OpenGL- Volcanic Environment Simulation - Report

Overview

Simulated a 3D World Using **OpenGL** in Python An interactive 3D environment A volcano, terrain, weather (clouds), a day-night cycle, a windmill - all demonstrating textures, lighting, particle effects and procedural generation.

Objectives

- Build an interactive 3D world.
- Use particle systems for eruptions and rain.
- For realism, work with textures and lighting.
- Showcasing day-night cycles and interactivity.

Features

Visuals

1. **HD textures**: You will find higher resolution textures applied to the terrain, the volcano, the lava, and the sky for better aesthetics.
2. **Dynamic lighting**:
   - Day-night changes in ambient and diffuse light intensity.
   - Implementation of lava glow using emissive materials and a second light source.
3. **Procedural terrain**: Using sinusoidal height functions.
4. **Interactive camera**: Using your keyboard you can control

Dynamic Environment

1. **Camera Movement**:
   - Move the camera using the **arrow keys**.
   - Move vertically up using the **Space** key.
   - Move vertically down using the **H** key.
2. **Volcano and Lava**:
   - The Volcano consists of a textured crater with glowing lava spilling and lighting its surrounding.
   - Particles represent eruptions giving it a game mechanic feel with physics and life span.

   - **Controls**:
      - Press **F** to toggle volcanic eruption mode.

3. **Weather Effects**:
   - Rain drops with varying trajectories to simulate a realistic downpour.
   - **Controls**:
      - Press **R** to toggle raining mode.

4. **Windmill**: A windmill with spinning blades makes the environment feel a bit more alive.

5. **Day-Night Cycle**:
   - Sun and moon transitions based on user input.
   - **Controls**:
      - Press **N** to toggle night-day mode.

Technical Details

Libraries Used

- **PyOpenGL**: Used for 3D rendering, lighting, transformations, etc.
- **NumPy**: For terrain heights generation and particle physics computations

Key Functionalities

1. **Texture Loading**: Import images to apply them to 3D objects: Function load_texture.

2. **Particle System**: The Particle class creates volcano eruptions with realistic motion and gravity.

3. **Procedural Terrain**: Heights generated using sinusoidal functions with multi-scale perturbations.

4. **Rain Simulation**: The Raindrop class contains logic to render and update rain particles with random velocities and respawn behavior.

5. **Day-Night Cycle**: Ambient & Diffuse Lighting ParametersNight And Day Mode Toggle

Challenges

- More optimization for performance to hold frame rates.
- Optimizing lighting and texture mapping.
- Blending interactions and animations easily.

Future Enhancements

- Use sound effects like thunder and fog.
- AI based events such as Earthquake.

Conclusion

This project used procedural generation, particle systems, and texture mapping to simulate realistic 3D environments. Planned improvements are intended to enhance interactivity and immersion.