MINISTRY OF EDUCATION AND RESEARCH

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA, ROMANIA

Graphic Processing

Project documentation

Moraru Alex-Mihai

Group 30431

Assistant teacher: Adrian Sabou

Table of contents

1. Subject specification

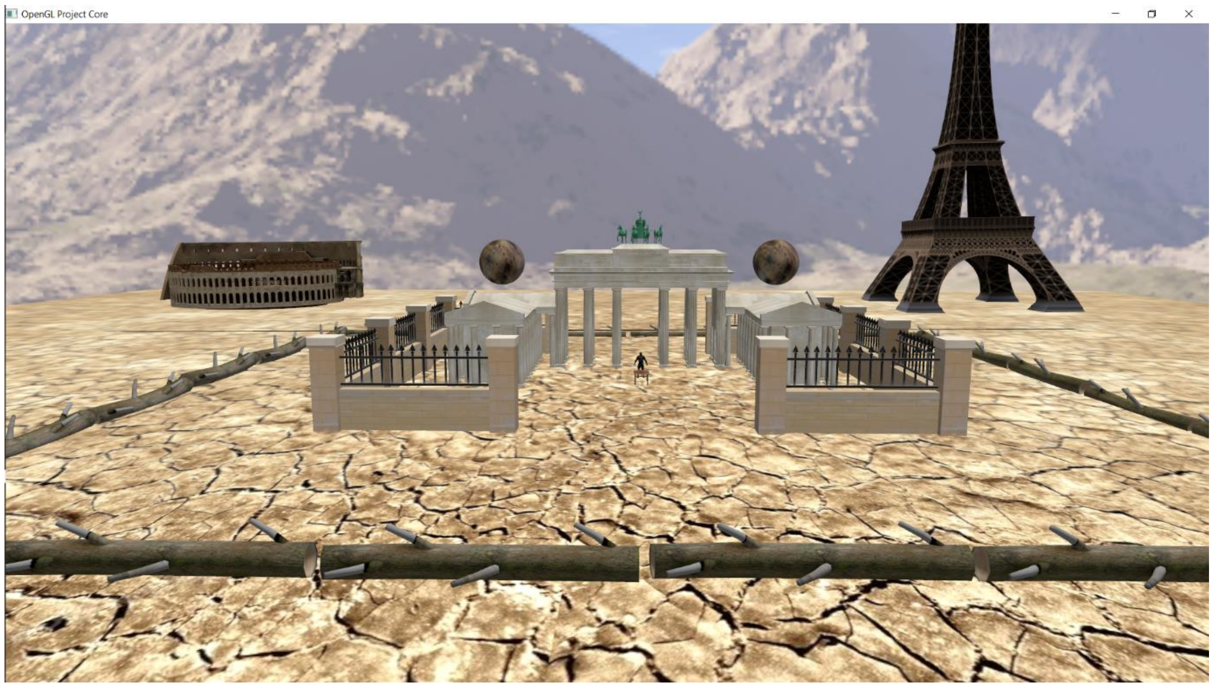
The subject of the project consists in the photorealistic presentation of 3D objects using OpenGL library. The user directly manipulates by keyboard the scene of objects.

The scene I chose to implement presents some real world buildings situated somewhere in a deserted area. The user is able to move through the scene with the help of the keyboard, in order to view it from different perspectives.

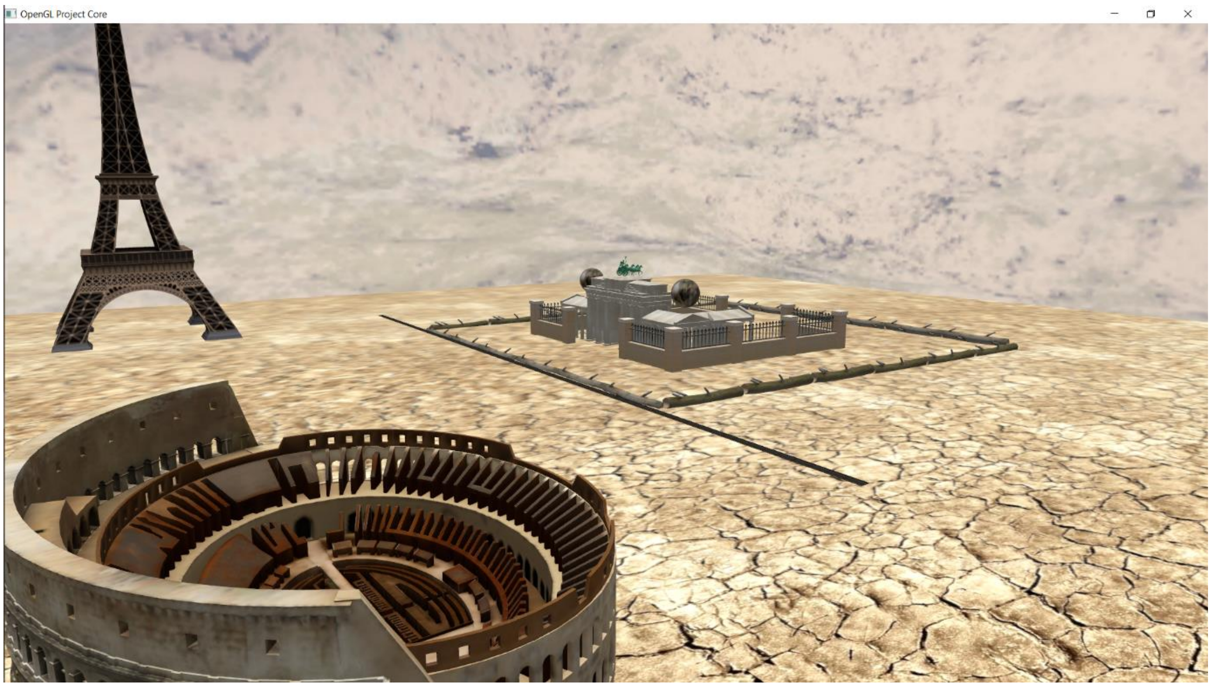
2. Scenario

2.1. Scene and objects description

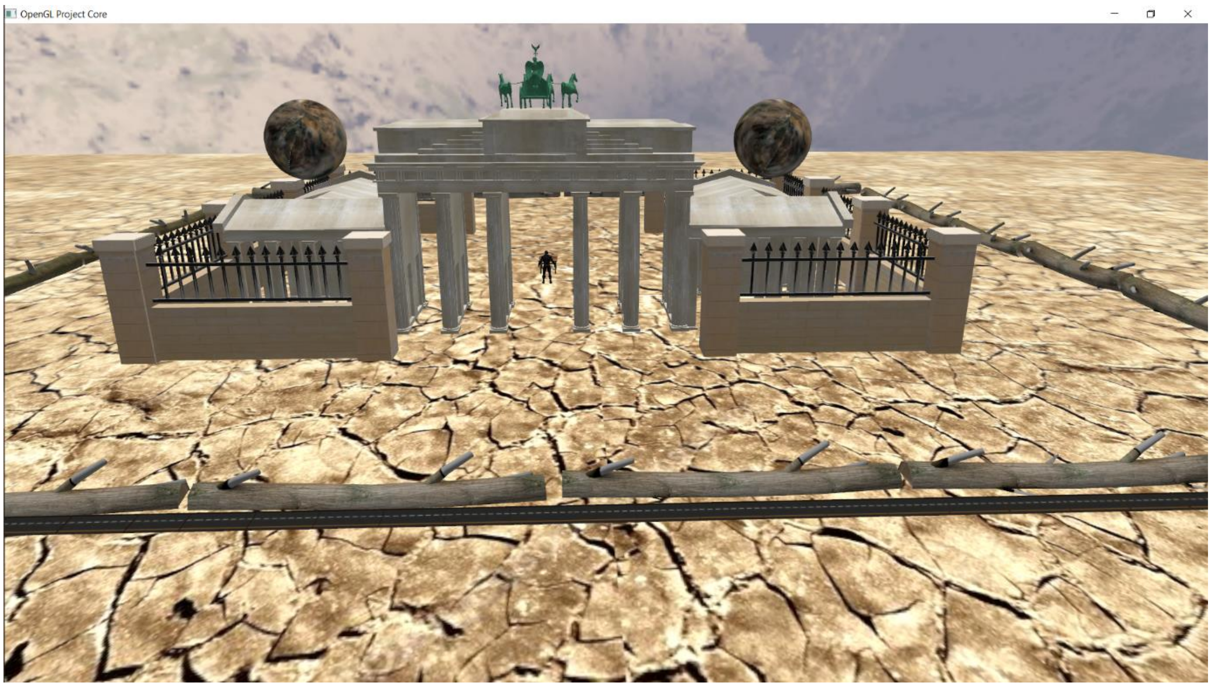
My scene resumes to a robot-like figure standing in front of the Brandenburg Gate. In his proximity one can notice a table with a bricked teapot on it. On the lateral top sides of the building there are two alien planets continuously spinning. The building is surrounded by walls and by tree trunks. In the far plane the Colosseum and the Eiffel Tower can be seen, with a road connecting them. Up in the sky, a plane can be noticed, as well as the mobile source of light.



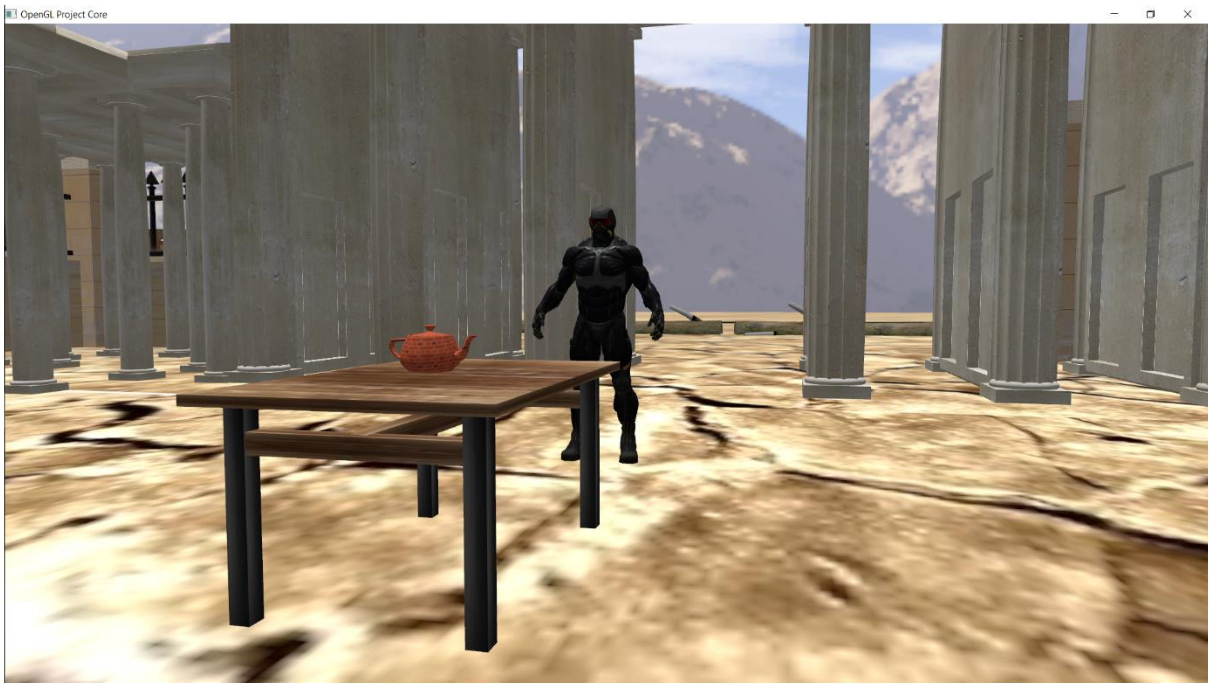
The view from the Colosseum's point:



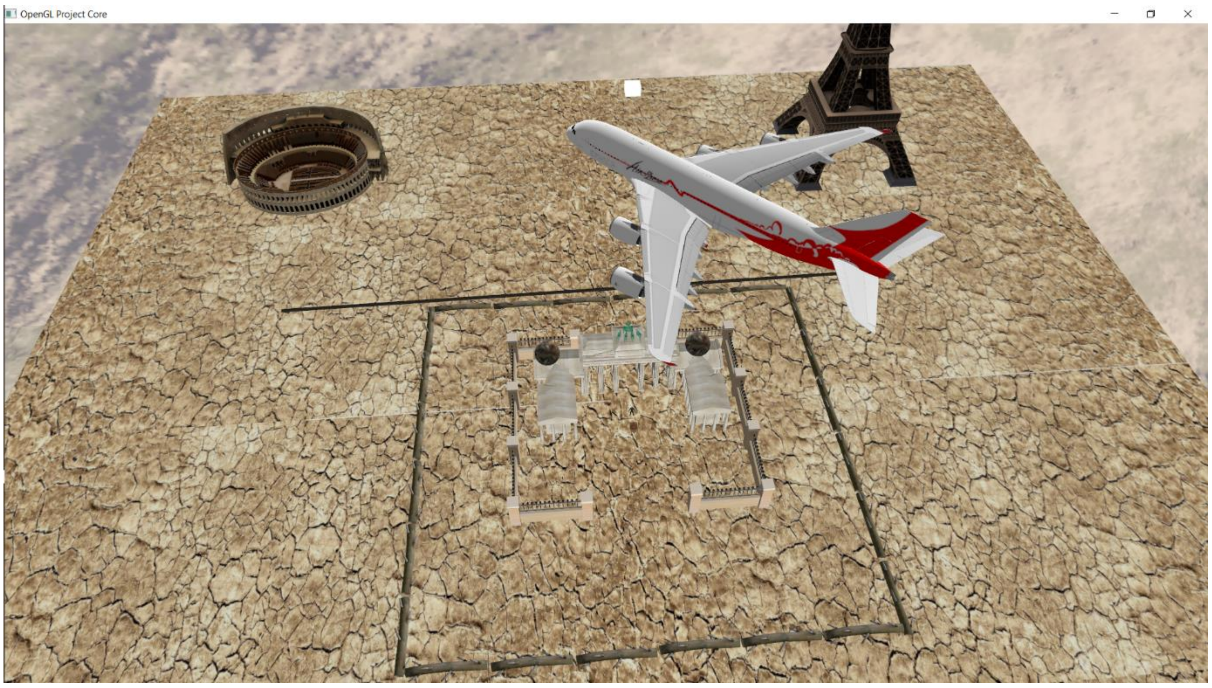
The view from behind the main scene:



The main scene:



The far up view:



2.2. Functionalities

The user is able to navigate through the scene in all three-dimensional directions. The scene can be viewed through a continuous animation too, which can be activated with the keyboard. The user can choose if collisions should be detected or not, through keyboard. These collisions only refer to not passing the margins (not leaving the ground).

Another feature is represented by the possibility to introduce fog into the scene, modifying its intensity.

I have introduced two sources of light: directional and spotlight. The spotlight focuses on the robot and can be turned on and off by the user, through the keyboard.

3. Implementation details

3.1. Functions and special algorithms

3.1.1. Possible solutions

In this project I used a lot of functions, some of which were already implemented, others implemented by me. All this functions contributed to the functionality of the project, but I could highlight the most important ones like this:

- renderScene() – it renders all the objects, placing them into the scene in different roles. It can also be used for computing shadows
- processMovement() – it deals with the keyboard data: assigns some actions to some of the pressed keys
- initModels() – loads all the models
- initShaders() – loads all the shaders
- initUniforms() – initializes some of the main components: model, view, projection

OpenGL has a lot of predefined functions. Some of the ones I used are:

- glUniformMatrix4fv() : for sending matrix data to shader
- glUniform1i() – activate texture
- glGetUniformLocation() – returns the location of a uniform variable
- glBindFramebuffer() – binds the specified frame buffer

3.1.2. Motivation of the chosen approach

I chose this approach because I wanted to follow the laboratory work. My project highly resembles the laboratory code. I was looking forward to taking it to the next level. To creating a whole scene. To put together all the puzzle pieces that we learned in the lab classes.

3.2. Graphics model

The most of the objects are taken from the free3d website and some of them (teapot, nanosuit) are taken from the laboratory work. I also consulted the link on moodle, from which I got my skybox. The majority of the objects have been imported into blender, so that I could properly apply the textures and eventually create a realistic scene.

3.3. Data structures

I did not use anything special in regard to data structures. An example would be the data structure used to store the sky box faces.

3.4. Class hierarchy

The project contains a few .c and .cpp files, all of them contributing to its functionality:

- main – holds the functions enumerated before
- Camera – contains functions used to manipulate the view side
- Window – initializes the window
- Model3D – fetches the objects
- Shader – manages the shaders
- SkyBox – fetches the skybox

4. **Graphical user interface presentation / User manual**

As I said before, the user is able to navigate through the scene and execute certain operations. The initial view is far enough from the scene so that the whole area can be seen.

The user manual looks like this:

W –	move front
S –	move back
A –	move left
D –	move right
R –	move down
T –	move up
1 –	start animation
2 –	stop animation
3 –	start fog
4 –	stop fog
5 –	increase fog
6 –	decrease fog
Q –	move planets left
E –	move planets right
N –	activate collisions
M –	deactivate collisions
C –	start spotlight
V –	stop spotlight
7 –	wireframe view
8 –	point view
9 –	flat view

5. **Conclusions and further developments**

In the end, I can conclude that it was a fun and useful experience, I struggled a lot but in the end I am proud of the results. This experience got me to like the graphics side of computer science even more. I learned the basics of OpenGL and I understood how the computation of lighting models is done.

Some further developments could consist in adding more objects, more animations and more special effects to the scene. Basically I think the main developments that can be made regard the realistic side of the project: shadow mapping, wind, rain, more lights and shadows for those lights.

6. **References**

- www.free3d.com
- https://moodle.cs.utcluj.ro/course/view.php?id=423
- https://wrf.ecse.rpi.edu/wiki/ComputerGraphicsFall2013/guha/Code/spotlight.cpp
- http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/#the-shaders
- https://learnopengl.com/Lighting/Light-casters