Reflections of Paradise

Student: Borzan Călina-Annemary

Group: 30424

Discipline: Graphic Processing

Contents

1. Theme Presentation

The aim of this project is to introduce the student to the world of OpenGL (Open Graphics Library) and the universe of Graphic Processing. OpenGL is a programable interface used for rendering 2D and 3D vector graphics. The API is mostly used on the part of hardware, typically used for interacting with the GPU. The GPU is a specialized a specialized electronic circuit initially designed for digital image processing and to accelerate computer graphics. [1]

Another interesting and helpful tool for creating this program was Blender which is a free and open-source 3D creation suite. It supports the entirety of the 3D pipeline—modeling, rigging, animation, simulation, rendering, compositing and motion tracking, even video editing and game creation. [2]

With the help of the presented tools, I succeeded in implementing a graphical and realistic scene, with the help of Blender I had the possibility to create my own scene, to add the objects I considered fit and to give each of them a specific position. Then, by rendering this scene in Visual Studio with the help of the OpenGL library I could add effects: rain, fog, multiple types of lights, animations, certain modifications of objects, giving my scene a more real perspective.

„Reflection of Paradise" is exactly what it says, a representation of a paradisiac space, a place of quietness, where the only thing around you is the nature and the ocean. There we can see a boat, the only way to reach this abandoned place, a path of torches that lead you to the most peaceful place, chairs, glasses and so on.

2. Scenario

2.1 Scene Description

My scene illustrates an island in the middle of nowhere, if you look around everything you can see is the ocean and the sky full of clouds. The only way in which you can reach this magnific place is by boat at the back of the island, which, if we follow the path of torches, takes us to the best part of the scene. Two beach chairs right at the shore of the island. Under a rotating umbrella the person can calmly gather his thoughts, drinking

from the glasses that wait for him right near the chairs. We can also observe the interesting fauna, the colored flowers and the gigantic palm trees. The scene was composed with the help of the tutorial, and the objects were downloaded from multiple sites: TurboSquid[3] and Free3D[4].

Figure 1 ,, The scene”

2.2 Functionalities

The functionalities that I successfully implemented are the following:

- The ability to visualize the scene in the point and wireframe views

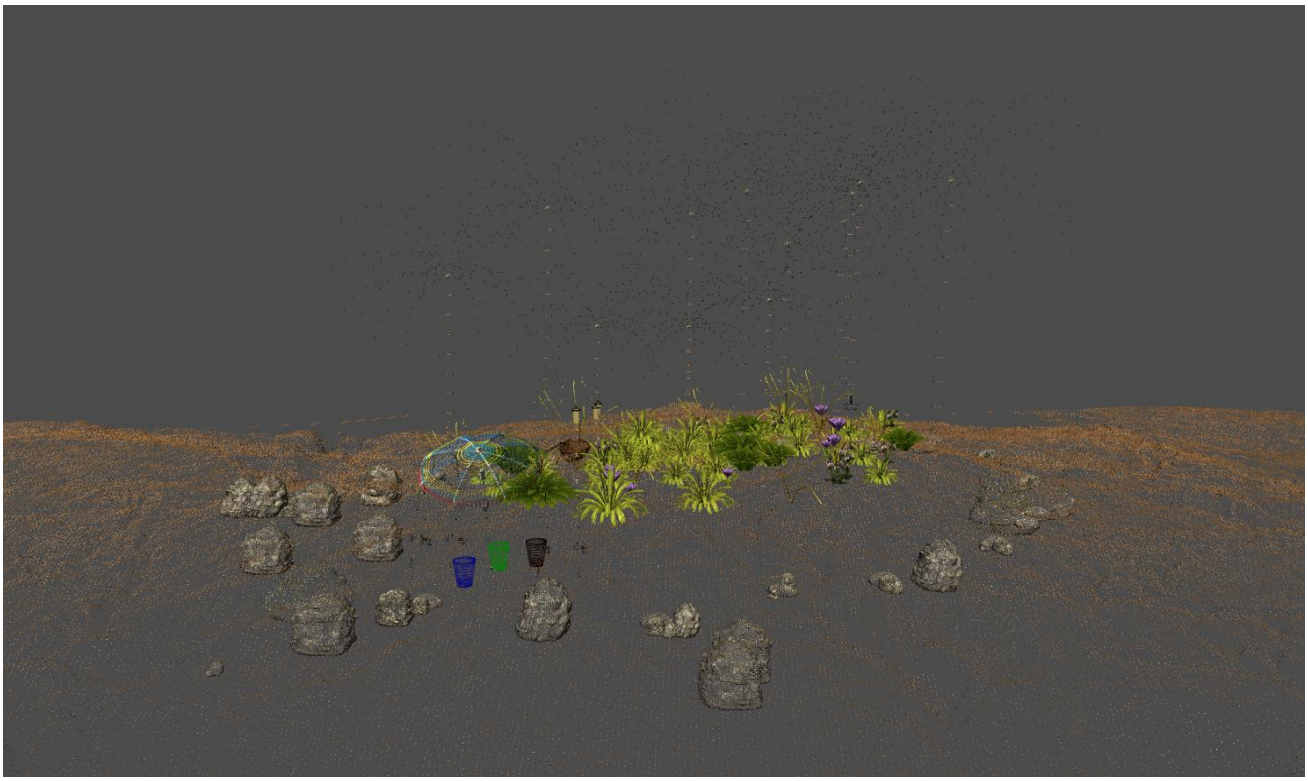

Figure 2 „ Point view”

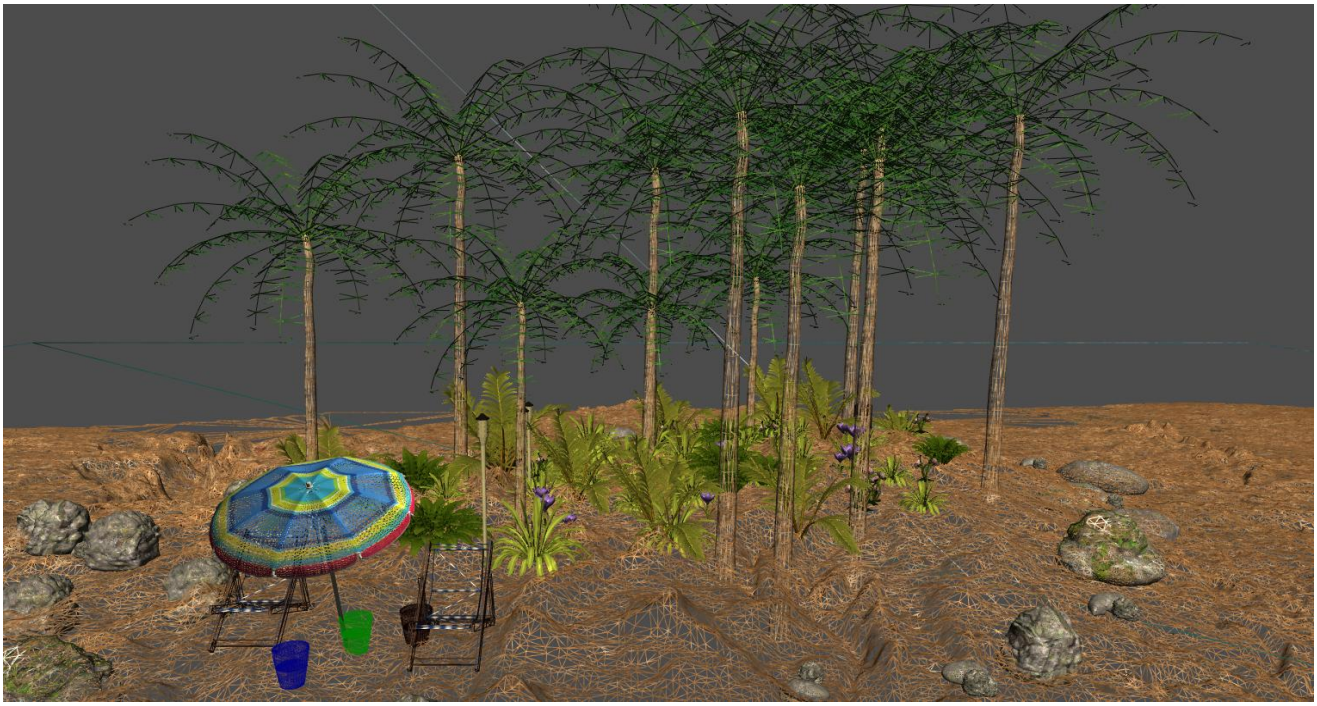

Figure 3 „ Wireframe view”

- The presence of two types of light source: directional and points light.

Figure 4 ,,The point light source”

- A full camera navigation, being able to visualize the whole scene with the help of keyboards and mouse

- An automated tour which gives the possibility to view the whole main parts of the scene

- Animation of umbrella which keeps rotating

- Activation of fog with the help of the keyboard

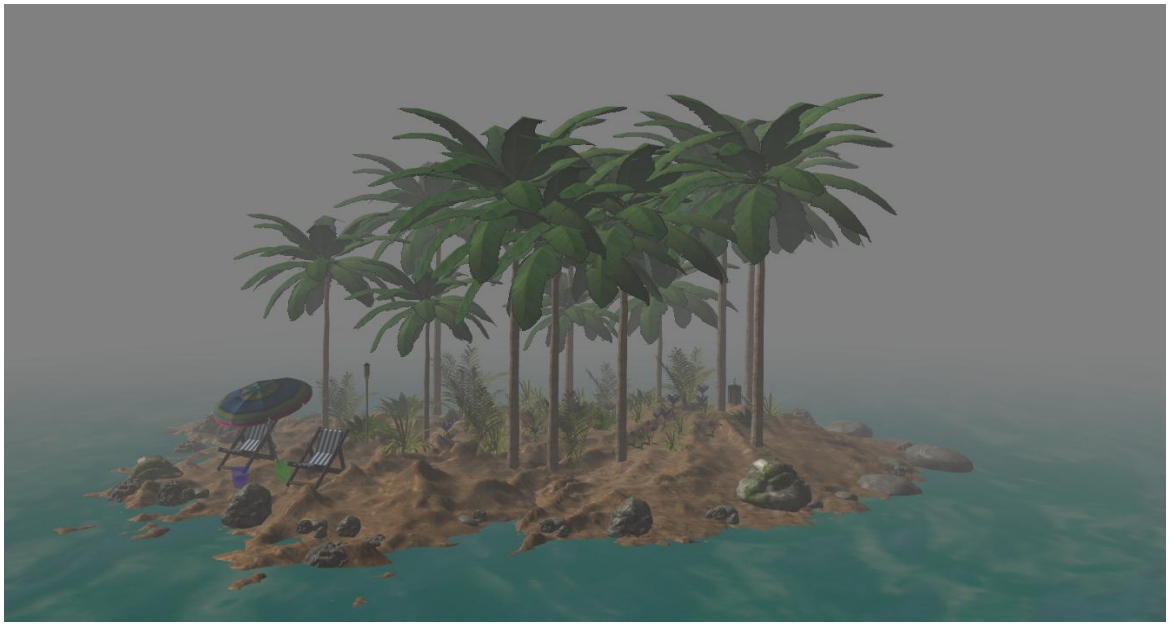

Figure 5 ,,The fog activated in the scene”

- Activation of the rain with the help of the keyboard

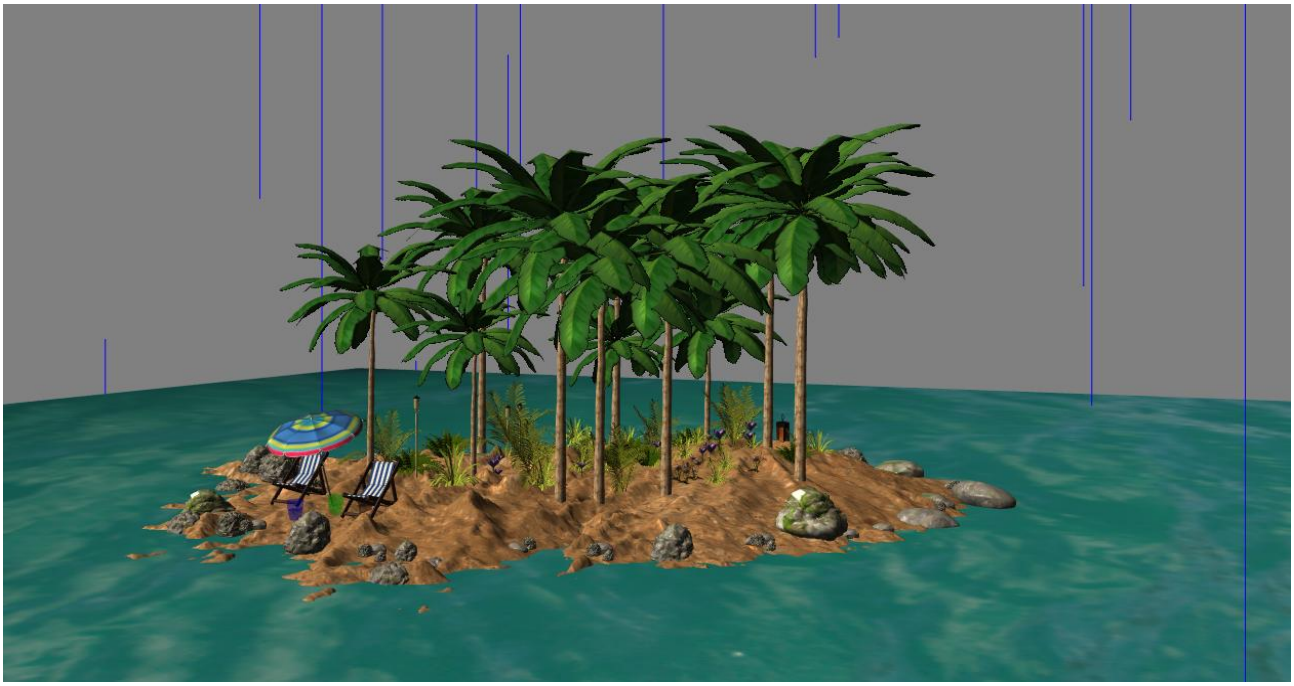

Figure 6 ,,Activation of the rain in the scene”

- The 3 torches that all emit point light

Figure 7 „The light in the front”

Figure 8 „The light in the back”

3. Implementation

3.1 Camera movement

The camera movement together with the automated tour was implemented with the help of the laboratory work 5 [5] which talks about all the types of camera transformations. I used a Camera object initialized with values for position, target and up vector. We use the view matrix which gets updated dynamically with the help of lookAt, to reflect every change of the camera position. For the movements we use normalization of vectors, sin, cos, angles and speed.

The automated tour was done by setting different target positions for the camera. These positions were chosen to take the viewer to all important parts of the scene, around all the objects.

3.2 Directional light

In my project directional light is implemented as the primary source of light, like sunlight. It affects all the objects in my scene, having parallel rays, regarding their position. I did this by calculating all the types of lighting: ambient, diffuse and specular and adding them to the final color of the fragment. These were used to contribute to the realistic appearance of the objects.

I also use a rotation angle to be able to move the light dynamically in my project, to simulate as realistic as possible the sun's movement. By adjusting this angle, and adding or taking from the light density I was capable to replicate the sun's properties.

3.3 Point light

In this project the point of light is demonstrated by a lantern and 3 torches. This type of light is a source of light that depends on the position because is a localized light, it will reflect only on a part of the surface and the intensity will differ on the distance. For this intensity we use attenuation which has the following formula: 1/ constantAttenuation + distanceOfLight* linearAttenuation + distanceOfLight ^2 * quadraticAttenuation. [6]

3.4 Wireframe and point view

The wireframe view was implemented with the help of this function: polygonMode = GL_LINE; and the point view by this one: polygonMode = GL_FILL. They are useful in learning how objects are composed, to see the polygons and points.

3.5 Animation

In this project I applied an animation onto the umbrella that stays between the two chairs. This was made by separating the object into multiple objects so as to be easier to control each part. Regarding the animation, I wanted the higher part, the umbrella, to rotate round and around. I implemented this by applying onto that part multiple rotations for a deltaTime for continuous movement.

The rotations happen to increment once 360 dregrees, allowing the umbrella to fully rotate around its axis. The speed of this rotation can be changed dynamically by modifying the deltaTime.

3.6 Fog

The fog was one of the most useful effect because this one really gave my scene a sense of reality and depth. This was implemented in the fragment shader by calculating the distance from the fragment pixel to the camera. This distance was used to determine how much fog will be applied. I also use a uniform called fogDensity, with the help of which I can control how dense the fog is.

To make the fog transition smoothly, I also used „mix” function to make sure that as you get further away from the fragment it will become less visible. This helped improve the sense of scale and the overall appearance.

3.7 Fragment Discarding

Before applying fragment discarding, my tree’s leaves were green but also had a black background around the green. This was not good because it reduced the realism of my scene. I fixed this using fragment discarding in the fragment shader by checking the texture’s alpha value. If the alpha value was below 0.1, I discarded those fragments.

This technique helped remove the unwanted black edges, making the leaves appear more natural and seamlessly blending with the background. By discarding fragments with

low alpha values, I also improved the rendering efficiency since unnecessary fragments were not processed. Overall, fragment discarding helped enhance both the visual quality and performance of the scene, allowing the tree's appearance to look more realistic while maintaining a good performance.

3.8 Transparent Objects

When it came to the cup glasses, I wanted to implement transparency for all three of them. The method I used was similar to how I handled fog, where I blended colors to create the transparent effect. By mixing the desired color for the glass with a weight factor, I could achieve partial transparency.

This technique allowed for a more realistic appearance, as the transparency could be adjusted dynamically. The weight factor controlled how transparent the glass would appear, enabling the effect to vary depending on the desired look and the environment around the glasses.

3.9 Rain

The rain simulation renders and creates falling raindrops. I did this by placing a bunch of raindrops above my scene by random spawning positions, delays and speed. Each drop has a randomized fall speed which makes it seem like the fall at different movements and speeds. Each of them has a velocity directed downward along the Y axis.

Additionally, a small delay exists before each of them drops, adding to the realism of the rain dropping and the rain process overall. The falling of the drop is done by updating every frame by adding its velocity multiplied by the frame's delta time.

3.10 Data Structures

As data structures I used in the most part the ones that we already had predefined in Project Core. I added new data structure for the camera tour which holds the target of the camera and position of the objects. I also added a struct data structure for the rain in which to store the states of the raindrops.

4. Graphical User Interface

In this project the movement of the user, to be able to visualize the entire scene, I gave the possibility of moving with the help of both mouse and keyboards.

The functionality of each keyboard is the following:

- UP key – moving ahead
- DOWN key – moving backwards
- LEFT key – moving left
- RIGHT key - moving right
- Q key – rotation to left
- E key – rotation to right
- A key – make the directional light less intense
- S key - make the directional light more intensea
- Z key – move the directional light to left
- X key - move the directional light to right
- C key – start automated tour
- F key – start fog effect
- M key – start rain
- P key – activate point view of the scene
- O key – activate wireframe view of the scene
- I key – activate normal view of the scene
- L key – activate the main point light source
- T key – move up
- Y key – move down
- K, J, H – activate each torch

5. Conclusions and further developments

This project helped me, in a practical and entertaining way, how to create my own graphical scene, the crafts and art of OpenGL, Blender and Visual Studio. I learned what rendering is, how to manage shaders, textures, and lighting models. I also now have a full knowledge of all the types of lighting sources and the way they can change the whole aspect of a scene.

For future developments I would like to make the ocean more realistic, adding waves, animations and a more realistic texture. Also, another development could be the changing of the skybox at the change of light: to have day mode and night mode.

In conclusion this project gave me the opportunity to strengthen my skills in OpenGL programming, 3D modeling integration, and scene optimization. It has challenged my creativity and my capability of understanding complicated techniques and algorithms. However, at the end the result is a beautiful, colorful and realistic scene.

6. References

1. https://en.wikipedia.org/wiki/OpenGL
2. https://www.blender.org/about/
3. https://www.turbosquid.com/Search/3D-Models/free/island
4. https://free3d.com/3d-models/obj
5. https://moodle.cs.utcluj.ro/pluginfile.php/202187/mod_resource/content/1/Laboratory_work_5.pdf
6. https://moodle.cs.utcluj.ro/pluginfile.php/202209/mod_resource/content/3/Laboratory_work_8.pdf