PROJECT PRESENTATION

RAYTRACER - A SIMPLE 3D RENDERING ENGINE

LANGUAGE: C++
COMPILATION: MAKEFILE

The goal of the Raytracer project is to build a **basic 3D rendering engine using the ray tracing technique,** from scratch. The engine must be capable of rendering 3D scenes by **simulating how rays of light interact with objects** — including reflection, refraction, shadows, and lighting — to generate realistic images.
This project serves as a **deep dive into geometry, optics, and mathematical modeling** of scenes, while also reinforcing skills in **clean C++ architecture**, file parsing, and performance optimization.
Ray tracing is a rendering technique where each pixel is calculated by tracing the path that a ray of light would take in a virtual scene. For each ray:

1. Intersection is computed with objects in the scene.
2. The color is determined by material properties, lights, and optional reflection/refraction.
3. The final pixel color is written to the screen or image buffer.

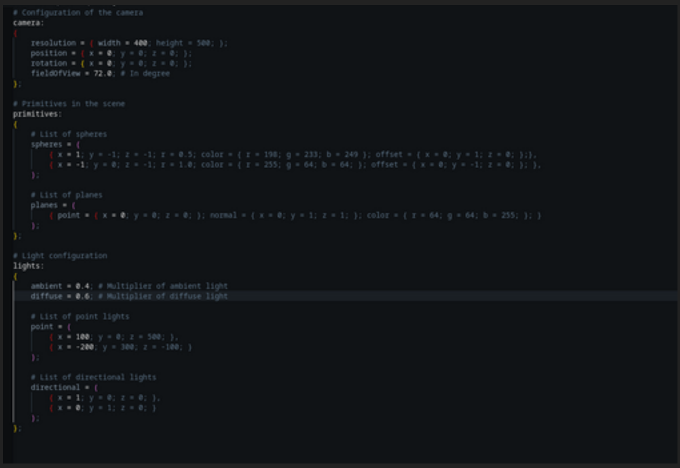
Features

- **3D scene rendering** using ray tracing.
- Support for **basic geometric primitives:** spheres, planes, cylinders, cones.
- **Multiple light types** (point, directional).
- Phong shading model (ambient, diffuse, specular lighting).
- **Scene file parsing** to define object placement and settings.
- Rendering to **SFML** window or image file.

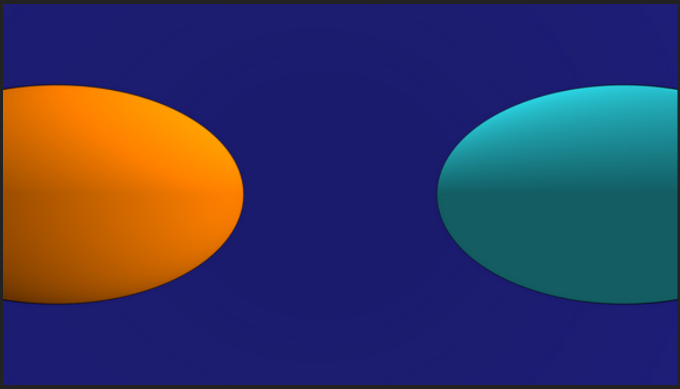
Why I like this project ?

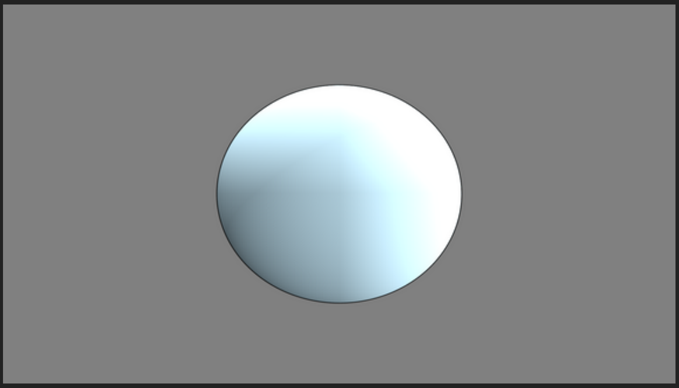
- Mathematics
    - **Ray-object intersection** algorithms.
    - Vector operations (dot/cross product, normalization).
    - Matrix transformations for camera & object rotation.
- Lighting
    - Basic shading: ambient, diffuse, specular.
    - Shadow rays to compute object occlusion.
- Architecture
    - **Modular design:** scene parser, math utils, renderer core.
    - **Object-oriented class hierarchy** (e.g. Object, Sphere, Plane, Light).

Scene example (.cfg file)



```
# Configuration of the camera
camera:
{
    resolution = { width = 400; height = 500; };
    position = { x = 0; y = 0; z = 0; };
    rotation = { x = 0; y = 0; z = 0; };
    fieldOfView = 72.0; # In degree
};

# Primitives in the scene
primitives:
{
    # List of spheres
    spheres = (
        { x = 1; y = -1; z = -1; r = 0.5; color = { r = 198; g = 233; b = 249 }; offset = { x = 0; y = 1; z = 0; };},
        { x = -1; y = 0; z = -1; r = 1.0; color = { r = 255; g = 64; b = 64; }; offset = { x = 0; y = -1; z = 0; }; },
    );

    # List of planes
    planes = (
        { point = { x = 0; y = 0; z = 0; }; normal = { x = 0; y = 1; z = 1; }; color = { r = 64; g = 64; b = 255; }; }
    );
};

# Light configuration
lights:
{
    ambient = 0.4; # Multiplier of ambient light
    diffuse = 0.6; # Multiplier of diffuse light

    # List of point lights
    point = (
        { x = 100; y = 0; z = 500; },
        { x = -200; y = 300; z = -100; }
    );

    # List of directional lights
    directional = (
        { x = 1; y = 0; z = 0; },
        { x = 0; y = 1; z = 0; }
    );
};
```





Click here to see the project repository