Grapple Log:

**Day 0:**  
I want to make a gameplay prototype to show my abilities in programming, game design, and motion design. The game is focused on freedom of movement and traversal. The main mechanic is going to be a grappling hook that lets the player move fast and far with a lot of freedom. But the handling of the grappling hook will require some thinking and skill. These are the basic ideas, and I will see where the journey takes me. I am motivated to do this project since it is the type of gameplay that I really enjoy myself. Nowadays a lot first person shooter games feature grappling hooks, but they are always very limited in to not interfere with the rest of the game. But grappling hooks and other mechanics that allow for free movement have always fascinated me and I always wondered how a game would look that would just let loose on all restrictions. Big inspirations for this project are Titanfall 2, Insomniac’s Spiderman and Just Cause 3.

10.10.2024:  
The standard stuff for making a project in Unreal Engine. I created a git repository, an Unreal Project. Since I don’t want to spend time on modeling or rigging, I decided to use Unreal’s Mannequin first person hands. I exported the unreal hands and modified the rig in Maya to make animating easier. I deleted most of the skeleton because it still contained joints for a full body even though it only contained a mesh for the arms and hands. It already contained a joint for a weapon which gave me the idea to add a weapon to the character.

I imported the modified rig and made a new scene and created a new Character Class. I will try my best to do everything in this project in c++ which is not prototyping since I want to deepen my knowledge in it.

11.10.2024:  
Today I cleaned up my git project and had to update my gitignore file. I also set up the first-person hands rig in unreal and in maya. I added a bone to the rig that is intended for the camera. Then I added the same camera in maya and in unreal and attached it to the rig. This enables me to have the preview in maya of how the animation will look in unreal. It also enables me to add premade camera shake to animations.

12.10.2024:  
I struggled with basic input reading since I am still not as familiar with the Advanced input system and it being a bit complicated in c++ made it even trickier. But I powered through and managed to create basic looking and moving input. I also made some test Animations for my arms.

13.10.2024  
I found a Gun model online and attached it to the player hands. I had to experiemt around with some constraints, but I managed to make a walking animation. And I started setting up an animation blueprint for my player character.

14.10.2024:  
I Spend the day animating the Hand that holds a gun. I made an Idle, Jump, fall start and fall loop. I also implemented them.

15.10.2024  
I deleted some old files and tweaked my animation logic.

16.10.2024  
I started with the grapple side of the project. I modeled a simple grapple shooter to attach to the arm. Making the attachment the same in unreal as in maya was a bit tricky since they have different coordinate systems and handle attachment a little bit differently. I also created a shoot start and shoot loop animation for the left arm and implemented those into the animation blueprint.

I also created an actor class for the grapple shooter and added to the player character. The grapple shooter shoots out the newly created grapple projectile actors.

I also already started implementing the inputs triggers and a basic shooting logic for the grapple projectile.

19.10.2024  
I made grapple pull animations and I implemented the grapple pull in game. I experimented more with constraints and used aim constraints instead of inverse kinematics.

20.10.2024  
I refined the pull animation and made reeling in animations and implemented the reeling in the game.

21.10.2024  
I redid the reeling. Before the projectile would fly back towards the player and when it was close enough, it would be reeled in. Now I have a system that blends the position of the projectile back to the player over time. The time is calculated based on the distance to the player and cant exceed a specified “maximum reel in time”. The issue before was that, if the player flew too fast, the projectile would not be able to catch up and couldn’t be reeled in, effectively punishing the player for going too fast; the opposite of what I want to achieve.

I also added the cool down and the soft cool down and made animations for it.

22.10.2024  
I added a boost function in air when the player presses jump while falling. The boost just adds an upward force to the player character. I experimented with making it instantly stopping the fall but it felt weird. I also added controller controls.

23.10.2024  
I started with making a wall run mechanic and have made a detection for the trigger so far.

28.10.2024  
I want to make wall running a custom movement mode, but it seems to be quite complex to add custom functionality to it. I had to look up how to even do it. So far I managed to create the new movement component, create a custom movement mode for wallrunning and trigger and stop in in game.

30.10.2024

I am working on the wallrun movement mode and it is quite the challenge. But I have figured out how to access and correctly use the players velocity. This will open up the possibilities of letting the existing velocity influence the wallrun behavior. This is important to me, because I want to maintain the players flow when they use the wallrun correctly.

26.11.2024

After a long break due to external circumstances I picked up the project again and continued to implement the wallrun. I think I managed to do the most complicated parts. The wallrun feels quite decent already. I added a bunch of features. The wallrun retains the initial vertical velocity and smooths it to 0. Giving the player a controlled slow down of their fall or rise.  
It also retains a part of the initial horizontal velocity, depening in which angle the wall is hit when the wallrun gets triggered. The more parallel the hit, the more velocity is maintained. This again serves to not interfere with the players flow.

Another issue was the question of how fast the wallrun should be. I managed to make the speed variable. Depening on the input, the player can accelerate or decelerate on the wall. This makes the wallrun technically infinite but the alternative would be to always slow the player down on each wallrun. This would make the wallrun feel more like a punishment then an opportunity.

Lastly, I added a jump off from the wall. When the player runs on the wall and jumps, They get launched of the wall and upwards as if they jumped off it.

Lots of control variables needed to be created and tweaked to make the wallrun fell right. But I wouldn’t say that its perfect yet, it does need some more tweaking. It would probably also benefit from different feedback but I am having a hard time thinking about something that is doable with my skills and time constraints. I also plan to move into making more feedback and a mission target soon.

27.11.2024

I implemented a field of view change that is based on the players speed. And I started to make a particle effect that is also supposed to indicate the players speed

28.11.2024

I implemented a tilt of the player mesh for the wallrun and made the character jump when the player jumps off the wall. Both are just visual feedback for the player but it already does a lot to make the wallrun feel better.

A lot of work has gone into the speed lines particle effect and I managed to find a workaround for the orientation. I had to write a custom Niagara node for it. But at the end I don’t think it was worth it. It does not look good at all. I probably need to look into alternatives. I have 2 ideas in mind. A UI based speed line animation and a particle effect that just shows passing by clouds. I think the second one would work better because it does indicate the velocity of the player quite well, even when they aren’t just going forward, without being too intrusive. There might also not be a need for it to be dynamic and react to the players speed. But that’s for tomorrow. Good night.