for our beach: half the square was covered in the ocean water material and a small space between the grass and the ocean was painted with the sand texture. After this, the landscape was sculpted to create some mountains and valleys and some rock material was added in certain areas as well. Palm trees (stock) were then added to the beach area and the Oculus Rift camera actor was placed on the beach near the water. The program packaged successfully and was tested with the Oculus Rift.

Our second prototype was similar to our first but it was created in Unreal Engine 4. This prototype was essentially a design model, one that has been a basis for all our subsequent designs. It had little to no functionality and all subsequent work has been built on top of this project. The environment is a tropical beach complete with realistic water, palm trees, and a crackling camp fire. This prototype and all subsequent prototypes were and will be developed in Unreal Engine 4 using visual scripting.[6] The performance specifications for our second prototype are as follows:

- The program runs at at least 40 frames per second on the Oculus Rift
- Level loads in under 1 second

Design of this prototype was begun by creating two adjacent, default-sized landscapes in Unreal Engine. All of the materials used were either from Unreal Engine's "Content Examples" project, or created from textures in there. The Ocean material was added to one of the landscapes, but for the other, we created our own multi-layered material from stock textures and normal maps. This material is composed of grass, rock, and sand. This layered material is necessary to create if you want to paint landscapes in Unreal Engine 4. The exact material make-up can be seen in the appendices. After the material was applied, we painted sand on the boundary beneath the water and sculpted mountains and valleys in the background. We also painted rock material onto parts of these mountains. After the landscapes were arranged, we created a campfire using a cylindrical brush for logs and the fire particle system. Lastly, we added the default fire sound effect and an ocean waves sound clip to the environment.

[6] "What is Unreal Engine 4?" *Unreal Engine*, Accessed November 16, 2014.

For our third prototype, we added some functionality to our beach environment. A day/night cycle was created and used for our second environment as well. We also added some basic logic that acted as a placeholder flight attendant call. When the spacebar is pressed, the program toggles text box saying that the flight attendant has been called. The main thing we will be testing with this prototype is if we can correctly receive user input to activate the call flight attendant option. The performance specifications for our third prototype are as follows:

- The program runs at at least 40 frames per second on the Oculus Rift
- Level loads in under 1 second

This prototype added a couple simple features to our first environment. The flight attendant call was very simple to make, although the design was altered in subsequent prototypes. In the HUD blueprint, we added an event tick that waits for a spacebar key press. This key press toggles a text box in the heads up display that notifies the user that the flight attendant has been called. The day-night cycle was added in this prototype as well, with a design that can be transferred to our other environments. The basework for this code was found online in a simple day-night cycle tutorial which can be found here.[7] Our blueprint is very similar to this implementation.

For the fourth prototype we created our second and third environment as add-on levels in our project. This will allow us to freely switch between the three environments later. These environments have the same interior design, a large living room area in a house, but different exteriors, one being rainy and the other snowy. The main purpose for this prototype was to ensure that we have multiple relaxing environments so we can have some variety in our program. The performance specifications for our fourth prototype are as follows:

- The program runs at at least 40 frames per second on the Oculus Rift
- Levels load in under 2 seconds

The house for these environments was build using square brushes for walls. We first created a large cube with these brushes on top of a default sized landscape. Then we separated a quarter of this cube off to be a separate room using two copied brushes. One more loft-like room was created by separating off a corner area of the larger room with two more brushes. Next up, we created windows and doors by overlaying subtractive brushed into the walls that were created in the previous steps. Default windows and door frames were added to these subtractive brushes as well as a glass texture to the windows. The walls, floors, and ceilings were textured using different stock materials. A spiral staircase was added going into the loft using the spiral stair brush. After that, lighting was added to the house using the default hanging and wall mounted lights.

With the interior done, we started the exterior by sculpting the landscape. A pond was added near the house by placing a small landscape with the lake water material in a hole sculpted into the main landscape. After this was done, a copy was made of the level so we could make one environment snowy and the other rainy. Painting the landscapes was done by making two multi-layered landscape

[7] https://wiki.unrealengine.com/Tutorial:_Time_of_Day

materials, one with snow, rock, and dirt and the other with grass, rock, and dirt. After the landscapes were painted, some dead trees from Unreal Engine's Content Examples project were placed around the exterior. The last thing that needed to be added to our environment was weather. Using the VFX Weather Pack from the Unreal Engine marketplace, we added a snowstorm to our snowy environment and rain along with lightning and clouds to the rainy environment. The day-night cycle from the first environment was used in these environments as well.

For our final prototype, one that reaches the guidelines set by our MVP, we added functionality to our environments. This includes being able to toggle a flight attendant call, being able to look at an in-flight menu, and seeing the flight tracker. This time, we did not create any new environments, but we did ensure that all of the functionality works in every environment and they can be seamlessly switched between. To do this, we are not using an input device. How this happens is that for each option, there is a corresponding object in the environment. By tracking the user's line of sight, the program activates options when its object is looked at for a short amount of time. We also wanted to ensure that our menus look presentable and that the user can switch between all three modes very quickly. The performance specifications for this prototype are the same as those outlined for our MVP

The implementation of our menu switching was relatively simple and blueprint code for it can be seen in the appendix. In our environments we placed four square tiles with floating text on them designating their functionality. Since we are not using an input device, the user cannot move around the environment. These tiles are placed and the environments were built with that in mind. While everything may look complete from the user's point of view, wandering around the environment will show make this design choice clear. These tiles include the options "call flight attendant", "view menu", "view flight tracker", and "switch level". The implementation of the activation of these menus relies on line of sight tracking and object visibility. In the program, we implemented an event blueprint that keeps track of the vector that points in the direction the player is looking. When this vector collides with an object, a function returns that object. In the code we have numerous if statements for checking which object the collision occurred with. When these statements return true, an event is activated that is directly correlated with the object that was looked at. These events toggle the visibility option of certain objects in the environment. Each event makes every object associated with the corresponding looked-at object visible while also making the objects from the other menus invisible. Adding in a short delay in the trigger and a fifth smaller tile that closes all menus makes this remarkably smooth and an effective no-input device solution. After adding visuals for each menu, our minimum viable product was achieved.

Testing for these prototypes was fairly straightforward. In each one we added certain functional elements that either worked or didn't. We had to go through each of these options and assure crashes didn't occur when switching between them. Aside from software issues like bugs, we wanted to have people try out the prototypes and give us feedback. The most important things we were looking for is if the environment is pleasing or relaxing, if they become motion sick at all especially if they are wearing it for an extended period of time, and if they feel that the input and the display are intuitive. Based on the user feedback, we may need to make changes to our prototype.

Summary

Our product is a virtual reality environment for the Oculus Rift that enhances passenger comfort and safety in a number of ways. We will focus on the software aspect of the implementation, but we have laid out design plans for the hardware aspect as well and will continue to assess and refine those designs. The environment itself allows the user to relax in a virtual world of their choosing as well as contact the flight attendant and view pertinent information to the flight such as in-cabin announcements, in-flight menu and ordering, bathroom occupancy, flight position and estimated time of arrival. Our MVP is complete, but there is still plenty of work left to do to improve our product and lots of potential next steps.

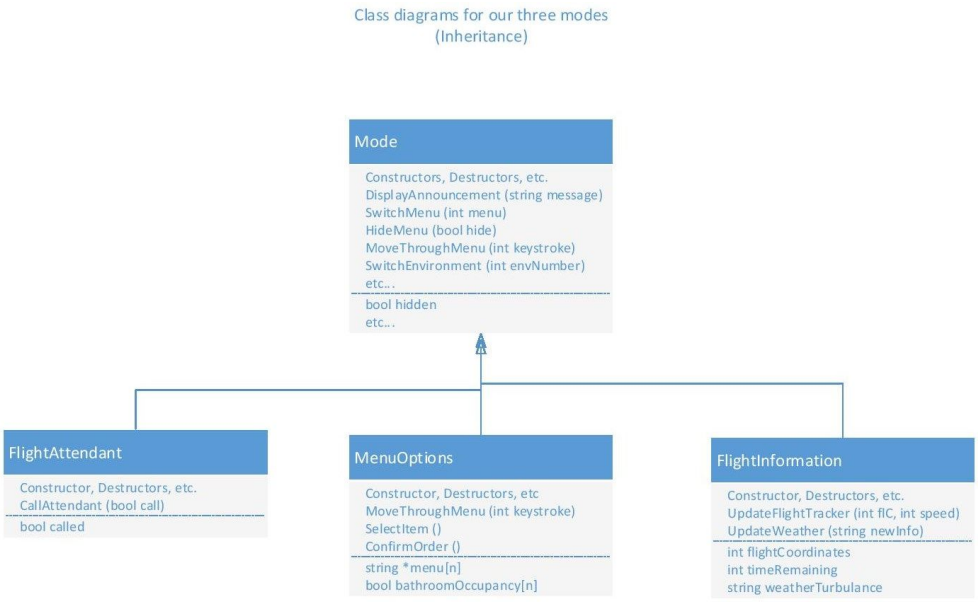
Appendices

Failure Modes and Effects Analysis

SEV = How severe is effect on the customer?
OCC = How frequent is the cause likely to occur?
DET = How probable is detection of cause?
RPN = Risk priority number in order to rank concerns; calculated as SEV x OCC x DET

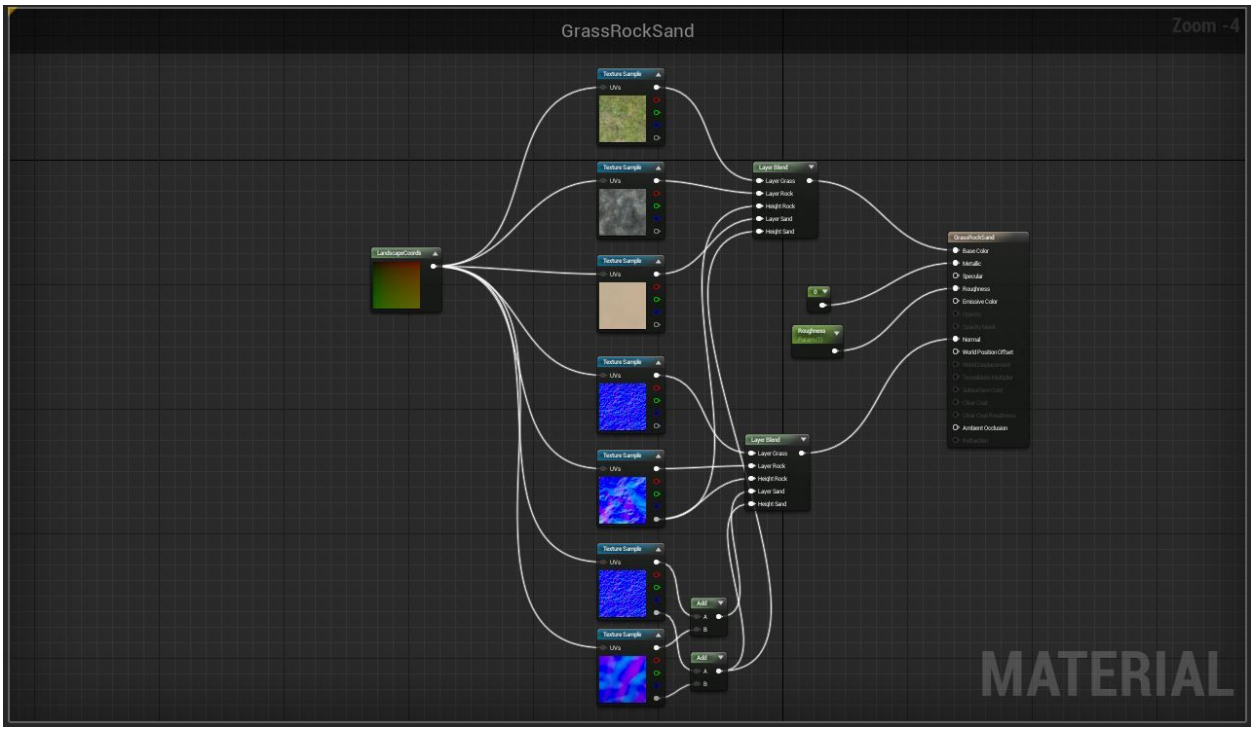
Process step	Potential failure mode	Potential failure effects	SEV	Potential causes	OCC	Current process controls	DET	RPN	Actions recommended	Responsibility (target date)	Actions taken
The environment is active	The environment crashes	The customer is dissatisfied	4	Infinite	7	Built in preventative measures. We must take every precaution to ensure crashes do not happen.	6	**168**	Debug the code and find the error. We can use the context of the crash to get a rough estimate of where the problem should be.		
	The environment gives the user motion sickness	The customer feels sick, will not want to use the program anymore	6	Overactive environment, overactive user, user is especially succeptable to motion sickness	8	Attempt to limit the motion of the environment, but can be uncontrollable. Many people get motion sick from VR environments.	2	**96**	Attempt to find where the problem occurred, and limit the motion even more, but can be unfixable.		
Displaying safety information	Our saftey measures malfunction, warnings are not correctly displayed	The customer is put in danger, is unaware of incoming turbulance or has not heard in-cabin announcements	10	Our environment is gathering or displaying safety information incorrectly	4	Implement a test to ensure warnings are being received correctly, if not display a warning that the information is not available to ensure the customner is aware	6	**240**	Debug the code to find the problem		
Displaying in-flight systems information	The stewardess cannot be called	The customer is must manually call the stewardess	2	Error in the code	5	Implement checks to ensure information is displayed correctly or to notify user if it is not	3	**30**	Debug the code to find the problem		
	The bathroom occupancy is displayed incorrectly	The customer may believe the bathroom is always full and will become uncomfortable	4	Error in the code	5	Implement checks to ensure information is displayed correctly or to notify user if it is not	3	**60**	Debug the code to find the problem		
	The menu is not able to be displayed	The customer must order manually	3	Error in the code	5	Implement checks to ensure information is displayed correctly or to notify user if it is not	3	**45**	Debug the code to find the problem		

UML Class Diagram

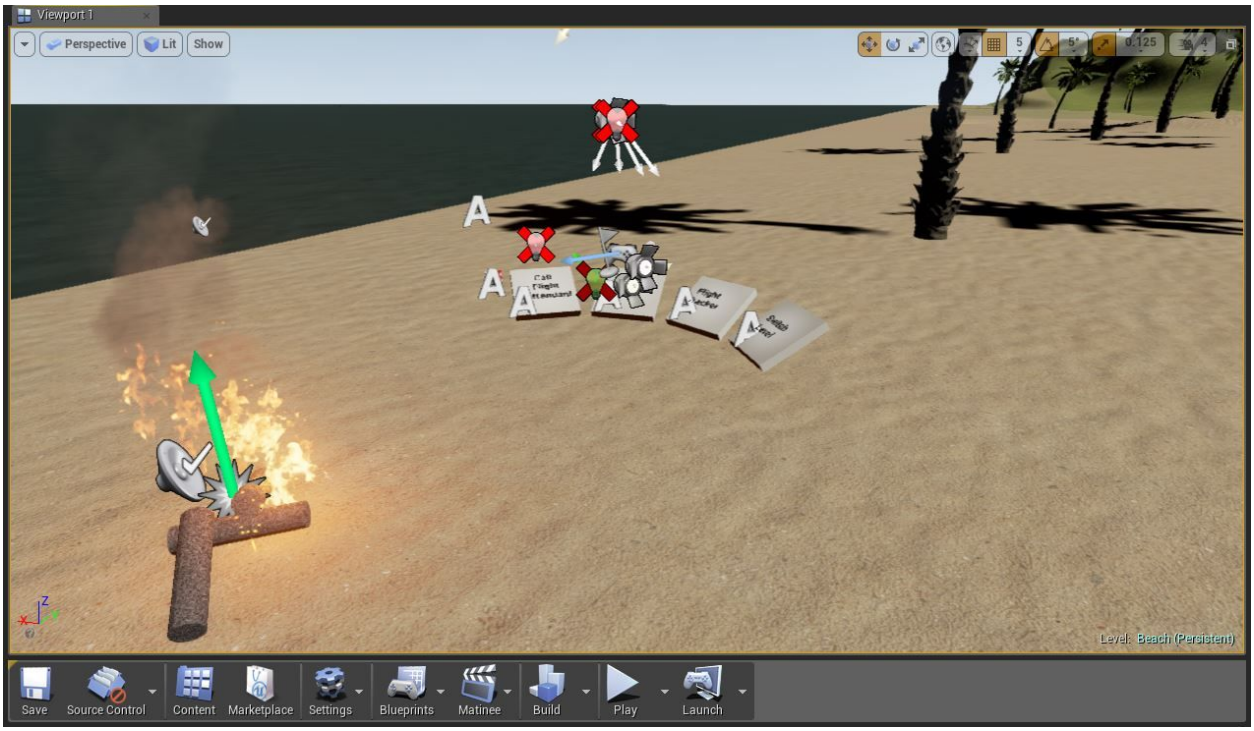


Menu classes like these (above) would be useful for the fully functional implementation

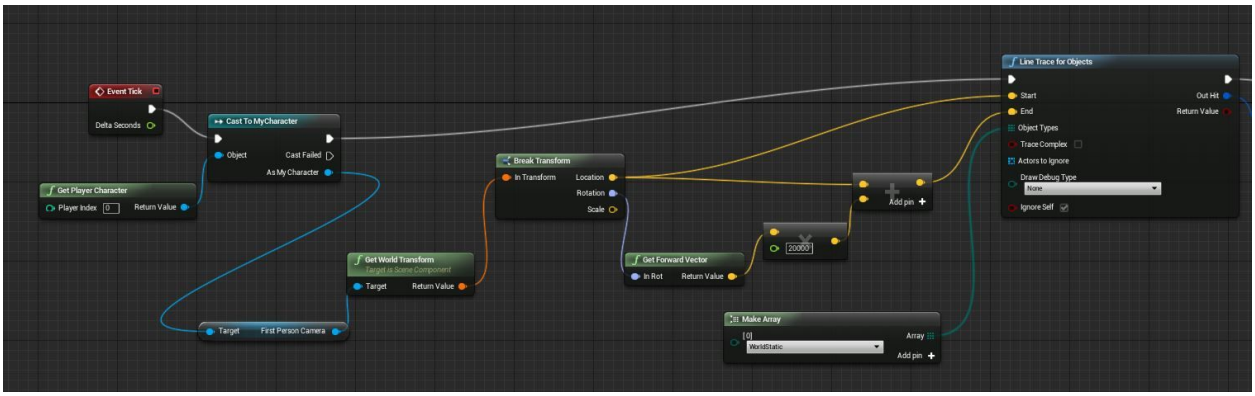
Development Process Images



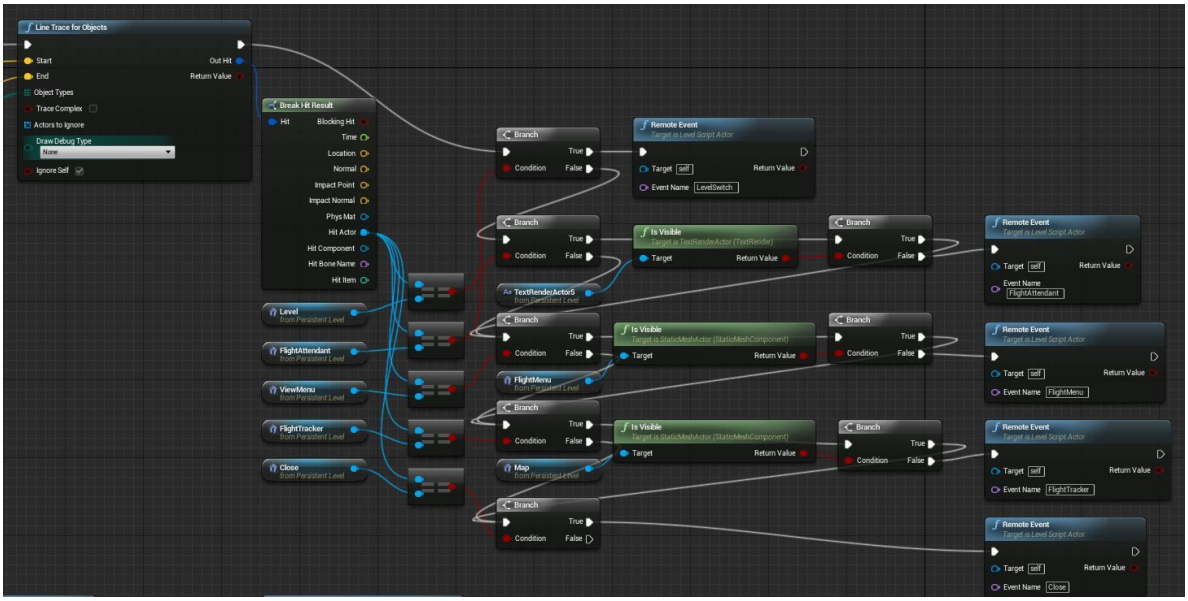
Landscape material from beach environment (above)



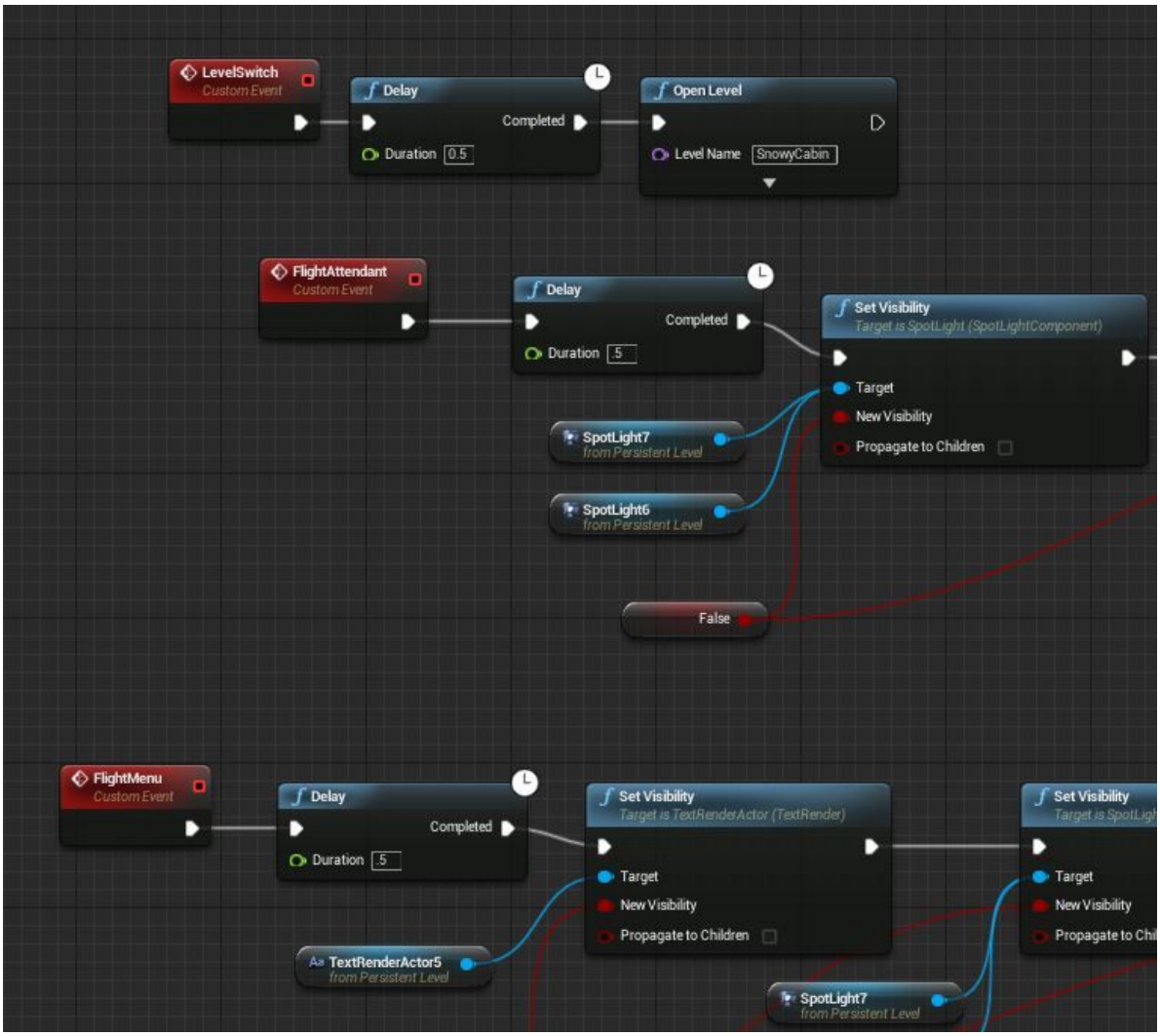
User area of first prototype (above)



Camera vector collision trace used for finding object being looked at (above)



Checks for which object is being looked at and if that object is currently visible (above)



Some of our remote events for the menus, all toggling object visibilities (above)

Citations

Clark, Taylor. "How Palmer Luckey Created Oculus Rift." *History, Travel, Arts, Science, People, Places* | *Smithsonian*. November 1, 2014. Accessed November 18, 2014.

“Dev Kit 2.” *Oculus.* Accessed November 15, 2014.

Kuchera, Ben. “Jaunt VR makes history repeatable, and every spectator becomes part of the show.” *Polygon*. October 2, 2014. Accessed November 14, 2014.

Kumparak, Greg. “A Brief History of Oculus”. *TechCrunch.* Last modified March 26, 2014. Accessed December 5, 2014.

Lang, Ben. "The Tantalizing Possibilities of an Oculus Rift Mounted Camera ." *Road to Virtual Reality*. May 14, 2013. Accessed November 18, 2014.

Nelson, Noah. “Virtual Reality’s Next Hurdle: Overcoming ‘Sim Sickness’.” *NPR*. August 10, 2014. Accessed November 15, 2014.

“What is Unreal Engine 4?” *Unreal Engine*. Accessed November 16, 2014.