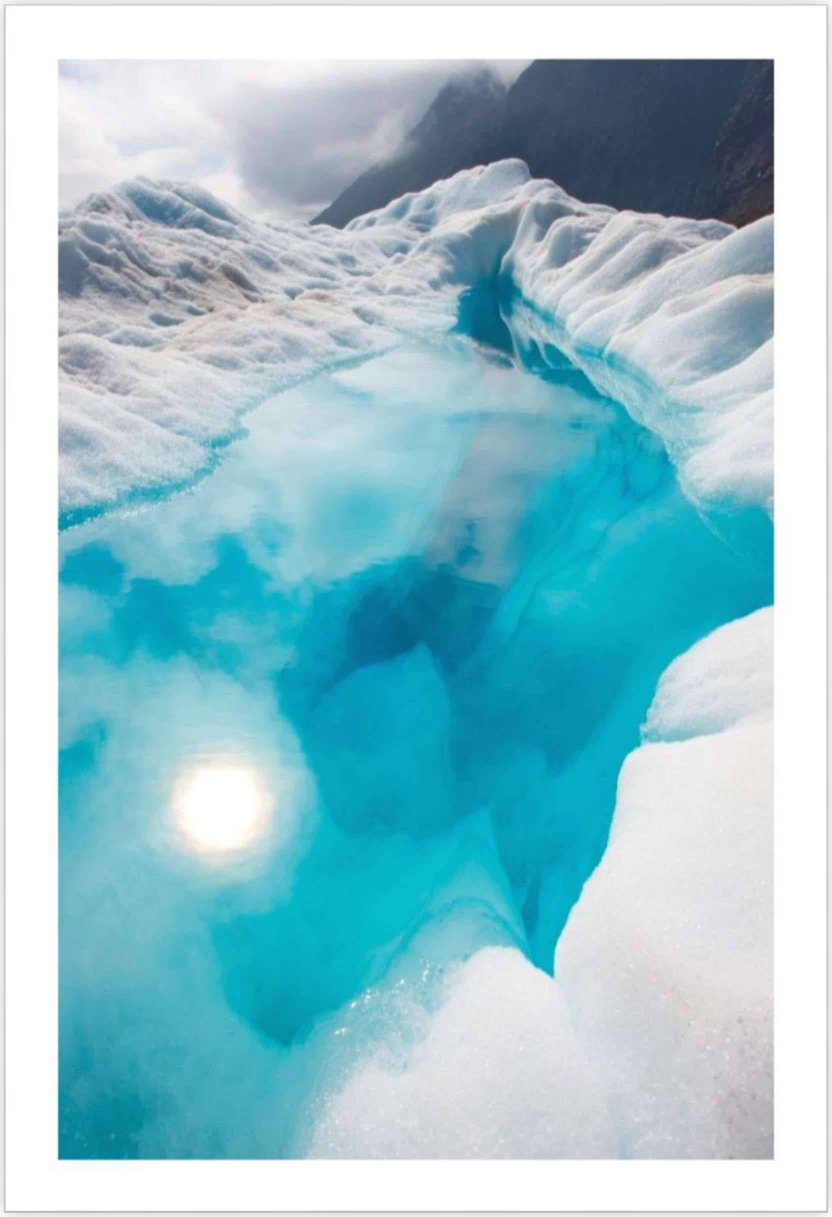


Space shooting project description

VR, GRAPHICS, ANIMATION - 2016

hxu001 | STE6249 | [Date]

Introduction

This is a development of a simple computer game, space shooting, which covers all elements of OpenGL. This project is made by using C++ language in Visual Studio 2015. It is implemented by 13 classes. In this game, it has three different types of enemies with different shapes and shader and moving in in different ways: normal way, sine and cosine. The plane can shoot in two different types of bullets. I hardcode the 3D model directly in the source code instead of using 3D model object. Following lib have been used from openGL:

- <GL/glew.h>
- <GL/glut.h>
- <GL/freeglut.h>
- <GL/gl.h>
- SOIL.h

Programming description

It has 14 classes. Following are the brief description of which I implemented:

1. BattleField
   It is a class for making the valley and the route for enemies and plane moving. It use muti-texture so that the battlefield is 3-dimensional. I draw a landscape based on the geometry from the battlefield. Use multi-texturing to render two textures. One of the textures should be stretched across the landscape. The second texture should be stretched over a quad and repeated across the landscape. This texture will represent the details in the landscape. And then use a fragment shader for texturing. A height field is a texture that is used to store height values, which is used to generate a terrain. Use one of the textures to generate a landscape. Send the texture to a vertex shader that uses the pixel value to specify the height of the landscape. Use a color map for texturing of the landscape. Use vertex buffer object and the geometry from the battlefield implementation.

2. BitmapText
   A bitmapis a rectangular array of the values 0 and 1, which are used as a mask for drawing to a rectangular portion of the window.

3. Bullet
   Made two different types of bullets. One is using glutSolidSphere and the other one is using GL_QUADS.

4. Camera
   It is a class given by teacher. I modified some line to fulfill my willing. For example

5. Enemy
   In this class, it has three different enemies moving in different ways. Each of the enemy has different shader.

6. GameManager
   It is the engine of the game. It controls all the function here. Collision detection is also made here.

7. ParticleGenerator
   Make the particle. Put it behinds the plane as a jet.

8. Shader
   This is the class for loading shaders. Textures can be accessed in both vertex and fragment shaders.

9. Skybox
   Create a unit-sized cube and texture the cube with a cube map. Position the camera in the middle of the cube each frame. Disable depth test.

10. SpaceObjects
    It is a class for organize all the variables for different Objects such as Space ship, Bullet, Enemy etc.

11. SpaceShip
    It is a class for the plane. I hardcode the geometry of the plane.

12. Water
    I create a tessellated water surface that can be used for rendering of a water simulation. It is 128*512 here. Using shaders I made a dynamic water surface. Use environment mapping on the water surface so that it has refection. I use following wave equation:
    v.y = sin(waveWidth * v.x + waveTime) * cos(waveWidth * v.y + waveTime) * waveHeight

13. Main
    It is a class given by teacher. I modified some line to fulfill my willing. For example I changed the keypress space to send bullet instead of moving the camera down.

Game function

In this game, most functions work by keyboard. For moving the camera:

W-> watching forward;

S-> watching backward;

C-> watching up;

X-> watching down;

A-> watching left;

D-> watching right.

For moving the plane, we use the arrow keys.

↑ : moving up;

↓ : moving down;

← : moving left;

→ : moving right;

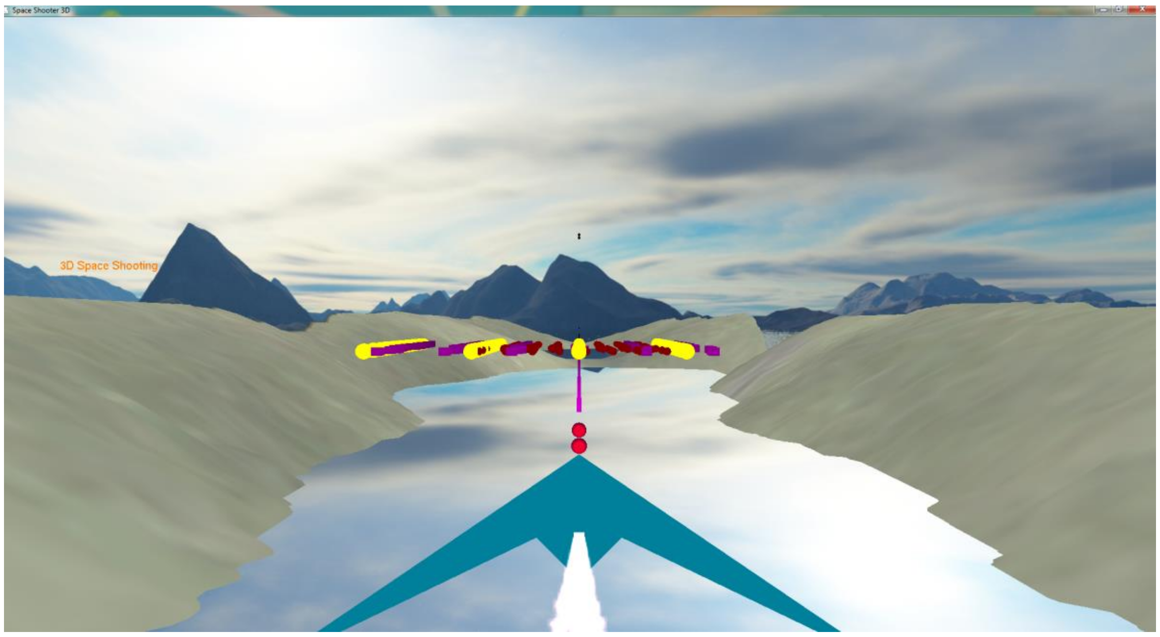
Space key for sending bullets.

Future work

If I have time, I am going to load 3D model such as trees, grass on the battlefield. I also need to make the bar for the plane so that it will show how long it will die. I also can make a life bar for all the enemies. I also need to implement the text so that it will show the life number of the plane, the level of the game and so on. It will be good if I make the particle to show the explosion. Implement shadow mapping in the GameManager. In this project, some of the code is made there, but it does not working as it should be since I do not have enough time to fix it.

Project result overview

Here is the screenshot of the result.


Space Shooter 3D
3D Space Shooting