Graphical processing systems - Final Project
- Graveyard 3D scene -

Horvath Andrea - Anett

13/01/2021

Contents

1 Subject specifications

The subject of the project consists in the photo-realistic presentation of 3D objects using OpenGL library. The chosen theme for the 3D representation is a graveyard. A series of animations and effects are also present in the project to enhance the photo-realism and to induce a proper atmosphere of anxiety, commonly created by graveyards.

2 Scenario

2.1 Scene and objects description

The scene contains the representation of a graveyard. The fenced-in area presenting the core of the scene is located in a remote desolate place between hills. In the enclosed region a chapel can be observed together with some graves and statues. Also some detail objects were added in the scene to create some complexity and realism for the scene.

A great part of the static part of the scene was created in Blender using various objects and exported as a single *.obj* file. This part contains the following objects:

- ground plane
- road cube
- *gate arch*
- *iron gate*
- *fence* (29 pieces)
- *fence column* (28 pieces)
- paved alley plane
- *chapel*
- *graves* (3 groups of graves)
- *dead tree - front*
- *dead tree - back*
- *angel statue*
- *tombstone statue*

Other objects were added individually:

- skydom sphere
- *bench*
- *lamp*
- *crow* - on parts(i.e. body, left wing and right wing)
- raindrops

Some overview images of the scene are presented in figures 1-4 at the end of the chapter.

2.2 Functionalities

The functionalities of the project are the following:

- visualization if the scene with camera movements using keyboard and mouse (mouse for rotations and keys W, S for moving forward and backward, A, D for moving left and right);
- visualization of the scene using animation (key T);
- commute between directional light and point light from the lamp (key L);
- commute between solid and wireframe visualization of objects (key M);
- commute between polygonal and smooth visualization of surfaces (key N);
- visualization of shadows;
- animation of crow (key C to make crow fly);

- visualization of fog effect (key F to disable/enable effect);
- visualization of rain (key Z to disable/enable effect);
- visualization of wind (key X to disable/enable effect; visible only when rain is enabled);
- collision detection (visible when rain is enabled; raindrops cannot pass through the roof or the walls of the chapel).

Images presenting some of the effects are presented in figures 5-11.

Figure 1: Overview image - 01

Figure 2: Overview image - 02

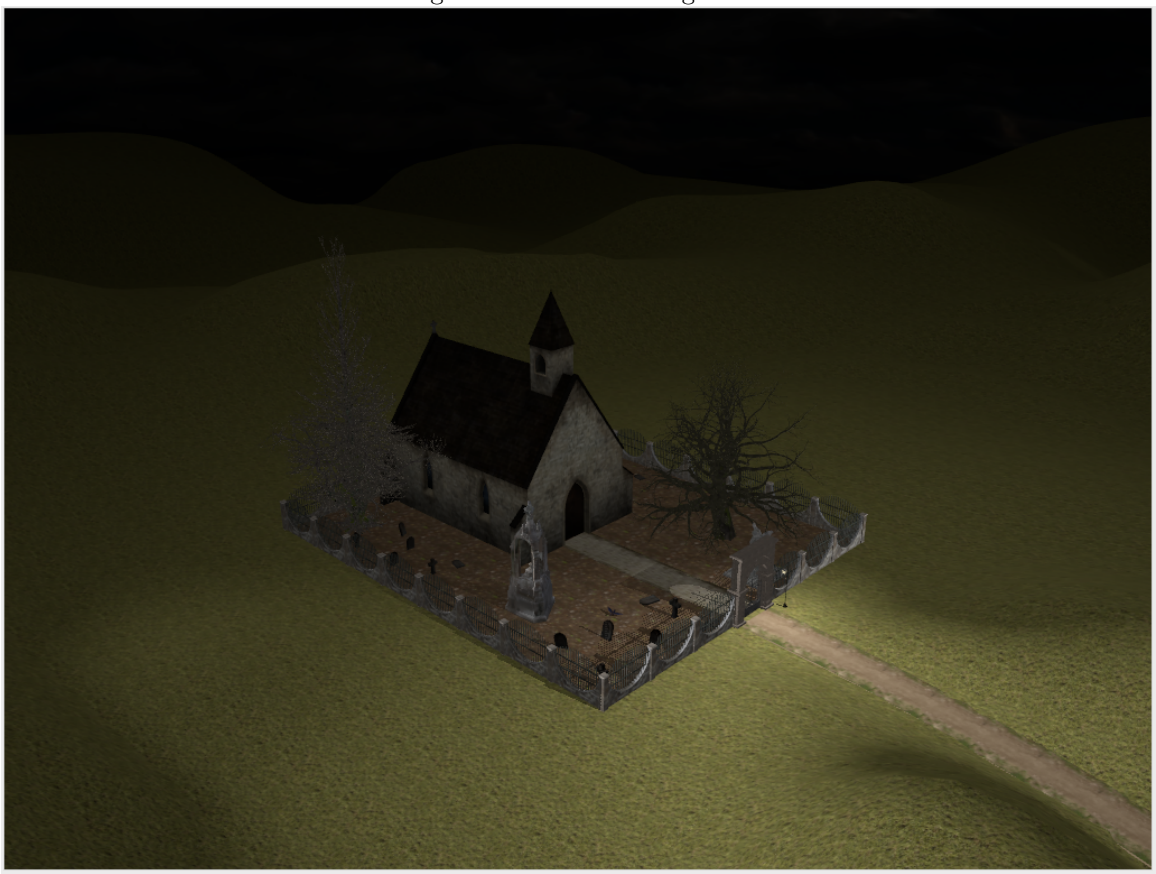


Figure 3: Overview image - 03

Figure 4: Overview image - 04

Figure 5: Directional light

Figure 6: Point light

Figure 7: Shadows

Figure 8: Fog effect

Figure 9: Rain

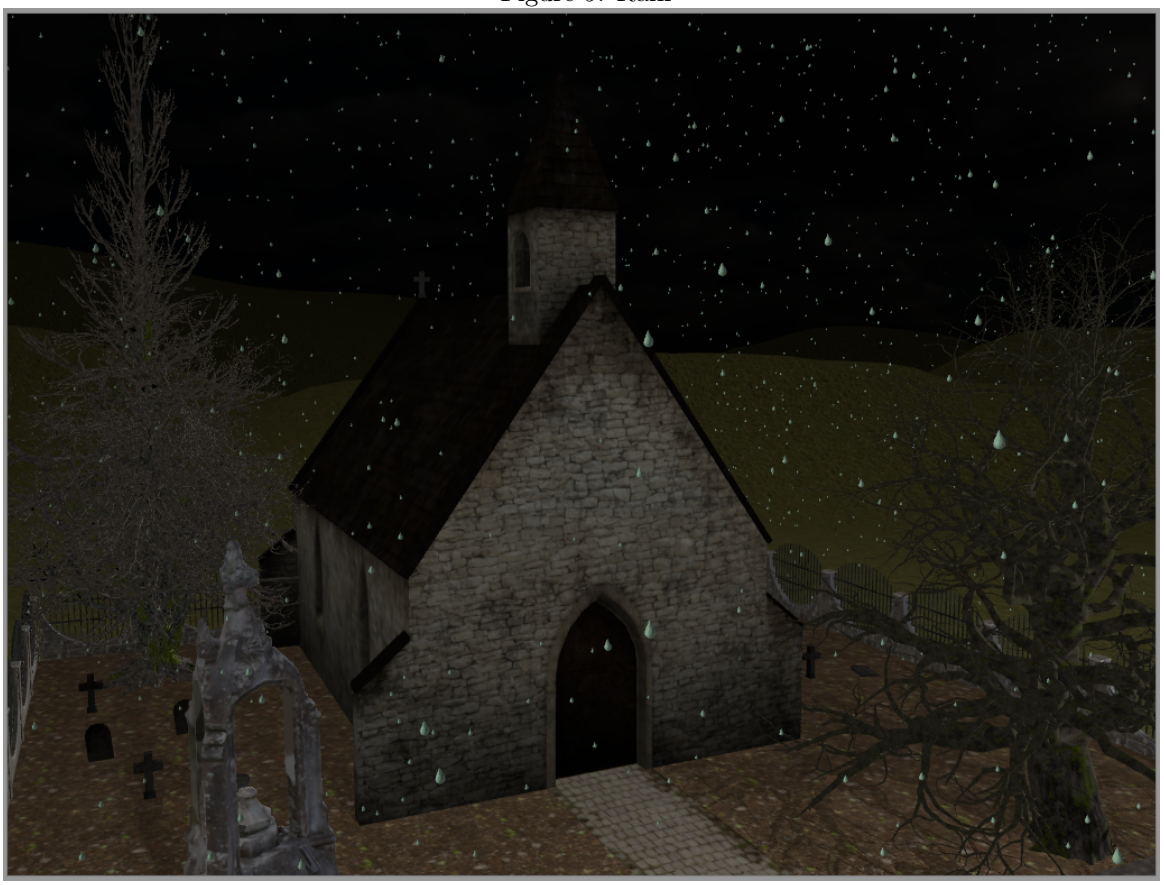


Figure 10: Wind effect

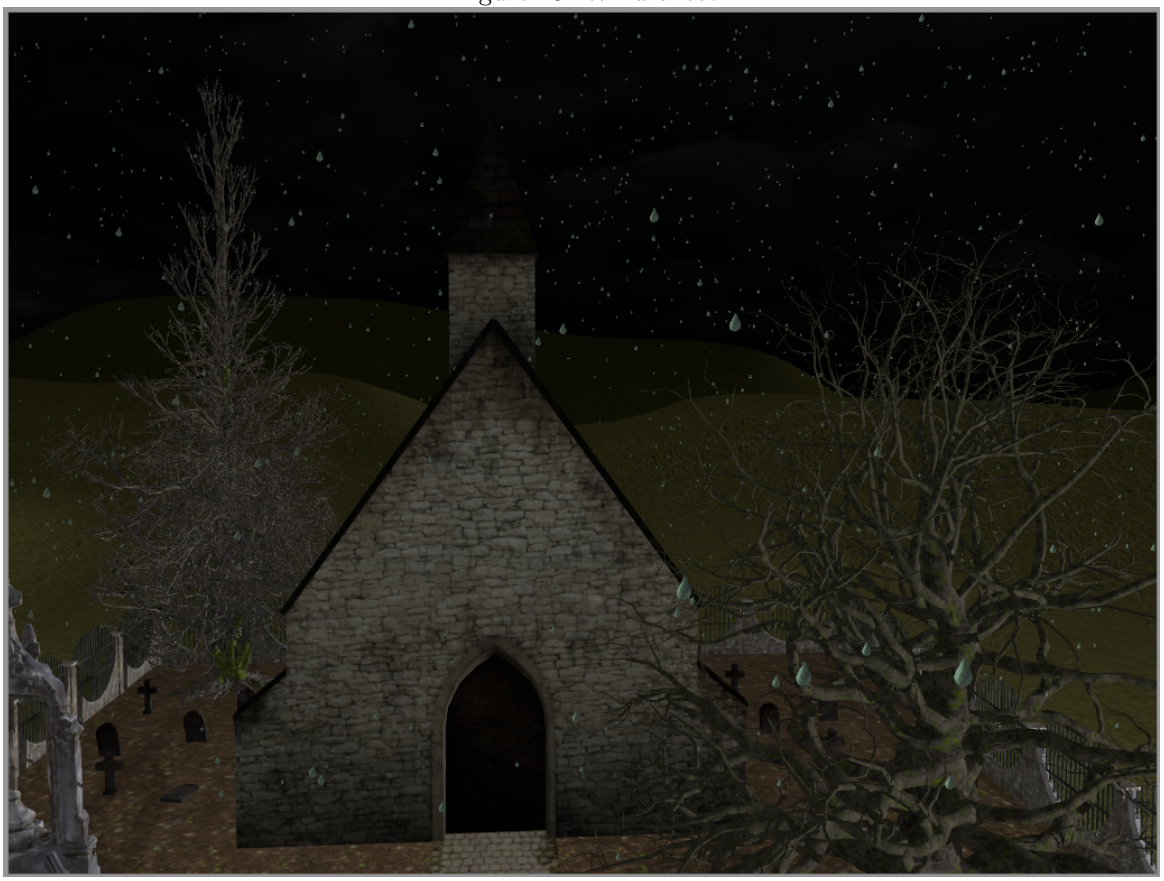
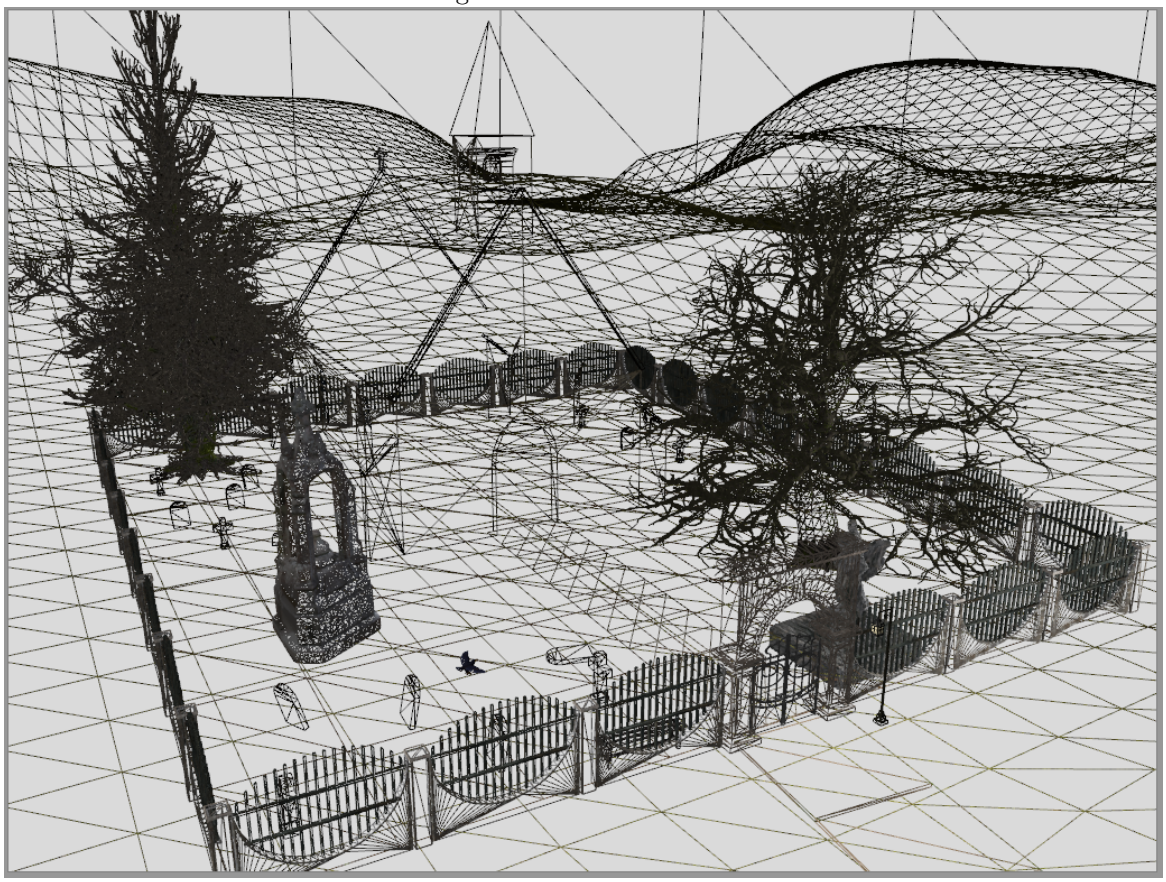


Figure 11: Wireframe view



3 Implementation details

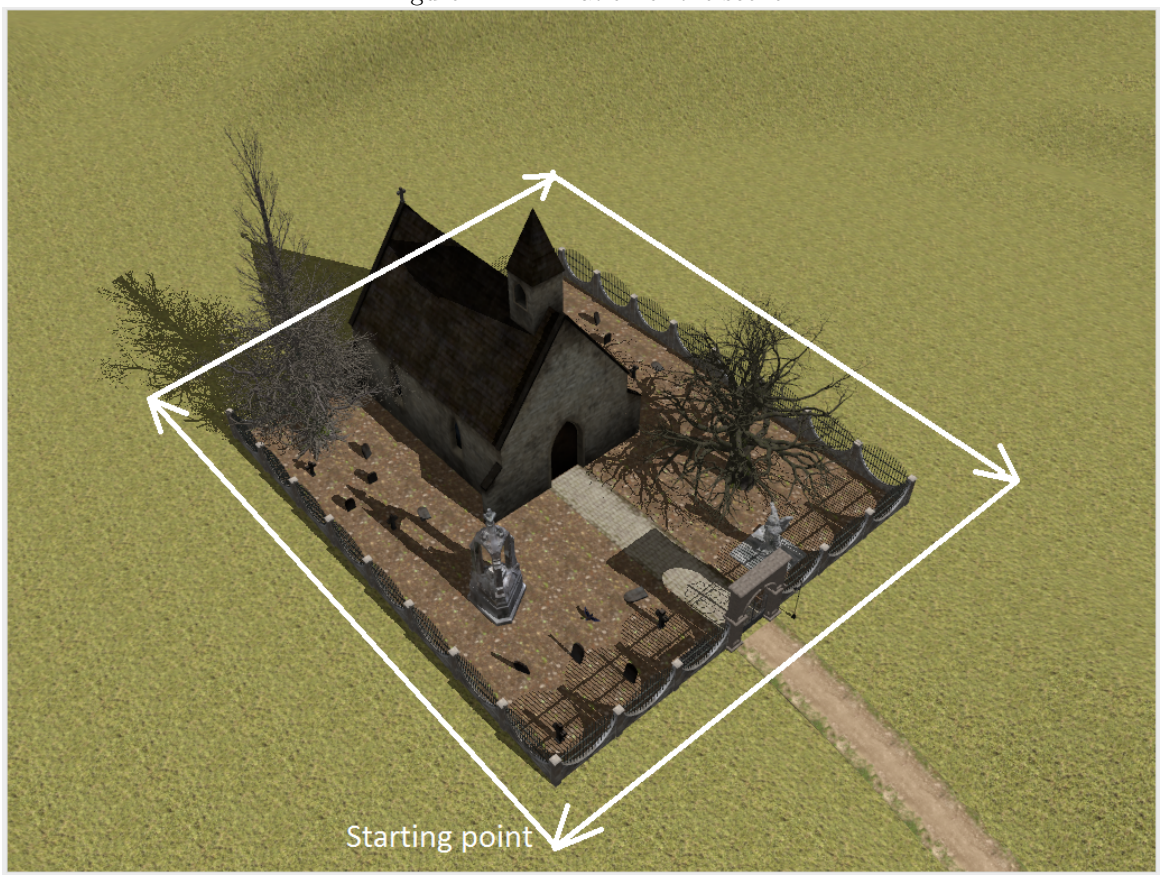
3.1 Functions and special algorithms

For improving the complexity of the scene, a series of functions an algorithms were used.

3.1.1 Animation of the scene

The animation of the scene was done using an array of positions along a predefined path surrounding the graveyard in the shape of a rectangle at a given height, parallel with the ground. The camera was then moved through all this positions and the scene was rendered in each of them, obtaining in this way a tour of the scene.
A description of the functionality is presented in figure 12.

Figure 12: Animation of the scene



3.1.2 Animation of an object - crow

The animation of the crow was made on parts. The body and the wings were animated separately. The body of the crow executes a translation from point to point along the diagonal in Y-Z plane. The wings were animated separately. They execute a rotation and the same translation as the body. The rotation is done incrementing the angle step by step until a certain angle for the upward rotation and then starts the reverse rotation until a certain angle for the downward rotation.
The functionality is presented in figure 13.

Figure 13: Animation of crow

3.1.3 Rain effect

The rain is implemented in the scene by rendering a big number of raindrop objects. When enabling the rain, a function initializes the positions of the raindrops randomly in a specific area. When rendering the rain, the raindrops are translated downward along the Y axis until they hit the ground or the roof/walls of the chapel, when they translate at the highest position on Y axis, keeping their initial X and Z coordinates.

3.1.4 Wind effect

The wind effect can be enabled during the rain. Besides the normal movement of the raindrops, a translation along Z axis is also added, such that the raindrops will have a diagonal trajectory.

3.1.5 Collision detection

The collision detection was implemented in the context of the rain effect. The aim was that the raindrops cannot pass through the roof and the walls of the chapel respectively. For this, a function *checkCollision()* was implemented to signal when the translation of the raindrops have to end, i.e. they hit the roof or a wall.

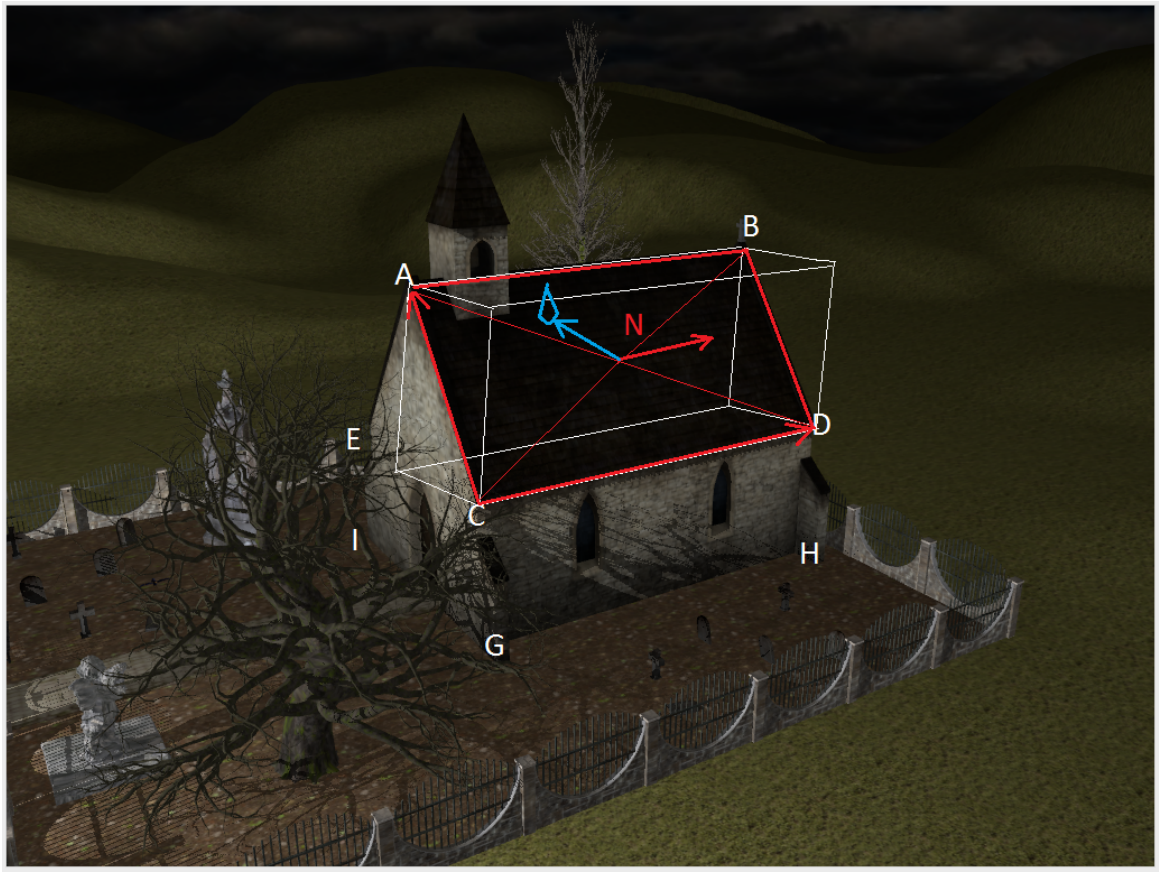
The following steps are done for each plane of the chapel (left and right side of the roof, left and right wall):

- compute a bounding box for plane;
- compute normal vector of plane using cross product;
- check whether the position of the raindrop is in the box;

- check whether the position of the raindrop relative to the plane is under it using dot product between a position vector and the normal;
- if the previous two conditions are met, a collision between the raindrop and the plane is detected and the translation of the raindrop is stopped.

The algorithm for the right side of the roof is visible in figure 14.

Figure 14: Collision detection



3.2 Graphics model

The scene was built using the *polygonal modeling* approach.

3.3 Class hierarchy

The classes included in the project are the following.

3.3.1 *Camera.hpp*

The class contains as data the following:

- *cameraPosition*
- *cameraTarget*
- *cameraFrontDirection*
- *cameraRightDirection*

- *cameraUpDirection*
- *yawAngle*
- *pitchAngle*

The functions of the class are:

- *getViewMatrix()*
- *move(...)*
- *rotate(...)*

3.3.2 *Mesh.hpp*

The class contains as data the following:

- *vertices*
- *indices*
- *textures*
- *buffers*

The functions of the class are:

- *getBuffers()*
- *Draw(...)*
- *setupMesh()*

3.3.3 *Model3D.hpp*

The class contains as data the following:

- *meshes*
- *loadedTextures*

The functions of the class are:

- *LoadModel(...)*
- *Draw(...)*
- *ReadOBJ(...)*
- *LoadTexture(...)*
- *ReadTextureFromFile(...)*

3.3.4 *Shader.hpp*

The class contains as data the following:

- *shaderProgram*

The functions of the class are:

- *loadShader(...)*
- *useShaderProgram()*
- *readShaderFile(...)*
- *shaderCompileLog(...)*
- *shaderLinkLog(...)*

4 User manual

In order to run and test the application one mus follow the next steps.

1. launch the provided executable file;
2. test the functionalities as presented in section *3.2 Functionalities*.

5 Conclusions and future developments

In conclusion, a photo-realistic representation of a 3D scene using OpenGL library was successfully developed. As future improvements the following can be considered:

- shadows for the point light;
- collision detection for the other objects in the scene;
- collision detection for the camera;
- Bezier curves for the movement of the crow or the animation of the camera.

6 References

6.1 Objects

1. *Arch* - https://sketchfab.com/3d-models/arch-743422dda7db46a081dcd48ae503da6c
2. *Iron gate* - https://sketchfab.com/3d-models/portao-antigo-2e1b9478831c46f7af08c56975a75235
3. *Fence* - https://sketchfab.com/3d-models/fence-a-1d9f74c87f254361aee933a984c64b3f
4. *Chapel and graves* - https://www.turbosquid.com/3d-models/free-chapel-celtic-video-3d-model/464431
5. *Dead tree front* - https://sketchfab.com/3d-models/old-mossy-tree-6f7f392accb54094ac1de1f2eec427c2
6. *Dead tree back* - https://free3d.com/3d-model/some-dead-trees-32880.html
7. *Angel statue* - https://sketchfab.com/3d-models/indian-springs-angel-figure-3a27d5d61b5342049896e6c9a2e34059
8. *Tombstone statue* - https://sketchfab.com/3d-models/bonaventure-cemetery-tombstone-scan-3-736c2de7194d478086160
9. *Bench* - https://sketchfab.com/3d-models/bench-02-815332bd71624fa29ed204f4bc969fed
10. *Lamp* - https://www.turbosquid.com/3d-models/free-street-light-3d-model/932849
11. *Crow* - https://sketchfab.com/3d-models/hello-neighbor-crow-bf711e537ff54be3b14eb6a767ff0896

6.2 Others

1. *Lights* - https://learnopengl.com/Lighting/Multiple-lights
2. *Shadows* - https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping
3. *Camera* - https://learnopengl.com/Getting-started/Camera
4. *Anti-aliasing* - https://learnopengl.com/Advanced-OpenGL/Anti-Aliasing
5. *Collision detection* - https://learnopengl.com/In-Practice/2D-Game/Collisions/Collision-detection