Logo, company name

Description automatically generated

Graphic Processing

Laboratory Project

3rd year

Student: Onaci Andreea-Maria

Group: 30431

Contents

[1. Subject specification 2](#_Toc155944053)

[2. Scenario 3](#_Toc155944054)

[a) Scene and objects description 3](#_Toc155944055)

[b) Functionalities 4](#_Toc155944056)

[3. Implementation details 4](#_Toc155944057)

[a) Functions and special algorithms 4](#_Toc155944058)

[**1.** **Possible solutions** 4](#_Toc155944059)

[**2.** **The motivation of the chosen approach** 5](#_Toc155944060)

[b) Graphics model 6](#_Toc155944061)

[c) Data structures 6](#_Toc155944062)

[d) Class hierarchy 6](#_Toc155944063)

[4. Graphical user interface presentation. User manual 6](#_Toc155944064)

[5. Conclusions and further developments 7](#_Toc155944065)

[6. References 7](#_Toc155944066)

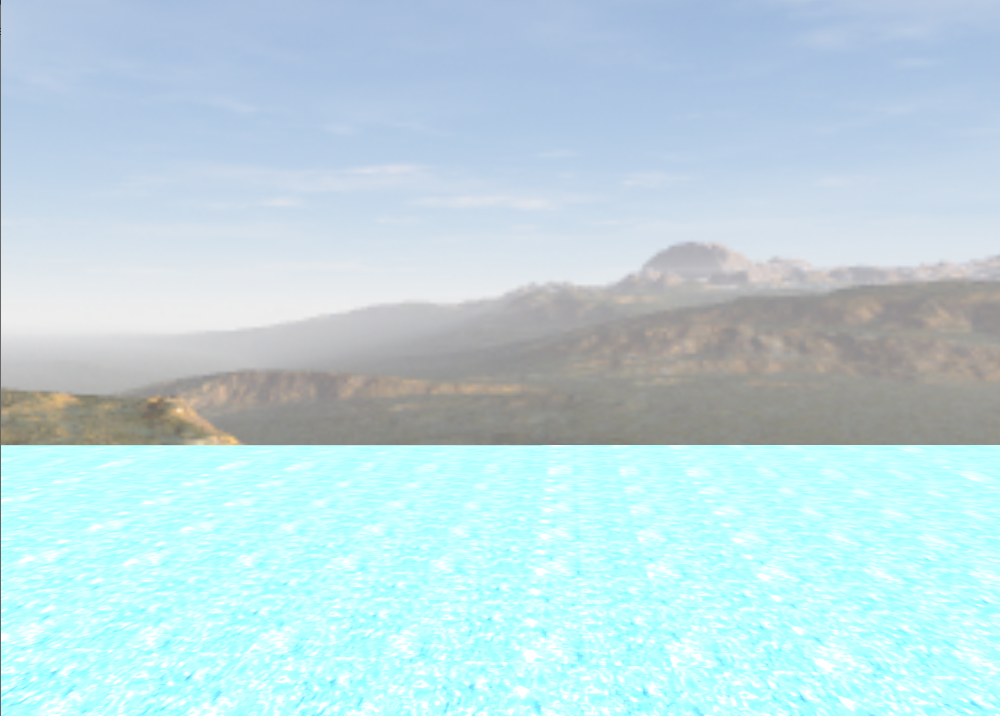
1. Subject specification

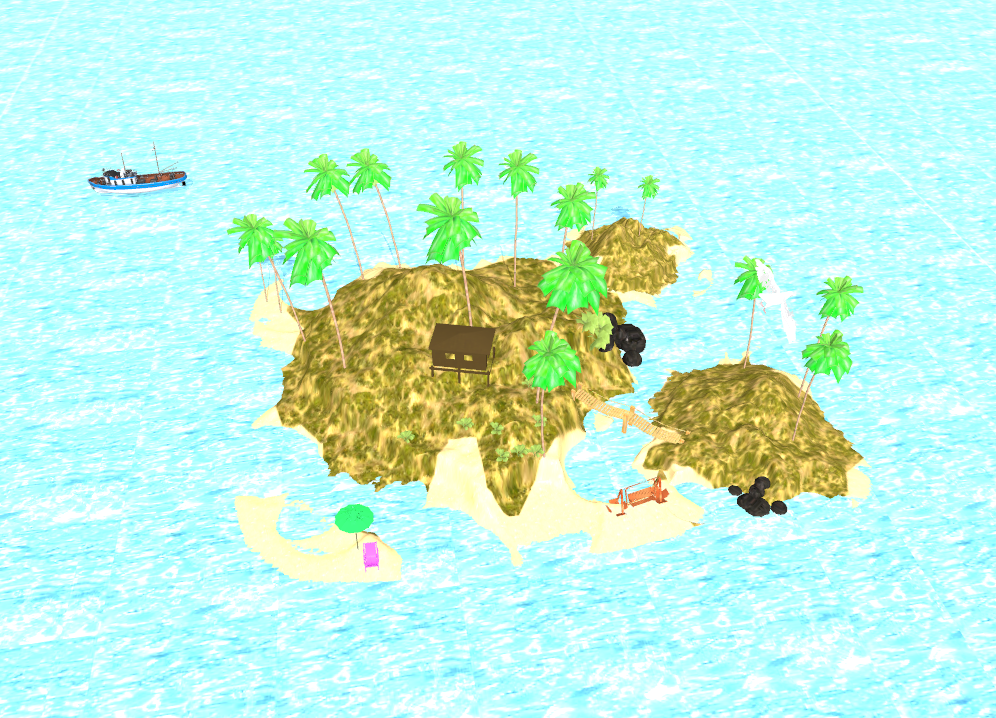
OpenGL is an open-source cross-platform graphics API (Application Programming Interface) that enables developers to create high-performance 2D and 3D graphics applications.

The scene I implemented is a Small Tropical Island in the middle of the ocean. The user interacts with the object scene directly using the mouse and keyboard, allowing for movement and exploration within the environment.

1. Scenario
2. Scene and objects description

The scene depicts a small island in the middle of the ocean. The mountains and the sky can be observed in the farest plan. On the island, we can notice a house, some trees, a bridge, some stones, a swing, a beach chair and an umbrella. On the ocean we can observe a ship and in the air a bird.





## Functionalities

Users can visualize the entire scene through mouse and keyboard inputs and toggle between solid, wireframe, polygonal, and smooth face views. They have control over the appearance of fog, adjusting its’ intensity as desired. The scene features various light sources, including directional, point (on the bird and chair) and spot lights (on chair), which users can toggle on/off using specific keyboard commands. The user has the ability to move the ship in 4 directions using keyboard controls, while the bird and the swing exhibit autonomous movement within the scene. The umbrella’s hat is also rotating using specific keyboard command.

1. Implementation details
2. Functions and special algorithms

### **Possible solutions**

The OpenGL library offers a diverse array of functions, including:

* + glViewport(...): Configures Viewport transform.
  + glTexParameterfv(...), glBindTexture(...), glTexImage2D(...): Used for various purposes, such as creating depth textures for Framebuffer objects.
  + And numerous others.

The central function in this project are void renderScene() and

void drawObjects(gps::Shader shader, bool depthPass), responsible for:

* + Sending data to shaders.
  + Computing values.
  + Creating models and shadows.
  + Computing normal matrices.
  + Handling rotations, translations, and scaling.
* void initUniforms(): Configures details related to lights, including point lights, directional lights, and spotlights.
* void initShaders(): Instantiates shaders for the project.
* void initObjects(): Instantiates all 3D models for use in the project.

Essential user interaction functions include:

* + void processMovement()
  + void previewFunction()
  + void mouseCallback(GLFWwindow\* window, double xpos, double ypos)
* void keyboardCallback(GLFWwindow\* window, int key, int scancode, int action, int mode)

These functions facilitate necessary changes to models or the camera, ensuring movements desired by the user.

* void mouseCallback function implementation leverages Euler Angles.

Implementation utilizes the GLM mathematics library tailored for OpenGL to execute functions and algorithms.

### **The motivation of the chosen approach**

The chosen approach of creating a Small Tropical Island scene in the middle of the ocean is motivated by the practical knowledge gained in dedicated laboratories for the subject. Through these laboratories, I acquired skills in computing light, shadows, and performing essential operations like translation, scaling, and rotation on various objects. With a solid foundation and a project skeleton in place, the implementation of this project follows a well-defined path. The aim is to apply the learned concepts in a real-world scenario, allowing for a comprehensive understanding of how these graphical techniques come together to create a visually appealing and interactive virtual environment. This project serves as an opportunity to practically apply the acquired knowledge and showcase the capabilities of OpenGL technology in crafting immersive and dynamic scenes.

1. Graphics model

During the graphics modeling phase, I browsed online model sources for various objects and textures, importing them into Blender for intricate editing and placement within the final scene. This aspect presented challenges, particularly with some textures not rendering correctly on the objects or the objects themselves facing display issues. This prompted an extensive search for appropriate objects and textures. Despite the complexities, I view this stage as an engaging and creative process, where the pursuit of optimal elements significantly contributed to the overall aesthetic finesse of the project.

1. Data structures

The data structures used were already implemented in the skeleton received from the laboratory, used for holding the attributes for different objects used in the project (implemented in std\_image.hpp and tiny\_obj\_loader.hpp).

1. Class hierarchy

* **Camera**: contains the implementation for camera movement (up, down, front, back, left, right and also the preview approach for camera animation)
* **SkyBox**: contains skybox implementation (load, draw, load textures, initialize)
* **Mesh**: represents a 3D object;
* **Model3D**: contains methods for printing meshes using a specified shader program
* **Shader**: contains methods for creating and activating shader programs

1. Graphical user interface presentation. User manual

* ESC – exit the program
* P – preview the scene (camera animation)
* M – show the depthMap
* Q – rotate the pigeon in counterclockwise
* E – rotate the pigeon clockwise
* J – rotate the light counterclockwise
* L - rotate the light clockwise
* W – move the camera forward
* A - move the camera left
* S – move the camera backward
* D – move the camera right
* R – start pointLight
* T – stop pointLight
* Y – start spotlight
* U – stop pointLight
* 1 – line view
* 2 – point view
* 3 – normal view
* 7 – polygon smooth
* B – start fog
* V – stop fog
* 4 – increase the intensity of fog
* 5 – decrease the intensity of fog
* I – rotate the umbrella
* Left – move the boat to left
* Right - move the boat to right
* Up - move the boat in front
* Down - move the boat back
* O – discarding parts of objects
* K – wet island

1. Conclusions and further developments

In summary, I would like to express that working with Blender presented its fair share of challenges, and the process of seamlessly integrating my scene into the OpenGL implementation proved to be a genuinely demanding task. Delving into the intricacies of Blender required a concerted effort, and navigating the complexities of its features was both a learning experience and a test of my problem-solving skills.

Moreover, tackling the integration of my scene into the OpenGL framework posed its own set of challenges. Debugging the code demanded a considerable amount of time and effort, yet the journey was undeniably rewarding. Despite the hurdles, I found the entire process fascinating and engaging, providing me with an opportunity to enhance my understanding of both Blender and OpenGL. This experience not only refined my technical skills but also instilled in me a sense of accomplishment for overcoming obstacles and successfully bringing my vision to life.

Future Improvements: rain, more objects in the scene, more animations provided, day&night approach, computing more accurate shadows for all types of implemented light, etc.

1. References

* <https://github.com/SonarSystems/Modern-OpenGL-Tutorials>
* Youtube
* <https://learnopengl.com/Advanced-Lighting/Advanced-Lighting>
* <https://free3d.com>