Project Report **Hanxiao Shen**
Graduate Computer Graphics, CSCI-GA 2270-001 Prof. Daniele Panozzo
Date: 12/18/16

How to build

The program is built on MacOS 10.12.2, and using CMAKE to organize the compilation. To build the executable file, execute the following instructions under the main directory of this repository.

```
mkdir build
cd build
cmake ..
make
```

Problem Background

The project is about grass simulation. The ultimate goal is to create a meadow and the blade of grass can swing in the wind. To modeling the scene, there are two central components, land and grass. In the next sections, I would like to present what is the idea and how to implement it.

Grass Modeling

The grass can be modeled in many different ways, one possibility is to subdivide it into multiple small segments, and formulate a layer system that supports the realtime updating. But here the layer system is not easy to construct. Yet a similar way is to handle the blades individually, which is they method I used in this project. I am using 2 curves to modeling the blade. To make the action of blades more natural, it is better to use splines whose control points are on the curve itself. And there is one family of spline, Catmull-Rom[3] spline, that has this property. For each blade, I am using 5 control points to generate 2 splines, and sampled along the curve, connected the sample points of same heights. Then the position and shape of the blade is directly depending on the position and action of the control points.

Hooke's law[2]

In reality, as the blade of grass moves/bend, it has the tendency to recover the displacement. This can be viewed as the same property of a spring. And there is one famous physical law named 'Hooke's Law'. It is defined as

$$F = -kx$$

where x is the displacement of the spring, k is a constant depending only on the spring's material or shape. And the displacement I am using, is the vector pointing from the initial position to the current position of each control point. Thus we can get the spring force for every control points at any time.

Movement of control points

Then we combine the force of wind and the force of spring, divided by the mass of each control points, we will get the acceleration.

$$F_c = F_w + F_s$$
$$a = \frac{F_c}{m}$$

Suppose at that time, the velocity of control point is v, then we can sample after a small period of time, e.g. 2ms, and assume the new position is exactly the next position of control points (although in this period the acceleration changes too, so this assumption is not precise). Then at the new position, first we need to check the deformation of segments, because we have to make sure the length of the blade does not change as it moves. And then we just shrink or elongate the length of each segment. But if the position of any control points is changed at this step, it makes the speed calculated before partially invalid, because only the speed in orthogonal direction is needed. This can be easily fixed by calculating the projection of the speed in the orthogonal direction.

Wind

The wind model here is very simple. I only consider the wind in x and y direction, and the wind is set to be constant (but adjustable).

Land

I am using a cube model to represent the land, and then I used the SOIL library to do the texture mapping (a dirt surface to simulate the ground).

Figure 1: Final result

The final result is shown in Figure 1, and note that the parameter of the grass, wind and ground are all adjustable.

Future work

I hope to improve the UI first, to support more interaction for and provide more information about the scene, such as the wind, the number of blades. Also, I did not do any collision detection, thus some movement may looks not realistic because blades may pass through each other. And shadow is not supported either in the demo. All this factors may make the scene more realistic.

References

[1] Making Grass and Fur Move Sven Banisch and Charles A. Wu, Journal of WSCG, ISSN 1213-6972, Vol.14, 2006

[2] Hooke's law from Wikipedia

[3] Introduction to Catmull-Rom splines

[4] Simple OpenGL Image Library

[5] GPU Gems, Chapter 7. Rendering Countless Blades of Waving Grass.

[6] Lecture Notes 07:Designing Interpolating Curves; Graduate Computer Graphics, CSCI-GA 2270-001 Fall 2016