



---

## Documentatie proiect

*Prelucrare Grafica*

---

Autor: Vlase Rafaella

Grupa: 30235

Profesor indrumator: Sabou Adrian

FACULTATEA DE AUTOMATICA  
SI CALCULATOARE

18 Ianuarie 2023

## Cuprins

<b>1 Prezentarea temei . . . . .</b>	<b>2</b>
<b>2 Scenariul . . . . .</b>	<b>2</b>
2.1 Descrierea scenei si a obiectelor . . . . .	2
2.2 Functionalitati . . . . .	2
<b>3 Detalii de implementare . . . . .</b>	<b>3</b>
3.1 Functii si algoritmi . . . . .	3
3.2 Modelul grafic . . . . .	3
3.3 Structuri de date . . . . .	4
3.4 Ierarhia de clase . . . . .	5
<b>4 Prezentarea interfetei grafice utilizator . . . . .</b>	<b>5</b>
<b>5 Manual de utilizare . . . . .</b>	<b>8</b>
<b>6 Concluzii si dezvoltari ulterioare . . . . .</b>	<b>9</b>
<b>7 Referinte . . . . .</b>	<b>9</b>

# 1 Prezentarea temei

**Tema proiectului:** Realizarea unei scene in OpenGL

Tema acestui proiect consta in generarea unei scene 3D in OpenGL, in care utilizatorul are posibilitatea de a se plimba, ca intr-un joc. Interactiunea cu scena se face prin intermediul mouse-ului si a tastaturii.

**Obiective:**

- Crearea unei scene 3D
- Adaugarea unor efecte vizuale precum umbra si ceata
- Implementarea camerei pentru vizualizarea scenei

**Tehnologii utilizate:**

- OpenGL pentru randarea obiectelor
- C++ pentru cod
- Modelul de iluminare Blinn-Phong
- Fragment si Vertex Shaders
- Blender pentru modelarea scenei si adaugarea de obiecte

## 2 Scenariul

### 2.1 Descrierea scenei si a obiectelor

Scena conceputa de mine se doreste a fi imaginea unui mic sat japonez, la poalele muntelui. In scena am adaugat diverse obiecte cu specific japonez, de la casute, poduri, bonsai si dojo-uri in copaci sakura, la poarta Shinto si un "shrine" (mic templu de rugaciune, gazda a zeilor). Accesul la templu se face traversand un pod, deoarece templul este inconjurat de apa. Am mai adaugat copaci, un mic loc de relaxare in jurul lacului, cu bancute si bonsai. Am mai adaugat si un personaj si cateva pisici, pentru o nota mai mare de realism.

In Blender am modelat muntii, iarba si apa. Pentru gardul rosu ce inconjoara o casuta si cele doua dojo-uri am folosit modifier-ul array, pentru a avea un singur obiect cu o singura textura. Am adaugat de asemenea si felinare cu specific japonez, pentru a avea si lumina punctiforma. Scena se poate vizualiza in diferite moduri: solid, wireframe, punctiform si smooth.

### 2.2 Functionalitati

Pentru a putea vedea scena "in lumina zilei", sursa principală de lumina este cea directională. Pentru a obține umbrele obiectelor din scena lumina se poate roti folosind tastele J și L. Pentru a ne deplasa în scena ne vom folosi de tastele W, A, S, D, dar și de mouse, pentru orientarea camerei. Pentru umbre am folosit tehnica "Shadow Mapping", prezentată în laborator, mult mai facil de implementat în OpenGL. Harta de adâncime poate fi vizualizată apasând tastă M. Atunci când scena este relativ umbrată se poate observa efectul pe care îl are lumina punctiformă, prezenta în felinar. Modul "cu umbre" se poate activa folosind tastă H iar cel "cu ceata" folosind tastă F.

### 3 Detalii de implementare

#### 3.1 Functii si algoritmi

Proiectul are la baza o serie de functii OpenGL default, dar si alte functii pentru desearea obiectelor, folosirea mouse-ului etc. Dintre cele mai importante functii amintim:

- `void keyboardCallback(GLFWwindow* window, int key, int scanCode, int action, int mode)`

Folosita pentru a asocia diferite actiuni cu diferite taste(e.g. tasta M pentru vizualizarea hartii de adancime)

- `void mouseCallback(GLFWwindow* window, double xpos, double ypos)`

Ne permite sa folosim mouse-ul pentru a ne orienta privirea in scena

- `void processMovement()`

Mapeaza pe taste ”misdarile”(e.g. A, S, W, D pentru deplasare stanga, dreapta, sus, jos)

- `void initModels()`

Gestioneaza incarcarea si plasarea obiectelor in scena

- `void initShaders()`

Incarca continutul shaderelor

- `void initUniforms()`

In aceasta functie se implementeaza detaliile despre lumina, perspectiva camerei, matricea de normale

- `void renderScene()`

Randeaza intreaga scena, apeland la shadere

De mentionat este si SkyBox-ul, pentru a oferi un fundal scenei, sa para ca deasupra este cerul. Este incarcat cu ajutorul functiei `initSkyBox()`.

#### 3.2 Modelul grafic

Am folosit Blender pentru a modela si genera scena. Folosindu-ma de subdividere pe plan si apoi de sculpt mode, am modelat muntii. Pentru pavajul de sub case am adaugat un nou plan peste cel de munti, iar pentru apa am adaugat al treilea plan, vizibil doar in locurile in care am dat ”crease” in planurile deja existente.

Toate obiectele au fost gasite pe site-uri de obiecte 3D si asezate de mine in scena. A trebuit sa dau split by material la unele obiecte pentru ca fiecare obiect sa aiba un singur material, conform cerintelor. Texturile planelor pentru munti, pavaj, iarba si apa au fost mapate folosind UV editor, deoarece initial se vedea urat, si prin scalare au ajuns sa se vada mai realist.

O singura bucată din gardul rosu era initial constituită din mai multe parti, carora le-am dat join. Apoi folosindu-ma de modifier-ul array am multiplicat acea bucată initială pentru a obține întreg gardul, din laturi de sine statatoare, facând astfel economie de obiecte și forme, pentru o optimizare mai bună.

De asemenea am adăugat ca animație o pasare care zboara deasupra scenei. M-am folosit de Blender pentru a lua coordonatele pasarii.



Figura 1: Scena Mountain City



### 3.3 Structuri de date

Printre structurile de date utilizate se numara vectorii specifici OpenGL de tipul `glm`, tipul `GLuint`, `GLfloat`, tipurile Model si Shaders, dar si tipuri clasice din C++, precum `bool`, `int` si `float`.

### 3.4 Ierarhia de clase

Clasa main este de sine statatoare. Restul claselor din proiect vin insotite si de headere.

- **main.cpp**: Punctul de pornire al programului. Contine initializari ale componentelor si creeaza instantia Window.
- **Camera.cpp** si **Camera.hpp**: Are ca scop gestionarea perspectivei si pozitionarii camerei. Cu ajutorul ei miscam si controlam camera in scena.
- **Mesh.cpp** si **Mesh.hpp**: Se ocupa cu triunghiurile care definesc geometria obiectelor tridimensionale.
- **Model3D.cpp** si **Model3D.hpp**: Gestioneaza un obiect 3D format din mai multe mesh-uri.
- **Shader.cpp** si **Shader.hpp**: Gestioneaza shaderele OpenGL. Ofera functionalitati precum incarcarea si compilarea acestora.
- **Skybox.cpp** si **Skybox.hpp**: Gestioneaza si afiseaza skybox-ul, un cub al carui fete sunt imagini cu cer si pamant, pentru a crea fundalul scenei 3D.
- **stb\_image.cpp** si **stb\_image.hpp**
- **tiny\_obj\_loader.cpp** si **tiny\_obj\_loader.hpp**
- **Window.cpp** si **Window.hpp**: Reprezinta fereastra aplicatiei in care se afla scena.

## 4 Prezentarea interfetei grafice utilizator

Moduri de vizualizare:

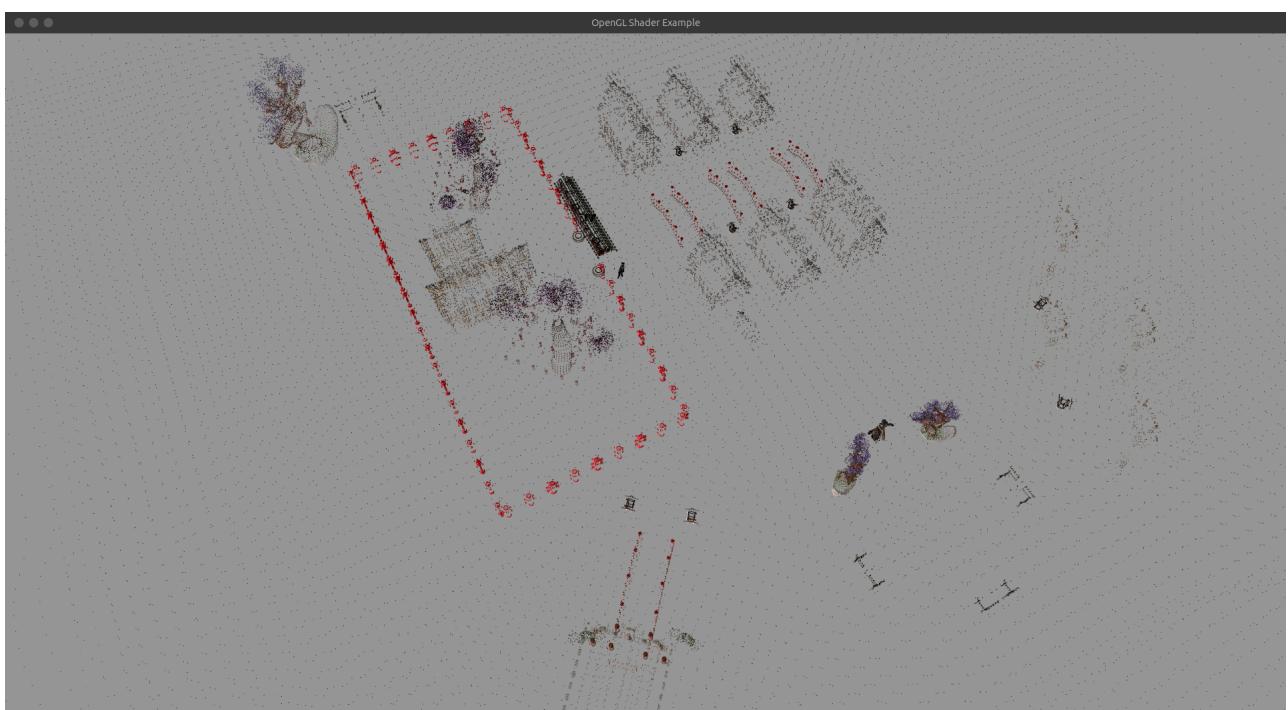
- Solid



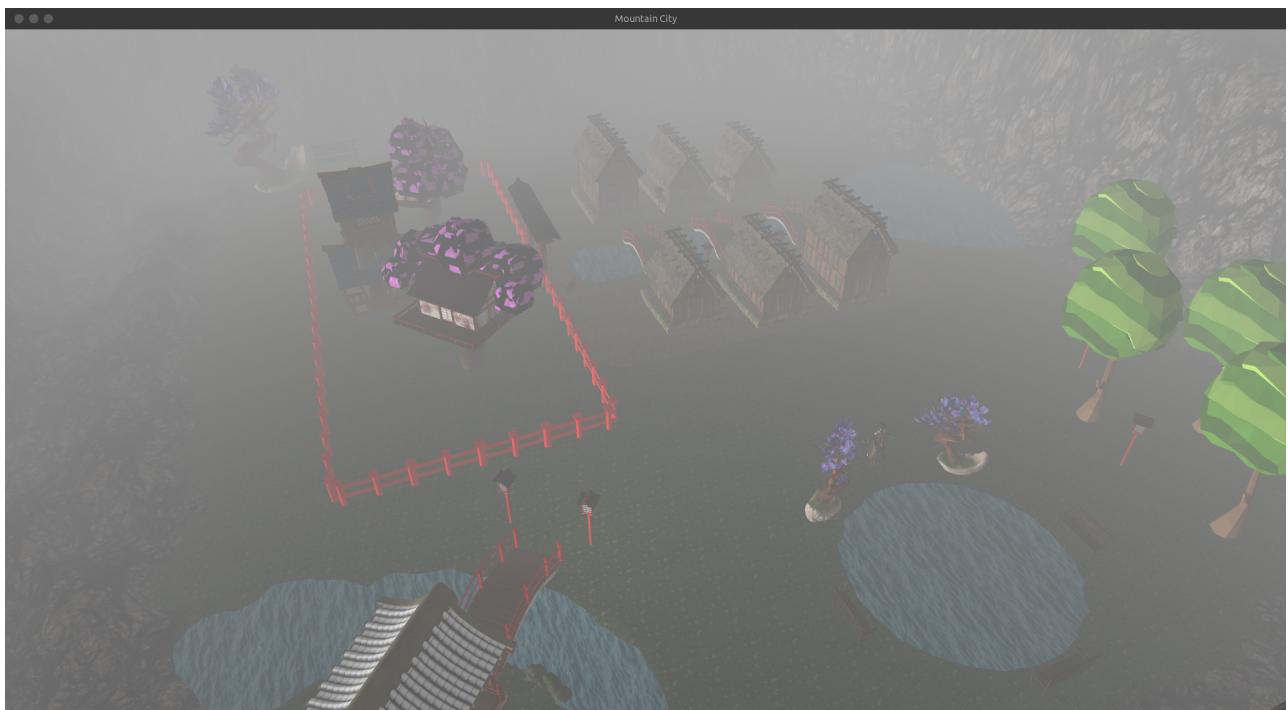
- Wireframe



- Point



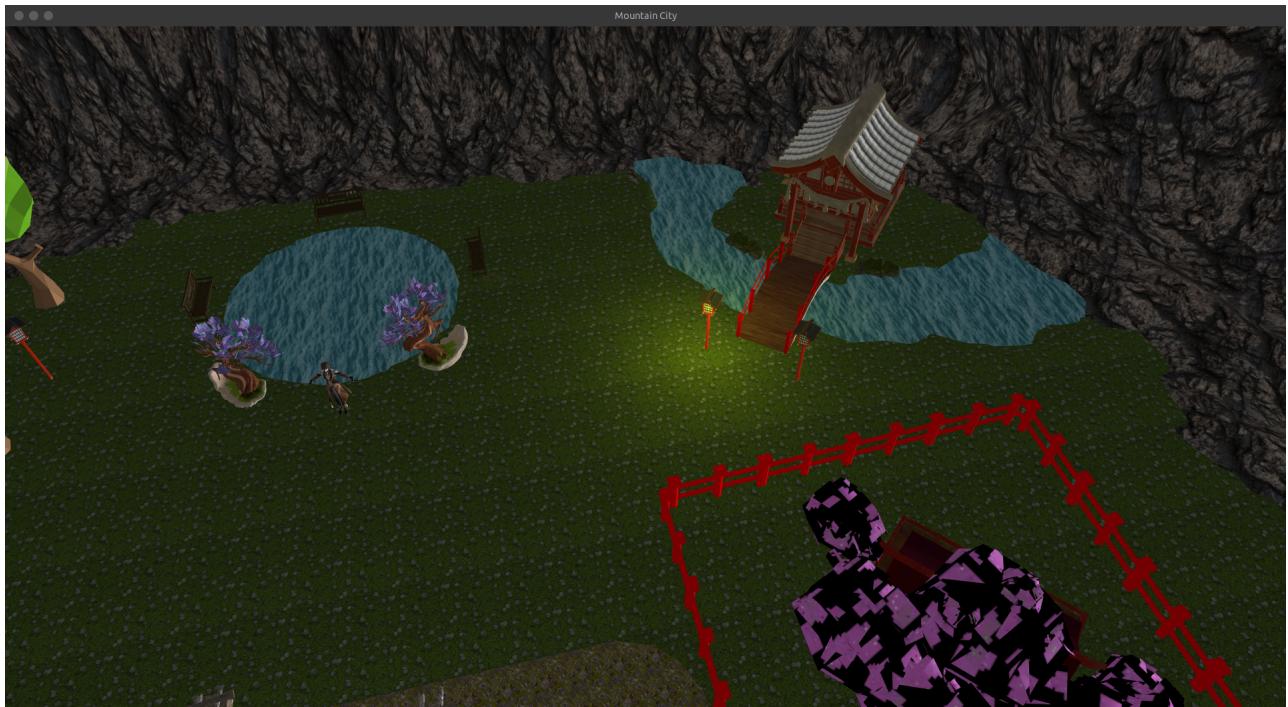
- Cu ceata



- Cu umbre



- Lumina punctiforma



## 5 Manual de utilizare

Utilizatorul are la indemana urmatoarele functionalitati:

- Tasta W: deplasare in fata
- Tasta S: deplasare in spate
- Tasta A: deplasare la stanga
- Tasta D: deplasare la dreapta

- Tasta J: rotire lumina directionala la stanga
- Tasta L: rotire lumina directionala la deapta
- Tasta T: vizualizare wireframe
- Tasta M: vizualizare harta de adancime
- Tasta P: vizualizare point
- Tasta F: activare ceata
- Tasta G: dezactivare ceata
- Tasta H: activare umbre
- Tasta Y: dezactivare umbre
- Tasta U: activare lumina punctiforma
- Tasta I: dezactivare lumina punctiforma
- Miscarea mouse-ului pentru orientarea privirii in scena

## 6 Concluzii si dezvoltari ulterioare

Acest proiect m-a ajutat sa imi aplic cunostintele despre OpenGL dobandite in cadrul cursurilor si laboratoarelor de Proiectare Grafica. Am invatat sa lucrez si in Blender, si mi-a stimulat creativitatea, permitandu-mi sa-mi modelez si asamblez scena dupa bunul meu plac.

Ca dezvoltari ulterioare ar putea fi incluse mai multe animatii, in special cea a personajului, care sa poata merge prin scena, folosind skeleton animation.

Consider ca acest proiect si toate cunostintele acumulate in urma implementarii sale pot constitui un punct de start pentru o dezvoltare ulterioara a unui joc 3D.

## 7 Referinte

1. <https://learnopengl.com/>: bazele OpenGL
2. <https://sketchfab.com/>: descarcare obiecte 3D gratuit
3. Tutoriale Blender: Tutorialele domnului profesor Constantin Nandra
4. Materialele didactice si codurile surse oferite la laborator