T1 Report

Team name: Go kalbi
Team member: Min-Seong Lee(2018311500),
Yi-Eun An(2017314510),
Ye-Lim So(2018310587)

1. Game Senarios

The game we made is a kind of survival game. The map consists of three layers of grid shaped like a beehive made of hexagons.
The player will move around the map created above. At this point, the hexagon that the player stepped on disappears after a certain period of time.(In the figure above, only the player above the missing hexagon is the player, and the rest are npc.) If the player falls into a hexagon, it falls down, but npc doesn't fall.

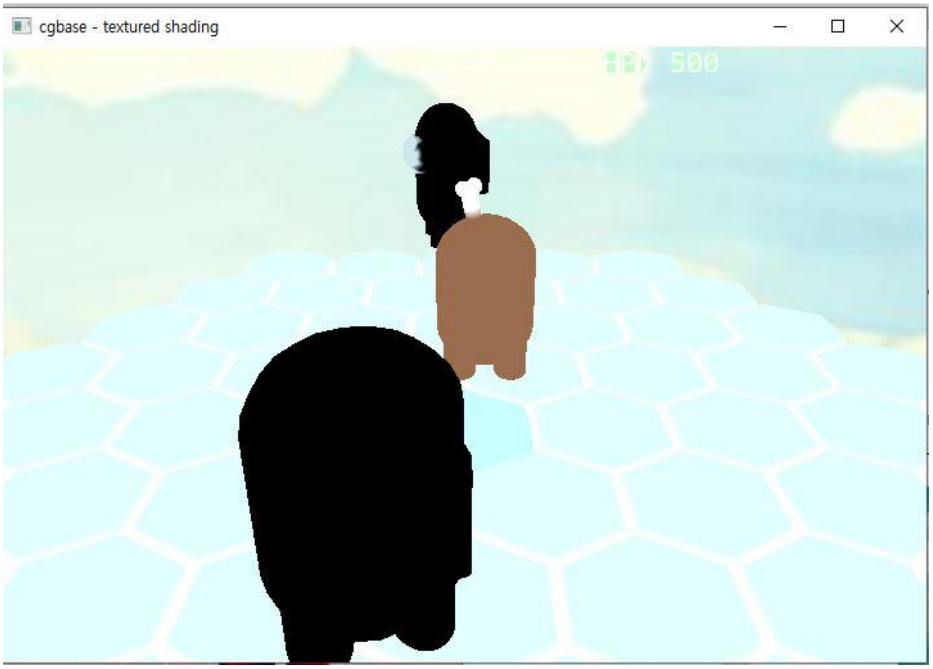

Figure 1. Game scene

Players moving around on the map that would be disappearing will be given missions to kill all npc on each floor. The only condition for a player to win is to complete this mission. You can throw ribs to npc by pressing the left mouse click and kill npc.

Else, If the player cannot kill all npc and falls to the ground, or if HP reaches zero, the player loses. If npc meets player, HP of player will decrease.

Figure 2 Game scene

2. Game Design

All the designs that appear in the game are handmade by us. First, the background, skybox. We designed to change the skybox background depending on the difficulty level. If the difficulty level is 1, 2, we edited the following photo and applied it to Skybox(Figure 2).

For the highest level of difficulty 3, we used skybox with a scary atmosphere to emphasize the difficulty.(Figure 3)

Figure 3. Skybox (lv.1-2)

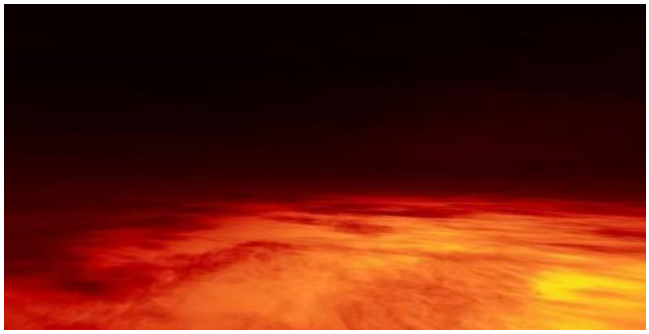

Figure 4. Skybox (lv.3)

Second is a 3D model for player and npc. We homaged the characters from other games to embody the 3D model. The figure below (Figure 5) is the reference we made.

Players and NPC characters were created by referring to the characters of the

Game 'Among us', and the weapon is shaped like a rib(갈비). Using this as a reference, we created the following 3D model. This one is for player and npc.

This one is the weapon player using in the game, shaped like a rib.

Figure 5. Character Reference

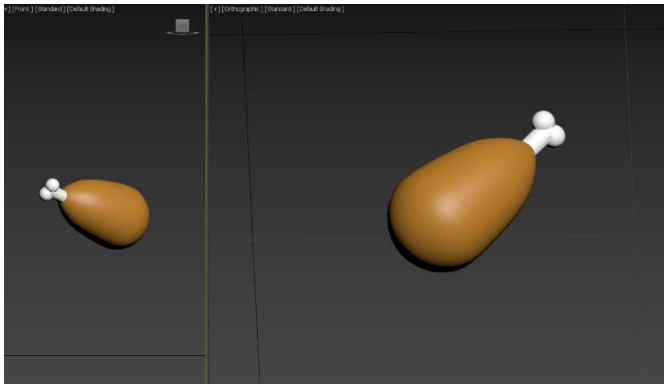

Figure 6. Rib model

Third, we made 3D text for the title in the game.

Figure 7. Title

For dynamic production, the characters were also made in 3d. Just like the above character models, we calculated the vertexes and put the 3D model on it.

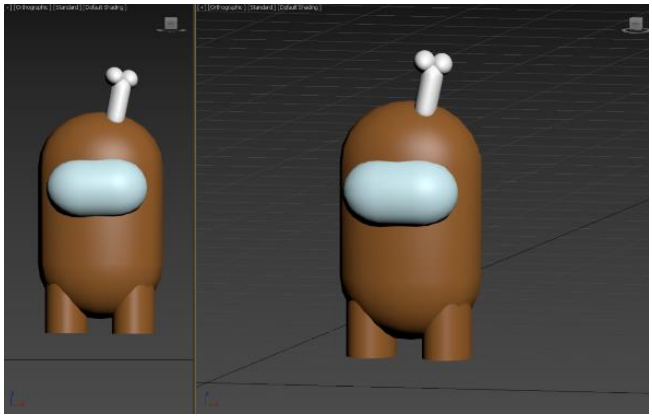

Figure 8. Player model

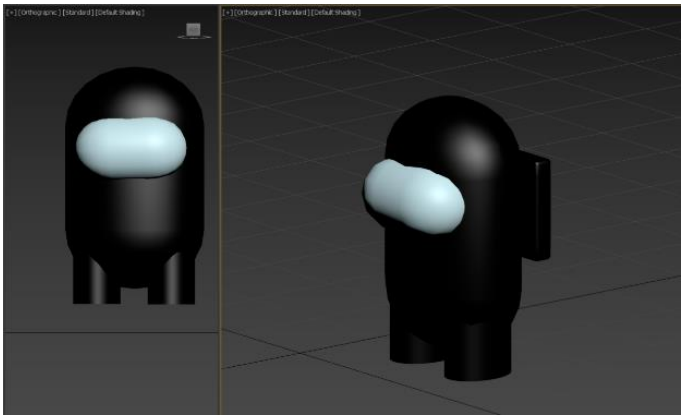

Figure 9. NPC model

3. Fidelity of Implementations

3-1. Title Screen

We made a "title screen" by attaching a brief description below along with the 3d characters mentioned above.

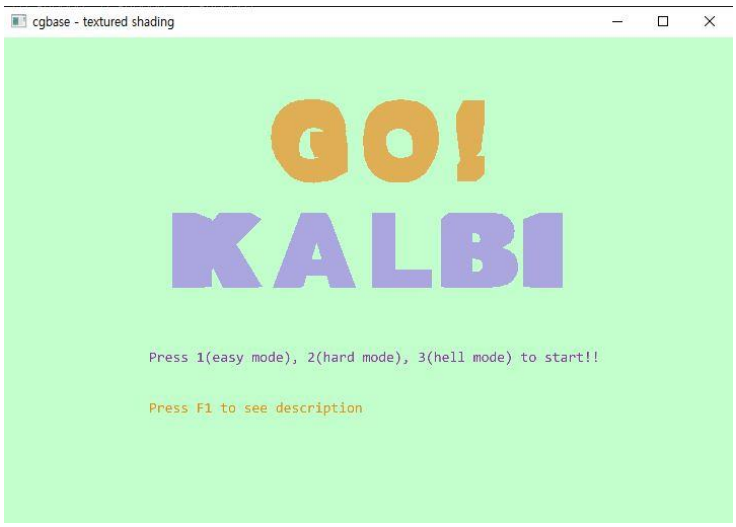

Figure 10. Title Screen

3-1. Graphical Help

Pressing f1 on the title screen brings up a screen with a brief description of the game. I tried to paste the photo into the hexagon above with textures, but it was a little difficult to understand because the photo was crushed.

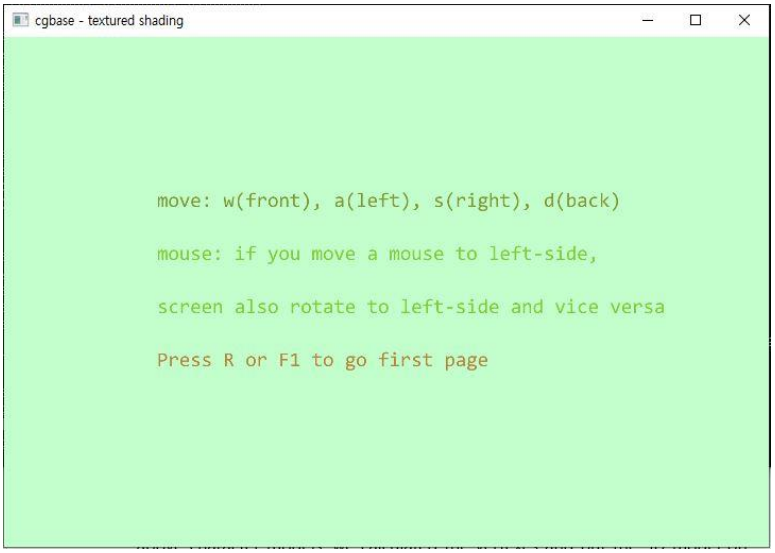

Figure 11. Description Screen

3-2. In-Game Reset

If you press R during or after the game, the screen switches to the title screen so that you can proceed with the new game again.

3-3. Resizable

You can continue to play games at the same rate as if you increase the screen.

3-4. Text Rendering

We wrote some description at the title screen, the game description part and when the game was over using text render. You can also see the player's HP in the upper right corner when playing the game.

3-5. Sound Rendering

A total of four backgrounds were used. First of all, two backgrounds were used

for the game play. We used background music separately depending on the level of difficulty, divided into levels 1 & 2 and 3. And the rest is sound effects. I set the sound effects for both jumping and throwing rips.

3-6. Dynamic 3D Camera Movement

When you start a game, the camera moves from where the game title is located to where the player is called. You can also use your mouse to toggle between left and right when playing games.

4. Extra points

4-1. Our own Hand-drawn Images

We used the 3d model that we drew in the player, rib, and title. The content is the same as previously shown

4-2. Our own physics simulation

Various physical aspects were used. First of all, in order to achieve a more natural crash when the player crashes, the acceleration crash was used using the concept of gravity acceleration.

To make NPCs wander inside the map, we used collision detection. The distance between boundary and NPC's center limit was calculated as shown in the Figure

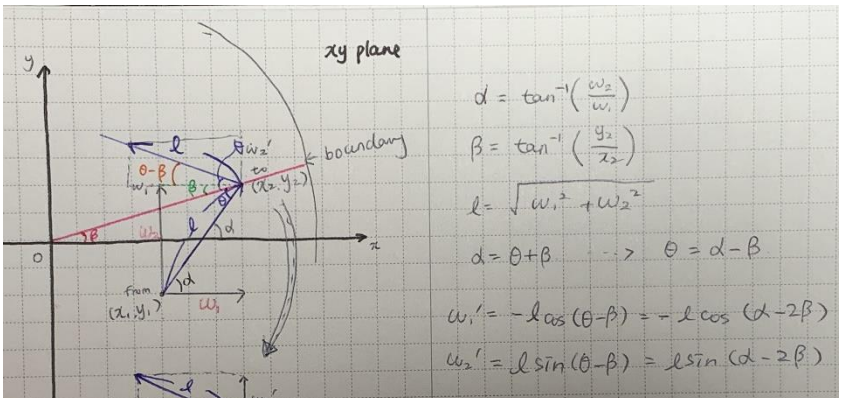

Figure 12. NPC boundary collision

below. Also, using distance between their centers, collision between NPCs was handled.

We also made rib(the weapon) to move in projectile motion. When throwing object with the angle of θ, Velocity after t seconds is calculated as

$$v_x = v_0 \cos\theta,\ v_y = v_0 \sin\theta - gt$$

And location after t seconds is calculated as

$$x = v_0 \cos\theta * t, \quad v_y = v_0 \sin\theta * t - \frac{1}{2} g t^2$$

where g refers to gravity.

Geometric aspects were also applied. In order to calculate whether the player is on top of a hexagon first, we calculated whether the player is on top of the hexagon by internally using the normal vector and the position vector of the player for each stool of the hexagon. These are all the parts that I used by applying the content that I dealt with in the previous class or calculating by ourselves.

4-3. Moving 3D NPC's with AI

We built a dock based on physics so that npc can move on its own. We implement it so that npc does not collide with each other. However, when the npc and the player came into contact with each other, we didn't have much equipment available to attach the player's HP.