Agile Grapple-Based Movement in VR

COMP 8037

Zoravar Lalli - A01019819
December 7th, 2021

Table of Contents

1. Student Background

My name is Zoravar Lalli, I am a computing student with an interest in AR/VR technology and user experience.

1.1. Education

I have graduated from the Computer Systems Technology diploma program at BCIT and I am now pursuing my Bachelor's in Technology degree with a specialization in games development.

1.2. Work Experience

I have had experience in AR development for the first HoloLens device. I along with 3 others created a warehouse guidance prototype for CAMS Software using Unity and the Mixed Reality Tool Kit. The prototype's purpose was to help workers find and pick items in a warehouse environment via guidance graphics and voice input in order to provide a hands free solution for item management. After this initial experience I began to look into AR and VR more and discovered that the development process is quite similar in VR.

2. Project Description

I'm looking at this project as an opportunity to learn more about VR development and to create a new style of movement to better cater to users who desire a more agile and precise movement system in VR. I want to create a grapple-based movement system for traversing VR environments quickly with agility and precision. This movement system is meant to allow the user to move in any direction via grapple points that are shot from a hand held device as projectiles. I plan to design this system to combine aspects from two existing movement systems into a new hybrid that should address issues in both of the older existing systems individually regarding agility and precision.

3. Problem Statement and Background

There are existing implementations that attempt movement via grappling; however, they do so in such a way that the user's agility, precision, and freedom of movement are limited significantly. These two existing implementations are the Spider Man swing-based movements and the grapple & reel based movement in the Attack on Titan mod for Blades & Sorcery.

Take the Spider Man swing-based movement for example. The way that it currently works is that the player can aim and shoot a grapple (in this case a web) on any point on a building perpendicular to the player, this will initiate a swing and the player can hold the grapple through until they release to shoot the next web. Now the issue here is that during the swing the player has little control over their path of travel, the angle at which they swing, and their momentum.
It also feels as if the player is constrained to an invisible track which mimics free movement, but does not provide true free movement.

On the other hand there is the grapple & reel based approach in the Attack on Titan mod. In this one the player can shoot grapples just like they can webs in the other implementation however, instead of

swinging along a track between two grapple points they can reel in towards them. In this case the player gains momentum as they reel in instead of as they swing downwards. The problem in this case is again related to freedom of movement and a sense of control. Once the player is off the ground and reeling toward something, they cannot adjust their trajectory or adjust their momentum direction anymore. They are essentially locked into the trajectory path as soon as they launch their grapples. On top of this, it is quite hard to accurately shoot and land grapples where the user desires in the first place, so the lack of adjustment once you are committed to a grapple is even more detrimental to the experience.

Overall both existing systems succeed in allowing the player to move with grappling devices however, they fail to provide a system in which the player can do so with an adequate sense of agility or for any practical purposes other than novelty. If a player wants to traverse complex geometry or attempt to maneuver between, around, or above and below objects, they will struggle as the mechanics do not allow for precise movements or adjustment.

My proposed solution is to create a new hybrid that combines the best aspects of both of these systems to provide that lacking agility and control. Individually swing grappling and reel grappling yield a hard to aim and imprecise system of control. However, I believe that providing both of these methods to the player to use in conjunction could be the solution to providing more agile and precise movement. The swinging grappling and the ability to interrupt said grapples to instead reel in toward a specified point is a method of control that is not used in VR grappling systems as of now. That core concept of interrupting one mode of grappling to switch to another is the oversight that has caused existing implementations to be hard to use and imprecise.

4. Scope and Depth

This project will run over the course of the final semester in the B-Tech program so there is roughly three months from January to March to work on this. I already have access to the necessary hardware and software to develop this project as I am using my own personal headset and development tools.

The time is limited and this is an exploratory project so de-scoping aspects that do not pertain to the primary purpose of the project is necessary.

As for feasibility of the project, at this point I have reason to believe the below is achievable once I have a basic knowledge of the development processes for the quest II in Unity. My reasoning is based on my previous experience developing for an AR project with no prior knowledge in a similar amount of time.

Below I have itemized a high level breakdown of in-scope and out-of-scope features for the project.

In scope:

Shoot grapple	The player can click the back and side triggers of either controller to shoot a grapple from the corresponding hand in the pointing direction of that hand.
Release grapple	The player can release the currently held grapple point by either releasing a held trigger or by clicking a trigger again like a toggle. (I will provide settings to

	set a preference for which release method the player uses)
Reel grapple	The player can press the A button of either controller to reel the corresponding grapple in, pulling the player towards the grapple point.
Swing	The player can enter a swinging state by holding the side triggers on the controller while grapples are latched onto points.
Reel	The player can enter a reeling state by using the reel grapple input while the grapples are latched onto points.
Gain momentum	The player can gain momentum by chaining swinging movements or by reeling in towards grapple points. (the reeling will gain momentum at a constant rate)
Adjust weight distribution	While the user is in a swinging state, they can move their head to manipulate their in-game body's weight distribution to adjust the swing closer or further from the axis of rotation.
Adjust momentum direction	The player can use their look direction while mid-flight between grapple latches to adjust the direction of their momentum to enable the player to have a sense of turning in the air and allow more precise movement.

Out of scope/Nice to have:

Graphics/Texturing/ Assets	I plan to include at least a few assets to give the player a proper sense of the 3D environment they are in. However, I will limit the development time into graphical polish or aesthetic overall as it does not pertain to the primary purpose of the project.
Sound	I plan to include some sounds to provide the user the audible aspect of having performed an action. For example I may add a grapple launch and latch sound.
Complex geometry (designed map)	When I say complex geometry I mean landscapes and geometry like that in a full map in a video game including alleyways and tunnels with enterable buildings. I will be using mostly rectangles in different sizes, orientations, and combinations to create complex obstacles to be maneuvered in order to test the features.
Additional headset support	Due to my lack of experience in VR development, I will be developing for only one platform which is the Oculus Quest II specifically. I will not have support for other VR controller schemes or sensor setups.
Hand gesture based inputs	I plan to create the functionality without hand/arm gesture inputs, however as a stretch goal I think it could be interesting to try and provide additional control with arm gesturing. I also feel this specific feature wouldn't work as well on a quest II as it could on other devices that use room scale sensors.

User Interface	Although I will include some basic UI elements or an options tab for the user, a polished UI or HUD is not necessary.

5. Test Plan

Since this is a project aimed at user-level interactions in a VR environment I have decided the primary testing method will be to host play testing sessions following the release of a major milestone.

Play Testing

The purpose of these tests will be to gain user feedback so that it can be taken into account and applied as development continues.

Every play test phase will be conducted like so:

Pre-Test

- A group of volunteers is selected prior to the testing phase
- A schedule is produced for testing sessions for each user (testing is likely limited to one person at a time due to COVID restrictions, limited space, and limited headset availability)
- A volunteer arrives at the play testing area and is provided a standard release form to participate in the test.
- The volunteer is given the testing equipment and if needed, a tutorial on how to use the device generally in a test environment. (The goal is to see how intuitive the controls are so only a basic tutorial is given for using a VR headset, not the controls of the game)

Conducting the Test

- Player is loaded into the play test level and told to attempt to maneuver between marked points in the environment. (Specific instruction will vary from test to test depending on test purpose)
- Player is observed by the test conductor for the entirety of the test; however, the conductor must remain silent and allow the player to test without distraction.
- Surprising or interesting behaviors/dynamics are taken note of to be reviewed later.
- Test conductor is on standby in case the volunteer has any questions or issues with the test.

Concluding and Feedback

- Upon completion of the instructions provided by the conductor the volunteer is now given a google forms survey.
- The survey contains questions pertaining to the purposes of the play test and also includes an open feedback section for suggestions and thoughts.
- Surveys are collected and results condensed into google forms survey statistics sheet
- Statistics and responses are logged for review when planning next sprint

High Level Test Criteria

Below I define the core criteria in terms of the final build's use cases.

Use Case	Description	Pass/Fail Criteria
Walking & Turning inputs	The player should be able to walk and turn without moving their actual body for ease of use or for when the player wishes to get to a desired location without using physics.	Player game object collides with the environment and analog inputs are used to adjust game object orientation and position to traverse said environment on the horizontal plane.
Grapple Aiming	Player should be able to aim grapple via their hand direction and angle (hand angle is the aiming vector and hand position is the grapple launch point)	User is able to accurately aim the grapples at other game objects (any collidable object with 'GrappleAble' property)
Grapple Shoot	Grapple is shot at the currently aimed direction when the user presses a button	User is able to shoot a grapple in their current aiming direction that attaches to the environment once colliding with it.
Head (leaning) Input for Swinging	Players can use their head position to lean left and right mid swing to influence the swinging direction and speed.	Leaning left and right updates the player game object's weight distribution variables which in turn adjust the path of the swing closer and further from the axis of rotation
Head (turning) Input for momentum direction	Players can use their head orientation mid flight to adjust the momentum direction in the direction they are turning their head.	Rotating the headset while in the air and without grapples latched updates the momentum direction of the player game object and guides the object in that turned direction
Reel and momentum gain	Player should be able to reel in latched grapples by holding a button and gain momentum while doing so	Player game object's momentum value is updated as the reeling occurs, player can trigger the reeling via button hold
Grapple swinging	Grappling while mid flight with a momentum value will result in an intuitive swinging motion. The user should be able to infer the direction they will go in based off of a sense of speed and corresponding torque when grappling	Players should swing in the direction calculated by using the orientation and position data of the hand sensors, head sensor, momentum, and the grapple latch positions relative to the

	at speed to manipulate path of travel.	player position.

6. Methodology

I plan to treat this project as a game development project similar to those I have worked on previously for desktop PC games on the Unity engine. The major difference of course will be learning the additional components of Unity that are required to communicate with the Quest II hardware.

I will be coding in C# as this is the native scripting language of the Unity development environment. For the editing tools I will be using Visual Studio Code.

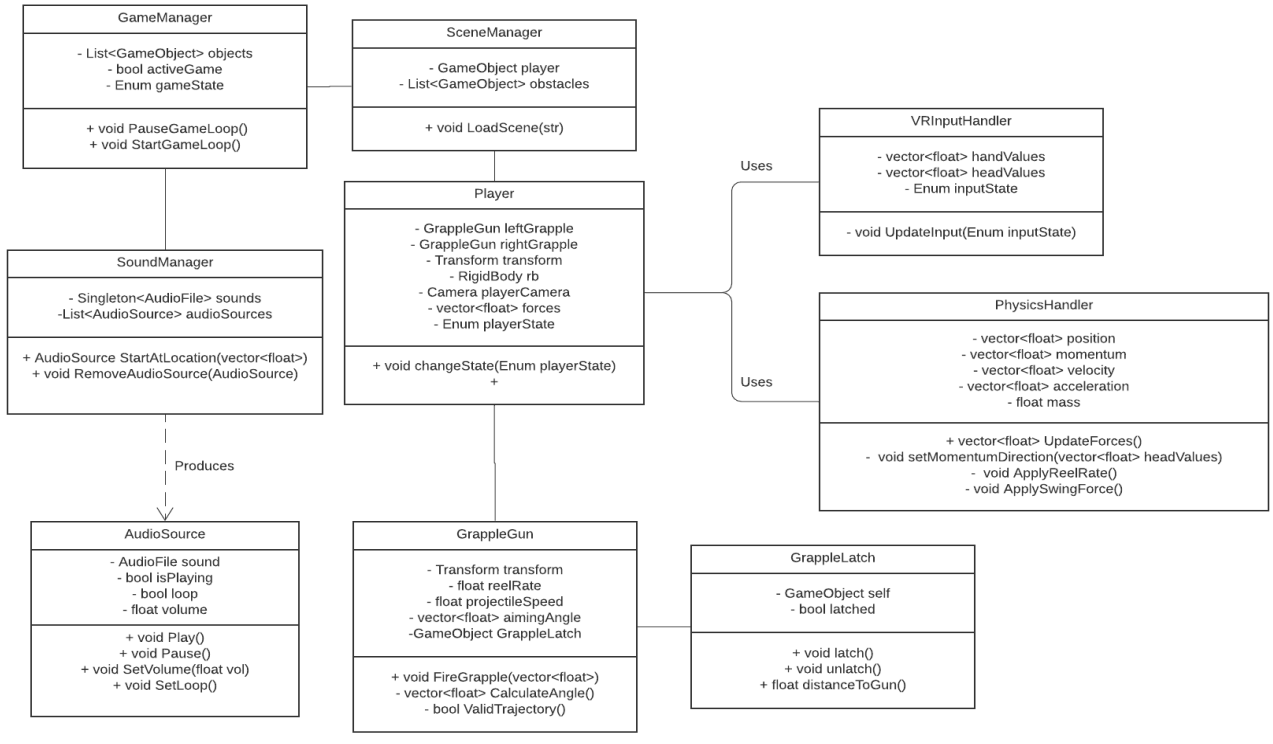
Task time will be split between development and tutorial in the beginning as I will be learning to use the Oculus Integration toolset while I develop this project.

For task management, I will make a personal Kanban board to utilize throughout the project and I will be following Agile Management Methodologies for planning my development phases in two week sprints.

As for any specific patterns or algorithms I am thinking of using, at this point I do not have any concerns to justify a specific pattern or algorithm choice. However, as I proceed with the project I will update accordingly as needed.

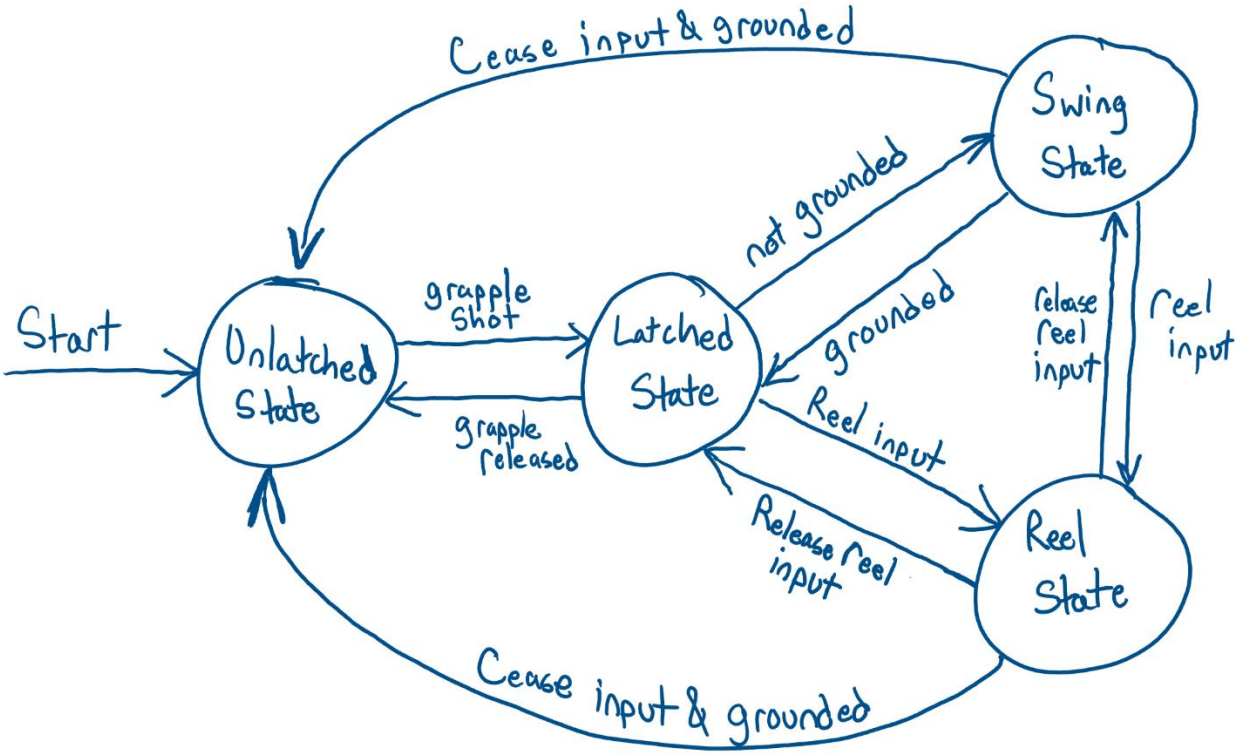
7. System/Software Architecture Diagram

Here I have provided a high-level breakdown for the planned classes in the project.



8. State Diagram Breakdown

This is a diagram representing how a user's movement state will flow.



Note:
Physics constraints will be coded to simulate real-world Newtonian physics with the exception that momentum can have its direction adjusted via introduction of "imaginary" forces applied as a result of the head turning input. There is potentially going to be an additional state integrated called "mid-air" for application of momentum direction adjustment however this is unincluded currently as the factors that contribute to allowing this momentum adjustment may change as I develop the project and in turn change the need for such a state.

Some examples of how I plan to code the physics:

Swing State:

```
// In a fixed loop for updating physics values of Player object
If(!atBottomOfSwing)
        Player.accel -= gravityForceValue;
// In a physics handler for the player object's physics values being changed
If(Player.state == States.Swing)
        // Function that considers active latch points to determine
        // a swing trajectory according to player's Mass and relative position from the points
        // This data will be used as inputs to calculate the trajectory and speed
        calculateSwingPath()
```

Reel State:

```
// In a fixed loop for updating physics values of Player object
if(grappleLength > minReelLength) // if not fully reeled in
        // calculateReel function will determine the rate of reeling depending on
        // existing forces from the player (if we have forces in the direction of
        // reeling they will be added onto and accounted for)
        vector<float> forces = calculateReelForceAdjustment()
        player.updateForceVector(forces)
```

9. Innovation

In both current systems the player will face issues with precision, agility, and they will feel they have a lack of control when attempting to traverse the environment. These problems become especially apparent when facing more complex geometry or attempting to maneuver around objects instead of past them or toward them. I want to innovate upon these systems by combining the mechanics of both to provide the player with the ability to swing from a grapple point and also reel in on the point they landed could be the key to providing the control necessary for more complex maneuvers. I would also need to introduce some new aspects of control to allow for precise adjustments of trajectory during flight to allow for more agility. The first aspect would be a weight distribution control during the swinging motions. This means that the player will be able to lean left or right during a swing in order to change the swinging angle via their headset orientation relative to their body (what the player would view as adjusting their weight distribution). Second, I would introduce the ability to use the look direction of the headset to adjust the momentum direction when flying from momentum gained via reeling in grapples.

10. Complexity

This project's difficulty is primarily related to my lack of experience in VR development, but also due to the fact that this movement in VR is still something developers are trying to create today. The two existing implementations I mentioned are genuine attempts at creating a good movement system, and I've highlighted where they fail. Creating a system that succeeds where these current systems fail will require research on physics, Unity VR development, and oculus hardware (specifically the quest II). I chose this idea because I've always wanted to do it but never had the means as a student. In the diploma program we had a lack of knowledge on physics, games development, and especially VR related software. Now that I've been in B-Tech for a year, I feel I have my foot in the door in that regard and that I have enough knowledge on those topics to be able to venture out and attempt this project.

11. Technical Challenges

I will need to learn a lot about the Unity development process for quest II applications, although I do have limited experience with the MRTK with HoloLens development on Unity, it has been years now and the processes of course have changed. I will also need to apply physics concepts and test them in a virtual space, something I haven't done much in previous projects, let alone a VR one.

Also, the use of physics in the project and fine tuning the user experience to feel satisfactory will be challenging as I haven't had much experience with physics in general and I understand translating those concepts into satisfying gameplay will be quite a challenge.

12. Development Schedule and Milestones

Below I have broken down the planned task schedule into hours. This task schedule will be followed over the three month duration of the project distributed evenly over bi-weekly sprint iterations.

Milestone/Phase	Tasks	Time
Project Start	Create Unity Project, set up KanBan board with initial tasks, initialize git repo and set up any other relevant tools for development	2 hrs
	Acquire and Integrate Oculus' tool set into the Unity project	6 hrs
	Utilize Oculus Integration demos/tutorials and understand fundamentals of how Unity communicates with VR hardware through it (5 days of development time)	40 hrs
	Create a base environment that works with the Oculus integration pipeline	6 hrs
	Test base environment to ensure that there are no performance issues or errors with the Oculus hardware running Unity in the base environment	2 hr
	Create and test input handling for Quest II controllers and sensors	4 hrs
	Total Project Start Time: 60 hours	
Prototype Phase	Create game object/Rigidbody to represent the player physically in game	1 hr
	Create simple model or acquire asset to be used for player in game (Important for a sense of self and orientation in a VR environment)	3 hrs

	Create a sensor interpreting script so that adjusting the position of Oculus sensors adjusts the mesh data in game to reflect the posturing of the player in the player's model.	6 hrs
	Create a collidable testing map with obstacles using ProBuilder in Unity	4 hrs
	Create movement mechanics for turning and walking using analog inputs	2 hrs
	Create/test collisions between the player, grapples, and environment	2 hrs
	Create or utilize a VR camera handler that will turn the unity camera according to the headset's orientation	3 hrs
	Create an aiming script that uses the wrist angle of both hands to calculate a grapple landing point on the environment	5 hrs
	Create a grapple shot mechanic that uses the aiming script to fire and latch a grapple projectile on the targeted position. Also test a hitscan approach and choose the better between the two	6 hrs
	Create a model & game object to represent grapple gun/grapple hook	4 hrs
	Create a script for drawing a grapple line based on grapple trajectory, this script should also check if the line would be valid or if it collides through other objects	2 hrs
	Create a script for reeling the player object towards a latched grapple position at a constant rate	2 hrs
	Code physics to be applied when performing grapple shot and reel. Apply momentum gain and loss to the reeling script	8 hrs
	Create calibration options for the player to set their height and arm span during gameplay (likely going to use a T Pose and button press)	6 hrs
	Total Prototype Phase Time: 54 hours	
Alpha Phase (Playtesting I)	Create a set of survey questions that are curated to identify concerns regarding the prototypes base functionality and sense of control	4 hrs
	Find candidates for the first testing phase and schedule them for testing. Also book facilities to use during the test for the scheduled dates at BCIT	8 hrs
	Conduct play testing to gain insight on what changes and focus areas should be applied in the Alpha build (Aiming to conduct 3 tests minimum, 5 maximum)	10 hrs

	Review the observation notes and survey responses and compile them into the play testing document. Gain insights and make decisions for development based on this review.	5 hrs
	Define a set of physics rules to govern the player's movement when in a standing state	2 hr
	Define a set of physics rules to govern the player's movement when in falling state	2 hr
	Define a set of physics rules to govern the player's movement when in flying state	4 hrs
	Define a set of physics rules to govern the player's movement when in a reeling state	4 hrs
	Define a set of physics rules to govern the player's movement when in a swinging state (this will be the most complex physical aspect and will take the majority of the time for physics)	8 hrs
	Create a finite state machine so that different sets of physics rules can be applied to the player for a given movement state and to define how state transitions should be handled between certain states.	10 hrs
	Add input handling for using the headset orientation's horizontal component as a steering input to adjust the momentum direction during a flying state.	6 hrs
	Add input handling for using the relative orientation between the headset and controller sensors to calculate the player's leaning direction and to quantify how much they are leaning as an input to angle in said direction during a swinging state.	10 hrs
	Integrate the swinging and reeling physics to have a combined adjustment of momentum when transitioning between them	4 hrs
	Total Alpha Phase Time: 77 hours	
Beta Phase (Playtesting II)	Create a set of survey questions that are curated to identify concerns regarding the alpha's controls being intuitive and satisfying to use and also to identify any issues regarding the physics of the game	4 hrs
	Find candidates for the second testing phase and schedule them for testing. Also book facilities to use during the test for the scheduled dates at BCIT	8 hrs
	Conduct play testing to gain insight on what changes and focus areas should be applied in the Beta build (Aiming to conduct 3 tests minimum,	15 hrs

	5 maximum)	
	Review the observation notes and survey responses and compile them into the play testing document. Gain insights and make decisions for development based on this review.	5 hrs
	Bug fixing session and application of development decisions made from playtest reviewal	8 hrs
	Further physics testing and refinement based on playtest reviewal and own continued testing	10 hrs
	Create audio loading and audio player singleton to play sounds/music	2 hrs
	Find/create and add sounds for wind and grappling to cater to player's senses	4 hrs
	Add basic textures to the environment and adjust lighting so that the experience can be more immersive	6 hrs
	Total Beta Phase Time: 62 hours	
Final Polish	Implement stretch goals if time permits otherwise use this time to complete as many of the remaining tasks as possible (3 days of dev time)	32 hrs
	Add cleaner UI for an options menu (volume, exit, calibration)	6 hrs
	Add supplementary sounds/graphics/effects if time permits	10 hrs
	Code cleanup and refactoring of final source code	10 hrs
	Documentation review and finalization	8 hrs
	Total Final Polish Time: 66 hours	
Total Project Duration: 319 hours		

13. Deliverables

Below I detail each deliverable and a description of what it provides or its purpose in the project.

Deliverable	Description
Play Test Documentation	Each play test that is conducted will yield a set of corresponding documentation that will be compiled into a single playtest document. In each test I will document observation notes, survey responses and any relevant consent forms or feedback.
Technical Design Document	A live document that will be added to and maintained throughout the project. Contains details for the overall design and details of the project from planning to implementation and results.
Prototype Build	This will be the first working build created that provides rudimentary functionality for the use cases except for leaning and turning. It provides base functionality to allow a user to perform basic grappling and movement in a testing environment. This build is to ensure that the user input and basic interactions are operating correctly before attempting more complex physics and movement.
Alpha Build	This build will be the product of conducting play tests on the prototype build and applying changes based on said test's results. This build will address the immediate concerns discovered by the first play testing phase. The testing will be curated to ensure the player use cases function as intended for users in multiple scenarios. This build is focused on the refinement of basic inputs and user interaction as well as the initial development of swinging physics and momentum adjustment controls.
Beta Build	This build will be the product of conducting play tests on the alpha build and applying changes based on said test's results. This build will address concerns raised by the second playtesting phase which is focused on physics and player agility. This build will be a more refined version in terms of the user's experience being intuitive and satisfying by integrating user feedback pertaining to the physics and feel of the game.
Final Build	The final build is essentially a polished version of the Beta build as the final phase of the project will be spent polishing and refining the project to be the best it can be by the time of submission. Although play testing is not conducted anymore for this deliverable changes are still made based on the overall playtesting data gathered from prior phases. Any stretch goals that can be integrated if time permits will be included in this final build.

14. Conclusion and Expertise Development

As a student of the Games Development option in BCIT's BTech program I feel that knowledge of VR development processes and the ability to create VR software solutions could open doors for me not only in terms of career opportunities but in terms of my own creative freedom as well going into the workforce. After playing these games for about a year myself I have realized that the potential innovations that are made possible by VR technology is immense and mostly unexplored.

VR is a new frontier of technology that has been growing for about half a decade and I want to be a part of it as early on as I can. I feel that exploring an idea such as this one that exists in popular culture but has yet to be properly executed in VR is a great way to accelerate my learning. I will be able to learn the development processes for VR while also creating an innovative prototype that pertains to my specialization of game development.

15. References

My primary source of research to support my proposal has been conducted by testing these existing software/mods which I will list here.

Attack on Titan Mod for Blades & Sorcery

- https://www.nexusmods.com/bladeandsorcery/mods/2401

Spider-Man: Far From Home Virtual Reality Experience

- https://store.steampowered.com/app/1067800/SpiderMan_Far_From_Home_Virtual_Reality/

16. Change Log

- October 20th, 2021
    - Changed formatting to be like a resume rather than an essay
    - Moved last part of Complexity section into first part of Technical challenges as it made more sense to categorize it as such
    - Added section 4, part of section 13, and updated section 14
- October 26th, 2021
    - Added methodology and test plan sections
- October 30th, 2021
    - Document revised and adjusted for proper formatting and grammar.
- November 10th, 2021
    - Added the development schedule and milestone information (section 11)

    - Added a deliverable breakdown (section 12)
- November 11th, 2021
    - Added architecture diagram for high level breakdown of the project
- December 7th, 2021
    - Added state diagram breakdown and high-level physics approach info