Universitatea Tehnica Cluj Napoca

Facultatea de Automatica si Calculatoare

Specializarea Calculatoare si Tehnologia Informatiei

An 3 semestru 1 , an scolar 2019-2020

**Proiect in cadul disciplinei**

**Sisteme de prelucrare grafica**

Student : Tusinean Andrei

Grupa : 30236

1.Cuprins

1. Cuprins
2. Prezentarea temei
3. Scenariul

3.1 Descriere scena

3.2 Functionalitati

1. Detalii de implementare

4.1 Functii si algoritmi

4.1.1 Solutii posibile

4.1.2 Motivarea abordarii alese

4.2 Modelul grafic

4.3 Structuri de date

4.4 Ierarhia de clase

1. Manual de utilizare
2. Conlcuzii
3. Referinte

**2.Prezentarea temei**

Proiectele au ca și scop realizarea unei prezentări fotorealiste a unor 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.

o (2p) vizualizarea scenei: scalare, translație, rotație, mișcarea camerei o utilizând tastatura sau mausul o utilizând animații de prezentare

o (1p) specificarea surselor de lumina (cel puțin două surse de lumină diferite)

o (0.5p) vizualizare scenă în modurile solid, wireframe, poligonal și smooth

o (1p) maparea texturilor și definirea materialelor o calitatea texturilor și nivelul de detaliu al acestora o maparea texturilor pe obiecte

o (1p) exemplificarea generării umbrelor

o (0.5p) exemplificarea animării diferitelor componente ale obiectelor

o (3p) fotorealism, complexitatea scenei, nivelul de detaliere al modelării, dezvoltarea diferiților algoritmi și implementarea acestora (generare dinamică de obiecte, detecția coliziunilor, generarea umbrelor, ceață, ploaie, vânt), calitatea animațiilor, utilizarea diferitelor surse de lumină (globală, locală, de tip spot)

o (1p) documentația (obligatorie)

**3.Scenariul**

Scene si descriere obiecte La rularea programului, utilizatorul poate vedea o fereastra deschisa dupa incarcarea tuturor obiectelor si texturilor, o fereastra ce contine o lume 3d cu obiecte si efecte .

**3.1 Descrierea scenei si a obiectelor**

Scena contine ([2],[3]):

* Un ground reprezentand pamantul pe care stau obiectele
* Soldati
* 6 masini humvee
* 3 elicoptere
* 6 cladiri distruse din caramida
* O cladire cu un etaj si soldati in interior
* Cladiri de diverse dimensiuni si suprafete
* Mai multi copaci

**3.2 Functionalitati**

Utilizatorul se poate plimba prin scena cu ajutorul tastelor w,a,s,d,e,q sau cu mouse – ul precum un specator .

Scena poate fi vizualizata in 3 moduri de prezentare :

* Solid
* Wireframe
* Point

**4. Detalii de implementare**

Abordarea aleasa pentru dezvoltarea aplicatiei aleasa are ca baza resursele disponibile la laborator .

Modelul grafic utilizat este modelul prezentat in lucrarile de laborator , model ce se bazeaza pe modelul ShadowMapping , pe modelul Phong , un model empiric al iluminarii punctelor pe o suprafata folosind componentele luminii : difuza , ambientala si speculara .

**4.1 Functii si algoritmi**

* IncrementFog() , decrementFog()
* GunFight() : sunet de batalie
* Sniper() : sunet de sniper
* StartAnimation(time) : controleaza animatia de prezentare

**4.1.1 Solutii posibile**

Se putea face si fara aceste functii sau in alt mod , de exemplu a nu le implementa sau a le lasa automate.

**4.1.2 Motivarea abordarii alese**

Aceasta abordare mi s-a parut cea mai usoara de implementat si de inteles .

**4.2 Modelul grafic**

O parte din scena cum se vede cand programul este deschis :



O parte din scena cu ceata ([6]) (care poate fi reglata cu butoane)



Elicopterul a carui elice se invart singure in continuu :



**4.3 Structuri de date**

Pentru realizarea calculelor am folosit structurile de date disponibile in biblioteca GLM , cum ar fi vec < n > , sau mat < n > , precum si structurile specifice opengl – ului cum ar fi Gluint , GLFWwindow , ect .

**4.4 Ierarhia de clase**  
Pentru a implementa functionalitatile mentionate anterior a fost necesara includerea de headere si cpp – uri precum :

- Camera.hpp – care realizeaza controlul camerei *[7]*

- Mesh.hpp – defineste varfurile unui obiect

- Model3D.hpp – creeaza un nou model 3D

- Shader.hpp – incarca un shader

- SkyBox.hpp – realizeaza functionalitatea cubemap-ului

- glm.hpp – biblioteca pentru calculul matematic

- GLEW.h si GLFW.h – pentru functionalitate/randari

**5. Manual de utilizare**

Actiuni ce se pot efectua folosind tastatura ([6]):

* W : miscare in fata
* A : miscare in stanga
* S : miscare in spate
* D : miscare in dreapta
* Mouse : rotatie camera
* E : rotatie camera spre stanga
* Q : rotatie camera spre dreapta
* J : rotatie sursa de lumina
* L : rotatie sursa de lumina
* , : decrementare ceata [1]
* . : incrementare ceata [1]
* 1 : sunet batalie [4]
* 2 : sunet sniper [4]
* P : stop sunet [4]

Taste de pe numpad :

* 1 : mod vizualizare smooth
* 2 : mod vizualizare line [4]
* 3 : mod vizualizare point [4]
* 4 : start animatie de prezentare[1]
* 5 : stop animatie de prezentare

**6.Concluzii si dezvoltari ulterioare**

Prin crearea unei astfel de aplicatii o persoana isi poate devzolta considerabil cunostiintele si abilitatile in modelare 3D , realizare de animatii , openGL si blender .

Imbunatatirea unei aplicatii poate fi un aspect foarte amplu . In cazul acestui proiect pentru o dezvoltare ulterioara s-ar putea imbunatatii texturile pentru a face scena mai realista , sa se adauge mai multe tipuri de obiecte si sa se implementeze mai multe animatii .

Proiectul poate fi dezvoltat de asemenea cu optimizari , detecare de coliziuni si adaugare de surse multiple de lumina .

**7.Referinte**

[1]<https://learnopengl.com/>

[2]<https://free3d.com/>

[3]<https://www.turbosquid.com/>

[4]<https://www.google.com/>

[5]<https://www.youtube.com/>

[6]<https://moodle.cs.utcluj.ro/course/view.php?id=185>

[7] <https://stackoverflow.com/>