WINTER WONDERLAND

Graphic Processing

Student: Danci Patricia Ioana

Group: 30434

Technical University of Cluj-Napoca

January 2025

Contents

1. Subject Specification

This project involves creating an interactive 3D scene using OpenGL and Blender. The primary objectives are to demonstrate the use of modern OpenGL features, such as:

- Lighting models incorporating (both directional and point lighting).
- Animations such as wave effects on water and wind on trees, as well as movement (rotations, scalations, translations).
- Shadow mapping.
- Visual effects such as fog, rain, snow, smoke, etc.
- Dynamic camera navigation.
- Implementation of skyboxes and skydomes.
- Design of an object in Blender using Sculpting and UV Mapping techniques.

The project aims to create a Winter Wonderland 3d scene, by incorporating multiple objects and textures, all designed to create a seamless environment.

2. Scenario

2.1 Scene and Objects Description

The 3D scene represents a simple winter landscape. It includes a variety of objects, placed one by one directly in OpenGL in order to have full control over each one.

- **Ground and Terrain:** A snowy ground sculpted with draw and crease brushes in Blender to create the illusion of realistic terrain and a lake.
- **Lake:** A semi-transparent frozen lake composed by overlapping 2 planes, one with wave animations to simulate water effects, and the other with an icy texture placed above.
- **Trees:** Pine trees made out of a bark object and a leaves object overlapped, in order to be able to apply wind to the leaves.
- **Cottage:** A wooden cottage with a snowy texture, which has a smoke effect applied to its chimney.
- **Playground:** A playground roundabout which has a rotation animation applied to simulate wind as well.
- **Deer:** Static model of deer.
- **Bonfire:** A bonfire with coal and log textures to simulate burnt wood.
- **Skybox:** A dynamic skybox transitioning between day and night based on keyboard inputs.

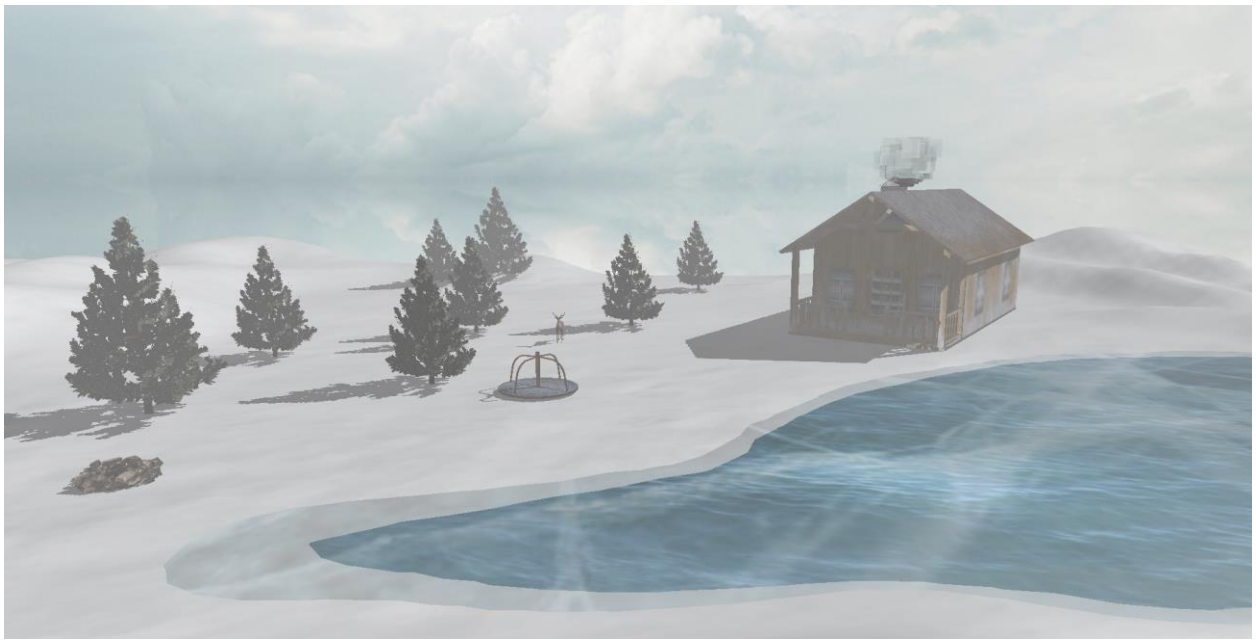
2.2 Functionalities

The project implements the following visual functionalities:

- Camera movement in all directions toggles with keypads W A S D for x-axis and z-axis movement, as well as R and F for y-axis movement
- Rotation of the scene using both mouse and keypad E Q T G
- Activation of solid mode using keypad 1
- Activation of wireframe mode using keypad 2
- Activation of point view mode using keypad 3
- Dynamic lightning:
    - Enable/Disable of the directional light (Daymode/Nightmode) using keypad N
    - Enable/Disable of the main point light (blue) using keypad P
    - Enable/Disable of 3 additional point lights (meant to incorporate Northern Lights with colors red, green, blue) using keypad N as well, since they can only be seen at night
- Animations:
    - Wind -> permanent on tree leaves and simulated on the roundabout through rotations
    - Wave -> computed in the same manner as the wind, but applied to the lower lake plane to simulate water movement
    - Rain -> toggled with keypad 9 and UP/DOWN for density
    - Smoke -> computed with particles of a PNG smoke texture and dynamic transparency
- Shadows that move based on a light angle that increases with each rendering
- Fog that is activated with keypad O
- Automated tour which follows a predefined camera path computed with coordinates from Blender and highlighting the main objects of the scene -> activated with keypad K

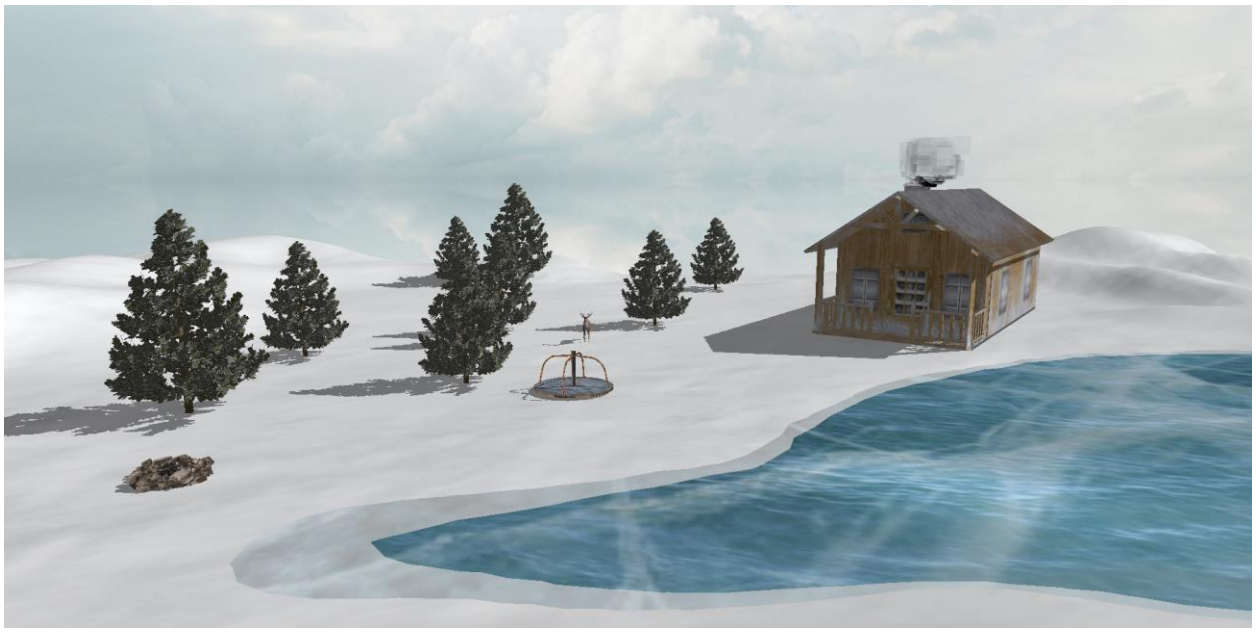
Attached below is the visualization of these functionalities:

Fog enabled:

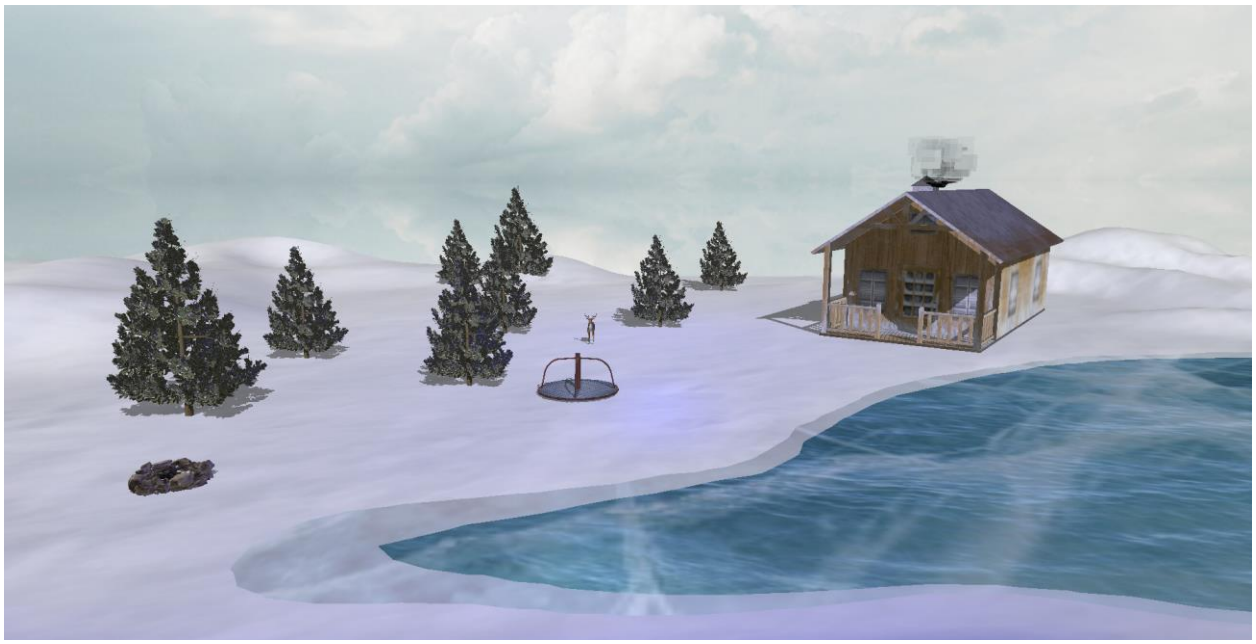


Night Mode enabled:

Shadows moving:



Point light enabled:



Rain:

3. Implementation Details

3.1 Lighting Model in OpenGL

Lighting in this project is based on the Phong reflection model, combining ambient, diffuse, and specular components. The scene incorporates:

- **Directional light**: simulates sunlight, affecting all objects equally, and is represented using a normalized light direction. The ambient and diffuse intensity of the light can be toggled for day and night modes.
- **Point light**: positioned at fixed locations, point lights create localized illumination. They also include an attenuation factor to limit their effect based on distance. In this project, each point light has a specific color, neither is plain white.
- **Shadow mapping**: shadows are calculated using a depth map from the perspective of the directional light source. This technique helps enhance depth and realism.

Each light type uses GLSL shader myBasicShader for real-time calculations by implementing bool type uniforms that control enabling/disabling.

3.2 Camera Navigation

The camera system uses a combination of keyboard inputs and mouse movements for seamless navigation:

- **Keyboard Input:** Implements movement in six directions (forward, backward, left, right, up, and down).
- **Keyboard Input (rotation):** Implements rotation in 4 directions (left, right, up, down).
- **Mouse Input:** Handles rotation based on yaw and pitch to simulate head movement.
- **Automated Camera Tour:** A pre-defined camera path guides the user through the scene.

The camera's position and orientation are updated each frame, while the implementation uses the glm library to manage view, model and projection matrices. Additionally, the camera is defined as an instance of the Camera class

3.3 Effects

In the Winter Wonderland scene, several effects are implemented to enhance realism:

- **Wave Animation**: The lake's base surface features a dynamic wave pattern, which is computed in the vertex shader using sinusoidal transformations. This effect is permanently enabled. It basically copies a wind effect, except its speed is much larger to simulate waves collapsing.

- **Tree Wind**: The tree's leaves swing gently, mimicking a slight wind. In order to synchronize all trees' movements, the time uniform has been sent only once to the shader, specifically before drawing the first tree.
- **Fog**: The main fragmentshader calculates exponential fog density, blending it with the scene.
- **Particles:** Rain and Smoke effects are generated using particle systems. Each particle's position, size, and transparency are updated dynamically and their values are given randomly. Moreover, the Rain effect can be toggled, whereas the Smoke effect is permanent.

3.4 Graphics Model

The graphics model utilizes a hierarchical approach to organize and render scene elements efficiently.

- **Models**: each object is loaded as a 3D model using external .obj files.
- **Shaders**: custom GLSL shaders handle specific rendering tasks, such as lightning, shadows, and animations.
- **Textures**: textures are applied to models via .mtl files to enhance visuals. They are mapped to model surfaces through UV coordinates.
- **Framebuffers**: a FBO object is used for shadow mapping to create depth textures.
- **Lightning Model**: the lightning is computed using Phong model.

3.5 Data Structures

The data structures used in this project are specific of GLM library and also OpenGL's.

- **Vectors**: standard C++ vectors to store particles (rain/smoke).
- **Matrices and Vectors** from the GLM library.
- **Custom Structs**: Raindrop for individual rain particles and SmokeParticle for smoke.

4. Graphical User Interface Presentation / User Manual

4.1 User Controls

The application provides intuitive controls to navigate the scene:

- Keyboard Controls
- Mouse Controls

4.2 Automated tour

The automated tour provides a guided experience through the scene while highlighting key objects. To activate this mode, press key “K”.

The interface design prioritizes simplicity, therefore, for using the application, you should run the executable file. Instructions regarding the functionality of the program can be found in Chapter 2.

5. Conclusions and Further Developments

Conclusions

This project successfully proves the capabilities of modern OpenGL in creating 3D environments by implementing advanced rendering techniques, dynamic animations and interactive elements, including camera navigation and real-time toggles for different features.

By implementing such technologies, I have been given a great opportunity to expand my knowledge in terms of graphic processing and OpenGL. Additionally, I was also able to learn how to create 3D object and apply textures using Blender.

Further Developments

Personally, these are the further developments I look forward to applying to my project:

- Adding a fire object / particles over the bonfire object and apply an effect/animation to make it more realistic.
- Find more decent 3D objects with more polygons/faces and more complex textures.
- Apply some animations to the cottage, such as opening the door/window.
- Add more objects to make the scene more complex
- Expand the scene
- Dynamic animal behavior, such as making the deer jump trough the forest.
- Add physics-based interactions.
- Add snow effect.
- Add reflective surfaces or transparent objects.

6. References

[1] Learn OpenGL, "Shadow Mapping"

[2] Cubemaps & Skyboxes

[3] Transparency & Blending

[4] Tutoriale Blender

[5] Graphic Processing Laboratory Guides

[6] Learn OpenGL, "Model"