# *PRELUCRARE GRAFICA*

# *~PROIECT~*

Student: Adăscăliței Andra

Grupa: 30234

Profesor Coordonator: C. Nandra

# Cuprins

# 1. Prezentarea temei

Proiectul are ca scop realizarea unei scene 3D, fotorealiste, utilizând librăriile OpenGL, GLFW și GLM pentru redarea și manipularea în timp real. Scena poate fi explorata prin intermediul interacțiunii utilizatorului, folosind mousse-ul și tastatura. Proiectul urmărește simularea unei scene cât mai apropiate de realitate, incluzând diverse surse de lumină, materiale texturate și animații dinamice ale obiectelor.

# 2. Scenariul

## 2.1. Descrierea scenei și a obiectelor

Scena realizată este o reprezentare a unui sat rural modern, care îmbină elemente naturale și arhitecturale, oferind o experiență vizuală detaliată. Aceasta combină structuri statice cu obiecte animate, oferind un peisaj dinamic care poate fi explorat.

Scena este construita pe un teren verde, străbătut de un drum luminat de mai multe felinare, de-a lungul căruia sunt amplasați mai mulți stâlpi. Pe drum sunt plasate diverse vehicule statice.  Scena include un elicopter funcțional, care adaugă un plus de dinamică.

Elicopterul este însoțit de un sunet specific provocat de învârtirea elicei. poate fi controlat pentru a decola și ateriza pe un punct de coborâre de tip heliport. De asemenea, în partea stângă a scenei este construită o cale ferată cu doua șine de tren unde regăsim un tren static și unul ce se mișcă înainte și înapoi, realismul fiind accentuat si de efectul sonor. Scena conține mai multe tipuri de clădiri ce redau diversitatea arhitecturală a unei localități rurale. Observăm case pe o parte și cealaltă a drumului ce traversează satul, clădiri vechi, o clădire industrială ce reprezintă o fabrică veche si o biserică în spatele căreia se află un cimitir. În scena observăm si un turn de apă amplasat într-o parte a scenei, adăugând un detaliu specific comunităților rurale. Un lac de dimensiuni considerabile este amplasat în scena, completat de nuferi și de un pod de lemn care traversează apa. În marginea scenei, lângă lac mai observam si un parc, destinat copiilor, cu diverse echipamente de joaca, precum tobogane și bănci. Scena este populata de copaci și vegetație, precum și de o fântână, iar în fața unei case observăm o cușcă de câine si un câine.

Toata aceasta scena este cuprinsă într-un skybox, ce accentuează ideea de continuitate si realism, adăugând un element important, cerul.

## 2.2. Funcționalități

Utilizatorul  poate naviga prin scenă cu ajutorul tastaturii, folosind tastele W, A, S, D pentru direcțiile standard înainte, stânga, înapoi și respectiv dreapta, precum si de a schimba direcția prin folosirea mousse-ului.

Scena are implementată și funcționalitatea de „automated tour” prin apăsarea tastei T, oferind utilizatorului un tur automat prin cadru și prezentându-i cele mai importante puncte.

Scena noastră este iluminată atât de lumina direcțională (de la soare), cât si de cele punctiforme (felinarele) și spot (farurile uneia dintre mașini), care se pot activa di dezactiva de la taste.

In ceea ce privește modurile de vizualizare, scena poate fi văzută în mod SOLID, WIREFRAME și PUNCTIFORM.

Se poate activa ploaia, tot prin apăsarea unei taste, insoțită de efectul sonor specific.

De asemenea, asupra scenei poate fi activat si efectul de ceață prin intermediul tastaturii.

Elicopterul are o elice ce se rotește, iar la input-ul utilizatorului poate decola pană la o anumită înălțime, iar prin reapăsarea tastei poate ateriza.

# 3. Detalii de implementare

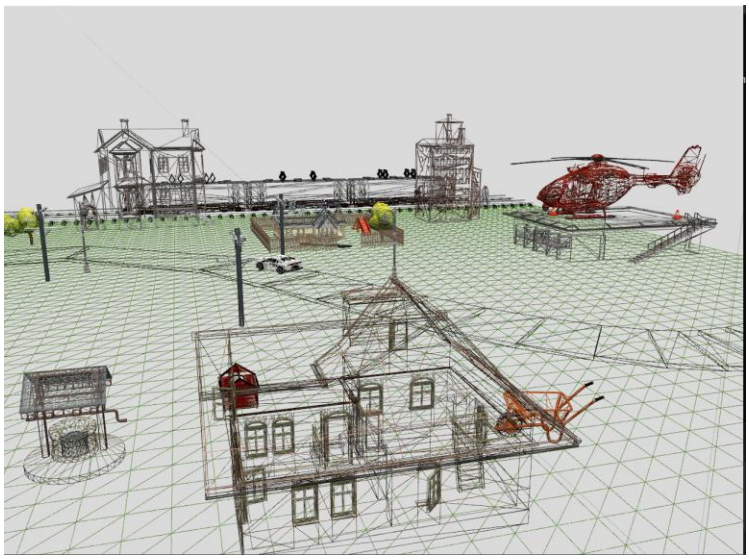
## 3.1. Funcții și Algoritmi

De-a lungul proiectului am implementat mai multe  funcții ce facilitează interacțiunea user-scenă, precum si funcții ce ajută la efectul fotorealist al planului, in fisierul main regăsindu-se cele mai multe dintre ele. Am implementat funcțiile keyboardCallback(), mousseCallback() și processMovement(), pentru a oferi utilizatorului abilitatea de a naviga ușor prin scena, în orice punct își dorește.

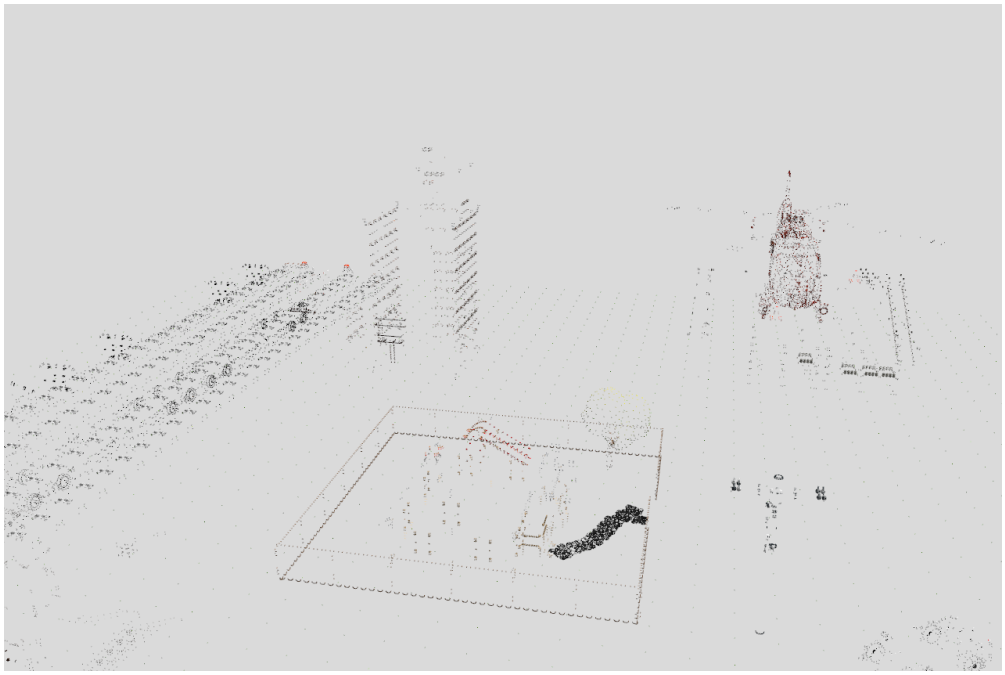
**Moduri de vizualizare**

Implementarea proiectului oferă utilizatorului 3 posibilități de vizualizare a scenei: solid, wireframe și punctiform. Pentru acestea, am folosit funcția glPolygonMode, alături de parametrul GL_FRONT_AND_BACK care specifica ce față a poligonului va fi desenată, alături de atributul ce specifica modul dorit:

- GL_LINE – modul WIREFRAME (se afișează doar muchiile poligoanelor)



- GL_POINT – modul PUNCTIFORM (afișează doar vârfurile poligoanelor)



- GL_FILL – modul SOLID (afișează suprafețele poligoanelor, cu texturi și iluminare)

**Lumini**

Iluminarea este implementată utilizând mai multe tipuri de surse de lumină, fiecare dintre ele având un algoritm specific de calcul al contribuției sale asupra fiecărui pixel (fragment) din scenă. Luminile sunt implementate în shader si sunt de 3 tipuri: lumina directională, lumina punctiformă și lumină spot. Am implementat aceste lumini, folosindu-mă de materialele de laborator precum si de noțiuni de pe site-ul learnOpenGl.com.

**Lumina directională** simulează lumina care vine de la o sursă îndepărtată, precum soarele, venind dintr-o direcție fixă, fără a-și pierde intensitatea în funcție de distanță. Funcția

computeDirLight() calculeaza componentele de iluminare necesare, adică componentele **ambientală**, **specular**ă și **difuză**.

**Lumina punctiformă** emite lumină în toate direcțiile dintr-un punct dat și își pierde intensitatea pe măsură ce distanța crește.  Pentru a implementa această lumină am creat o structura pentru a declara parametrii necesari, ce urmează a fi trimiși în shader.

```
struct PointLight {
    vec3 position;

    float constant;
    float linear;
    float quadratic;

    vec3 ambient;
    vec3 diffuse;
    vec3 specular;
};
```

Atenuarea este un aspect important al iluminării punctiforme. Intensitatea luminii scade pe măsură ce distanța dintre sursa de lumină si pixel crește.

Se poate observa lumina punctiforma la felinare:

**Lumina Spot** este o lumină ce se emite sub forma unui con de iluminare. Aceasta este utilă pentru simularea în proiectul meu, a luminii generate de farurile unei mașini. Pentru a implementa aceasta lumină am extins structura luminii Point prin adaugarea de noi parametri.

```
struct Light {
    vec3 position;
    vec3 direction;     //pentru spot light
    float cutOff;       // pentru spot light
    float outerCutoff;  //pentru spot light

    float constant;
    float linear;
    float quadratic;

    vec3 ambient;
    vec3 diffuse;
    vec3 specular;
};
```

In aceasta nouă structură, parametrul cutOff se referă la unghiul interior al conului de iluminare, unde lumina este cea mai puternică, iar parametrul outerCutOff reprezintă unghiul exterior al conului de lumină, în afara căruia lumina scade treptat. Lumina de tip spotlight are un comportament special care depinde de unghiul dintre direcția luminii și vectorul către fragmentul iluminat. Am folosit aici parametrii theta si epsilon pentru calculul intensității, unde theta ține cont de poziție și direcție, iar epsilon de cele doua unghiuri ale conului.

**Implementarea turului automat**

Am implementat un tur automat al camerei prin scena, folosind o lista de puncte de referință prin care trece camera în timpul animației. Camera se deplasează între aceste puncte prin interpolare liniară. Camera merge în linie dreapta de la un punct la altul datorită implementării ecuației liniare parametrice.

P0 * (1 - t) + P1 * t;

Turul automat poate fi activat la apăsarea unei taste specifice (T) și se întrerupe automat când camera ajunge la ultimul punct din vectorul de puncte.

**Implementarea ploii**

Ploaia din scena 3D este implementată utilizăand un sistem de particule, unde fiecare picătură de ploaie (picătura este un .obj texturat si modelat în Blender) este reprezentată de o particulă individuală cu poziție si viteză aleatorii. Funcția initRainDrops inițializează pozițiile și viteza picăturilor de ploaie, iar updateRainDrops actualizează continuu pozițiile acestora pe măsură ce cad. Când o picătură ajunge la sol, aceasta este resetată într-o poziție aleatorie, simulând un flux continuu de ploaie.

**Implementarea SkyBox-ului**

In cadrul proiectului, cerul este pus in scena prin intermediul SkyBox-ului. Proiectul foloseste 2 variante de skybox: unul de zi si unul de noapte. Skybox-urile sunt create utilizând texturi predefinite aplicate pe un cub care înconjoară întreaga scenă, oferind astfel o perspectivă infinită asupra cerului. Am implementat shadere-ele corespunzatoare, iar utilizatorul poate schimba modurile din taste. Am scazut si lumina (lightColor) pentru un efect mai realist.

**Implementarea efectului de ceata**

Utilizatorul poate aplica efectul de ceață asupra scenei, folosind algoritmul implementat in funcția computeFog din fragment shader. Funcția calculează factorul de ceață pe baza distanței dintre fragment si camera de vizualizare si densitate. Dacă efectul de ceață este activat (fogEnabled == 1), culoarea finală a fragmentului este un amestec între culoarea calculată și o culoare de ceață, în funcție de acest factor. Astfel, efectul creează un grad de estompare vizuală care devine mai intens pe măsură ce distanța față de observator crește.

## 3.2. Modelul Grafic

Scena a fost modelată folosind Blender și importând acolo obiecte descărcate de pe internet in format .obj, asupra cărora am adăugat texturi. Texturile sunt imagini .jpg sau .png descărcate odată cu obiectele, făcute in Paint sau descărcate de asemenea de pe internet. Blender oferă o mulțime de funcționalități care te ajută sa muți obiectele oriunde in scenă, sa le rotești/scalezi/translatezi si sa le așezi așa cum iți dorești. Obiectele dinamice au fost exportate direct in OpenGL in format .obj alături de .mtl si textura. Acestea au fost plasate in scena prin aplicarea transformărilor necesare pentru ca obiectul să ajungă in locul dorit. Scena din Blender a fost ulterior exportată, iar folosindu-ne de formatele .obj si .mtl alături de toate texturile folosite am putut sa punem scena in OpenGL.

## 3.3. Structuri de date

În proiectul meu am utilizat o structură de date pentru lumini, utilizată in shader-ul fragment. Structura Light contine toate informațiile necesare pentru calcularea efectelor de iluminare pe obiectele din scenă. Aceasta conține parametrii pentru pozitia luminii, pentru directia acesteia, precum si anumiți parametri specifici pentru lumina punctiformă necesari in calcul ( constant, linear, quadric). Cel mai important, aceasta structura conține componentele luminii, respectiv ambientala, difuza și speculara, care determina modul în care lumina interacționează cu obiectele.

## 3.4. Ierarhia de clase

Pe lângă clasa main.cpp care se ocupă în mare parte de implementarea scenei, proiectul utilizează OpenGL pentru a desena si manipula obiecte 3D. Clasa **Mesh** gestionează informațiile despre modelele 3D, incluzând vârfuri, indicii și texturi, și le desenează folosind buffer-e OpenGL. Clasa **Camera** permite mișcarea și rotirea camerei, oferind metode pentru actualizarea poziției și obținerea matricei de vizualizare. Clasa **Model3D** se ocupă cu încărcarea și desenarea modelelor 3D din fișiere .obj, folosind **tinyobjloader** pentru parsing și **stb_image** pentru încărcarea texturilor. Așadar, ierarhia de clase din proiectul C++ facilitează gestionarea eficientă a elementelor 3D într-o scenă OpenGL.

# 4. Prezentarea interfeței grafice utilizator / manual de utilizare

Aplicația trebuie rulată fie din Visual Studio, fie prin intermediul executabilului. Odată ce aplicația este rulata, scena va apărea pe ecran. Cursorul este dezactivat pentru a facilita mișcarea prin scena. Pentru a naviga prin scena, utilizatorul trebuie sa folosească cateva taste importante si mousse-ul.

Manual de utilizare

- Mișcare înainte - **Tasta W**
- Mișcare înapoi - **Tasta S**
- Mișcare spre stânga - **Tasta A**
- Mișcare spre dreapta - **Tasta D**
- Închidere aplicație - **Tasta Esc** (Escape)
- Activarea ceții - **Tasta** F
- Activare lumina direcțională - **Tasta L**
- Activare lumina punctiformă - **Tasta P**
- Activare lumina spot - **Tasta O**
- Activare ploaie + sunet ploaie - **Tasta R**
- Activare Night Mode - **Tasta N**
- Activare Day Mode (by default) - **Tasta M**
- Vizualizare WireFrame - **Tasta K**
- Vizualizare punctiform - **Tasta J**
- Vizualizare Solid - **Tasta I**
- Animatie Elicopter decolare/aterizare - **Tasta H**
- Automatic Camera Tour - **Tasta T**

## 5. Concluzii și dezvoltări ulteriore

Datorită acestui proiect am înțeles mai bine cum sa folosesc biblioteca OpenGL si de asemenea am învățat sa folosesc programul Blender si să mă familiarizez cu facilitățile pe care le oferă. Acest proiect mi-a testat creativitatea, mai ales in etapa de creare a scenei. Am reușit să implementez o scenă 3D dinamică, cu control al camerei, iluminare realistă, texturare și animații ale obiectelor.

În continuare, există mai multe direcții de dezvoltare care ar putea îmbunătăți semnificativ realismul și interactivitatea scenei 3D, cum ar fi implementarea umbrelor (shadow mapping), implementarea reflexiilor si a transparentei obiectelor, efecte mai complexe cum ar fi efectul de fum, foc, explozii, implementarea ninsorii si poate chiar mărirea sau adăugarea unor plane noi.

# 6. Referințe


- https://learnopengl.com/Lighting/Light-casters
- Laboratoare Prelucrare Grafica
- https://www.youtube.com/playlist?list=PLrgcDEgRZ_kndoWmRkAK4Y7ToJdOf-OSM