**Proiect**

*Materia: Prelucrare Grafica*

**Contents**

[**1 Prezentarea temei** **3**](#_Toc5944)

[**2 Scenariul** **4**](#_Toc5945)

[2.1 **Descrierea scenei si obiectelor** 4](#_Toc5946)

[2.2 **Functionalitat,i** 4](#_Toc5947)

[**3 3. Detalii de implementare** **6**](#_Toc5948)

[3.1 **Functii si algoritmi** 6](#_Toc5949)

[3.1.1 Alte solut,ii posibile 7](#_Toc5950)

[3.2 Modelul grafic 7](#_Toc5951)

[3.3 Structuri de date 7](#_Toc5952)

[**4 4. Prezentarea interfetei grafice utilizator/manual de utilizare** **8**](#_Toc5953)

[**5 5. Concluzii si posibilitati de dezvoltare ulterioara** **9**](#_Toc5954)

**Chapter 1**

# Prezentarea temei

Tema acestui proiect consta ın realizarea unei scene 3D, folosind elemente de fotorealism, care sa aduca scena cat mai aproape de realitate s,i de detaliile din viat,a de zi cu zi. Scena pe care am ales sa˘ o construiesc este un mic sat aflat ıntr-o depresiune, fiind inconjurat de dealuri si munti. Satul are cateva case, un grajd pentru animale s,i cont,ine cateva elemente care sporesc idea de fotorealism.

Scopul acestui proiect a fost ınvatarea principiilor din OpenGL s,i familiarizarea cu utilizarea practica a acestora ın mediul de programare Visual Studio, utilizand librariile prezentate la laborator (OpenGl, GLFW, GLM, etc.), de asemenea utilizarea mediului de modelare 3D Blender. Utilizatorul trebuie sa aiba posibilitatea de a controla scena prin intermediul mouse-ului s,i al tastaturii.

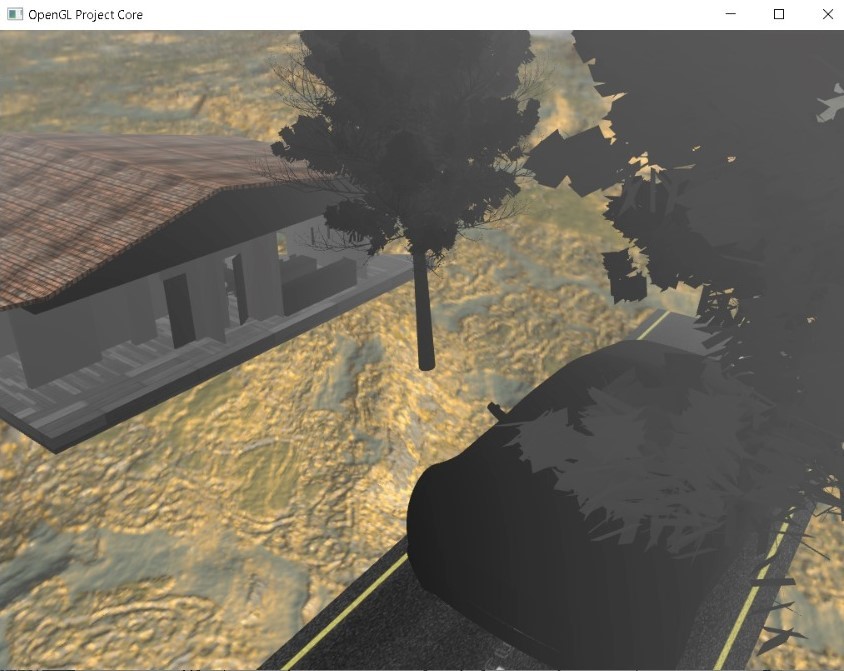
**Chapter 2**

# Scenariul

## 2.1 Descrierea scenei s, i obiectelor

Scena a fost construita cu ajutorul software-ului grafic ”Blender” s,i cu ajutorul unor modele 3D cu extensia ”.obj” preluate de pe diferite site-uri care propun diverse modele 3D. Aceasta este relativ simpla, cont,inand un numar mediu de obiecte, prezentand un peisaj urban cu o strada, o masina pe strada, o casa si cativa pomi.

Figure 2.1: scena



## 2.2 Functionalitat,i

In cadrul proiectului realizat ın Visual Studio se poate nanviga folosind operatiile de move s,i rotate, care au fost implementate ın camera de vizualizare: navigarea prin intermediul mouseului, prin intermediul tastaturii (folosind WASD pentru deplasarea ın fat,a, ın spate, la stanga s,i la dreapta, s,i R s,i F pentru deplasarea ın sus s,i ın jos). Au mai fost implementate functionalitat,ile cheie pentru un realism sporit, lumini directionale si punctiforme, precum si cateva seturi de shadere (pentru obiecte, pentru crearea hartii de adancime, pentru SkyBox, etc), inclusiv un efect de ceat,a.

De asemenea, scena inclusa ın proiect poate fi vizualizata ın mai multe moduri, folosind tastele 7, 8 s,i 9: ın modul solid, wireframe s,i punctiform.

**Chapter 3**

# Detalii de implementare

## Functii si algoritmi

In primul rand, avem functiile implementate pe parcursul laboratorului: *windowResizeCallback, keyboardCallback, mouseCallback, processMovement, initOpenGLWindow, initOpenGLState, initObjects, initShaders, initUniforms, initFBO, computeLightSpaceTrMatrix, drawObjects, renderScene cleanup*.

#### Ceata

Pentru implementarea efectului de ceata ın acest proiect, am decis sa implementez ceat,a exponent,iala patratica. Factorul de ceat,a este procesat ın funct,ia ”computeFog()” din fragment shader cu ajutorul formulei:

*fogfactor* = *e*−(*fragmentDistance*∗*fogDensity*)2

In vertex shader a fost calculata pozit,ia obiectului s,i distant,a relativa la camera de vizualizare, iar mai apoi s-a ınlocuit ın formula de mai sus.

#### Lumina globala si lumina punctiforma

Pentru cele doua surse de lumina am ales sa implementez o lumina globala direct,ionala s,i doua lumini de tip punctifroma, care au fost amplasate pe masa din scena de vizualizare. Luminile au culoarea verde, respectiv rosu.

Pentru lumina globala, ın cadrul fragment shader-ului exista funct,ia ”computeLightDir()”. Pentru aceasta lumina sunt folosite componentele: ambientala, difuza s,i speculara, calculate fiecare individual. Acest tip de lumina este calculat t,inand cont s,i de pozit,ia observatorului (a camerei) care conteaza pentru componenta speculara. Lumina ambientala , nu vine dintr-o anumita direct,ie, ea fiind de fapt aproximarea luminii globale, dar ˆımpras,tiate ˆın jurul scenei s,i reprezinta zona umbrita a obiectului.

Lumina difuza este cea imprastiata in toate directiile de o sursa de lumina, intensitatea luminii fiind mai puternica pe fetele orientate direct catre sursa de lumina (deci conteaza normala in punctul respectiv/ fragmentul deoarece folosim modelul de iluminare Phong, si directia sursei de lumina), iar cea speculara este reflectata direct de suprafata si se refera la cat de asemanatoare este suprafata cu o oglinda, deci depinde si de ”reflection”, pentru a

ımbunatat,ii efectul de iluminare.

Pentru lumina punctiforma, ın cadrul fragment shader-ului exista funct,ia: ”computeLightDir()”,ˆın care se calculeaza˘. Pentru realizarea acesui tip de lumina a fost nevoie de 3 parametrii (constant, linear s,i quadratic) preluati din main s,i transmisi funct,iei din fragment shader si de un vector care sa specifice pozitia luminilor. Cu cat cei 3 parametrii au o valoare mai mica, cu atat intensitatea luminii va creste. Aceste valori pot fi modificate din tastatura si lasa impresia cresterii intensitatii luminii. Factorul constant este lasat la valoarea 1, si se modifica numai linear si quadratic, care sunt initializati cu 0.35 si 0.44. Atenuarea se calculeaza astfel:

*Attenuation* = 1*/*(*Constant* + *Linear* ∗*Distance* + *Quadratic*∗*Distance*2) iar acest coeficient va fi inmultit cu fiecare componenta: ambientala difuza si speculara.

#### Miscarea camerei

Pentru explorarea scenei sunt folosite mouse-ul s,i tastatura. Rotatia camerei se realizeaza cu ajutorul mouse-ului, rotatia scenei se poate face prin intermediul tastelor Q si E, iar mis,carile la stanga sau la dreapta, ın fata, ın spate, sau ın sus s,i ın jos se pot face cu ajutorul tastaturii. Pentru rotat,ia camerei cu ajutorul mouse-ului exista ın main funct,ia ”mouseCallback” care preia coordonatele init,iale ale mouse-ului, iar mai apoi realizeaza rotat,ia camerei ın funct,ie de mis,carile mouse-ului, coordonatele pointer-ului acestuia. Pentru navigarea ın scena cu ajutorul tastaturii a fost implementata ın ”Camera.cpp” funct,ia ”move” care primeste ca s,i parametru directia de miscare a camerei s,i viteza cu care aceasta sa se deplaseze ın direct,ia indicata de primul parametru. In interiorul acestei functii, cu ajutorul acestor doi parametrii se vor modifica parametrii camerei:”cameraPosition” s,i ”cameraTarget”, ın funct,ie de directia ın care se dores,te deplasarea camerei.

### Alte solutii posibile

Solutiile posibile sunt cele prezentate si explicate mai sus, exista si alte solutii posibile, de exemplu pentru calcularea coeficientilor ambiental, difuz s,i specular, necesari pentru lumini, se putea calcula cu ajutorul modelului de iluminare Blinn-Phong. De asemenea, exista s,i alti algoritmi care pot fi folosit,i pentru implementarea efectului de ceata.

## Modelul grafic

Pentru a modela s,i a edita obiectele s,i scena a fost utilizat programul Blender. Terenul prezentat ın scena a fost modelat de catre mine, iar restul obiectelor au fost descarcate s,i importate ulterior ın proiect.

## Structuri de date

Nu au fost implementate structure de date aditionale sau clase in plus fata de cele care erau deja existente ın template-ul primit la laborator. Toate structurile de date utilizate ın acest proiect au fost incluse din librariile framework-ului OpenGL (vec3, vec4, mat3, mat4 etc.).

**Chapter 4**

# Prezentarea interfetei graficeutilizator/manual de utilizare

Pentru navigarea prin scena, utilizatorul are doua variante pe care le poate utiliza pentru a se deplasa s,i pentru a se roti:

1. Vizualizarea scenei cu ajutorul tastaturii:
   * ”W” va deplasa camera ın directia ınainte (efect de zoom-in)
   * ”A” va deplasa camera in directia stanga
   * ”D” va deplasa camera ın directia dreapta
   * ”S” va deplasa camera ın directia ınapoi (efect de zoom-out)
2. Vizualizarea scenei prin intermediul mouse-ului: mis,carea mouse-uluiınainte si ınapoi va deplasa camera ın sus s,i ın jos, schimband direct,ia de privire a scenei, iar prin mis,carea la stanga s,i la dreapta a mouse-ului, camera se va roti in jurul axei y, permit,and astfel vizualizarea completa a terenului s,i a obiectelor din scena. Mai sunt si alte taste care au diverse functionalita˘t,i precum:

* ”7” pentru vizualizarea scenei ın modul solid
* ”8” pentru vizualizarea scenei ın modul wireframe
* ”9” pentru vizualizarea scenei ın modul punctiform

**Chapter 5**

# Concluzii si posibilitat,i de dezvoltare ulterioara

Prin realiza rea acestul proiect, consider ca am reus,it sa ma familiarizez cu libraria OpenGL s,i cu principiile utilizate pentru randarea scenelor s,i obiectelor. De asemenea, am ınvatat sa utilizez produsul software dedicat modelarii obiectelor grafice 3D, Blender. OpenGL prezinta un pipeline grafic foarte puternic, utilizat pentru dezvoltarea de aplicatii 2D sau 3D.

Pentru dezvoltarea ulterioara a proiectului, se pot implementa animat,ii pentru mai multe obiectele din scena, se pot implementa act,iuni care permit interact,iunea utilizatorului cu obiectele din scena (de exemplu, deschiderea usilor s,i patrunderea ın interiorul clasei). De asemenea, se pot adauga mai multe efecte care sa sporeasca ideea de fotorealism: ploaie sau ninsoare, un efect de ceat,a care dispare s,i reapare ın funct,ie de trecerea timpului.

<https://free3d.com/>

<https://www.cgtrader.com/free-3d-models><https://www.turbosquid.com/Search/3D-Models/free><https://learnopengl.com/Introduction>