Programming Assignment 1: VR Deserts, Gazebos, and Minecraft

Aidan Brem, Ruby Tseng, Joost W. Vonk

VR Minecraft

Aidan's project, "VR Minecraft", provides a simple VR experience in a minecraft like world, featuring a small "chunk" of terrain a few blocks deep, a tree, a house and a model of the Minecraft character "Steve" (with accompanying 2D label, Minecraft username style). The house contains a small torch that shows off shadow rendering, and a simple day night cycle provides lighting and darkness for the scene. The scene contains several textures taken from Minecraft to provide an authentic experience, and a functional cubemap skybox to make the scene feel much more natural.

A feature that existed in the 160 version of the project that does not exist in the VR version is the ability to break and place blocks - the techniques used for this, raycasting and bounding boxes, does not work in VR, and will need an alternate implementation to be fully functional, likely with some integration with the Meta Quest VR wands. Additionally, the size of the world had to be reduced significantly for performance concerns, as in this project every block is its own object, which was acceptable in a standard rendering environment, but prohibitively expensive in a VR one. There are a few ways around this (namely chunking, or the practice of splitting all visible terrain into "chunks' ' and only rendering the blocks that are visible) that Aidan has gotten working in a simpler project, and may explore. implementation in VR at some point in the future.

Night with the Mystery Cat

Ruby's project, titled "Night with the Mystery Cat," immerses players in a tranquil, secluded park setting during the nighttime. Away from the bustling city, players find solace and tranquility as they navigate through the park. Despite the peaceful ambiance, the park is not devoid of life. Positioned in the center of the park is a gazebo, housing a mysterious cat surrounded by a mesmerizing, circling ball of magic. This magical ball serves as a rotating light source, illuminating the dark surroundings.

To add to the atmospheric allure, the park is adorned with streetlights casting a gentle glow, alleviating any eerie undertones. The project revolves around two main OBJ objects: the gazebo and the street lamp, both seamlessly integrated using OBJLoader. The enigmatic cat is crafted using various three.js shapes, with its animated tail gracefully moving up and down. The captivating ball of magic employs the Lens flare technique from the Lensflare library.

Texture plays a pivotal role, with most objects being textured squares. Shadows are strategically incorporated to amplify the interplay between light and shadow, contributing to the overall mood of the project. Additionally, distant objects are veiled in fog to create a subtle blur, although this effect is not discernible in the VR version. While no code removal was necessary for the VR adaptation, a crucial adjustment involved converting the camera into a camera group to ensure it didn't end up beneath the floor plane in the VR experience.

Sail the Sands Part A

In the first phase of this project Joost's experience, "Sail the Sands", had the user follow a pirate ship as it sailed through a desert. It featured two procedurally generated pyramids composed of cubes, each with a rotating polyhedral gemstone suspended above the tip. Each of these gemstones served as the source of a colorful spotlight that followed the ship as it sailed. These spotlights combined with both ambient lighting and a directional light that matched the perceived trajectory of the sun on the textured skybox created a nuanced lighting environment, which was further enhanced by real-time shadows cast onto the terrain by the ship and pyramids. This terrain was composed of a plane augmented by a normal map, as well as a displacement map. The ship itself was loaded from a set of .obj and .mtl files.

Prior to VR integration, the project featured two distinct camera views—one located at a fixed position high above the scene which the user could orient using orbital controls, and one attached to the ship itself. With the transition to VR compatibility came the need to pare down the number of cameras to just one, so it was decided that the camera attached to the ship would be more fitting for a VR experience. Along with this change came an enhancement to the rendering of the shadows cast by the ship; whereas before there were a set of invisible blocks which moved with the ship that were used to generate the shadows, now that the user could inspect them much more closely it became worthwhile to generate the shadows based on the actual ship's geometry.

Sail the Sands Part B

In the second phase of this assignment, we chose to expand upon Will's "Sail the Sands" experience. This is because we felt as a team that this project benefitted the most from being converted to VR due to the fact that the environment depicted in this project was somewhat fantastical in nature. Ultimately, we decided that the best way to add VR interactivity was to allow the user to explore/interact with the world in the form of steering the ship. We accomplished this by incorporating a mechanism that enables players to manipulate the ship's wheel in virtual reality (VR). To accomplish this, we divided our project into distinct components.

Part 1: Scaling Up the Environment

In the previous phase, it was clear the environment was not designed for VR experience due to the fact that the ship, pyramids, dunes, etc were far too small. Therefore, it was necessary to scale the world up to allow for a more immersive experience. This came with certain challenges, the most significant of which being the increased performance cost. We ended up having to drastically reduce the user's view distance in addition to removing shadows.

Part 2: World Border

With the decision to give the player the ability to traverse the environment at their own discretion came the need to restrict their movement to be within the bounds of the environment.

Part 3: Importing controller

To actually steer the wheel in the VR world, we need to import controllers and assign commands to the controller.

Part 4: Making the Wheel

The pirate ship model we used for this project came with a steering wheel already attached, but we decided that for our purposes it would be easier to add our own steering wheel on top of this and configure it to be interactive rather than detach/identify the steering wheel from the model and configure that.

The process of procedurally generating the ship wheel itself was somewhat simple due to the symmetrical nature of the object. The difficulty here came from needing to position it in the scene just right in order to cover the model's steering wheel.

Part 5: Rotating the ship

The primary function involved in navigating the ship is to adjust its orientation by rotating it. Our program follows a sequence to achieve this. Initially, it retrieves the ship object's current rotation. Subsequently, it applies a vector to propel the ship forward in the direction it's facing. This vector is then scaled according to the ship's speed. Initially, we attempted to use preset x, y, and z speeds for the ship's movement. However, we encountered a challenge where rotating the ship would result in backward movement at certain angles. Consequently, we pivoted towards a new approach that involves calculating the ship's angle and vector dynamically to ensure forward movement post-rotation.

Integration

The three projects are consolidated through a common landing page. The landing page provides players with the opportunity to explore three distinct worlds at their own discretion and pace. Players have the freedom to choose and navigate through these diverse environments, each offering a unique experience tailored to their preferences. However, it's worth noting that there was no effort invested in creating cohesion among the three projects, as their themes diverge significantly. Each scene maintains its distinct and diverse theme, without a concerted attempt to unify them under a cohesive narrative or visual motif.