Introduction to Computer Graphics
Project Part 3 - Interaction & Animation

Group 23 - Florian Junker, Louis Séguy, Mathieu Monney

May 28, 2014

1 Issue

We added some improvements in the last few days but unfortunately there are some performance issues we don't have time to solve. Sorry for the low framerate :(

2 Overview

During this part we developed new ways to change the view in our world. We implemented two additional modes: fly through and camera path. Unlike the trackball that moves the world by changing the view matrix, these two modes move the camera.
The fly through mode let us move everywhere in the world using the q, e, w, s, a, d keys. Keys q and e are used to change the z-axis camera view direction while a and d allow us to change the xy-axis. Finally by pressing w or s, we can move forward or backward.
The camera path automatically changes the camera position along a Bezier curve. And thus in this mode it isn't possible to explore all the world.
You can see a preview of our final render in Figure 1.

3 Implementation

3.1 Fly through mode

We dropped the provided trackball to implement a fly through mode instead. We had to write a camera class that move according to the keys we press. It also eased the Bezier curve implementation as it was then easy to set the viewpoint and direction, it also helped in the same way to implement water reflection. Unfortunately we didn't have time to implement a proper FPS mode.

3.2 Computing and rendering a Bezier curve

We began with a naive implementation of a bezier curve. To approximate our Bezier curve we used LINE_STRIP with a parameter that allows us to change the number of lines that describe the curve. Then we can render the curve.

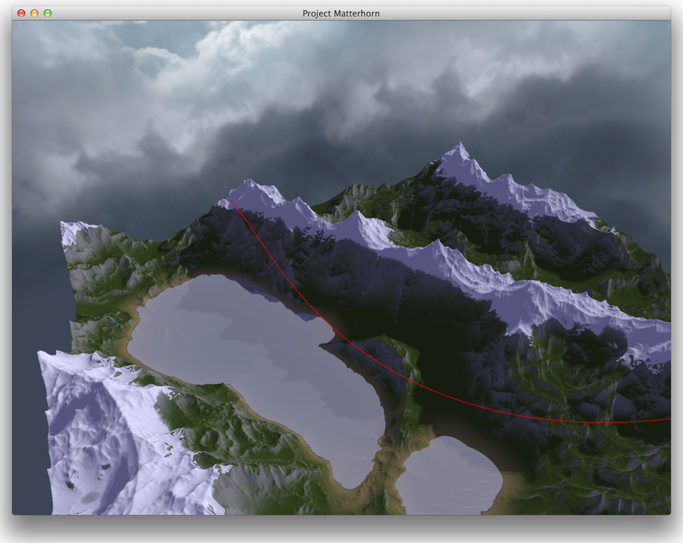

Figure 1: Out final render

3.3 Animating the camera

We simply had to change the camera position according to our Bezier curve. The hard part was to move linearly on the curve. Since the curve is approximated by N lines, we store the lengths of line segments to find the position in fonction of a parameter between 0 et 1 and thus obtaining a lineare movement.

4 Results

4.1 Fly through mode

We didn't find any difference in performance. It might be slightly faster as we don't have to compute the matrices that rotate and translate the model but just change the viewpoint and look direction.

4.2 Computing and rendering a Bezier curve

We first descide the curve we wanted to. And then try render it by changing its parameters. This was straightforward. Render a bezier curve with the navie algoritm may cost a lot due to cubic terms. De Casteljau improvement fix it.

4.3 Animating the camera

There are no special parameter to tweak for this implementation. Animating the camera doesn't add a big computational complexity.

5 Improvements

5.1 Physically realistic movements

We added inertia to the camera movements.

5.2 Water reflections

We implemented water reflection and refraction. To do this, we move the camera and render to a texture to obtain the reflexion. We then map it onto the water plan. For the refraction part, we simply add some transparency to see the underwater terrain for the sake of simplicity. Finally, we can blend the reflection and refraction parts according to the view angle. If we're viewing the water from above, there is no reflection and if we're looking at it from a grazing angle there is full reflection.

5.3 De Casteljau

The principle of De Casteljau is to find the middle point and split the curve in two. And then iteratively continue with the two new curves. And thus find enough to approximate it. It has the adventage to be simple to compute.

5.4 Pictorial representation

We draw a pictorial representation of the camera. Draw the eight lines (+ one triangle) of the pictogram was pretty easy. Be able to change its possible was also trivial. However we add some issues when we tried to change its orientation. This actually doesn't completly works.

5.5 Animation of view direction

We simply had to add a second bezier curve to animate the view direction of the camera. This is not complicated, simply some code to add. Our main problem was to use it with the pictorial camera because when had (and still have) some problem to rotate it.