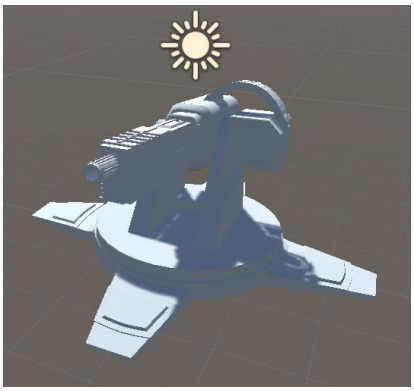
Midterm project Writeup

So far in Unity I learned mostly about rendering optimization. As I already used the software multiple times, I skipped the basics and went straight to more advanced notions. In this domain, I learned about Level of Detail (LOD) optimization. LOD lets you simplify the 3D model you're displaying as the camera goes farther away from it. You can have multiple levels of LOD on your model which progressively gets less detailed. This method makes rendering faster because the engine doesn't have to render every detail (less triangles) when you can't even see them. Since our game will have many objects rendered at once, it was only natural to optimize their rendering.

I also learned about Unity project management. I already did management on some projects in the past but never on a game, so that was new for me. I learned that the Unity Git plugins and cloud sharing functionalities were counterproductive, and that one should use standard git instead.

When my group and I discussed about how the game should work, I proposed several ideas. First, I suggested a PvP game where each player's goal was to destroy the other players' base, with liquid as a primary weapon. Because it was too much work, we decided to do a single player game instead with hostile AI instead of humans. My classmate had a really good game idea, so we went with that. We tweaked it a bit with other propositions from members of my group. I came up with multiple ideas for difficulty progressions and what the goal of the game should be. These ideas have since been slightly changed but they basically remain the same. For example, I came up with the principle of "map conquering" which essentially means the player will have to build units all over the map to win. I also recommended that we have mobile units capable of scouting and gathering materials.

For assets, I found this Turret model online, and I used it as a base for my animations and LOD implementation.



It was kind of hard to find because most free models online are either too detailed or too futuristic. This model fits our art direction (military design) very well while not being too detailed. The only issue is it didn't come with an assigned texture, so I will either have to use materials or shaders (or both) to make it look good in-game. I used this model to do all my different tests so far.

As far as implementation goes, I worked on the animations for the turret. This was a big part of my work because I wanted it to be as realistic as possible. After many tests I came up with this code.

```
// Update is called once per frame
void Update()
{
    float angle = Mathf.Sin(Time.time) * 40;
    Vector3 temp = transform.position;

    temp.y = Mathf.Sin(Time.time * speed) * amount;
    transform.rotation = Quaternion.AngleAxis(angle, Vector3.up);
    transform.position = temp;

}
```

This, assigned to the right parts of the model, will make it look like the turret is rotating sideways back and forth, all while vibrating a lot. The usage of the Sin function really helps to get a nice rotating curve that slows down at the edges. As mentioned before, I worked on the LOD implementation. I can't give myself too much credit here because Unity does most of the work, but creating the appropriate models is a fairly tedious task. Another minor thing I did was create part of the behavior for our game camera. I needed to be able to zoom out to test if my LOD was working, so I implemented it. It is worth to note that the zoom functionality doesn't use any FOV tricks and moves with position variations, otherwise LOD wouldn't work.

```
// Update is called once per frame
void Update()
{
    float scrollAmount = Input.GetAxis("Mouse ScrollWheel");

    transform.Translate(0, scrollAmount * scrollSpeed, scrollAmount * scrollSpeed, Space.World);
}
```

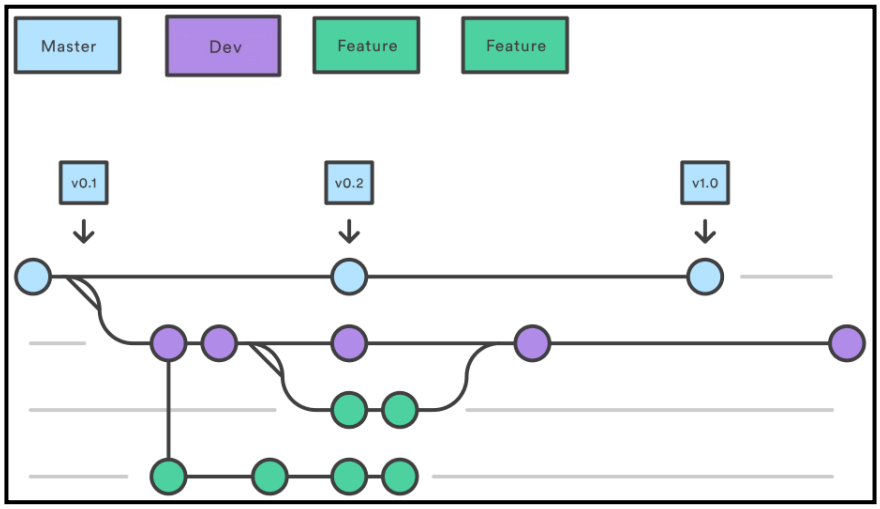
I also did most of the project management so far. This ranges from setting up the repository to assigning tasks to the other group members. First, I did a bit of research on what was the best way to work remotely. As I said earlier, Unity's version control manager is not suitable for such a big project, and the Unity Git plugin was obsolete. So, I simply chose to work with standard Git. I chose the most suitable Git workflow for a video game and set it up on the repository.

Next, I plan to work on a lot of things. First, I will automate the mesh simplification process. Until now, I did it manually on Blender, but I recently discovered that you can create Blender scripts to do it automatically. This script will take an object as an input and will produce an LOD group for this object with multiple levels. I also plan to work on the general UI of the game. Since our game will be mostly resource management and architecture planning, it is of the utmost importance to have a user-friendly UI/UX. I will also try to implement mipmapping in our textures, which will optimize rendering even more.

Documents and assets

LOD tutorial I followed: https://www.youtube.com/watch?v=ifNyVS2_6f8

Git workflow:



Turret with and without LOD optimization: