



UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

Scena fotorealista a unui parc

Proiectare Grafica

Zubascu Ileana
CTI an 3 2024

FACULTATEA DE AUTOMATICA
SI CALCULATOARE

15 Ianuarie 2024

Cuprins

1 Prezentarea temei	2
1.1 Introducere	2
2 Scenariul	2
2.1 Descrierea scenei și a obiectelor	2
2.2 Funcționalități	3
3 Detalii de implementare	3
3.1 Funcții și algoritmi	3
3.1.1 Soluții posibile	3
3.1.2 Motivarea abordării alese	4
3.2 Modelul grafic	4
3.3 Structuri de date	5
3.4 Ierarhia de clase	5
4 Prezentarea interfeței grafice utilizator / manual de utilizare	5
5 Concluzii și dezvoltări ulterioare	7
6 Referințe	7

1 Prezentarea temei

1.1 Introducere

Proiectul are ca și scop realizarea unei prezentări fotorealiste a unei scene de obiecte 3D utilizând librăriile prezentate la laborator (OpenGL, GLFW, GLM, etc.). Utilizatorul trebuie să aibă posibilitatea de a controla scena prin intermediul mausului și tastaturii.

Scena aleasa este reprezentarea unui parc de-a lungul unui rau în care se află multiple obiecte, având un leagan care poate fi manipulat de utilizator.

2 Scenariul

2.1 Descrierea scenei și a obiectelor

In realizarea parcului, am folosit multiple obiecte de tip .obj descarcate din Free3D (banci, felinare, pod, barca, fontana). Unele obiecte au fost modelate de către mine în Blender (leagan, pavaje, copaci, cosuri de gunoi). Pentru fiecare obiect, am aplicat o textură unică în Blender, potrivita pentru un nivel de realism mai ridicat. Am folosit skydome pentru proiecțarea cerului.



Figura 1: Scena prezentată în ansamblu din Blender

Pentru realizarea copacilor, am folosit două plane la care le-am aplicat o textură pentru reducerea numărului de varfuri și fețe.

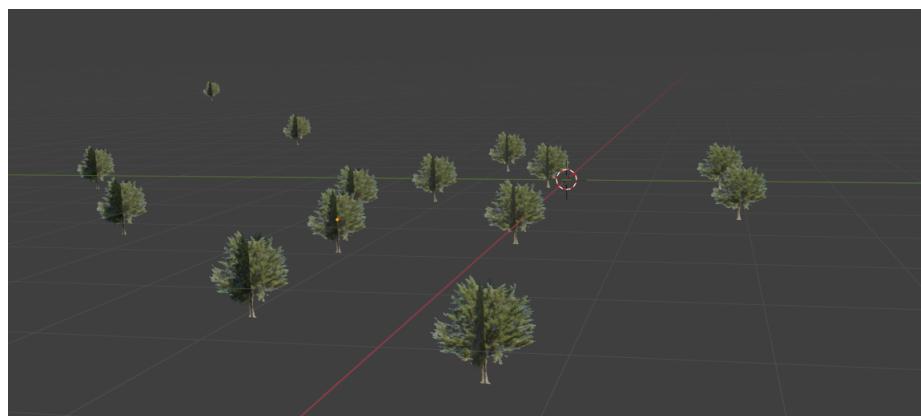


Figura 2: Copaci prezentati in Blender

Am ales sa export ca .obj separat o parte din leagan pentru manipularea lui in proiect.

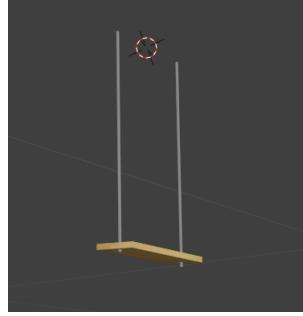


Figura 3: Leaganul in Blender

2.2 Funcționalăți

Utilizatorul poate vedea o previzualizare completă a scenei înainte de a începe să se deplaseze liber pe hartă. El poate avea o vizualizare a întregii scene prin apăsarea tastelor și prin utilizarea mouse-ului pentru a direcționa camera în poziția dorită. Obiectul leagan se poate roti în sus și jos, oferind efectul de leganare. De asemenea, scena poate fi vizualizată în modurile solid, wireframe, poligonal, apasând tastele corespunzătoare.

3 Detalii de implementare

3.1 Funcții și algoritmi

3.1.1 Soluții posibile

In implementarea proiectului, pentru vizualizarea scenei am implementat in Camera.cpp functia de move si rotate pentru ca utilizatorul sa poata sa se deplaseze in scena prin apasarea tastelor corespunzatoare. Am modificat Camera.hpp, adaugand inca doua elemente (MOVE_UP,MOVE_DOWN), astfel utlizatorul sa se poata deplasa in sus si jos.

Codurile pentru functiile move si rotate:

```
1 void Camera::move(MOVE_DIRECTION direction, float speed) {
2     glm::vec3 cameraFront = glm::normalize(cameraTarget - cameraPosition);
3     glm::vec3 cameraRight = glm::normalize(glm::cross(cameraFront, cameraUpDirection));
4     glm::vec3 cameraUp = glm::cross(cameraRight, cameraFront);
5
6
7     switch (direction) {
8         case MOVE_FORWARD:
9             cameraPosition += cameraFront * speed;
10            break;
11        case MOVE_BACKWARD:
12            cameraPosition -= cameraFront * speed;
13            break;
14        case MOVE_LEFT:
15            cameraPosition -= cameraRight * speed;
16            break;
```

```

17     case MOVE_RIGHT:
18         cameraPosition += cameraRight * speed;
19         break;
20     case MOVE_DOWN:
21         cameraPosition -= cameraUp * speed;
22         break;
23     case MOVE_UP:
24         cameraPosition += cameraUp * speed;
25         break;
26     default:
27         break;
28     }
29
30     cameraTarget = cameraPosition + cameraFront;
31 }
32 void Camera::rotate(float pitch, float yaw) {
33     glm::vec3 cameraFront = glm::normalize(cameraTarget - cameraPosition);
34
35     // Update yaw (around y-axis)
36     glm::mat4 yawRotation = glm::rotate(glm::mat4(1.0f), glm::radians(yaw), cameraUpDirection);
37     cameraFront = glm::normalize(yawRotation * glm::vec4(cameraFront, 1.0f));
38
39     // Update pitch (around x-axis)
40     glm::vec3 rightAxis = glm::normalize(glm::cross(cameraFront, cameraUpDirection));
41     glm::mat4 pitchRotation = glm::rotate(glm::mat4(1.0f), glm::radians(pitch), rightAxis);
42
43     cameraFront = glm::normalize(pitchRotation * glm::vec4(cameraFront, 1.0f));
44
45     cameraTarget = cameraPosition + cameraFront;
46 }
```

Pentru adaugarea obiectelor in scena am modificat si adaugat unele functii din main.cpp.Ca sa fac leaganul sa se ”legene”, am folosit functia de rotate si translate.

Pentru copaci, am adaugat un nou shader in care am adaugat metoda ”Eliminarea fragmentelor” prezentata in laboratorul 12.

3.1.2 Motivarea abordării alese

Am ales sa export separat leaganul pentru a-l manipula in scena, folosind tastele UP si DOWN.Pentru copaci, am ales sa creez un nou shader, deoarece era necesar o imbunatatire pentru ca obiectele sa fie afisate corect.

Obiectele care sunt pentru construirea scenei (felinare, banchi, etc.) am ales sa le export din Blender ca un singur .obj.

3.2 Modelul grafic

Majoritatea obiectelor au fost descarcate din Free3D (adaugate in folderul ”scena” din proiect).Texturile au fost descarcate de pe net.Toate obiectele au fost importate in Blender pentru a fi editate si adaugate in scena finala.

3.3 Structuri de date

Structurile de date utilizate în acest proiect sunt structurile de date comune utilizate în C/C++ și structurile de date din biblioteca glm (cum ar fi vectori, matrici etc.).

3.4 Ierarhia de clase

- Camera: conține implementarea mișcării camerei (sus, jos, față, spate, stânga, dreapta)
- Mesh
- Model3D
- main: conține metode pentru adaugarea, vizualizarea și editarea obiectelor în scenă
- Shader: conține metode de creare și activare a programelor de shader
- Shader Copaci: conține metode de creare și activare a programelor de shader pentru copaci

4 Prezentarea interfeței grafice utilizator / manual de utilizare

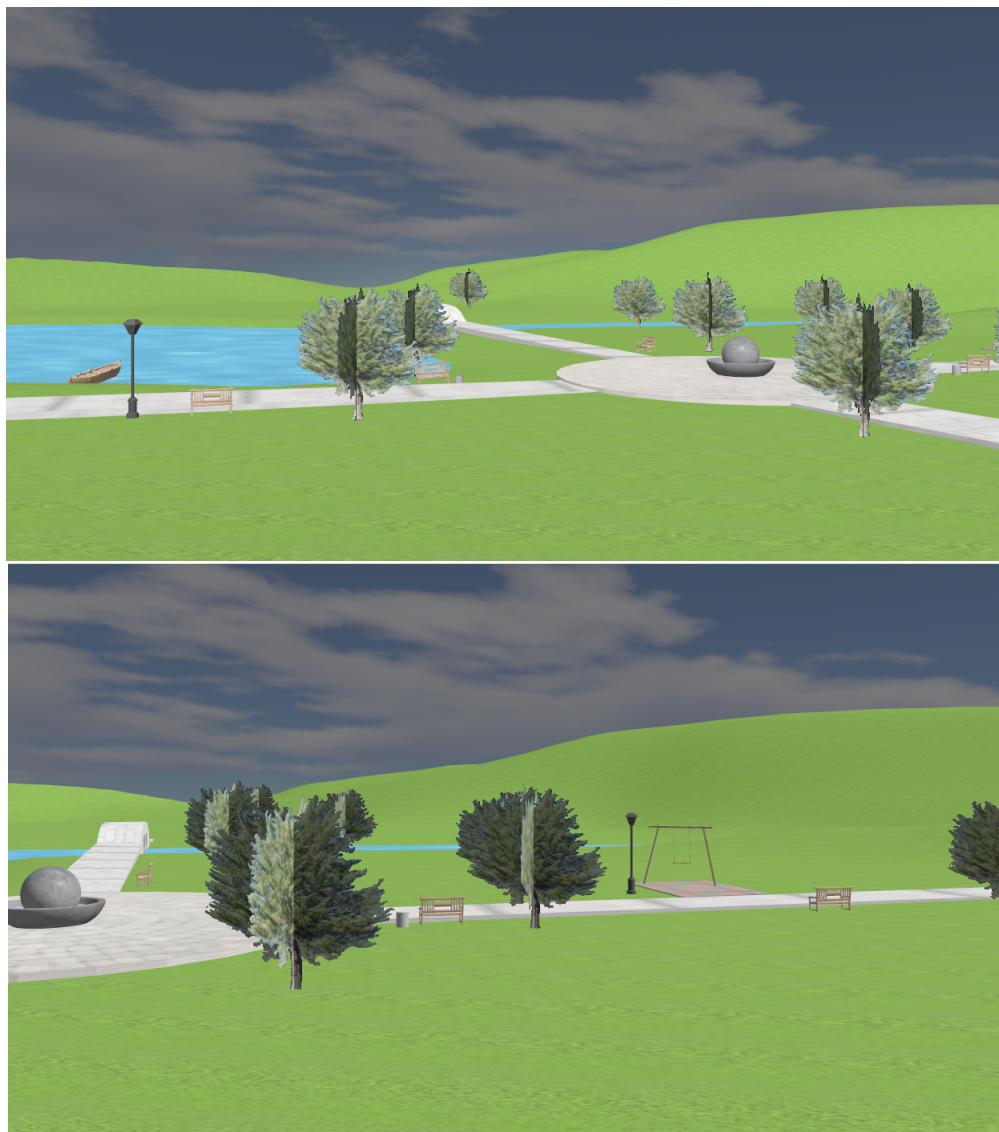


Figura 4: Scena finală



Figura 5: Night mode

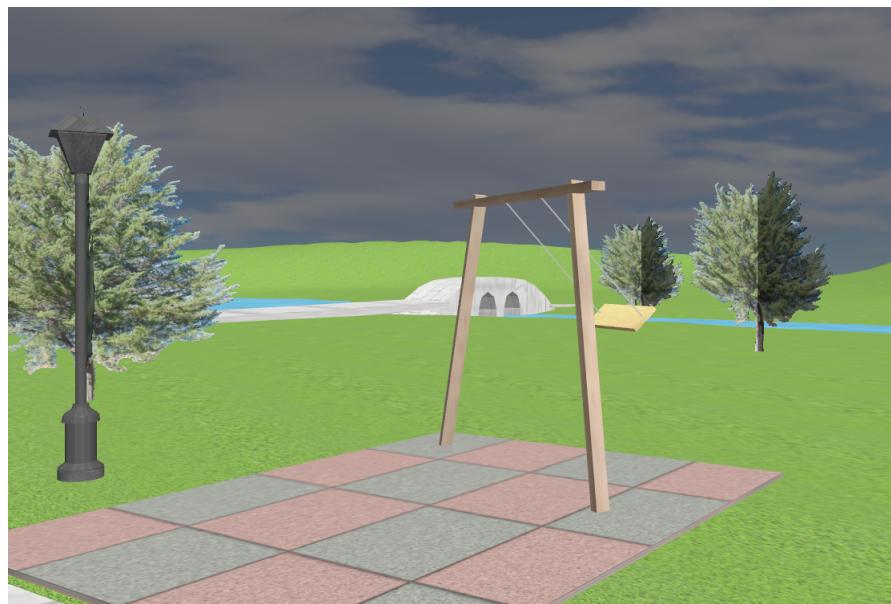


Figura 6: Leganul afisat in scena(UP)



Figura 7: Leganul afisat in scena(DOWN)

Manual de utilizare:

- miscarea mouse-ului - și pozitionarea camerei
- W, S, A, D - și miscarea camerei (față, spate, stânga, dreapta)
- U - ridica camera
- I - coboara camera
- Q, E - rotatia camerei in stanga si in dreapta
- UP, DOWN - rotirea leaganului in sus si jos
- Z - modul solid
- X - modul wireframe
- C - modul poligonal
- N - night mode
- M - day mode

5 Concluzii și dezvoltări ulterioare

In concluzie, proiectul realizat folosind Blender și obiecte 3D utilizând librăriile prezentate la laborator (OpenGL, GLFW, GLM, etc.) mi-au dezvoltat abilitatea de a lucra în Blender și OpenGL. Codul a fost implementat în Microsoft Visual Studio. Posibile dezvoltări ar fi adăugarea unor lumini la felinarele din scenă, night mode, manipularea mai multor obiecte în scenă, apă curgătoare.

6 Referințe

- <https://free3d.com>
indrumatorul de laborator
<https://learnopengl.com/>
https://www.youtube.com/playlist?list=PLrgcDEgRZ_kndoWmRkAK4Y7ToJdOf-OSM/