Blenderman Report

Shyheme Days, Sean Gai, Seho Young

May 14, 2019

Introduction

The purpose of our final project is to integrate what we have learned over the year in class into a fun game. The game is modeled off of a popular indie first-person survival horror game called *Slender: The Eight Pages*. The premise of the game is to collect pages of a journal before a scary figure called Slender Man catches you or time runs out. We thought this game would be a good way to learn more about rendering a procedurally-generated scene, adding interactive gameplay elements, and creating interesting visual effects like a flashlight.

In previous assignments, we created a detailed scene of geometric primitives and interesting materials using raytracing, but as suggested and noted by our TA Reilly, this method would be too slow and graphics intensive for interactive gameplay. So instead, we are using the standard rendering framework provided by `three.js` and `WebGL`.

Methodology

The implementation of our game is based on an example 3D platformer demo from the `three.js` website that helped us add the basic physics and gameplay controls to our game.

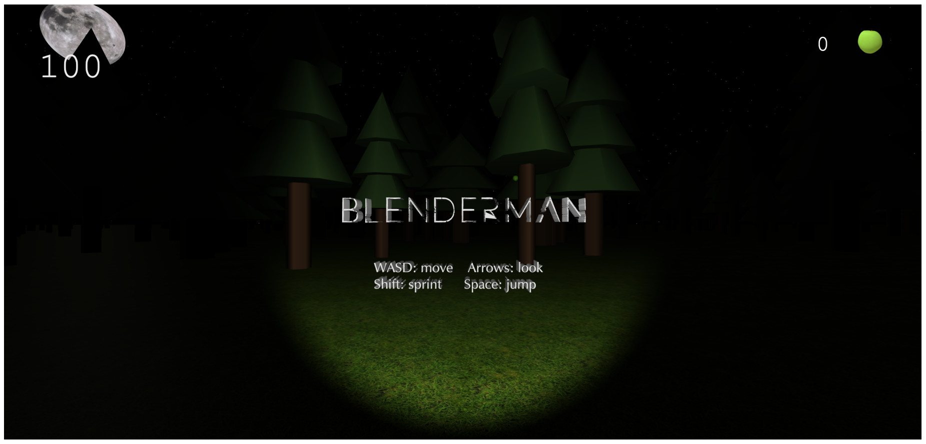
To modify this example and create our own scene to make the game look and feel like a forest at night, we implemented our own functions to add unique elements, textures, and lighting into the scene. Our basic scene is comprised of a ground plane with a grass texture, randomly placed and scaled tree meshes, randomly placed green apple models (gltf files), a cube map loaded with a starry night sky texture, a moon that is added as a sphere mesh in the sky with a lunar texture, and a villain that randomly moves around the map. For environmental lighting, we added ambient light using `THREE.AmbientLight` and a `THREE.DirectionalLight` placed at the position of the moon. For the character to see in the dim night lighting, we added a flashlight which is a `THREE.SpotLight` that moves with the camera position. Lastly, we added a title and controls text explaining how to play the game using `THREE.TextGeometry` that loads at the initial camera target and fades over time.

To add to the immersion, we added walking and running noises to simulate the sounds of moving through grass using `THREE.Audio`, `THREE.AudioListener`, and `THREE.AudioLoader`. And to give the game an objective, we added gameplay functions to check if the character (camera position) is within a certain distance of an apple (which plays the role of journal pages in the original *Slender Man* game), in which case the game plays a beep as audio feedback to notify the player that they have collected an apple and it increments the apple count in the upper right hand corner. And to create an element of challenge, we also created a health tracker in the upper left hand corner which informs the player how many more ”run-ins” with the villain they have left before their health runs out and the game is over. So the score is based on how many apples the character collects before they run out of health.

Some implementation challenges we ran into included using Javascript's Audio objects, which caused loading errors, so instead we switched to `THREE.Audio`. Other challenges we faced involved parameter tuning to make the game look how we wanted it to. For example, it took awhile to find the right values for the various `THREE.SpotLight` properties like `angle, penumbra, distance, decay, etc.` to limit the player's view to create enough of a challenge while also letting the player see far enough to avoid making the game frustrating.

We also had to find a medium between computational complexity and generating a more realistic and visually appealing scene. One approach to generate the terrain was to use a noise function to get a height map and then to use this map create a terrain with different elevations. We could also potentially generate larger terrains and allow the player to roam more freely. Generating a terrain with different elevations increases complexity of checking player-ground interactions and as we increase the size of the map it becomes easier to survive.

Results



Shown above is a screen capture of the title screen for our game. We think we did a good job creating an immersive experience for the player with the audio, choice of textures and meshes, and creation of the background scene. Given more time, we would make the gameplay more challenging and varied. Nonetheless, we surpassed our baseline deliverables of creating a procedurally-generated landscape and adding a first-person view camera that can move around the scene. We completed one of our reach goals by adding a scoring/reward system (apple collection) and a way to lose (a villain that follows the player and causes the player to lose health when in a close enough vicinity).

Discussion

The approach we took struck a balance between ambitiousness/complexity and simplicity. The framework and tools we ended up choosing were easy to work with and gave us enough flexibility to create our own unique world and gameplay.

Other approaches we could have used to create our scene and may have optimized the rendering framerate include using 3D model loaders from `three.js` like `THREE.GLTFLoader` and `THREE.OBJLoader` to add preloaded, more complex models that we could have randomly scaled and rotated without having to create each tree from simple geometries.

This leads us into possible improvements for our game. We would have liked to increase the variety of meshes and add other objects like rocks, flowers, and buildings to make the world more interesting. We could also allow players to adjust the settings of the game such as difficulty and to include more animations. We could also add eerie horror-style music that quickens and sharpens as the villain approaches to increase the suspense and tension.

Conclusion

Overall, this project was a valuable learning experience that taught us how to render a scene and make it interactive by adding controls and moving the camera. The assignments tended to feel static, so it was refreshing to work on something that feels more physical. We gained some insight into how games are created and the choices that developers have to make related to optimization.