3D OpenGL Pygame Lylia Documentation

This script initializes a 3D scene rendered with OpenGL in a Pygame window. It loads models and textures, applies animations and glow effects, and allows interactive camera control.

# Setup and Initialization

- Pygame and OpenGL initialization:  
 Initializes Pygame, its mixer for audio, and sets up an OpenGL context with double buffering and depth testing enabled for 3D rendering.  
  
- Shader program:  
 Loads and compiles GLSL shaders via create\_shader\_program(). This controls how the 3D models are rendered including lighting and textures.  
  
- Model loading:  
 Models and their textures are loaded from text files in the "parts" directory using load\_model\_from\_txt.  
  
- Projection matrix:  
 Sets a perspective projection matrix based on config parameters (FOV, near/far planes).  
  
- Background color:  
 Set to black (0,0,0).

# Camera and Viewing Controls

The camera position and orientation are controlled by three parameters:  
- distance: Zoom level (distance from camera to the origin).  
- rot\_x: Rotation angle around the X-axis (vertical rotation).  
- rot\_y: Rotation angle around the Y-axis (horizontal rotation).  
  
These parameters define the viewpoint of the 3D scene.

## Initial Camera State

- Initial values:  
 - distance = 5.94 units  
 - rot\_x = 42.5 degrees (tilt up/down)  
 - rot\_y = 20.5 degrees (rotate left/right)  
  
- Target values: These represent where the camera is moving smoothly toward, allowing smooth transitions.

## User Controls for Camera Movement and Viewing

|  |  |
| --- | --- |
| Input Method | Effect |
| Mouse Drag (Left Button) | Rotate the camera view by changing rot\_x and rot\_y: rot\_y increases with horizontal mouse movement (left/right drag), rotating the scene horizontally. rot\_x increases with vertical mouse movement (up/down drag), tilting the scene vertically. |
| Mouse Scroll Wheel | Zoom in/out by adjusting distance: - Scroll Up (wheel forward) → zoom in (decrease distance, minimum 1.0) - Scroll Down (wheel backward) → zoom out (increase distance) |
| Keyboard Left Arrow or 'A' key | Cycle to previous preset camera view in views list. Updates target\_distance, target\_rot\_x, and target\_rot\_y. |
| Keyboard Right Arrow or 'D' key | Cycle to next preset camera view in views list. Updates camera targets. |
| Keyboard 'P' key | Prints current camera parameters (distance, rot\_x, rot\_y) to console and appends to a file view\_log.txt. Useful for debugging or saving viewpoints. |
| Exit program | Close window (X button) |

## Preset Camera Views

- Stored in the list views as tuples of (distance, rot\_x, rot\_y).  
- You can cycle through these views with left/right arrow or A/D keys for quick perspective changes.

## Smooth Camera Movement

- Current camera parameters smoothly interpolate towards the target values using a linear interpolation (lerp) function with speed defined by lerp\_speed = 0.04.  
- This creates fluid, animated transitions when changing zoom or rotation instead of abrupt jumps.

## How Camera Matrices Are Calculated

- The camera is positioned at (0, 0, distance).  
- The glm.lookAt function sets the view matrix pointing the camera at the origin (0,0,0) (or a defined target in config).  
- Model rotations (rot\_x, rot\_y) are applied to all scene objects via their model matrices, effectively rotating the scene relative to the camera.

# Model Animation and Effects

- Objects in the scene animate with slight floating and orbiting motions (except the world object).  
- Some objects have emissive glow colors that pulsate with time for visual effects, controlled in the main loop.

# Summary of User Controls

|  |  |
| --- | --- |
| Action | Control Input |
| Rotate camera view | Click and drag mouse (LMB) |
| Zoom in/out | Mouse wheel scroll |
| Switch to next preset camera | Right arrow key or 'D' |
| Switch to previous preset camera | Left arrow key or 'A' |
| Print current view info | 'P' key |
| Exit program | Close window (X button) |

# Additional Notes

- The camera rotation limits are not clamped, so you can freely spin around the model infinitely.  
- Zoom cannot go closer than 1.0 units to avoid clipping or going inside the model.  
- The smooth interpolation makes controls feel responsive but not jerky.