CS324 Computer Graphics Report

u2000231, Henry Ha

January 23, 2023

1 Instructions

To run the game, first unzip the zip folder. Then simply run a Python HTTP server from the command line, within the directory of the source folders. If your browser does not recognise the Javascript files, use the included `test.py` by using the command `py test.py`. When this is done, you should then access the Solutions folder, which should redirect you to `index.html`. Please make sure you use the correct port number when accessing the local host. You will then be greeted by the game menu.

2 Blender Model

For my blender model, I decided to make the scene for my first level. The idea is to create a room and a simple environment around it. I first began with the window walls by creating the basic window frame shape. Using the Inset Faces tool, I easily achieved this effect and was able to duplicate this wall to place my two remaining window walls.

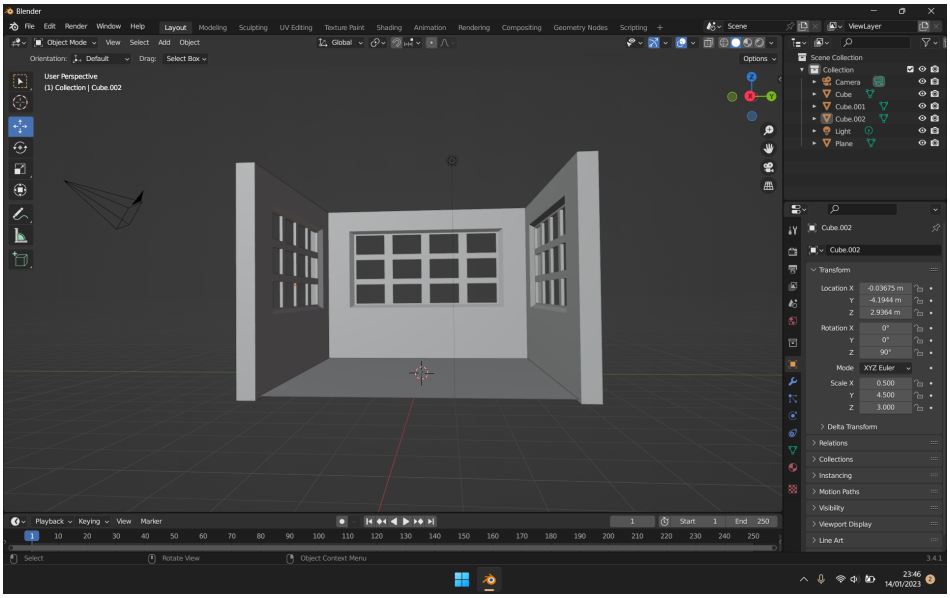

Figure 1: WBS for our system

I then created the door frame, using the same Inset Faces tool to create the obtrusion of the door outline. As you can see, deleting the faces from the mesh caused holes. This was filled using the fill hotkey, as shown in lectures.

Afterwards, I created a cabinet next to the door. As I didn't have enough edges to work with on the mesh, I used the Loop Cut tool. As I created the outline for a possible cabinet, I then extruded the edge regions. This creates a concave structure, making up the rough shape of a cabinet.

Proceeding, I then added Image Textures to each mesh to give it a sense of realism. My initial colouring with basic colour materials did not fit the style I was going for, so textures were an improved alternative.

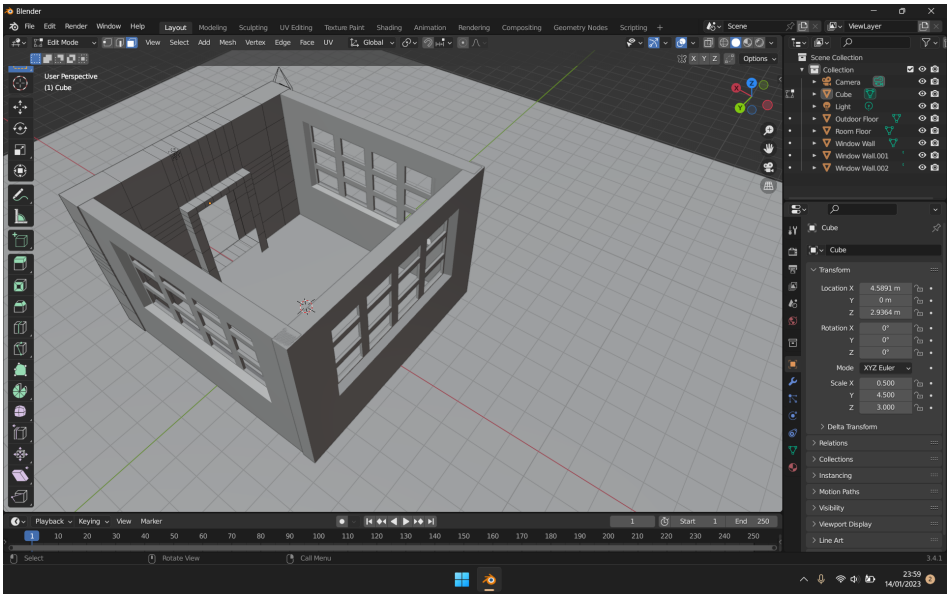

Figure 2: WBS for our system

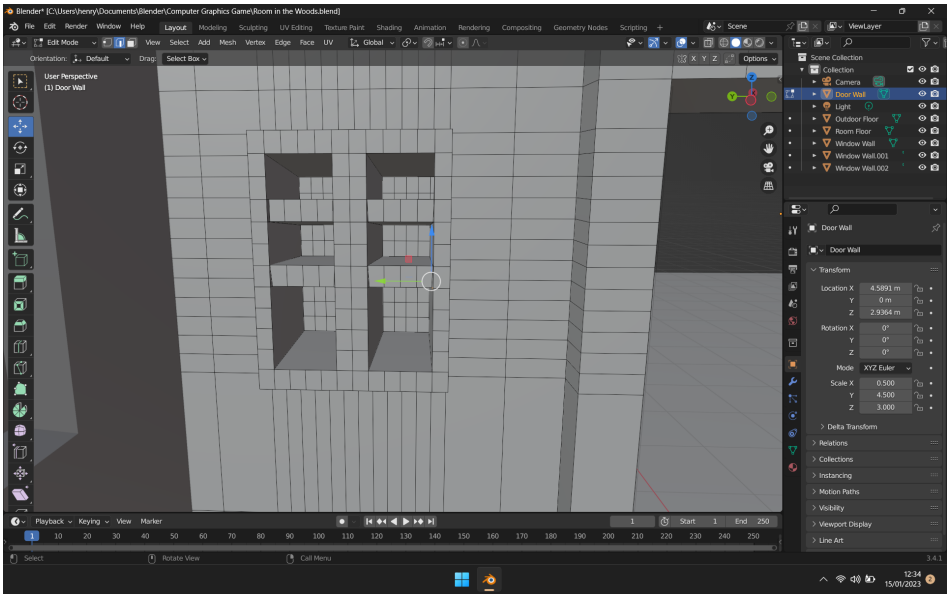

Figure 3: WBS for our system

To finish, I populated the scene with lighting. This served as a blueprint for where I want to put my lights when working in Three js. This is in addition to some of the models used to populate the empty space of the level. Lastly, I also modelled the cabinet doors and main door.

3 Implementation

The game's development consists of various component implementations. As such, this section will be divided by the discussion of each of them.

3.1 Flashlight

To create the flashlight effect, a new SpotLight was created. This SpotLight was added to the camera as a child object, alongside its target component. The target was then offset in the negative Z direction,

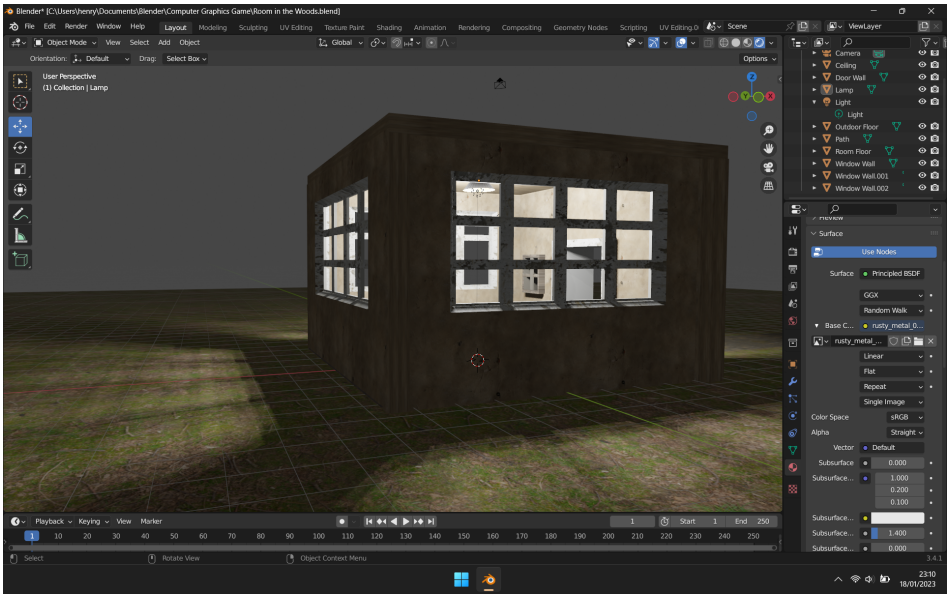

Figure 4: WBS for our system

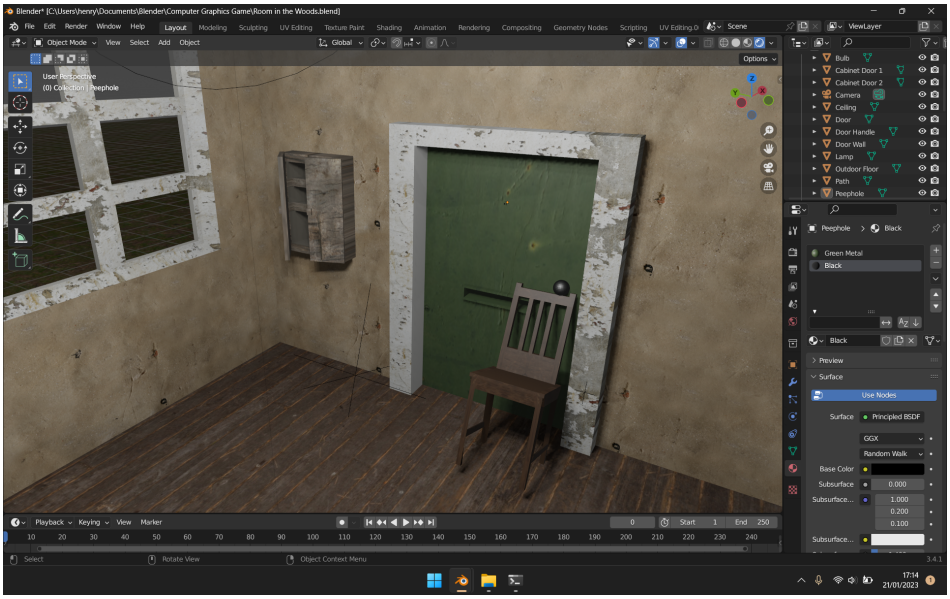

Figure 5: WBS for our system

causing the spotlight to beam in the direction of the camera.

The flashlights have two modes of usage: the default wide-angle beam, used to view and navigate the environment, and the focused beam, used to aim at eyeballs in order to destroy them. This was implemented simply using an event listener and changing the properties of the SpotLight. A boolean variable `UVLight` is used to track what is being used. Moreover, a raycast is also used to detect what is at the center of the camera and flashlight.

3.2 Eyeballs

Since the eyeballs all follow the same behaviour and model, they were grouped into the `eyeGroup`. This was used alongside raycasting to determine when the player was looking directly at the eyeballs. Using an `if` statement, the eyballs health was able to be reduced when being looked at. When the health of an eye reaches 0 or below, the mesh accessed by the raycast is turned invisible and a score

is incremented. This score determines the win condition of the game – destroy five eyeballs and the game is won.

3.3 Watcher Entity

The watcher event was used to disturb the player in-between searching for and destroying eyeballs. A timer is used to determine when the event occurs, which is approximately every 20 seconds. This timer counts down using a delta variable, which tracks the seconds between each concurrent frame, allowing me to create a timer pace and movement pace consistent despite variations in FPS.

When the event occurs, another timer counts down, denoting how long the watcher stays. If the player is not under a table or bed, another separate counter counts down denoting when the player will get caught if they stay in sight. Moreover, the lighting of the television changes to red and the entity model teleports to a random point defined by me. This is usually a window (or letterbox) and is quite creepy as he also tracks where you are moving using the `lookAt` function. These are done to show the player they are in danger. When the timer is up, the original timer resets and various variables are reset.

3.4 Fail and Win Conditions

The player wins when the 5 eyeballs are destroyed, giving the player 5 score points. Once this is done, player controls are unlocked and the win screen is shown. The fail condition occurs when the player has been in sight for roughly 5 seconds during a single watcher event. When this happens, the player controls are unlocked and the camera is rotated to view the watcher. The fail screen is then displayed. Player controls are unlocked for both to prevent further play after a fail or win condition is met.