UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Graphic Processing

Anca-Maria Giurgiu

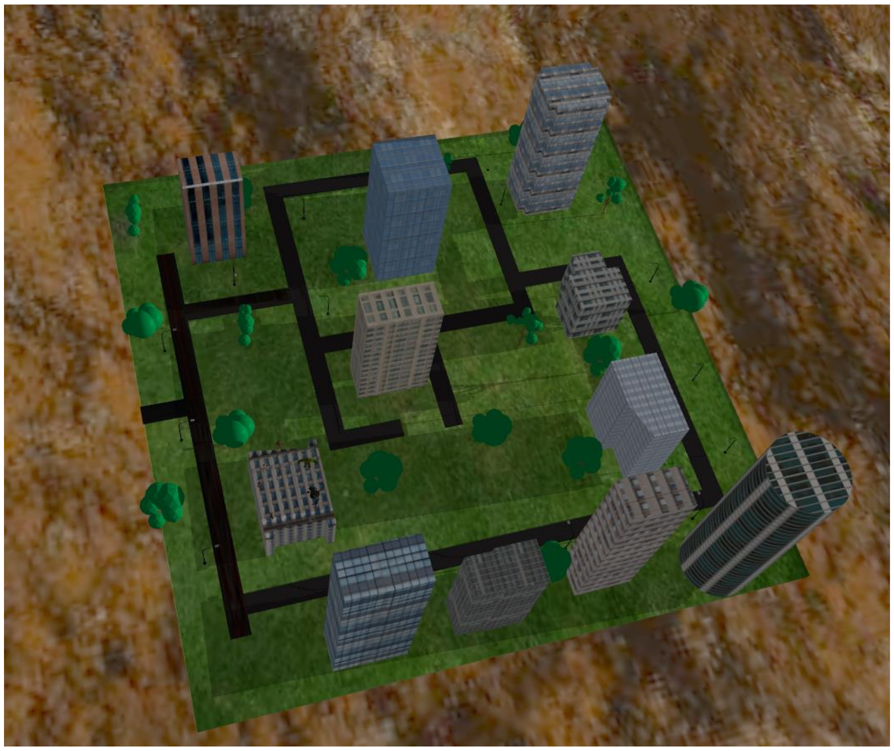
Group 30434

Content:

1. Project overview

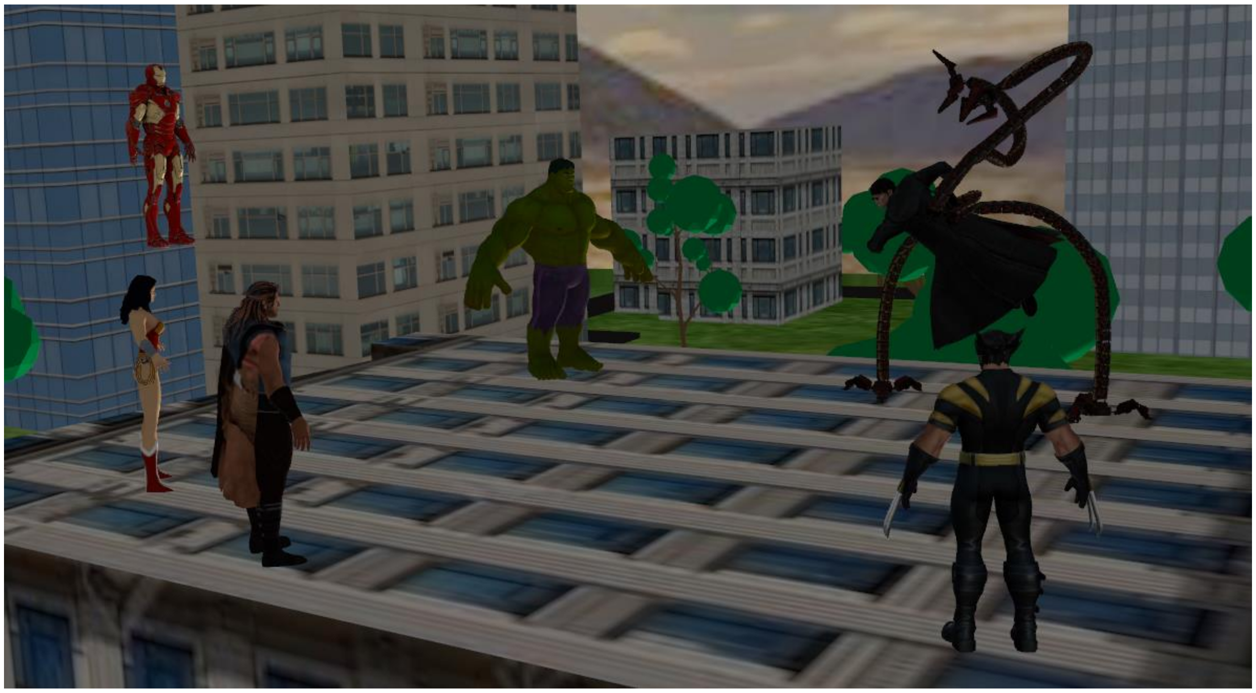
The aim of this project is to create a realistic 3D rendering of a town using the OpenGL library. OpenGL is a cross-platform standard that defines an application programming interface (API) for 2D and 3D graphics programming. The town I have designed is a modern urban area with skyscrapers, bridges, roads, and parks. The user can interact with the 3D scene using the mouse and keyboard and explore the town from different perspectives.

2. Scenario

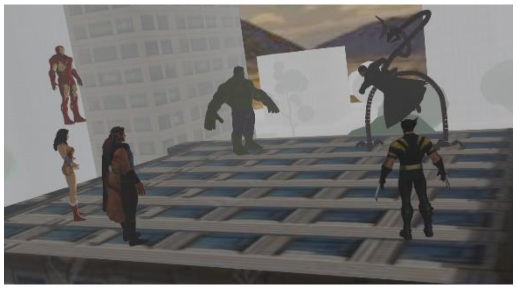


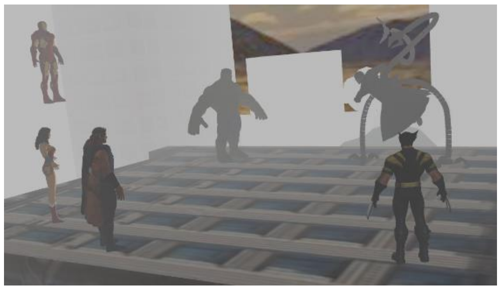
The image shows a 3D model of a modern city, with various skyscrapers and green spaces. The city is surrounded by dark roads and pathways that form a grid pattern. The buildings are grey-scale and have different heights and shapes. Some small, round trees add a touch of nature to the urban setting. The view is from above, giving a clear overview of the city layout. The background texture is a rough surface, with no sky visible.

I also included a fight scene between superheroes.

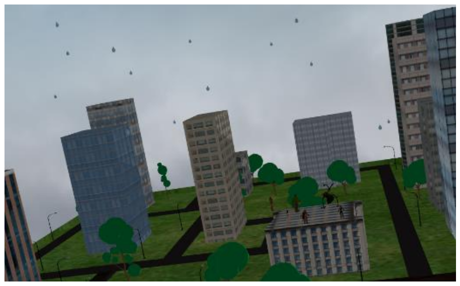
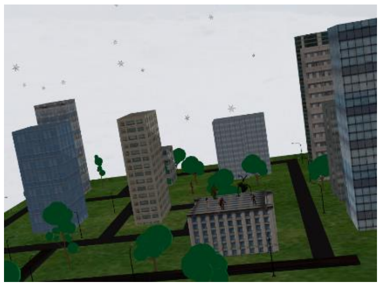


The user can have a visualization of the entire scene by pressing the keyboard keys or the mouse. One can also switch between solid view and wireframe or polygonal and smooth face using keyboard keys. Fog also appears or disappear by the choice of the user. Its intensity can be increased.

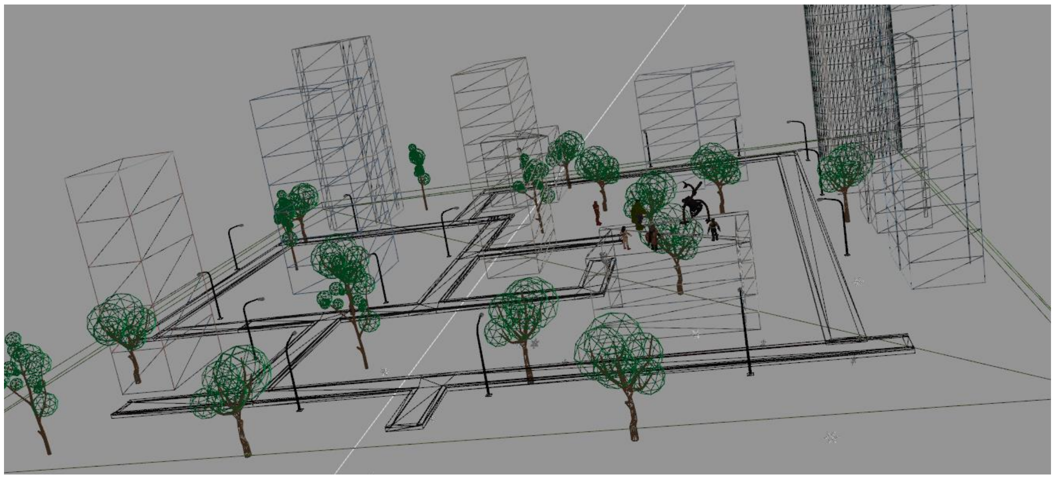




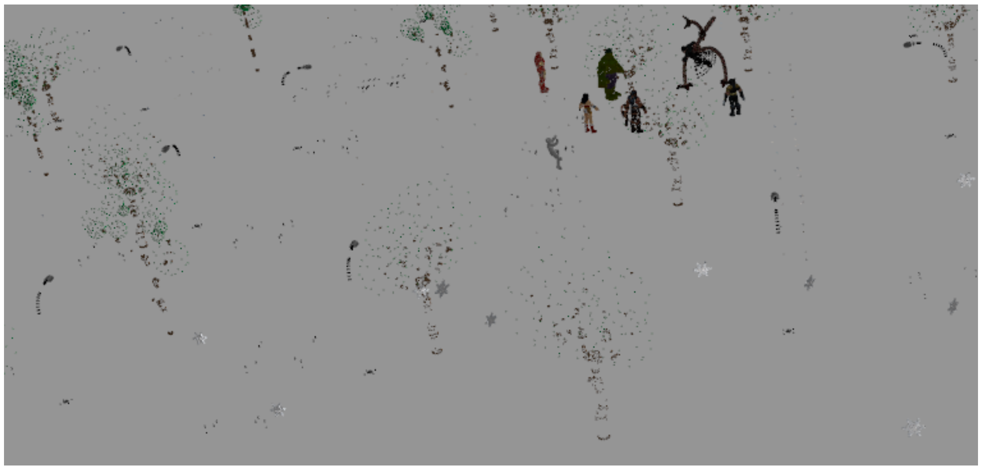
By pressing the u key, you can change the skybox. If you are in the first skybox, there is a sun that rises and sets. If you are in the second skybox and press the s key, it will rain. If you are in the third skybox and press the s key, it will snow.

If the user presses the M key, the program sets the polygon mode to GL_LINE, which means that only the edges of the polygons are drawn as lines.

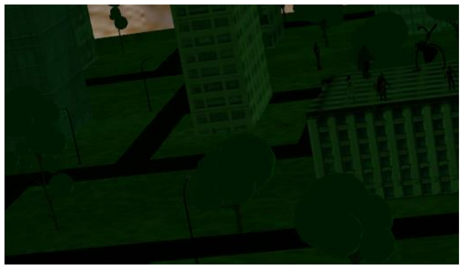


If the user presses the N key, the program sets the polygon mode to GL_POINT, which means that only the vertices of the polygons are drawn as points.



If the user presses the B key, the program sets the polygon mode to GL_FILL, which means that the polygons are filled with solid colors.

If the user presses the I key, the light color of the scene is changeing. The program uses a switch statement to cycle through four different light colors, each represented by a glm::vec3 object. The program also uses the glUniform3fv function to pass the light color to the shader program. The variable control keeps track of the current light color index. The program sets bNoapte to false after changing the light color.



3. Implementation details

Using the OpenGL library, I implemented various functions for this project, such as: glViewport(…) to set the Viewport transform, glfwCreateWindow(…) to create the window, glGenTextures(…), glBindTexture(…), glTextImage2D(…) to create the depth texture for the Framebuffer objects. One of these functions is void renderScene(), which sends all the data to the shaders and calculates all the values. This is where I create all the models and their shadows, compute the normal matrix, and perform the rotation, translation, and scaling. Another function is void initUniforms(), which sets the details of the lights – the point light, the directional light, and the spotlight. The function void initShaders() instantiates the shaders, and the function void initModels() instantiates the 3D Models.

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 make the necessary changes to the models or the camera according to the user's input. The mouseCallBack function uses Euler Angles. I also used a mathematics library designed for OpenGL – GLM – to implement the functions and the algorithms.

4. 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)

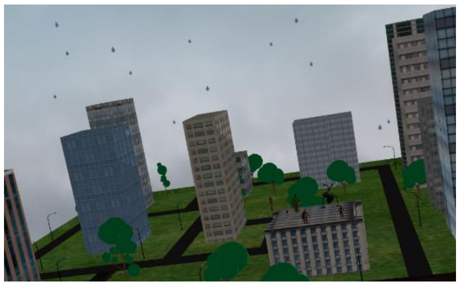
Texture (id, type and path); Material

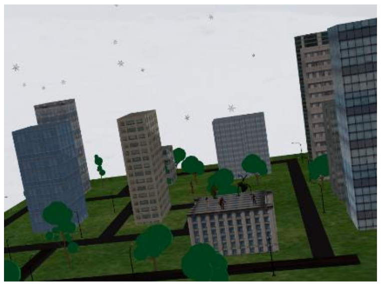
Model3D: contains methods for printing meshes using a specified shader program

Shader: contains methods for creating and activating shader programs

5. Animations

As animations, I have the sun that rises and sets, the rain and snow that fall, and Ironman that floats up and down.





6. References

https://learnopengl.com/

https://www.udemy.com/course/blender-environments/?gad_source=1&gclid=Cj0KCQiAtaOtBhCwARIsAN_x-3KSvK7QFJPQM47nCOmMRXk6Ls6eny09X2VxvkXIPESY4a1X2L64O90aAsjcEALw_wcB&matchtype=b&utm_campaign=LongTail-New_la.EN_cc.ROWMTA-B&utm_content=deal4584&utm_medium=udemyads&utm_source=adwords&utm_term=_._ag_101378300380_._ad_635753873653_._kw_blender+course_._de_c_._dm__._pl__._ti_kwd-346507641812_._li_1011806_._pd__._

https://www.youtube.com/watch?v=Rqhtw7dg6Wk

https://www.humus.name/index.php?page=Textures

https://moodle.cs.utcluj.ro/course/view.php?id=613

https://sketchfab.com/cimec/collections/obiecte-scolare-de-altadata-8402d6ec89814dacae0237a23df86fa3

https://free3d.com/3d-models

https://www.cgtrader.com/free-3d-models

https://www.turbosquid.com/3d-model/free