OpenGL Project

Graphic Processing

Author: Ghile Patricia
Group: 30433

FACULTY OF AUTOMATION AND COMPUTER SCIENCE

January 2026

Contents

1 Presentation of the theme

The theme of this project is the exploration and application of photorealism concepts in interactive 3D environments using OpenGL. The project focuses on rendering a complex scene composed of multiple objects, blending technical precision with artistic design to produce a realistic nocturnal atmosphere. For implementation, the project relies on essential libraries such as OpenGL, GLFW for window management, and GLM for mathematical transformations. Interactivity is a key element, providing an immersive experience through a sophisticated camera control system.

2 Scenario

2.1 Description of the scene and objects

The scenario combines traditional medieval architecture with atmospheric environmental elements to create a cohesive gothic setting. The objects included in the scene are:

- **Castle:** The central architectural piece, featuring detailed stone textures and complex geometry.
- **Ground:** A cobblestone surface serving as the floor for the entire courtyard.
- **Dead Forest:** A collection of barren tree models distributed around the castle to enhance the eerie atmosphere.
- **The Moon:** A rotating celestial body that provides global light and serves as a focal point in the sky.
- **The Lantern:** A point light source located within the courtyard, providing localized orange illumination.
- **The Wolf:** A 3D model strategically positioned within the scene, using custom materials.
- **Skybox:** A starry night background that creates the illusion of a vast, expansive environment.

2.2 Functionalities

The application includes several technical functionalities designed for a dynamic user experience:

- **Scene Navigation:** A 3D camera allows exploration using WASD keys and mouse movement.
- **Object Manipulation:** Implementation of rotation, translation, and scaling transformations.
- **Lighting System:** Management of directional (Moon) and point (Lantern) light sources.
- **Atmospheric Effects:** Implementation of distance-based fog and a real-time rain particle system.
- **Rendering Modes:** Switching between solid, wireframe, and point rendering.

3 Implementation Details

3.1 Functions and Algorithms

Matrix Transformations (Model, View, Projection)
The transformation pipeline is fundamental for 3D rendering:

- **Model Matrix:** Positions, scales, and rotates 3D objects in the global scene. Each object has its own model matrix, which transforms local coordinates into world coordinates.
- **View Matrix:** Represents the camera's position and orientation. It transforms world coordinates into camera-space coordinates, simulating movement through the scene.
- **Projection Matrix:** Defines the field of view and aspect ratio. A perspective projection matrix is used to create the illusion of depth, making distant objects appear smaller.

Camera Control and Interactivity
The camera is controlled through a combination of keyboard input for translation (W, A, S, D) and mouse input for rotation. The rotation is calculated using Euler angles (Yaw and Pitch), which are constrained to prevent gimbal lock. This allows the user to explore the gothic environment freely from any angle.

Lighting and Attenuation
The lighting system is based on the Phong reflection model, which calculates ambient, diffuse, and specular components. The scene uses:

1. **Directional Light:** Simulates the Moon, providing a constant light direction across the entire scene.
2. **Point Light:** Simulates the Lantern. It uses quadratic attenuation to decrease intensity based on distance:

$$f_{att} = \frac{1.0}{constant + linear \cdot dist + quadratic \cdot dist^2} \tag{1}$$

Figure 1: Visual demonstration of point light attenuation on the castle walls.

Rain Simulation System
To enhance the atmosphere, a real-time rain system was implemented. It manages a pool of 2000 particles (`RainDrop` objects). Each particle has its own velocity and vertical position.

When a drop reaches the ground level, it is immediately reset to a random height above the scene, creating a continuous loop. For rendering efficiency, each drop uses a thinned and scaled version of a simple mesh.

Exponential Fog

Fog is used to increase depth perception and add mystery. It is calculated in the fragment shader by blending the final pixel color with a fog color (dark gray) based on the distance from the camera:

$$fogFactor = e^{-(distance \cdot density)^2} \tag{2}$$

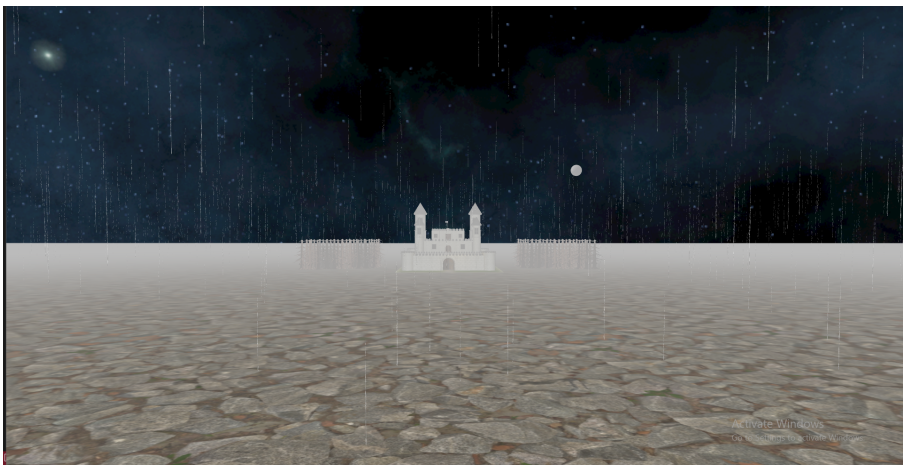

Figure 2: Atmospheric fog effect applied to the 3D scene.

Skybox and Environment

The Skybox uses a cube map with six high-resolution textures. It is rendered first with depth testing modified so that it always stays in the background, providing a realistic starry sky that moves with the camera.

OpenGL Error Handling and Callbacks

The `glCheckError` function is called after major OpenGL operations to ensure stability. Additionally, a window resize callback is registered to update the `glViewport` and the projection matrix whenever the user changes the window size, ensuring the scene is never distorted.

3.2 Graphic Model

The project architecture is modular, using specialized classes for shaders, textures, and models. The following figures show the different visualization modes implemented:

Figure 3: Solid Mode

Figure 4: Wireframe Mode

Figure 5: Point Mode

4 Presentation of the graphical user interface / user manual

The application offers an interactive interface with the following controls:

- **Movement:** Use **W, A, S, D** to navigate through the courtyard.
- **Orientation:** Move the **mouse** to look around (360-degree rotation).
- **Animations:** Press **R** to toggle the automatic castle rotation. Use **Q/E** for manual rotation.
- **Scaling:** Use **Z/X** to increase or decrease the scale of the castle.
- **Rendering Modes:** **G** for Solid, **F** for Wireframe, **M** for Pointform.
- **System:** **ESC** to terminate the application.

5 Conclusions and Future Development

This project successfully demonstrates how OpenGL can be used to create an immersive, photorealistic atmosphere. By combining complex lighting, particle systems, and fog, the gothic theme is effectively conveyed. Future improvements could include shadow mapping, localized sound effects for the rain, and camera collisions with the castle walls.

6 References

- **OpenGL Tutorials**: `https://learnopengl.com/`
- **Free3D**: `https://free3d.com/`
- **WoW Tools Export**: `https://wow.tools/export/`
- **OpenGL Laboratories**: Provided by the Technical University of Cluj-Napoca.