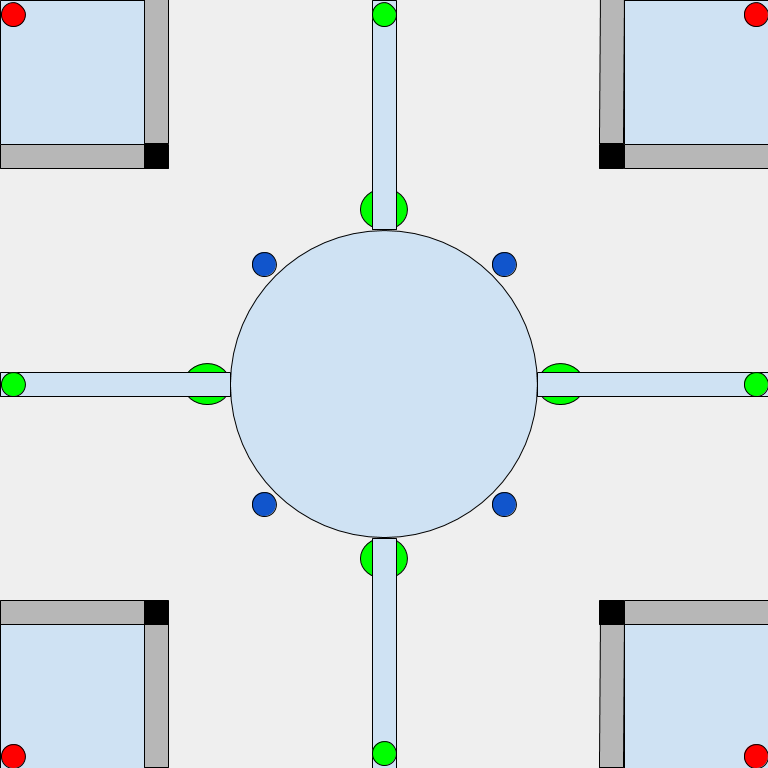
Game and Software Design

* Map and Key
  + Blue- raised height, standardized\*
    - \*The blockers between corners START at this raised height and end well above them
  + Dark gray- ramp connecting standard height and raised height
  + Black- wall from floor to ceiling
  + Red circles- player starting spawns, as well as a potential respawn location
  + Green circles- potential player respawn locations after death
  + Dark blue circles- represent the gravity lifts that raise a player to the upper level

The Development and Playtesting Process

***Setbacks***

This project was a doozy. The late decision to implement AI over turning this into a networking multiplayer game cost me a lot of time, and worse, the deletion of all my scripts set me back a lot more. This game’s vision is heavily compromised as a result.

Given that Cursed Halo is based upon Halo 2’s mechanics, I went through a lengthy process to calculate the Unity equivalent to Halo 2’s numbers. Jump height, player size, gravity, movement speed, and weapon behavior was studied and converted as best I could. Because the weapons are supposed to be weird and “cursed,” I used their existing converted numbers and used modifiers to “crank up” the effects. As a result, the Burst Rifle fires twice as fast as its Battle Rifle equivalent, the Micro Rocket fires 33% faster between shots than its Rocket Launcher counterpart, and the micro rockets themselves travel twice as fast. The explosions for the rockets and player terminal velocity I could not find a way to easily ascertain and decided to spend the time on development, using guesstimates of what I wanted in-game instead.

Ignoring the complete reversal of networking implementation, the first cut I made was the motion tracker. It was a lot of extra time and resources for a feature that wasn’t too helpful for this map. The second cut was the longsword; the combination of weapon behavior complexity and newfound implementation of AI made it a fetid limb. Due to a general lack of time weapon pickup and ammo mechanics were scrapped, as well as the melee attack. Given the strength of the burst rifle I doubt the melee was going to see much use anyway!

***Playtesting***

Due to a lack of interested parties (i.e. family not being available for being testers when I needed them), I had to be the sole playtester. The first thing I noticed with the AI was its tendency to get stuck between the slopes for the corners, so I built invisible walls into the slopes. I couldn’t just get the AI to recognize the slopes, as if they were not ignored for the pathing detector then the AI would refuse to go up any slope.

Another detail I noticed later once the AI could shoot: they could only shoot straight, and the burst rifle was *way too good at hitting the player*. I figured giving the Burst Rifle for AI a larger default spread would give it inaccuracy, but this proved to make it way too useless. So, I sharply reduced the minimum default spread, and instead had the AI prefer to only use it as a close range weapon. This made AI that used the Burst Rifle far less oppressive and more manageable opponents. As for the AI shooting straight, it turned out that copy/pasting the player’s version of the weapon code caused various oddities, resulting in a two-day process fixing the AI such that it would properly shoot out of its fake camera head rather than using its chest as the firing point.

Alas, once I did fix the AI Burst Rifle and implemented the AI Micro-Rocket, I encountered a new problem: the rockets would self-impact against the AI and spin around all crazy-like when they fired, and sometimes they would bounce off the AI’s body and float into space! It took me a full day to figure out why: PlayerController is also a collider, and for some reason only known to the Unity gods disabling both the PlayerController and the capsule collider still had the rockets self-impact the AI. Eventually I figured out that removing the AI’s capsule collider fixed the issue and the rockets would appropriately ignore its creator. What’s especially strange is that this wasn’t an issue at all for the player version of Micro Rockets.

There were far more oddities and bugs and “Unity spaghetti” I had to untangle, but that is the gist. Other highlights include the AI floating off into space forever once they entered a Gravity Lift (rigidbody was a bad idea), the AI aggressively tracking the player even after death (the Burst Rifle was firing forever!), and my personal favorite: backpedaling and jumping would inexplicably cause the player to get hit by their own Burst Rifle bullets and micro-rockets (no clue how to fix the former!)

***Features***

Despite the things I had to cut, I did add some polish. Every player death is announced in the announcement feed on the top left, which will automatically clear itself depending on how old the announcement is. If you’re trying to figure out who to put in place, that announcer could be a big help. The group score is printed in the top-right, which is actively updated as players get kills or die. I set active, then deactivate, a camera on the player’s camera position whenever they die so they can look around and take things in. The gravity lifts are my personal favorite feature. They reverse gravity at double potency, but have a lower cap on upwards vertical speed. The simple implementation created a surprisingly deep interaction through the physics system. Because the AI would just bounce off of the gravity lifts, I set a box that, when triggered appropriately, would make the AI beeline up the gravity lift. That way, you aren’t safe with your height advantage.

***AI***

The AI is a modified version of the basic AI from Program 2. The AI will always track where every single player is, and will target the closest player if they are within the AI’s “field of view” in front of it and within detection range. If the AI enters the area around the gravity lifts (marked with an invisible collider), they will prioritize going up the lift rather than their normal pathing. However, they won’t do so if they are aggressing on a player, if they used a lift recently, or if they are in the air.