Name: Jing Wang
ID: 21330849

Project 1 Final Report

Module: CS7GV6
January 10, 2022

1 Introduction

In Project1, I chose to make a 3D game referring to "Gaunlet". "Gaunlet" is a 2D dungeon crawl arcade game. The player controls a fantasy-based character to explore a dungeon. In the adventure, the players will encounter many monsters. Every character has his own skills. For instance, the player can shoot some fireballs to kill monsters. Moreover, there are many treasures scattering throughout the dungeon. The more treasure chests are collected, the more final score the player gets.

Considering the complexity of the game system of "Gaunlet", only one dungeon will be designed for this project. The goal of this game is also changed to collecting a key to open the gate and escape from the dungeon. After clearing this game, the collected treasures will be converted to the final score. According to the concept, some main components are considered as below:

1. Character. The player will control a mage girl in this game. Figure 1 is a screenshot shows the appearance of this girl. During combat, the mage girl can shoot ice balls to fight against monsters. Figure 2 shows the appearance of the ice ball.

2. Monster. The monsters are cursed diamond people. Figure 3 shows the monster. They are always trying to catch the mage girl. Once the monster catches the mage girl, the mage girl will take damage and this monster disappears. They can be killed if attacked by the mage girl's ice ball. However, the cabins scattering in the dungeon will spawn new monsters every several seconds. Figure 4 shows the cabin.

3. Key and gate. Figure 5 and Figure 6 are the key and the gate respectively. There is only one key in the dungeon. The mage girl needs to collect the key to open the gate and escape from the dungeon.

4. Gem. Figure 7 shows the gem. There are five gems in the dungeon. The more gems collected, the higher mark you get. Meanwhile, trying to collect more gems means more dangers in such a dungeon.

Figure 1: Mage girl.

Figure 2: Ice ball.

Figure 3: Monster.

Figure 4: Cabin.

Figure 5: Key.

Figure 6: Gate.

Figure 7: Gem.

2 Game Play

Control character. Press W/S/A/D key to move forward/backward/left/right in third person view, while to move up/down/left/right in top down view. Figure 8 shows the movement. Press space key to shoot a ice ball. Figure 9 shows the attack.

Figure 8: Movement (move left, move right, move forward, move back).

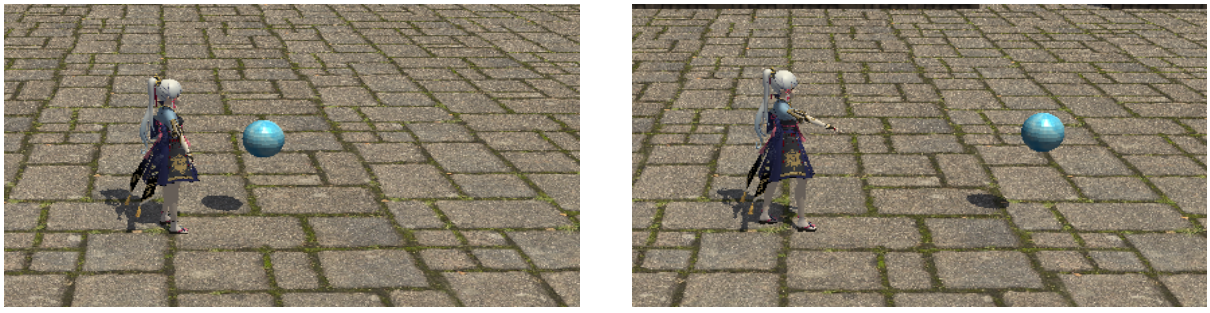

Figure 9: Shoot.

Combat with Monster. An ice ball can kill a monster. Figure 10 shows the the attack from the mage girl kills a monster. The hearts on the upper-left is the number of lives. If a monster catches the mage girl, she will loss one life. After the mage girl lost all the lives, the game is over. Figure 11 shows the process.

Figure 10: Kill monster.

Figure 11: Attacked by a monster.

Collect items. If the mage girl touched a gem or key, she collected the item. The number of gems and key will be displayed on the lower-left of the screen. Figure 12 shows the process.

Figure 12: Collect item.

Open the gate. If the mage girl does not have a key, the gate cannot be opened. After she found the key, she can open the gate and escape from the dungeon. After she get rid of the dungeon, the final mark will be shown. Every gem collected will be converted to a star. The max mark is 5

stars. Figure 13 shows the process.

Figure 13: Open the gate (left image: without key cannot open the gate, centre 2 images: open the gate, right image: congratulation scene).

3 Technology

3.1 Compulsory OpenGL elements

3.1.1 Interactive manipulation

Player can control the mage girl to move and attack, as shown in Figure 8 and Figure 9. When press W/S/A/D key, the mage girl will move. When press "space" key, the mega girl will shoot a ice piton. Moreover, when the mage girl is near to a gem/key, she can collect the item (see Figure 12).

3.1.2 complex model

This game used several complex models. For example, the mage girl (Figure 1), the cabins (Figure 4), the key (Figure 5) and so on.

3.1.3 Hierarchical structure undergoing transforms

The monsters have hierarchical structure. A monster has 5 parts: body, left arm, right arm, left leg and right leg. They will wave their arms and legs while moving to attack the mage girl. This action is using hierarchical structure undergoing transforms. Figure 14 shows their action.

3.1.4 Shading and light

The Phong model is used for shading. Regarding the diffuse reflection, it calculates the dot product of the normal of the fragment and the light direction as the intensity of the colour for the corresponding fragment. As for the specular reflection, it calculates the dot product of reflected ray

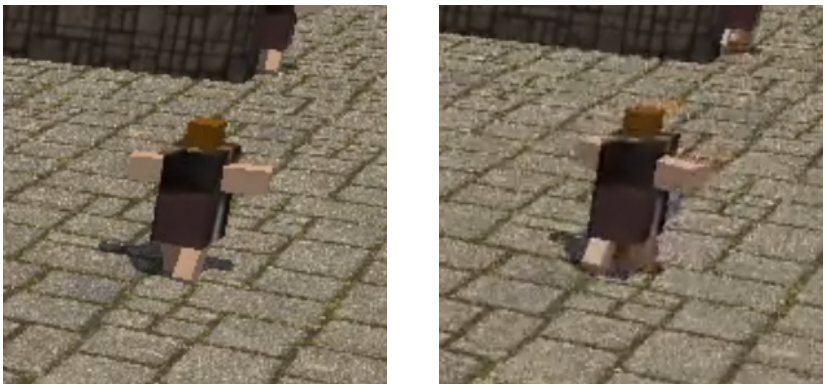

Figure 14: Monster action.

and the vector from the fragment position to the camera position as the intensity of the specular reflection we can receive. There is only one directional light in this game. Figure 15 shows the shading results. The fragments facing to the incident light are brighter than those backing to the incident light. In addition, some ambinent colour is added for the mage girl, so the fragments back to the light are not black. As for the specular reflection, if we look at the highlight on the girl's hair, you can see the position of the highlight changes following the camera viewpoint.

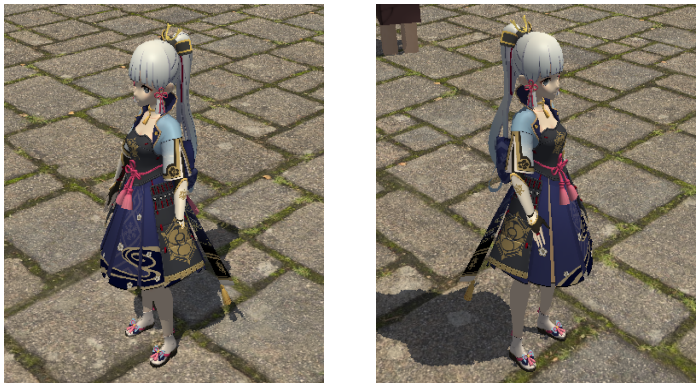

Figure 15: Shading results.

3.1.5 Camera view

Two camera views are implemented. One is a top down view, which is simply look at the player from the top. Another one is a third person view. Figure 16 shows the two camera views.

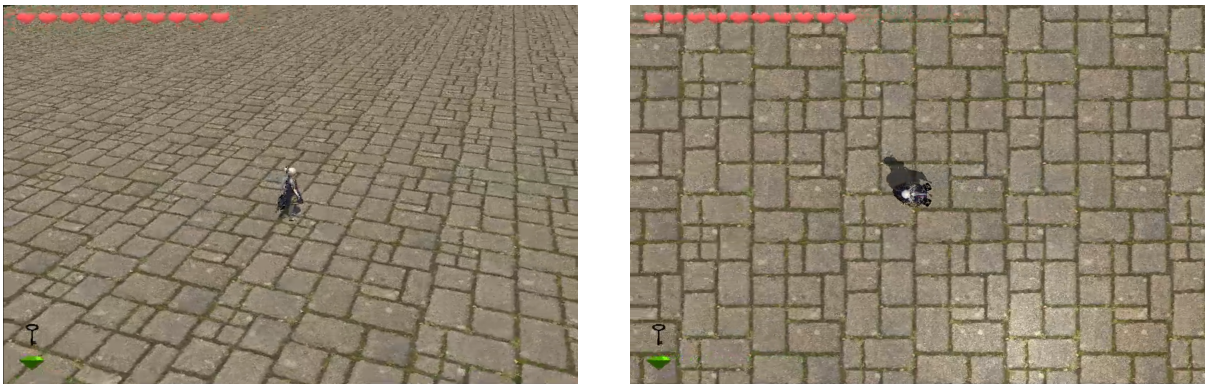

Figure 16: Two camera views (left image: third-person view, right image: top-down view).

3.2 Advanced OpenGL elements

3.2.1 Skybox

The function of rendering skybox is implemented in this game. A texture buffer for cube maps is created and 6 images corresponding to the six surface of a cube are loaded into the texture buffer. After that, a unit cube using the texture will be rendered as a skybox. A cube map of dusk scenery is chosen for the sky box. Figure 17 shows the skybox.

Figure 17: Skybox.

3.2.2 Collision detection

2 types of collision detection are implemented for this game. One is using the distance between two objects. Once the distance between the two objects is smaller than a specified value, the collision happens. The collision between the monsters and the mage girl is using this type. Another one is using bounding boxes for collision detection. Once the bounding boxes of the two objects are overlapping, the collision happens. The collision between the walls and the mage girl is using this type.

3.2.3 Shadow rendering

A shadow map is prepared for rendering shadows. First, all the objects in this game will be rendered once from the viewpoint of the light position. In this rendering, the depth information will be recorded as a depth map. Second, all the objects will be rendered again from the viewpoint of the camera position. This time, the depth of every fragment from the light position will be calculated. If the depth is larger than the corresponding depth recorded in the depth map, the fragment is in shadow. When rendering a fragment in shadow, only ambinent colour will be used. Figure 18 shows the rendered shadows.

Figure 18: Rendered shadow.

3.2.4 Skeletal animation

In order to make the action of the mage girl more smooth, skeletal animation was added. The mage girl model has 213 bones. When calculating the influence of the bones, every mesh of the model can be affected by as most 4 bones. The mage girl model is saved as a fbx file. The file include bone positions of key frames for every animation. When playing the animation, the frame between key frames will be linearly interpolated. In this game, totally 3 skeletal animations were designed for the mage girl. The first one is idling (Figure 19). The second one is moving (Figure 20). The last one is attacking (Figure 21).

Figure 19: Skeletal animation: idling.

Figure 20: Skeletal animation: moving.

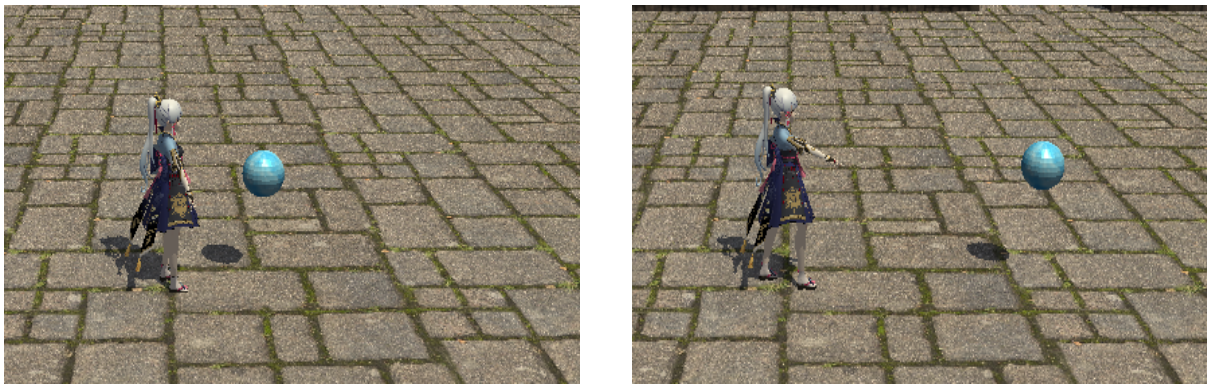

Figure 21: Skeletal animation: shooting.