A logo with black and red text

Description automatically generated

**FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE**

**Proiect Programare Grafica**

**Indrumator de proiect: Mihon Danut**

**Student: Igna Alexandra Andreea**

**Data: 18.01.2024**

**Grupa: 30235**

**CUPRINS**

[1. PREZENTAREA TEMEI 3](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917554)

[2. SCENARIUL 3](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917555)

[Descrierea scenei si a obiectelor 3](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917556)

[Functionalitati 4](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917557)

[3. DETALII DE IMPLEMENTARE 5](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917558)

[Functii si algoritmi 5](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917559)

[Modelul grafic 6](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917560)

[Structuri de date 6](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917561)

[Ierarhii de clase 7](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917562)

[4. PREZENTAREA INTERFETEI GRAFICE 7](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917563)

[5. MANUAL DE UTILIZARE 8](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917564)

[6. CONCLUZII 9](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917565)

[7. REFERINTE 9](file:///C:\Users\alexa\Downloads\Documentatie(1).docx#_Toc92917566)

1. **Prezentarea temei**

Proiectul se concentreaza pe explorarea vizualizarii fotorealiste in lumea jocurilor si graficii 3D, oferind o platforma educativa si interactiva pentru intelegerea procesarii scenelor 3D de catre calculatoare. Utilizand biblioteci si tehnologii avansate precum OpenGL, GLFW si GLM, proiectul creeaza o scena virtuala in care utilizatorii pot naviga liber folosind tastatura si mouse-ul. Aceasta experienta nu numai ca imbogateste cunostintele despre grafica 3D, dar serveste si ca un punct de plecare spre intelegerea mai complexa a lumii extinse a jocurilor si simularilor pe calculator.

1. **Scenariul**

Descrierea scenei si a obiectelor

Scena virtuala este o incursiune intr-o lume naturala. Ea cuprinde o cabana rustica amplasata langa un lac, cu barca asezata strategic pe mal, o fantana veche si copaci inalti care vegheaza peisajul. Cerul senin, ales intentionat pentru a adauga un sentiment de liniste si claritate, completeaza aceasta scena.



Functionalitati

Interactiunea cu scena este intuitiva si flexibila, permitand utilizatorilor sa navigheze prin peisaj folosind tastele WASD. Rotatia scenei este accesibila prin tastele Q si E, oferind o perspectiva dinamica si cuprinzatoare asupra intregului mediu. Modurile de vizualizare - solid, wireframe si poligonal - sunt comutabile prin Z, X si C, permitand astfel o explorare detaliata a complexitatii structurale a obiectelor.

Ceata, care poate fi activata sau dezactivata prin tastele V si B, adauga un strat de mister si adancime scenei, in timp ce interactiunea cu luminile (activata prin F si dezactivata prin G) demonstreaza modul in care lumina poate transforma si defini o scena. Functia speciala de pe tasta O scoate in evidenta mecanismele sofisticate ale umbrelor, iar animatia simpla de rotatie a scenei, activata prin P, ofera o vedere panoramica, amplificand frumusetea mediului creat.

1. **Detalii de implementare**

Functii si algoritmi

Proiectul meu in OpenGL ilustreaza un peisaj scenic, combinand tehnici si algoritmi avansati pentru a crea o experienta vizuala imersiva. Iata cateva aspecte cheie ale implementarii:

1. Initializarea Ferestrei: Am folosit functia myWindow.Create(glWindowWidth, glWindowHeight, "OpenGL Project Core") pentru a initializa fereastra unde se va afisa scena. Aceasta functie este cruciala pentru a seta dimensiunile si titlul ferestrei in care vom desena.

2. Desenarea Scenei: Functia renderScene() joaca un rol central in proiect, fiind responsabila pentru randarea obiectelor in scena. Aici, in functie de interactiunea utilizatorului, se decide daca sa se afiseze scena completa sau doar harta de adancime pentru a vizualiza umbrele.

3. Uniforms si Shadere: Cu ajutorul functiei initUniforms(), am stabilit si initializat locatiile uniformelor din shadere. Aceasta include detalii precum lumina, perspectiva camerei si matricea normala, esentiale pentru functionarea corecta a programului. De asemenea, initModels() si initShaders() sunt folosite pentru a incarca modelele 3D si shaderele.

4. Controlul Camerei si Interactivitate: Functiile mouseCallback() si processMovement() permit miscarea camerei cu ajutorul tastaturii si mouse-ului, oferind utilizatorului posibilitatea de a explora scena din diferite unghiuri.

5. Sky Box: Pentru a adauga profunzime si a extinde vizual spatiul scenei, am implementat un skybox prin functia renderSkyBox(). Acesta ofera o iluzie de lume infinita si contribuie la estetica generala a scenei.

6. Shadow Mapping: Utilizand initFBO() si renderShadow(), am implementat maparea umbrelor, care adauga realism scenei prin simularea efectelor de lumina si umbra.

7. Loop-ul Principal: In functia main, am stabilit bucla principala a aplicatiei, care gestioneaza actualizarile si randarea continua a scenei. Aici, sunt gestionate evenimentele de la utilizator si se asigura actualizarea constanta a scenei.

Prin combinarea acestor tehnici si algoritmi, am reusit sa creez o scena 3D detaliata si interactiva, care nu doar demonstreaza capacitatile tehnologice, ci si creativitatea in domeniul graficii pe calculator.

1. **Modelul Grafic**

Pentru realizarea modelului grafic al proiectului meu OpenGL, in loc sa creez fiecare obiect de la zero, am decis sa folosesc modele 3D preexistente, descarcate de pe diverse site-uri specializate. Aceasta alegere mi-a permis sa economisesc timp semnificativ si sa ma concentrez pe alte aspecte ale proiectului.

Structuri de date

In proiectul meu am utilizat o varietate de structuri de date pentru a gestiona diferitele aspecte ale mediului 3D. Acestea includ:

1. Clasa Camera: Aici, am folosit structuri de date pentru a stoca informatii esentiale precum pozitia camerei, directia de vizualizare si coordonatele de pozitie. Aceste date sunt vitale pentru a asigura o experienta de navigare fluida si intuitiva in cadrul scenei.

2. Variabile GLint: Folosite pentru a stoca adresele variabilelor uniforme din shader-e. Aceste adrese sunt esentiale pentru a lega datele din aplicatie cu logica de procesare grafica din shader-e.

3. Vectori GLM: Am folosit vectori de diferite dimensiuni pentru a stoca si manipula coordonatele spatiale, directiile si alte valori matematice necesare in grafica 3D.

4. Tipuri de Date Personalizate: Am folosit structuri de date precum Model si Shader pentru a organiza si gestiona modelele 3D si shader-ele. Aceste structuri au facilitat o mai buna organizare a codului si un acces mai usor la resursele necesare.**Ierarhii de clase**

**Pentru o structurare eficienta si un cod curat, proiectul meu se bazeaza pe o ierarhie bine definita de clase, fiecare avand un rol specific:**

**1. Clasa Camera: Gestionarea si actualizarea starii camerei in timp real, inclusiv miscarea si orientarea acesteia.**

**2. Clasa Window: Inglobeaza toate functionalitatile necesare pentru crearea si gestionarea ferestrei de desenare OpenGL.**

**3. Clasa SkyBox: Se ocupa cu incarcarea, desenarea si manipularea skybox-ului, contribuind la crearea unui mediu imersiv.**

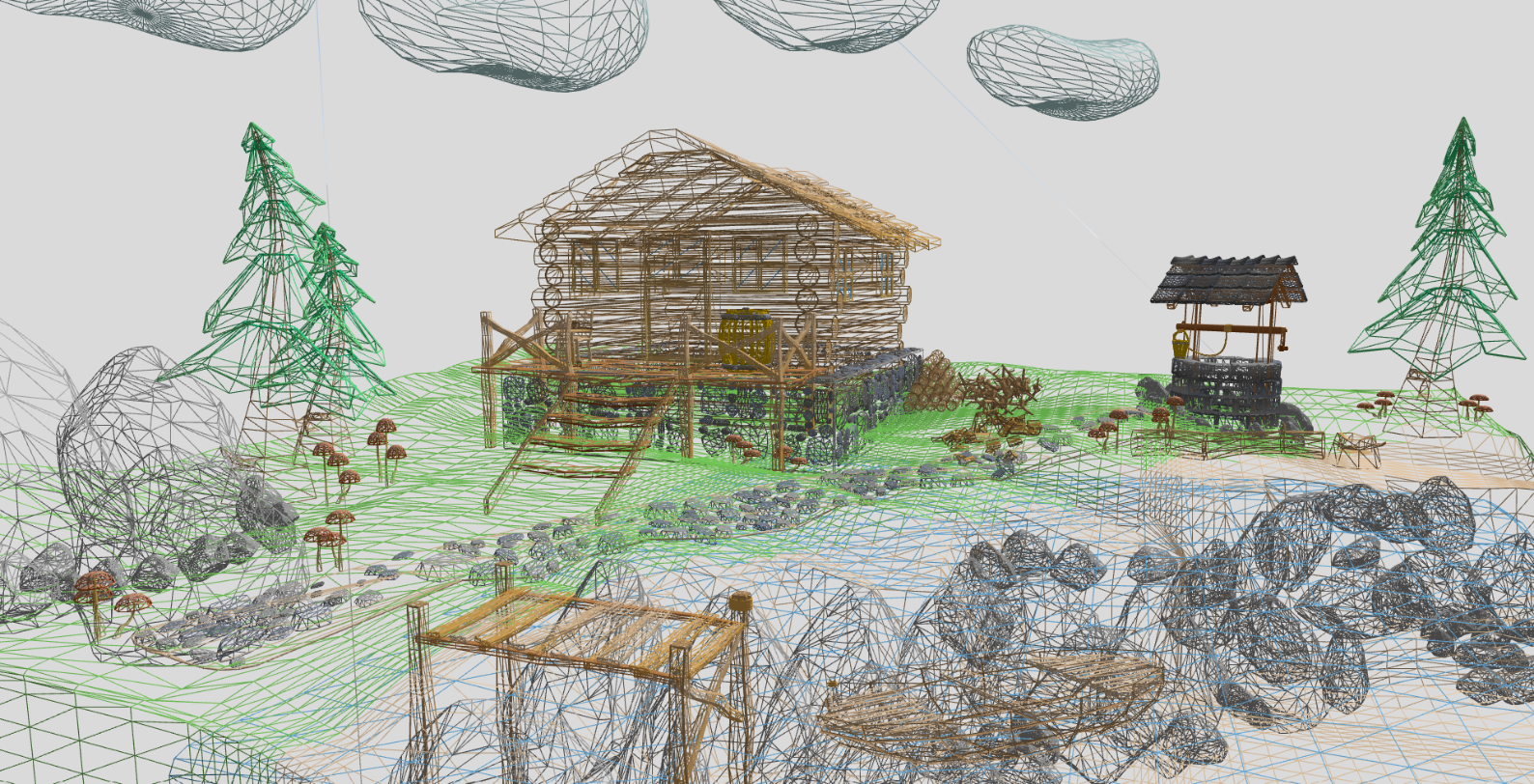
**4. Clasa Mesh: Responsabila de desenarea si gestionarea diferitelor obiecte 3D in scena.**

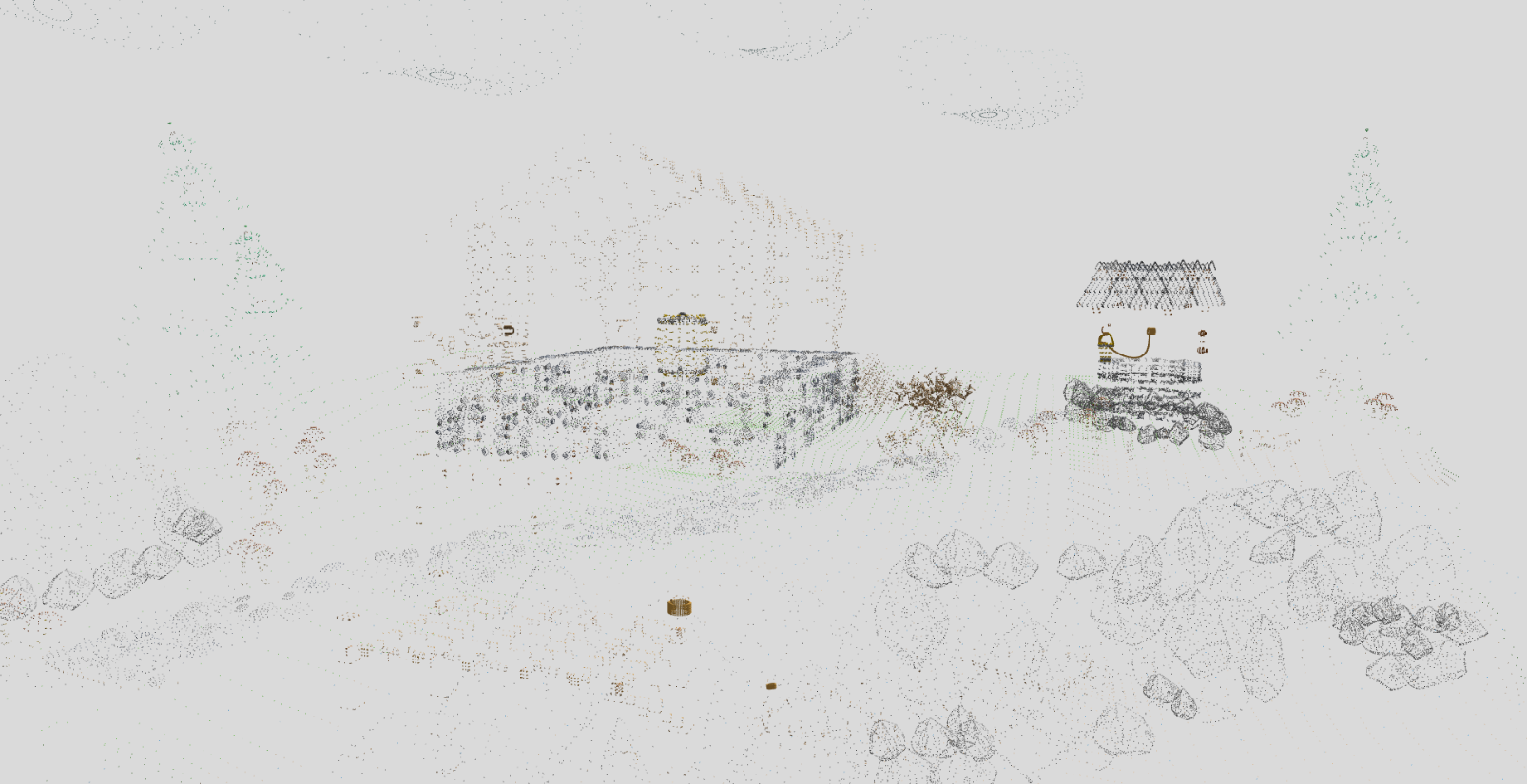
**5. Clasa Shader: Indeplineste toate operatiunile necesare pentru a implementa si gestiona shader-ele, incluzand incarcarea, compilarea si legarea lor cu datele din scena.**

**6. Biblioteci Auxiliare: Utilizeaza stb\_image pentru incarcarea texturilor si tiny\_obj\_loader pentru importul modelelor 3D, facilitand integrarea resurselor externe in proiect.**

**Aceasta structura de clase si organizarea datelor m-au ajutat sa mentin codul ordonat, scalabil si usor de inteles.**

1. **Prezentarea interfetei grafice utilizator / manual de utilizare**

Moduri de vizualizare:





Peisajul cu ceata:



Peisajul cu doua surse de lumina diferite:





* W - deplasare in fata
* S - deplasare in spate
* A - deplasare in stanga
* D - deplasare in dreapta
* E - rotatie in dreapta
* Q - rotatie in stanga
* V - activare ceata
* B - dezactivare ceata
* F - sursa lumina 1
* G - sursa lumina 2
* O - animatie umbre
* P - animatie rotire

1. **Concluzii**

De-a lungul acestui proiect, am dobandit competente valoroase in lucrul cu biblioteci grafice complexe si am inteles mai bine cum sunt create si renderizate elementele in jocurile video. Implementarea unei camere de tip first-person si realizarea unui mediu interactiv au fost provocari care mi-au permis sa aplic teorii si concepte tehnice intr-un context practic si angajant.

1. **Referinte**

* <https://www.cgtrader.com/>
* <https://learnopengl.com/Getting-started/Textures>
* Laboratoarele sursa de pe moodle