Project6: Firework

Chunxu Xu

Contents

1 Required Tasks

The goal of this project is to create a firework scene. It's basically a particle system with simple using of texture, playing sounds etc.

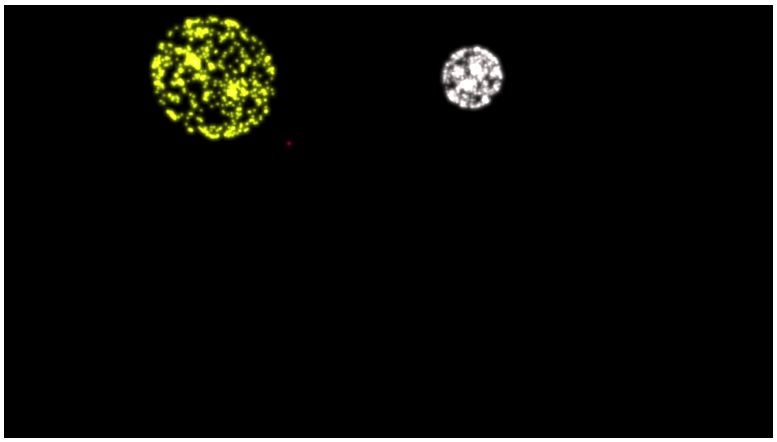

Figure 1: Overview of a firework scene

2 Detail Description

This part will talk about the implementation in detail.

2.1 Particle System

Particle system is usually used to simulate a large number of small objects which are probably blurry or dim, like in fire flame, smoke, water wave, cloud and so on. Like physical system, every "particle" in particle system has its own location and speed in most cases. However, since particle system focuses on the appearance of particles more than the motion, what's more important here is not physical principle, but the visual effect of the whole system.

Lesson 19 of NeHe tutorial is a good example of particle system, and you're allowed to do your own work by starting from this lesson's code. However, since there are still a lot of differences, the following part will describe a framework of a simple, independent particle system.

2.1.1 Texture

Using texture will achieve a better blurring effect than pure color. We can use a simple image with "star" shaped white region as a single particle. Since for most particle systems, the overlap of particles are very common (in our case for example, different firework particles can overlap each other

before and after explosion when projected on the 2D screen), you need to choose the blending function carefully, otherwise weird effects may occur. For example, here we use a "star" shaped picture for each particle, but the picture is not transparent at the black region, thus a bad effect shown by Figure 2 may occur due to an inappropriate choice of blending function.

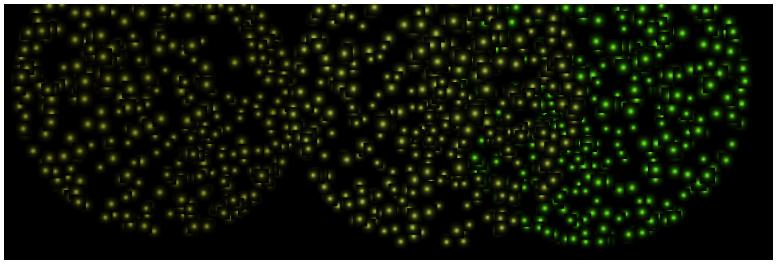

Figure 2: Bad effect, the opaque background mask other particles.

2.1.2 Fading effect

There is no standard way to achieve a fading effect in a firework scene. One way to do that is to use a *life* field to indicate the time that each particle's time to exist. It can be set to a positive floating point value, and with a random fading speed varies in a certain range. In that case, each particle will disappear one after another which looks better than the case where all particles fade out together.

2.1.3 Movement

To achieve a real-life like firework, we still need a simple physical engine system to simulate the motion of particles. The basic working flow of the physical system is to calculate the shifting of particles between two adjacent frames and refresh, which is a simple discretization of the continuous time. To achieve a smooth effect, the most important thing here is to count the elapsed time between two adjacent frames and calculate the shifting of particles in this duration. Thus to make the program give the (almost) same effect (like the existing time of the firework is roughly the same) on different machines, you need to measure the time elapsed during the interval of two frames.

In detail, each particle can be assigned with a tiny mass, There are only two external forces we need to take into account here: the gravity and the explosion force. For gravity, there is no point using the real gravity acceleration and an adjusted parameter may do a better job. For the explosion force, you can apply a sudden force at a certain time (for example, when the particle slows down to zero speed at vertical direction), but a much easier way to do this is to change the speed directly (such as adding a random

speed towards an arbitrary direction if you want to create a normal sphere effect).

2.1.4 Explosion

There are different styles of explosion shapes you can create. such as sphere, ellipsoid, heart-shaped, smile-faces and so on. In general, different initialization of speed when explosion happens can be an easy way to achieve the effects, and the only thing you need to care is how to assign an additional speed to different particles. This may involves a lot of mathematical calculation.

2.2 Sound

A bomb sound can be sent out when the explosion occurs. In Windows, it is easy to play a .wav file by calling PlaySound() function (a Windows API). To do this, you need to add *windows.h* header file. Meanwhile, *winmm.lib* is also needed when linking the program. You can check out MSDN to see the usage of PlaySound() function. And there are similar APIs on other platforms to help you play sounds.

3 Bonus

Any extra feature can be considered for extra credits. The difficulty and effects will determine how many extra grades you'll get. Please describe the features you implement in the final report in detail.

Bellow are some suggested options:

- You can choose to create different kinds of explosion effect.
- The explosion position can be randomly decided by the initial speed. On the other hand, you can design to specify a position of explosion manually. Note that it may be difficult to do it with the picking functionality provided by OpenGL. You can choose to do an accurate picking yourself. It is more complex if you're using perspective projection.