Graphic Processing

- Project Description -

Utiu Monica Iulia

Group 30433 – was in Erasmus

Content

1. Subject specification

The subject of this project is the photorealistic presentation of 3D objects using OpenGL library. Open Graphics Library is a specification of a standard which defines a multiplatform API, widely used for programming the 2D and 3D components of computer programs.

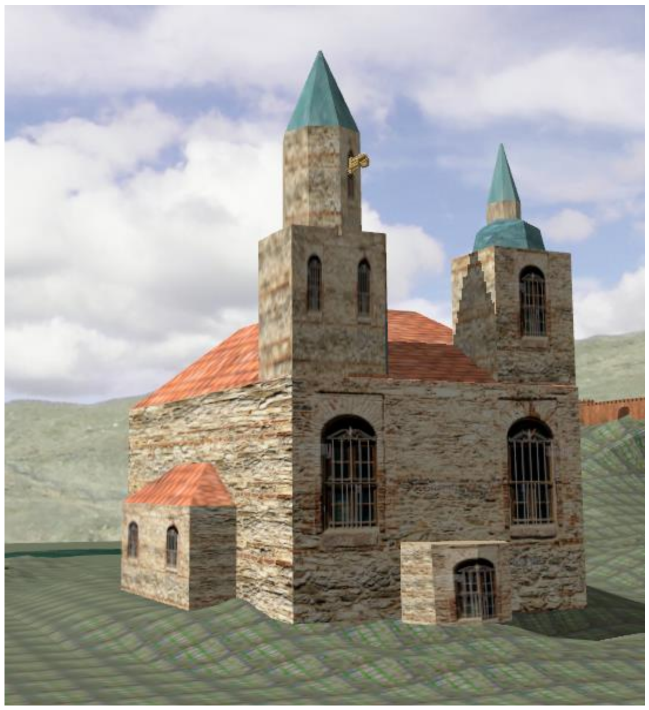
The scene I chose to implement is an ode to Krakow, where I spent my Erasmus this past semester and some of its main characteristics. The user manipulates the object scene directly by entering the mouse and keyboard and can move around it.

2. Scenario

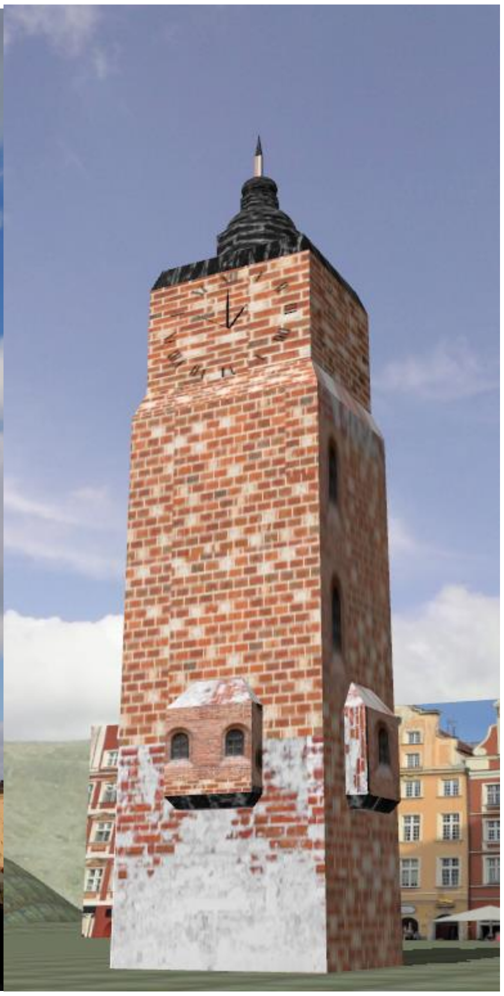
a. Scene and object description

The scene depicts a simplified version of Krakow.

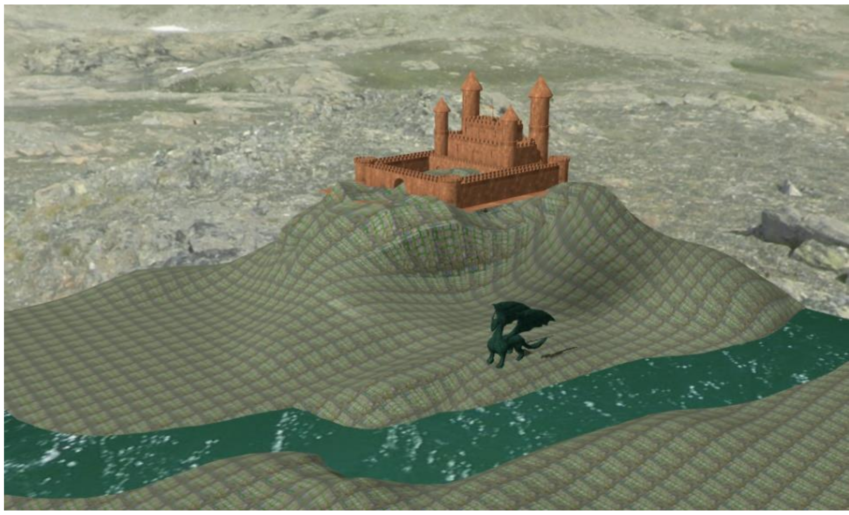
Some of the objects were constructed by me in Blender to mimic the base shapes of the representative buildings of the city center: The St Mary Basilica, The Barbican, the Tower with the clock, the river, and the European buildings. Others were imported from online sources: the castle for the Wawel Castle, the dragon to represent the legend of Smok- the dragon living under Wawel castle near the Wisla River.

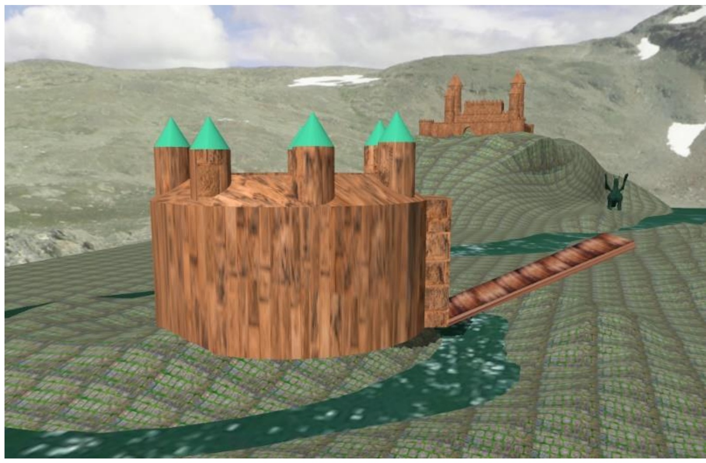


- St Mary’s Basilica -

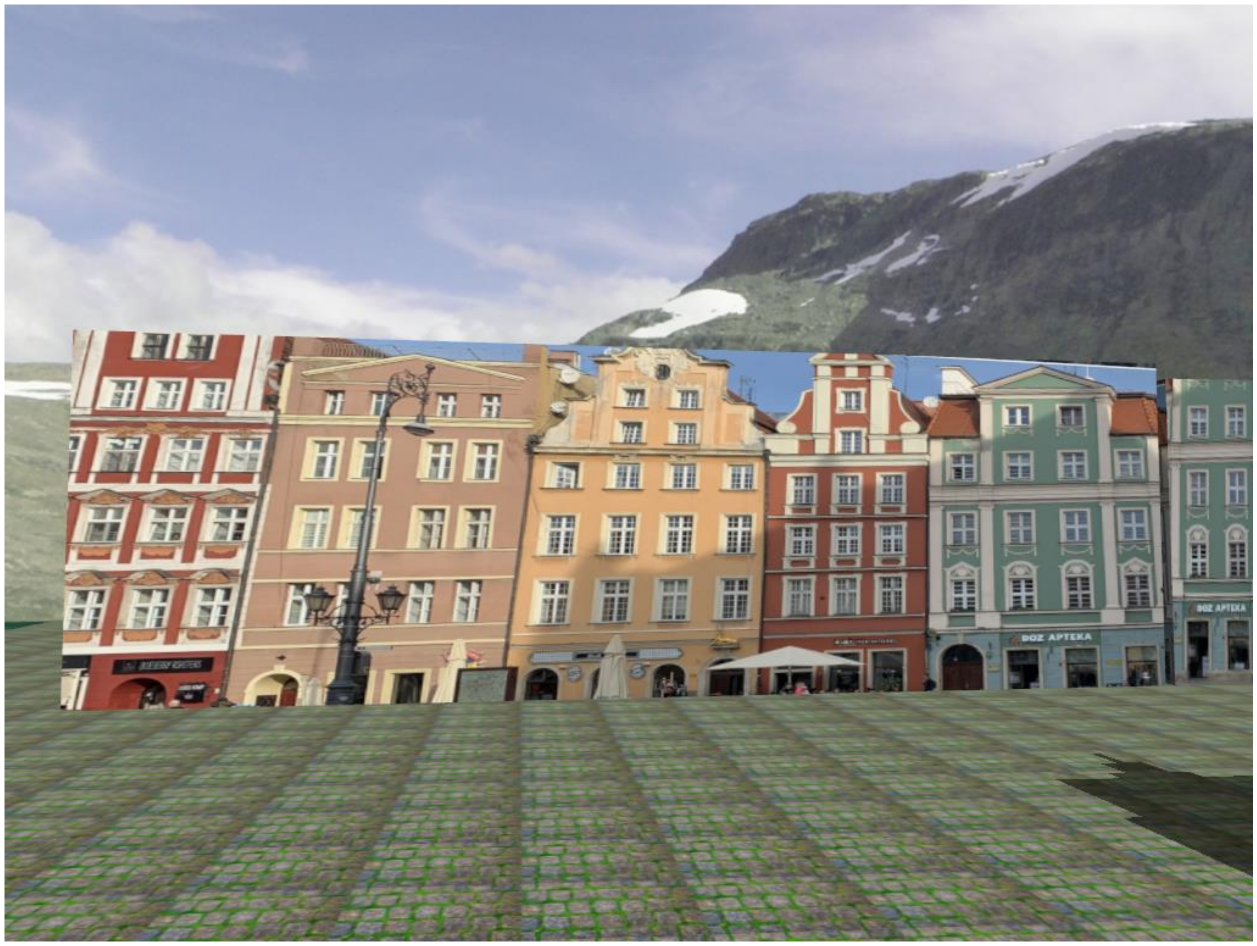


- Clock Tower –
- Wawel Castle and the Wisla river -





- Barbakan –
-



- European buildings created with the help of photos I took in Poland and Czech Republic that we're straightened (as much as I could) and cropped to create the feeling of the architecture without spending a lot of time creating the structural details of the objects –

The photos:

b. Functionalities

The user can have a visualization of the entire scene by pressing the keyboard keys or the mouse. One can also switch between solid view and wireframe or polygonal and smooth face using keyboard keys.

Fog also appears or disappear by the choice of the user. Its intensity can be increased or decreased.

There are multiple sources of light: directional light, the point light (on the Christmas tree in the main square). The point light can be turned on/off by pressing the corresponding keyboards.

The bridge of the Barbican swings up and down by itself, the trumpet does a similar motion to represent the periodic singing of the musician in the basilica at specific hours.

3. Implementation details

a. Functions and special algorithms

i. Possible solutions

OpenGL library consists in a large variety of functions such as:

- glViewport(...) used for setting Viewport transform
- glfwCreateWindow(...) used for creating the window
- glGenTextures(...)
- glBindTexture(...)
- glTextImage2D(...) used for creating depth texture for the Framebuffer objects
- many others.

One of the functions I've used in this project is void renderScene() which is the function used to send all the data to the shaders and compute all the values. Here is where all the models and their shadows are created, where the normal matrix are computed and where the rotation, translation and scaling are done.

Another function is void initUniforms(), where the lights' details are all set – the point light, the directional light.

The function void initShaders() is a function where the shaders are instantiated.

The function void initModels() is a function where all the 3D Models are instantiated.

Other important functions are:

- void processMovement()
- void move(gps::MOVE_DIRECTION direction, float speed)
- void mouseCallback(GLFWwindow* window, double xpos, double ypos)
- void keyboardCallback(GLFWwindow* window, int key, int scancode, int action, int mode)

These functions perform all the necessary changes on the models or on the camera so that they perform the moves desired by the user.

The mouseCallBack function is implemented by using Euler Angles.

A mathematics library tailored for OpenGL – GLM – is also used in order to implement the functions and the algorithms.

ii. The motivation of the chosen approach

In the laboratories dedicated to this subject I had on hand there where tutorials of how to compute light, shadows, to perform operations of translation, scaling and rotation on different objects. Having this starting point and also a skeleton of the project, the work I had to do in carrying out this project had a fairly well-defined path.

b. Graphics model

Some objects and textures have been downloaded from the internet. Most objects were imported into the Blender to be edited and placed in the final scene.

Almost all the textures were edited to comprise pictures for all the elements of an object and mapped in Blender to their specific zone, or multiplied for a better texture in the case where I couldn't just rescale the objects on the UV Map because they would overlap an unwanted zone.

I liked working in Blender with creating the objects and the textures, although I opted for simple designs because I did not have experience in it and was also scared about the numbers of polygons in my final scene.

All the objects taken from the internet (castle, dragon, trumpet) were decimated with a modifier to decrease the number of polygons. The same with the planes(ground,water).

c. Data structures

The only data structures required were some simple ones, to hold together all the attributes of the different types of light, as for point light. They have already been implemented in the project provided to the laboratory, so it was quite useful to have it. In addition, I added some functions to calculate the lights and fog.

d. Class hierarchy

- Camera: contains the implementation for camera movement (up, down, front, back, left, right)
- SkyBox: contains skybox implementation (load, draw, load textures, initialize)
- Mesh: represents a 3D object; includes the following data structures:
- Vertex (position, normal vector and texture coordinates)
- Texture (id, type and path); Material
- Model3D: contains methods for printing meshes using a specified shader program
- Shader: contains methods for creating and activating shader programs

4. User manual

The user can interact with the scene using the following keys on the keyboard:

- 1 – fog intensity step (from none to max then to none again)
- 2 – wireframe view
- 3 - point view
- 4 – solid view
- 5 – on/off point light on tree/ city center
- N – change skybox to night/day
- A – move camera to the left
- W – move camera to the front
- S – move camera backwards
- D – move camera to the right
- Z - rotate the camera to the left
- C - rotate the camera to the right

5. Conclusions and further developments

In conclusion, I would like to say that I found it quite difficult to work with a Blender because I did not deepen the work in this program. I think it’s a very powerful tool and you need many hours in it to master it, but I would like to work with it in the future for animations

Referring to the code part, it was more complicated than I expected. I spent a lot of time trying to understand certain code structures given in the lab.

This project required a lot of work, attention and especially time. I consider it to be one of the most difficult projects I have done so far.

As further developments, I would’ve liked:

- To have made the architecture more complex and create an authentic and real feel of Krakow so I could immerse in it when I miss it
- To have made the clock animation work
- To have synchronized the trumpet animation with the clock hour
- To animate the dragon
- Better, more complex lighting
- Etc.

6. Refrences