



# **UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA**

---

**Prezentare fotorealiste a unor scene de obiecte 3D**

*Prelucrare grafica*

---

Autori: Frincu Ioan - Cristian

Grupa: 30237

FACULTATEA DE AUTOMATICA  
SI CALCULATOARE

14 Ianuarie 2024

# Cuprins

<b>1 Prezentarea temei . . . . .</b>	<b>2</b>
<b>2 Scenariul . . . . .</b>	<b>2</b>
2.1 Descrierea scenei si a obiectelor . . . . .	2
2.1.1 Padurea . . . . .	2
2.1.2 Terenuri . . . . .	3
2.1.3 Ferma . . . . .	4
2.1.4 Cerul . . . . .	5
2.2 Functionalitati . . . . .	6
<b>3 Detalii de implemntare . . . . .</b>	<b>6</b>
3.1 Funcții și algortmi . . . . .	6
3.2 Modelul grafic . . . . .	9
3.3 Structuri de date . . . . .	9
3.4 Ierarhie de clase . . . . .	9
<b>4 Prezentarea interfeței grafice utilizator / manual de utilizare . . . . .</b>	<b>10</b>
<b>5 Concluzii și dezvoltări ulterioare . . . . .</b>	<b>10</b>
<b>6 Referinte . . . . .</b>	<b>10</b>

# 1 Prezentarea temei

Tema acestui proiect constă în realizarea unei prezentări fotorealiste a unor scene de obiecte 3D folosindu-mă de IDE-ul Microsoft Visual Studio și de ajutorul librariilor prezentate și invățate la laborator precum OpenGL, GLFW, GLM, etc.

Am ales să creez o aplicație care simulează o fermă deoarece este un domeniu familiar și îndragit de mine. Ea cuprinde o mini padure cu diverse animale, un râu, mai multe animale domestice, utilaje agricole, construcții specifice unei ferme, terenuri agricole. Am creat animații de prezentare și constante, efecte de lumina și ceata. Am modelat și texturat obiecte astfel încât să simuleze cât mai mult realitatea.

## 2 Scenariul

### 2.1 Descrierea scenei și a obiectelor

Pentru modelarea scenei și a obiectelor am folosit software-ul grafic "Blender" care mi-a permis să îmi mapez mult mai usor scena și să o import în programul meu. Majoritatea obiectelor folosite le-am gasit pe internet pe diferite site-uri, însă unele le-am modelat direct în Blender precum: planul principal, raul, unul dintre terenurile agricole, porumbul.

Mi-am împărțit scena în mai multe capituloare:

#### 2.1.1 Padurea

Padurea am modelat-o folosind niste copaci, diverse animale precum: vulpe, urs, caprioare, cerbi. Am separat de restul fermei printre rau la marginea caruia am plasat niste pietre pe care se află niste broaste.



Figura 1: Padure



Figura 2: Broaste + Pietre

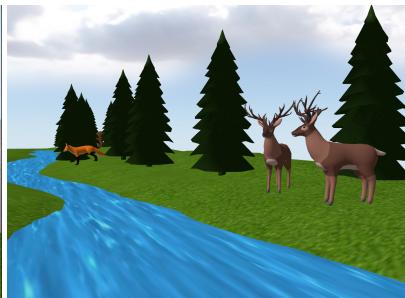


Figura 3: Vulpe



Figura 4: Urs + Cerbi

### 2.1.2 Terenuri

Portiunea cu de terenuri contine 2 terenuri, unul pe care se afla porumb si unul liber pe care am un tractor cu animatie. Porumbul si terenul liber au fost create in blender, de asta calitatea detaliilor este mai slaba.



Figura 5: Terenuri



Figura 6: Teren porumb



Figura 7: Teren liber

### 2.1.3 Ferma

La aceasta a fost cel mai mult de lucru deoarece este cea mai complexa parte. Contine obiecte specifice unei ferme precum hambar, siloz, garaje, un stalp de iluminat, utilaje agricole, o gramada de balori si multe animale precum cai, gaini, iepuri, gaste, pui, vaci, oi, bizoni, caini, pisici, pasari. In plus mai sunt detalii precum custile cainilor, mancarea si jucariile lor, gardurile si portile de la animale.



Figura 8: Caption



Figura 9: utilaje

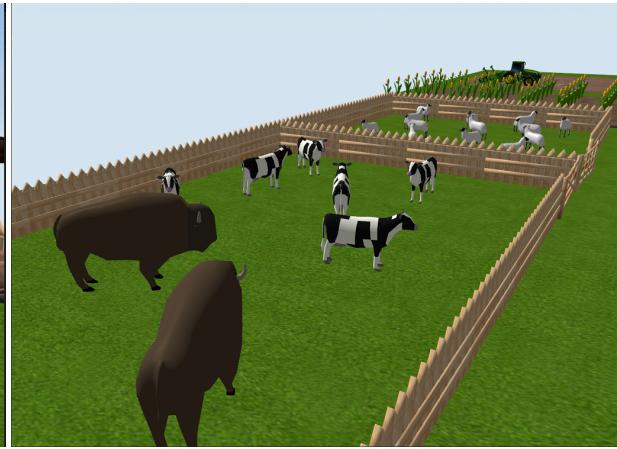


Figura 10: vaci + oi



Figura 11: Cai



Figura 12: Gaini + iepuri

#### 2.1.4 Cerul

In plus am creat si skybox-uri reprezentative atat pentru zi cat si pentru noapte.



Figura 13: Zi



Figura 14: Noapte

## 2.2 Functionalitati

In proiectul realizat odata cu pornirea executabilului se porneste automat prin animatia de camera in care se prezinta putin scena. In timpul animatiei nu poti interveni deloc, nicio miscare nu este permisa. Odata ce s-a terminat se revine la modul normal in care te poti plimba prin scena cu tastele si mouse-ul, poti modifica intensitatea luminii directionale, se poate modifica nivelul de ceata, schimbarea din modul zi in modul noapte, astfel schimbându-se skybox-ul si scazând intensitatea luminii.

## 3 Detalii de implementare

### 3.1 Functii si algoritmi

In realizarea acestui proiect am folosit functiile implementate la laborator pe parcursul semestrului pe care le-am mai adaptat dupa functionalitatea actuala si am adaugat functii noi pentru realizarea altor optiuni.

**keyboardCallback:** este functia care se apeleaza de pentru o noua apasare de tasta, se verifica ce tasta s-a apasat si se efectueaza sarcina specificata pentru tasta respectiva.

**mouseCallback:** se apeleaza la fiecare miscare a mouse-ului si se iau noile unghiuri de rotatie pentru a abdata directia de orientare a camerei.

**Camera:** contine *cameraPosition*, *cameraTarget*, *cameraUpDirection* si 3 functii. **getViewMatrix** pentru a returna cei 3 vectori, **move** pentru a realiza modificarea pozitiei si **rotate** pentru a modifica directia de orientare.

**processRotation** este functia care actualizeaza noua directie de orientare a camerei.

**processMovement** actualizeaza noua pozitie a camerei in scena.

**initSkybox:** pentru aceasat functie ne-am declarat 2 variabile care memoreaza cele 2 skybox-uri, pentru zi si pentru noapte. Initial este incarcat in scena cel de zi, iar pentru a se face schimbarea trebuie sa apasam tasta **K**. In momentul in care schimbam din zi in noapte sau invers se modifica intensitatea luminii directionale pentru a crea impresia de noapte.

**renderScene:** este folosit pentru a incarca scena.

**renderTractor:** incarca tractorul, dar tot aici avem codul si pentru animatia tractorului. Aceasta porneste dintr-un punct initial, se deplaseaza doar pe aza Z pana ajunge la un alt punct setat, in momentul respectiv realizeaza o rotatie de 180 de grade si se intoarce spre punctul de unde a pornit dupa care face din nou o rotatie de 180 de grade. Asta se intampla incontinuu. Pentru animatie verificam in ce directie se face miscarea in momentul respectiv, astfel stim daca crestem/scadem valoarea cu care se face animatia. Daca am ajuns in punctul in care dorim sa facem rotatia, translatam obiectul in pozitia principala, ii facem o rotatie si il translatam la loc, dupa care facem translatia obisnuita si incarcam din nou obiectul.

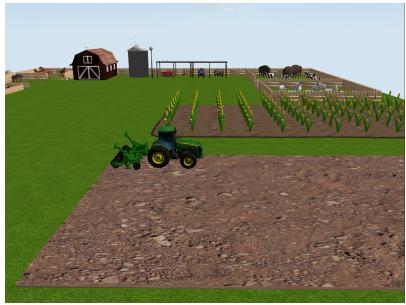


Figura 15: Poz.1



Figura 16: Poz.2



Figura 17: Poz.3

**animatie-inceput:** pentru realizarea animatie de camera in care se prezinta scena in momentul pornirii programului, salvam timpul in care a pornit executia si la fiecare ape nou preluam noua valoare a timpului si verificam daca diferenta nu depaseste 15 secunde(atat tine animatia). Ne-am folosit de functia **move** a clasei **Camera** pentru a updata pozitia. Pentru cateva secunde se deplaseaza drept in fata, cand ajunge aproape de marginea scenei face o rotatie, iar merge inainte dupa care din nou o rotatie si tot asa, astfel incat sa se realizeze o miscare pe marginile scenei, cat sa se observe toate obiectele. In plus cat timp tine animatia toate comenzile si optiunile de pe taste sau mouse sunt suspendate, ele putand fi folosite odata ce se incheie animatia.

**Ceata:** pentru realizarea efectului de ceata am preluat codul din laborator si l-am abdatat pentru a putea modifica intesitatea de pe tastele '**3**' si '**4**'. Codul necesar a fost scris fragment shader, iar valoarea intensitatii trimisa din programul principal.



Figura 18: Ceata

**Lumini:** am creat 2 surse de lumina:

- **Directionala:** simuleaza soarele si am setat ca de pe tastele '**1**' si '**2**' sa se poata micsora sau marii intesitatea luminii, modificand valoarea sa si transmitand-o la shader.

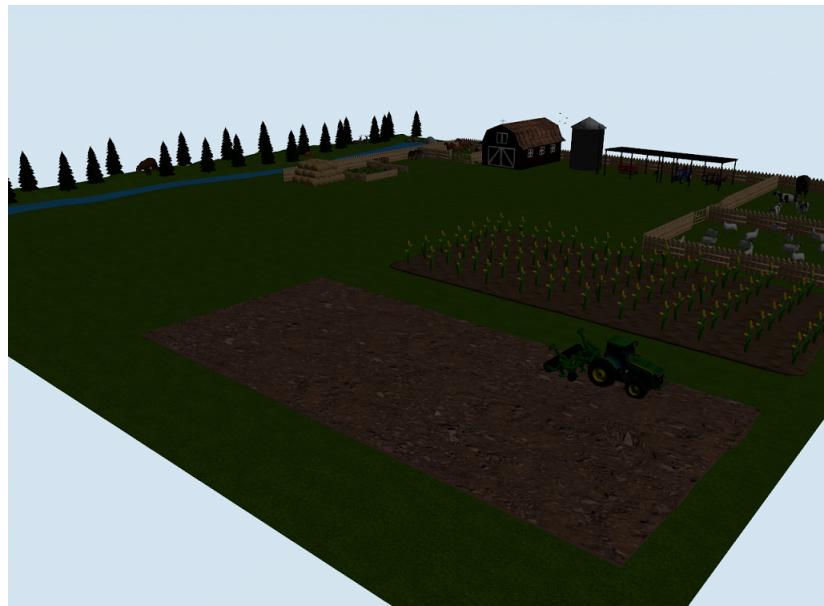


Figura 19: Lumina scazuta

- **Point Light:** m-am inspirat din laborator si de pe *"LearnOpenGL"* pentru a crea o doua sursa de lumina plasata intr-un punct dorit de mine care sa simuleze un felinar. Locatia unde doresc sa plasez aceasta lumina este trimisa din programul principal catre fragment shader.



Figura 20: Point light

### Vizualizare scena in moduri diferita

Scena poate fi privita in 3 moduri diferite:

- **Solid:** este modul standart in care se observa obiectele texturate. Se activeaza apasand tasta 'Z'

- **Wireframe:** se observa toate punctele. Se activeaza apasand tasta 'X'
- **Polygonal:** se observa toate poligoanele. Se activeaza apasand tasta 'C'



Figura 21: Solid

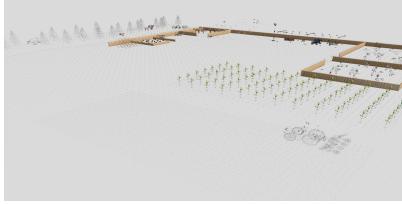


Figura 22: Wireframe

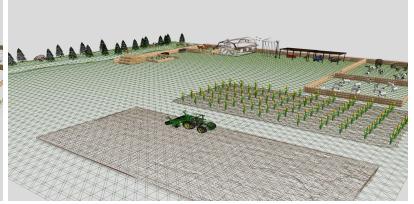


Figura 23: Polygonal

### 3.2 Modelul grafic

Cum am mentionat si mai sus, majoritatea obiectelor le-am gasit pe net si le-am descarcat, unele avand textura iar pentru altele le-am texturat eu. Folosindu-ma de 'Blender' am creat scena asezand pe un plan toate obiectele si plasandu-le unde am dorit. Le-am scalat si rotit pentru a pastra realismul. In plus am creat propriile obiecte precum porumbul, unul dintre terenuri, planul pe care se afla obiectele si raul. Pentru animatia tractorului am exportat toata scena fara obiectul respectiv, pe care l-am exportat ca un obiect separat, acesta fiind la randul sau format din 2 obiecte diferite, fiind necesat sa modul sau de texturare. Ca surse de lumina am 2, una directionala si una de tipul point light, prima reprezentand soarele, a doua pentru un felinar.

### 3.3 Structuri de date

Nu am folosit structuri de date specifice, am folosit doar variabile pentru pozitia si intensitatea luminilor, a cetei, pe care le trimitem catre fragment shader pentru a desena obiectele. Am mai folosit 2 variabile in care sa memoram cele 2 skybox-uri, **day**, respectiv **night**.

### 3.4 Ierarhie de clase

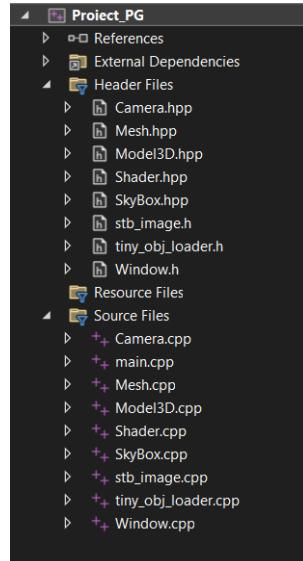


Figura 24: ierarhie

## 4 Prezentarea interfeței grafice utilizator / manual de utilizare

In momentul executiei programului se porneste automat o animatie de camera de 15 secunde, timp in care nu se mai poate efectua nicio alta optiune. Odata ce s-a terminat animatia se pot efectua urmatoarele actiuni:

- 'A' - deplasare in stanga;
- 'W' - deplasare in fata;
- 'D' - deplasare in dreapta;
- 'S' - deplasare in spate;
- 'Q' - rotatie la stanga;
- 'E' - rotatie la dreapta;
- 'Z' - vizualizare in modul solid;
- 'X' - vizualizare in modul wireframe;
- 'C' - vizualizare in modul poligonal;
- '**LEFT-SHIFT**' - mareste viteza de deplasare;
- 'P' - dezactiveaza cursorul mouse-ului;
- 'O' - activeaza cursorul mouse-ului;
- 'K' - comutare intre skybox-uri;
- '1' - scade intensitatea lumini;
- '2' - creste intensitatea lumini;
- '3' - creste intensitatea cetei;
- '4' - scade intensitatea cetei;
- **mouse** - in functie de cum miscam mouse-ul se calculeaza noile valori pentru pitch si yaw cu care se face rotatia camerei;

## 5 Concluzii și dezvoltări ulterioare

In concluzie am reusit sa implementez o aplicatie care sa simuleze realitatea, m-am familiarizat cu software-ul '*Blender*', dar si cu librariile OpenGl, GLFW, GLM. Si mi-a placut sa lucrez la proiect si eram mandru pentru fiecare realizare noua.

Ca si dezvoltari ulterioare se pot adauga umbre, alte animatii, unele chiar controlate de la anumite taste. Crearea efectului de ploaie, vant, adaugarea de noi surse de lumina diferite, modelarea mai complexa si detaliata a obiectelor, rezolvarea problemei coliziunilor si multe altele.

## 6 Referinte

- <https://free3d.com/>
- <https://www.turbosquid.com/>
- <https://www.cgtrader.com/>
- <https://learnopengl.com/Lighting/Light-casters>
- Indrumatoarele de laborator