MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Facultatea de Automatica si Calculatoare

Catedra Calculatoare si Tehnologia Informatiei

Prelucrare grafica

Documentatie

Student: Maria-Alexandra Buzila

Grupa: 30236

Indrumator: Adrian Sabou

Cuprins:

Prezentarea temei

Am creat un sat virtual folosind Blender pentru modelele 3D si texturi, iar OpenGL pentru a gestiona miscarea camerei si urmarirea scenei. GLM este folosit pentru operatii matematice, cum ar fi calculul matricelor si vectorilor, esentiale pentru transformarea obiectelor in scena 3D. Proiectul permite explorarea satului din diverse unghiuri si interactiunea cu acesta.

Scenariul

Descrierea scenei si a obiectelor

Scena creata reprezinta un sat pitoresc, cu o varietate de obiecte ce includ case, animale, o fantana, casute de lemn, o sectie de politie si o biserica. Pe langa aceste structuri, am inclus si elemente ambientale un cer senin, un lac linistit si dealuri/podisuri care adauga profunzime peisajului. Toate obiectele au fost modelate in Blender, folosind diverse tehnici de modelare. Blender a permis nu doar crearea geometriei, dar si aplicarea de texturi care dau viata scenei, completand astfel atmosfera rustica a satului.

Functionalitati

Codul prezentat reprezinta o aplicatie 3D interactiva scrisa in C++ utilizand OpenGL si ofera o serie de functionalitati pentru a crea o experienta vizuala captivanta. Totul incepe cu configurarea ferestrei si initializarea contextului OpenGL, care permite afisarea graficii pe ecran. Dimensiunile ferestrei sunt setate la 800x600 pixeli, iar codul este pregatit pentru a detecta si ajusta automat modificarile dimensiunilor ferestrei, astfel incat sa pastreze proportiile corecte.

Un element central al aplicatiei este camera, care poate fi deplasata prin tastele W, A, S si D pentru a explora scena. Camera este setata initial sa priveasca spre centrul unei scene 3D, iar rotatia acesteia este controlata prin miscarile mouse-

ului. Sensibilitatea miscarii mouse-ului poate fi ajustata pentru un control mai fin. Aplicatia suporta iluminare directionala si iluminare punctiforma. Directia si culoarea luminii sunt trimise la shader pentru a crea efecte vizuale realiste. In plus, utilizatorul poate activa sau dezactiva diferite efecte, iluminarea suplimentara, folosind taste dedicate.

Modelele 3D sunt incarcate in scena prin clasa Model3D. Avem mai multe modele predefinite, inclusiv un cub luminos care reprezinta pozitia sursei de lumina. Modelele sunt desenate in functie de o matrice de model care permite rotatii si alte transformari geometrice.

Utilizatorul poate schimba modul de afisare al obiectelor intre solid, wireframe sau puncte, folosind tastele 1, 2 si 3. De asemenea, tastele Q si E sunt folosite pentru a roti obiectele din scena in jurul unei axe verticale. Aceste functii ofera o flexibilitate sporita pentru a explora si modifica scena in timp real.

Un mod "automat" poate fi activat prin tasta B, care face ca scena sa se miste automat fara interventia utilizatorului. Aceasta este o functie interesanta pentru a vizualiza scena in totalitate.

Pentru a asigura stabilitatea aplicatiei, exista o functie glCheckError() care verifica erorile OpenGL si ofera informatii detaliate despre posibilele probleme. Acest lucru faciliteaza depanarea si permite dezvoltatorului sa se asigure ca grafica este redata corect.

Per ansamblu, acest cod ofera o baza solida pentru dezvoltarea de aplicatii 3D, combinand functionalitati de randare avansate, interactiune utilizator si optimizare pentru performanta.

Detalii de implementare

Functii si algoritmi

Librariile OpenGL contin o serie de functii utile pentru dezvoltarea aplicatiei, iar prima functie esentiala in proiectul meu este cea care initializeaza fereastra pentru afisarea hartii. Aceasta functie,

myWindow.Create(glWindowWidth, glWindowHeight, "OpenGL Project Core"), seteaza dimensiunea ferestrei si titlul aplicatiei.

O functie cheie in proiect este renderScene(), care are rolul de a desena obiectele in scena.

Un alt element important este initUniforms(), care seteaza locatiile variabilelor in shadere si le initializeaza cu valori specifice, precum informatii despre lumina, perspectiva camerei si matricea de normale. Fara aceasta functie, aplicatia nu ar putea functiona corect. De asemenea, functiile initModels() si initShaders() sunt esentiale pentru importarea obiectelor 3D in OpenGL si incarcarea shaderelor necesare pentru desenarea acestora.

Pentru a permite miscarea camerei prin tastatura si mouse, am implementat functiile mouseCallback() si processMovement(), facilitand navigarea in scena si vizualizarea acesteia din diverse unghiuri.

Modelul grafic

Aranjarea obiectelor in scena a fost realizata in totalitate de mine. Obiectele au fost descarcate de pe internet, site-ul pus in referinte, la fel si texturile.

Structuri de date

Structurile de date sunt fundatia oricarui proiect software serios, iar in contextul OpenGL ele joaca un rol esential.

In proiectele de grafica, matricile sunt folosite pentru a efectua transformari precum translatii, rotatii si scalari. In codulnostru, avem mai multe tipuri de matrici gestionate cu ajutorul bibliotecii GLM:

- Model Matrix: Aceasta defineste pozitia, orientarea si scara unui obiect in scena.
- View Matrix: Este folosita pentru a determina perspectiva camerei si pentru a seta modul in care vedem scena.

- Projection Matrix: Aceasta controleaza modul in care scena 3D este proiectata pe un ecran 2D.
- Normal Matrix: Se ocupa de transformarea normalelelor pentru a corecta luminile in functie de transformarile aplicate obiectului.

In OpenGL, aceste matrici sunt trimise catre shadere ca uniforme, folosind functii precum glUniformMatrix4fv.

Lumina este un element cheie in redarea obiectelor 3D, iar in acest proiect gestionam mai multe tipuri de lumini:

1. Lumina directionala: O sursa de lumina care vine dintr-o directie specifica (ex. razele soarelui). Directia este definita de un vector precum glm::vec3(0.0f, 1.0f, 1.0f).
2. Lumina punctiforma: O sursa de lumina care emite in toate directiile dintr-un punct fix. Este utila pentru a simula becuri sau alte surse de lumina localizate.

Proprietatile acestor lumini (directie, culoare, intensitate) sunt definite ca variabile si trimise catre shadere.

Pentru a incarca si reda obiecte 3D, folosim clasa gps::Model3D. Aceasta incarca modele din fisiere externe (ex. .obj) si gestioneaza desenarea lor. In codul nostru, modelele scene3 si lightcube sunt exemple clare:

- scene3 reprezinta o scena mai complexa cu obiecte multiple.
- lightcube este un cub mic folosit pentru a marca pozitia unei surse de lumina.

Camera este controlata cu ajutorul clasei gps::Camera. Pozitia, directia de privire si axa "sus" sunt definite prin vectori precum:

```
gps::Camera myCamera(

    glm::vec3(0.0f, 0.0f, 3.0f),  // pozitia camerei
```

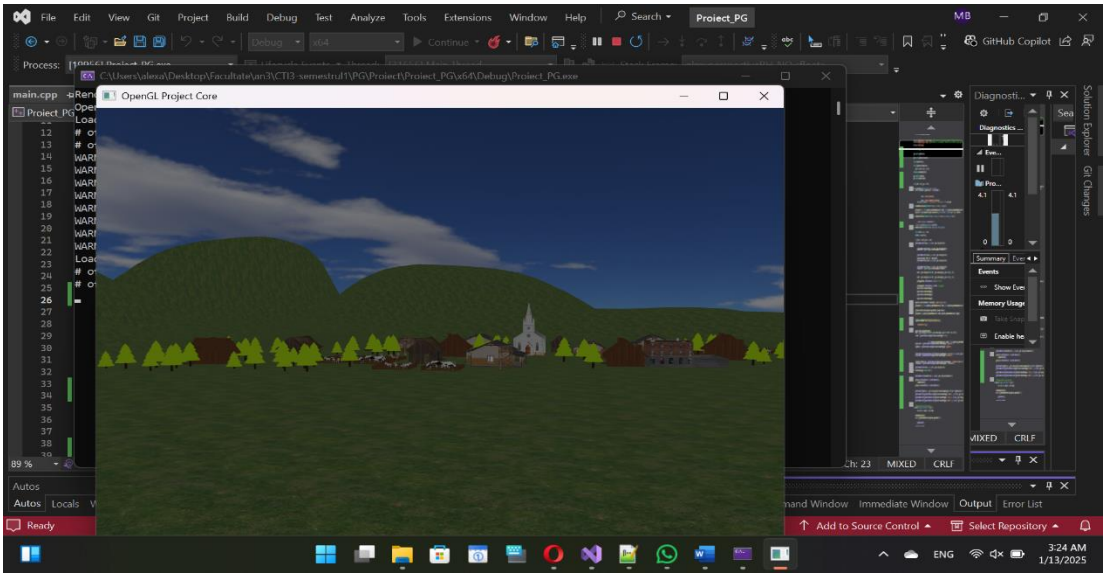
```
glm::vec3(0.0f, 0.0f, -10.0f), // directia in care priveste

glm::vec3(0.0f, 1.0f, 0.0f)   // vectorul "sus");
```

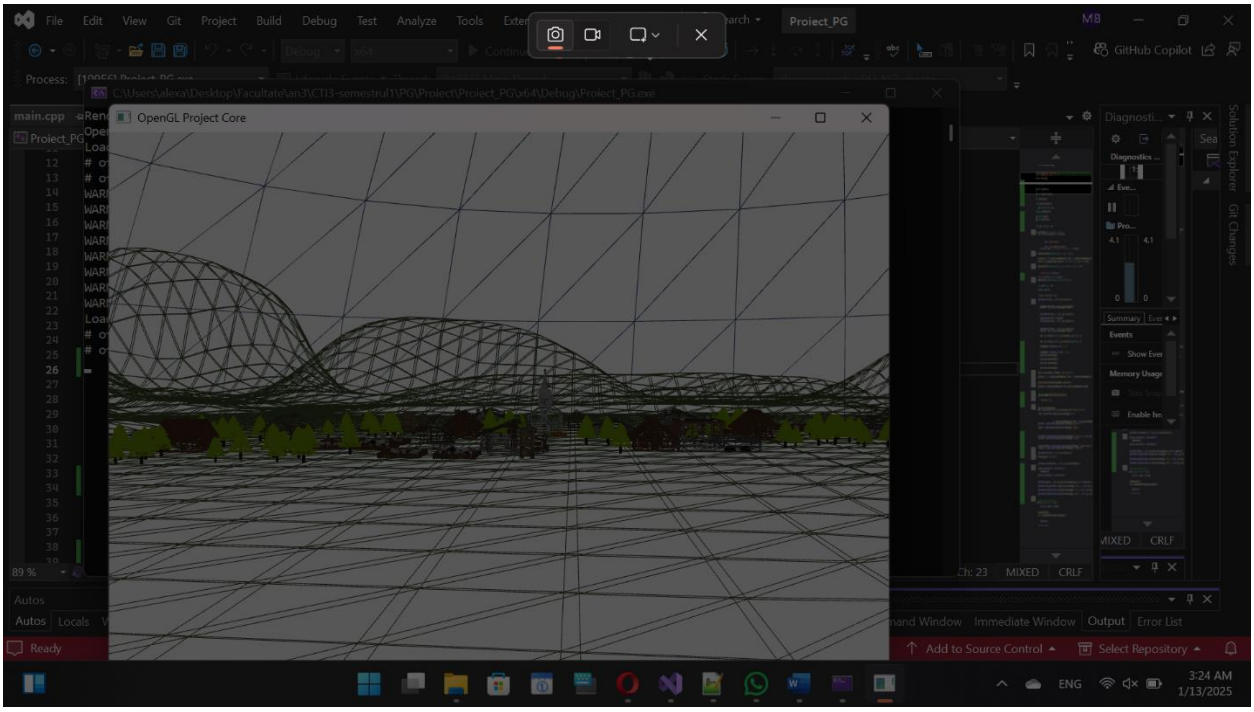
De asemenea, miscarile camerei sunt gestionate prin functii specifice pentru deplasare in fata, spate, lateral si pentru rotatii.

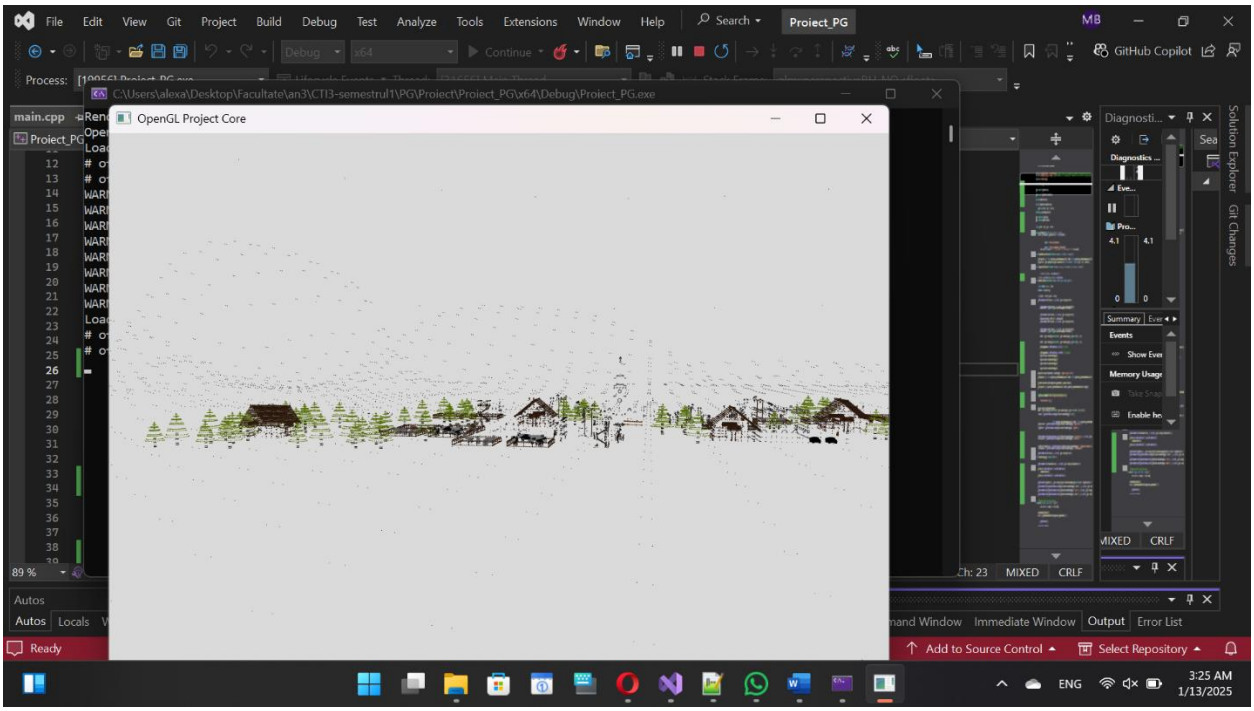
Ierarhii de clase

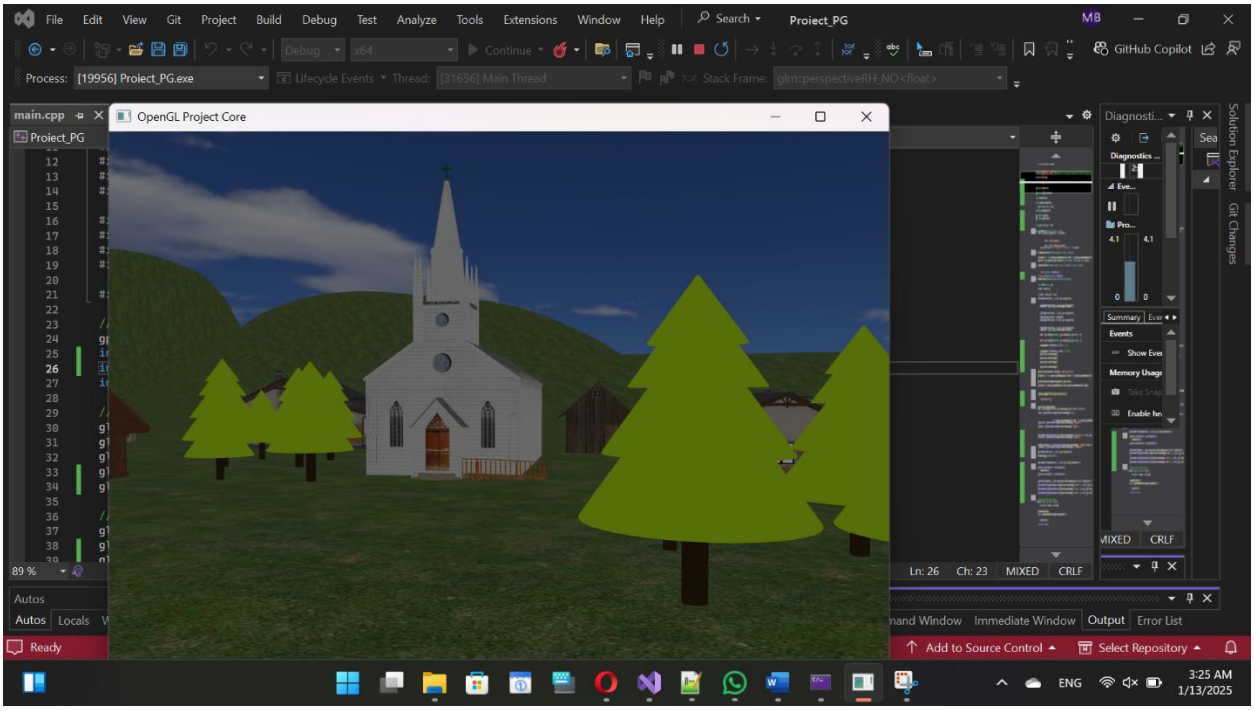
Proiectul este structurat modular, fiecare clasa avand un rol bine definit pentru a simplifica gestionarea codului si a oferi flexibilitate. Clasa **Camera** gestioneaza pozitia, orientarea si miscarea camerei, oferind o perspectiva interactiva asupra scenei. Clasa **Window** se ocupa de crearea si stergerea ferestrei OpenGL, inclusiv gestionarea evenimentelor de input si eliberarea resurselor. **Mesh** este responsabila pentru desenarea obiectelor, configurarea bufferelor si gestionarea datelor geometrice, in timp ce **Shaders** implementeaza functionalitati pentru incarcarea, compilarea si utilizarea shaderelor, permitand personalizarea efectelor vizuale. Bibliotecile externe **stb_image** si **tiny_obj_loader** completeaza proiectul, facilitand incarcarea texturilor si a modelelor 3D. Organizarea modulara, cu fisiere header pentru fiecare clasa, imbunatateste lizibilitatea, extensibilitatea si testarea codului, integrand toate componentele prin clasa main.

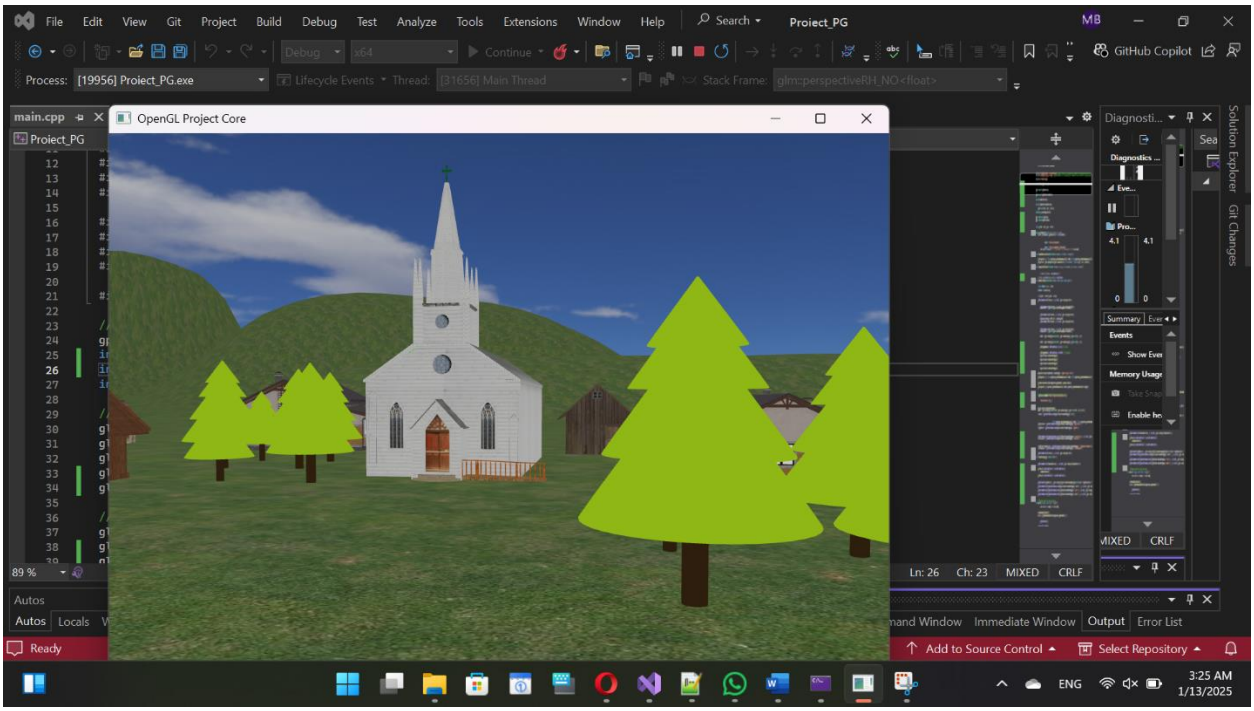
Prezentarea interfetei grafice

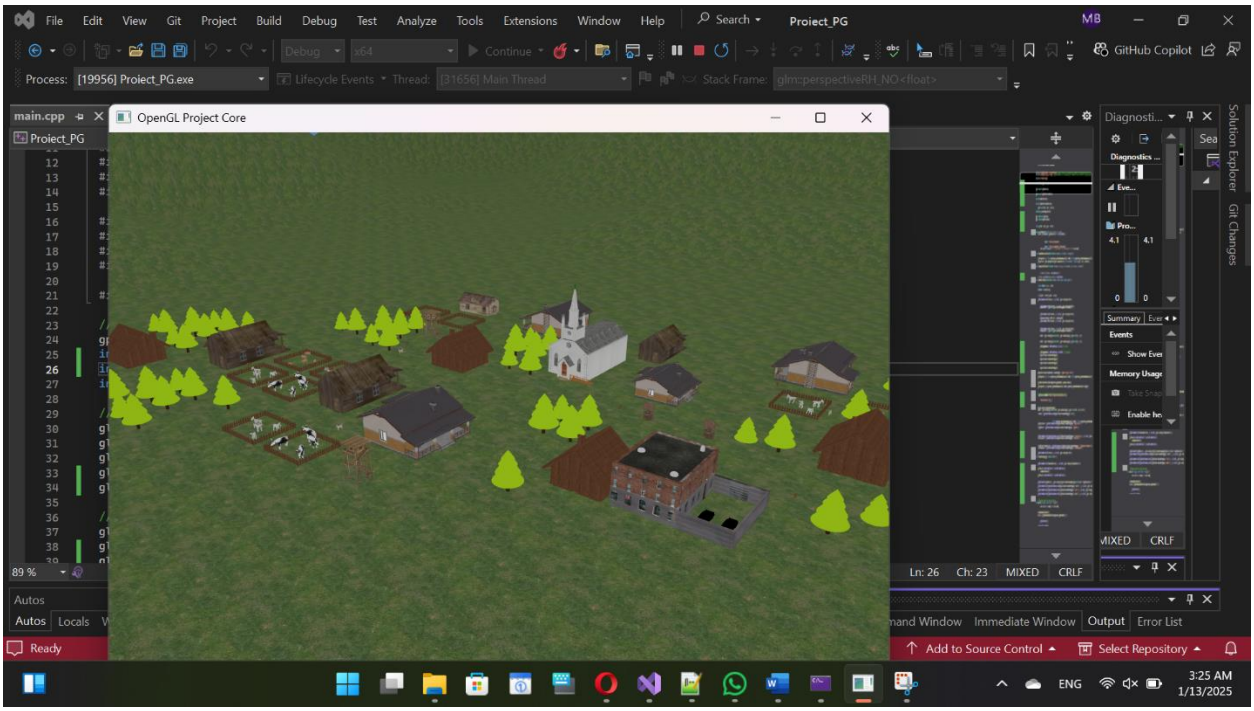


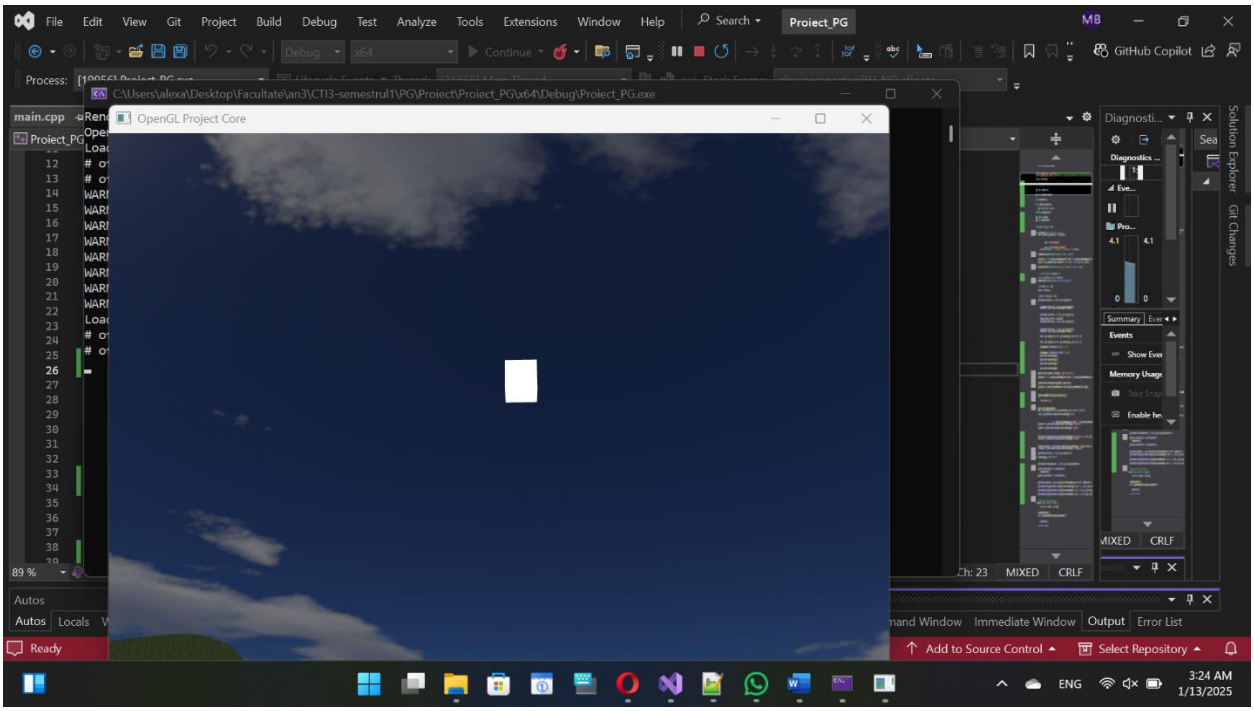



OpenGL Project Core
Proiect_PG
GitHub Copilot
Ready
Add to Source Control
Select Repository
3:25 AM
1/13/2025


OpenGL Project Core
Proiect_PG
GitHub Copilot
Ready
Add to Source Control
Select Repository
3:25 AM
1/13/2025


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
Proiect_PG
GitHub Copilot
Process: [19956] Proiect_PG.exe
main.cpp
Proiect_PG
OpenGL Project Core
Ln: 26 Ch: 23 MIXED CRLF
Autos Locals
Output Error List
Ready
Add to Source Control
Select Repository
ENG
3:25 AM
1/13/2025


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
Proiect_PG
GitHub Copilot
Process: [19956] Proiect_PG.exe
main.cpp
Proiect_PG
OpenGL Project Core
Ln: 26 Ch: 23 MIXED CRLF
Autos Locals
Output Error List
Ready
Add to Source Control
Select Repository
ENG
3:25 AM
1/13/2025



Manual de utilizare

Pentru a rula aplicatia, este necesar sa o deschideti prin intermediul unui fisier executabil sau al unui terminal, in functie de sistemul de operare utilizat. Odata deschisa, scena va fi incarcată automat, iar utilizatorul va putea explora satul virtual.

- W - deplasare camera in fata
- S - deplasare camera in spate
- A - deplasare camera in stanga
- D - deplasare camera in dreapta
- 1 – vizualizarea scenei in modul solid
- 2 – vizualizarea scenei in modul wireframe
- 3 – vizualizarea scenei in modul polygonal
- K – aprindere lumina punctiforma
- L – stingere lumina punctiforma

Concluzii

In concluzie, acest proiect m-a ajutat sa dobandesc cunostinte solide despre OpenGL si Blender, aplicand tehnici esentiale de grafica 3D, cum ar fi gestionarea ferestrei, iluminarea si umbrele, si miscarea camerei. Dificultatile intampinate au contribuit la aprofundarea intelegerii acestor tehnologii. Proiectul mi-a oferit o perspectiva mai clara asupra procesului de creare a unei experiente vizuale interactive intr-un mediu 3D.

Referinte

[1] - https://learnopengl.com/

[2] - https://free3d.com/

[3] – Laboratoare PG