Bardet Mike,
Petrescu Diana,
Bouvet Quentin

Introduction to computer graphics
Project features

Flat terrain

We went slightly further than a standard fractal brownian motion noise, and implemented a way to flatten the terrain in order to have plains, not only mountains and lakes. We did so by using another perlin noise (with values in [0, 1], which we query at each vertex and multiply the height value of this vertex with, pushing it toward the 0-altitude point of our choice.

Infinite terrain

We kept the camera still and simulate movement around the terrain by creating our Fbm noise with an offest matching this displacement, and query the textures according to this offset too.

Level of details

We increase the number of vertices of the terrain geometry the closer it gets to the camera. We should have used a tesellation shader for better performances, but didn’t know about it so we made our own implementation

Random noise for waves

we used a dudv map texture to represent the waves. During our second pass, we query the refraction and reflection textures with a small offset that we get from the map.

Fresnel algorithm for water

We implemented a fresnel effect to balance reflection and refraction depending on the angle at which we look at the water

Grass / Snow transition

We implemented a smooth transition between grass and snow. For that we used another noise than the one used for the height-map. This noise is used to determine some location where the terrain will converge to a height close to the surface.

Fog

The second new feature is fog. It's a mix between the terrain and the sky, the amount of terrain and sky is determined by an exponential law, where we used gradient, density, and distance, as parameters

Birds

We added 5 birds, a few untextured vertices, that fly together, with their wings following a sin function.

Camera inertia

The camera has intertia when it rotates and when it moves back and forward.

Notes

We did some optionnal “features” early on, that we ended up not maintining throughout our developement cycle. I refer specifically to the parameter “explore mode” in our gui, which when unchecked, shows the world in a trackball, but which is buggy.

We did not remove these “features”, because they don’t influence how the project runs as long as you don’t activate them, but we don’t intend to present them as features, nor be graded on these.

External code

We re-used directly or took inspiration from :

- The ImGui C++ library, to provide a gui that tunes the parameters of our terrain
- The algorithm for the Perlin noise and the Fractal Brownian motion
- The Fresnel algorithm
- Various snippets of code from stackOverflow, especially, the pseudorandom mathematical function used in rand() in screenquad fshader

Tasks repartition

During this project, Diana focused more on optionnal features, Mike did a lot of optionnal features too and some compulsory features, and Quentin took care of compulsory features, and some compulsory features. The repartition would be as follows : Mike 45%, Diana 30%, Quentin 25%.