

**Calculatoare și tehnologia informației**

**Proiect Prelucrare Grafica**

**Laborant Proiect realizat de**

**Sonia Eufrosina Maria Grigor Codrescu Răzvan-Gabriel**

**Grupa 30234**

Contents

[**2. Prezentarea temei** 3](#_Toc61214630)

[**3. Scenariul** 3](#_Toc61214631)

[**3.1.Descrierea scenei si a obiectelor** 3](#_Toc61214632)

[**3.2.Functionalitati** 3](#_Toc61214633)

[**4.Detalii de implementare** 4](#_Toc61214634)

[**4.1. Functii si algoritmi** 4](#_Toc61214635)

[**4.2. Modelul grafic** 7](#_Toc61214636)

[**4.3. Structuri de date** 9](#_Toc61214637)

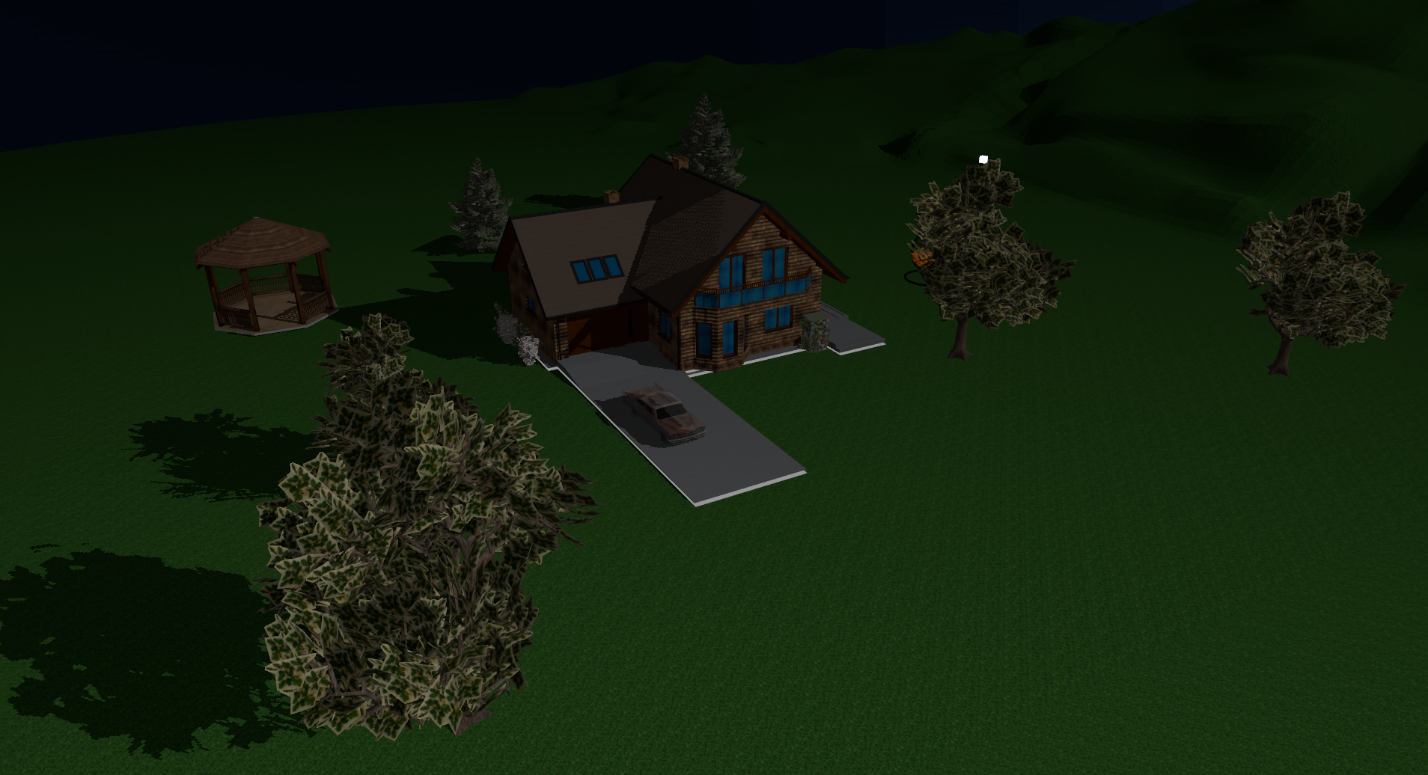
[**4.4. Ierarhia de clase** 9](#_Toc61214638)

[**5. Prezentarea interfetei grafica utilizator / manual de utilizare** 10](#_Toc61214639)

[**6. Concluzii si dezvoltari ulterioare** 11](#_Toc61214640)

[**7. Referinte** 11](#_Toc61214641)

# **2. Prezentarea temei**



Tema proiectului este reprezentarea modelului tridimensional a unui obiect, obiectul ales de mine fiind o casa mobilata si cu gradina.

# **3. Scenariul**

# **3.1.Descrierea scenei si a obiectelor**

In cadrul gradinii avem un inceput de deal in partea dreapta, realizat in blender cu ajutorul functiei de “Sculpt”, ca si obiecte separate sunt : casa impreuna cu gradina exportate sub un singur obiect denumit scena, masina din fata garajului, cubul de iluminat de deasupra casei si leaganul din partea dreapta a scenei. In cadrul obiectului “Scena” avem un foisor situat in partea stanga, in spatele casei si 6 copaci (4 copaci comuni si 2 brazi).

# **3.2.Functionalitati**

Ca si functionalitati in cadrul aplicatiei s-au implementat urmatoarele :

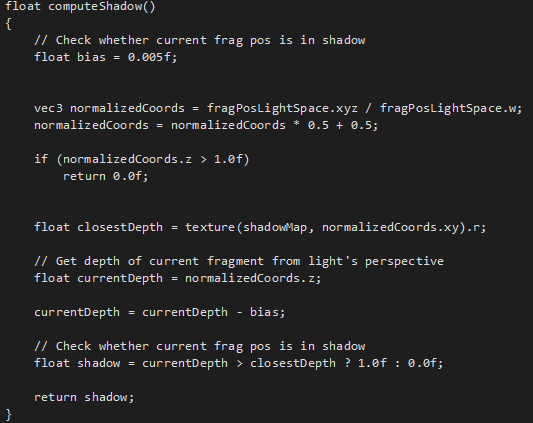
* Posibilitatea ca utilizatorul sa se deplaseze prin scena pe toate cele 3 axe cu ajutorul operatiilor de translatie si rotatie, miscarea putand fi realizata fie cu ajutorul tastelor, fie cu ajutorul mouse-ului.
* Posibilitatea de vizualizare a scenei prin intermediul unei animatii de prezentare, realizata prin pozitionarea camerei in scena si prezentarea ei.
* Posibilitatea de a vizualiza scena in cele 3 moduri precizate : fill, wireframe, punctiform.
* Posibilitatea de a controla sursa punctiforma de lumina pentru a vedea procesul de creare al umbrelor.
* Transparenta masinii. Masina trece de la transparenta la opacitate in continuu.
* Posibilitatea aparitiei cetii sau a ninsorii prin apasarea butoanelor corespunzatoare.
* Posibilitatea de deplasare a masinii pe axa Z.

# **4.Detalii de implementare**

# **4.1. Functii si algoritmi**

Un prim algoritm este **Shadow Mapping**. Acest algoritm este necesar pentru realizarea umbrelor pentru fiecare obiect din scena. Acest algoritm are 2 etape prin care va trece :

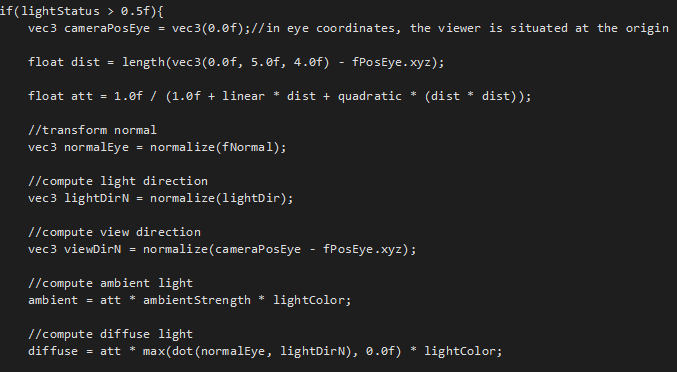
1. Incarcarea scenei din punctul de vedere al sursei de lumina. In cadrul acestei etape, singurul lucru relevant este tabelul cu valori de adancime. Aceste date sunt stocate in harta de adancime, fiind obtinute prin atasarea unei texturi de adancime unui obiect.
2. Incarcarea scenei din punct de vedere al observatorului. Se compara fragmentele de adancime a fragmentelor vizibile cu valorile din harta de adancime. Fragmentele cu o adancime mai mare decat cea anterioara. Fragmentele cu adancime mai mare nu sunt direct vizibile din perspectiva luminii asa ca se vor afla in umbra.



**Bias** este pus in aplicatie pentru a rezolva problema esantionarii aceleasi valori din harta de adancime de catre fragmente diferite.

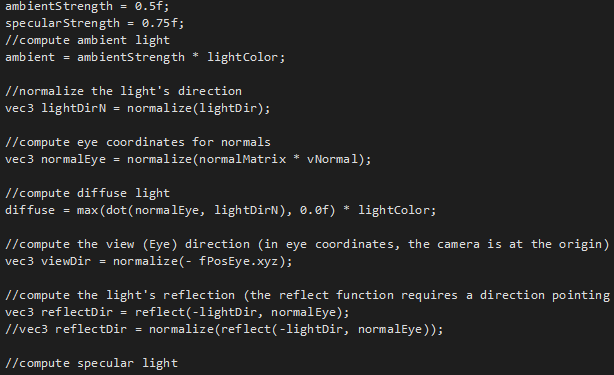
**Lumina punctiforma** este o sursa de lumina cu o pozitie data care ilumineaza radial si uniform in toate directiile.

Implementarea **luminii punctiforme** a fost implementata cu ajutorul algoritmului Blinn-Phong, reprezentat in laborator. Acesta este o versiune superioara a algoritmului Phong imbunatatind performanta prin evitarea calculului vectorului de reflexie.



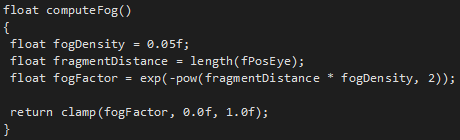
In cadrul luminii punctiforme, att are ca scop atenuarea luminii asupra obiectelor, prin urmare, acesta se va inmulti cu compomentele ambient, diffuse si specular.

Pentru **lumina ambientala** am folosit algoritmul Gourand din cadrul laboratorului. Pentru alegerea luminii care urmeaza sa fie folosite a fost realizat un if in functia main din shader, valoarea din if transferandu-se de din aplicatia principala sub forma uniform. Valorile folosite pentru aceasta sunt diferite de cele din laborator : ambientStrength de 0.5, specularStrength de 0.75 si shininess de 32. In cadrul main-ului se va calcula puterea luminii, directia de propagare si se va ajusta dupa directia privitorului.



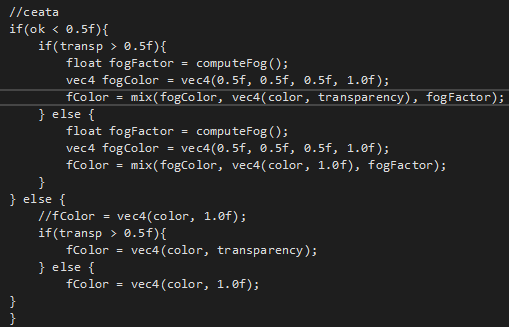
Pentru realizarea cerului am folosit un **skydome**, conferind astfel acestuia o forma rotunjita.

Algoritmul de implementare a **cetii** a fost realizat cu ajutorul tutorialului din cadrul laboratorului. Astfel, ceata are o valoare pentru densitatea acesteia si se calculeaza in mod exponential patratic in cadrului unui parametru **fogFactor**. Rezultatul returnat se va calcula cu ajutorul functiei **clamp** care va mentine valoarea returnata intre un minimum si un maximum.



Realizarea **tansparentei** se face prin intermediul a doi parametri transmisi din main catre shader. Un prim parametru este nivelul de transparenta, pentru a ajusta cat de transparent sa fie obiectul iar un al doilea parametru este cand sa se transmita gradul de transparenta ca parametru al culorii(deoarece nu vrem ca toate obiectele sa fie transparente). De asemenea pentru transparenta am avut nevoie de pornirea urmatoarelor functii :

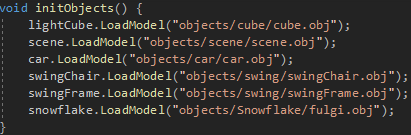
* glEnable(GL\_BLEND);
* glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA);



In main avem if-ul cu ok care verifica daca este ceata, iar transp care verifica ca obiectul curent sa fie cel ce trebuie desenat transparent.

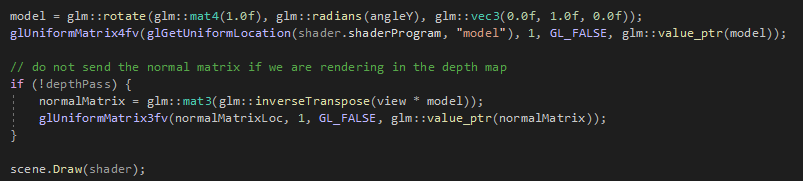
Am realizat cele 3 moduri de prezentare a scenei si anume : fill, wireframe si punctiform.

# **4.2. Modelul grafic**



In cadrul proiectului am importat 5(6 daca ar fi sa luam leaganul ca 2 componente diferite) obiecte diferite :

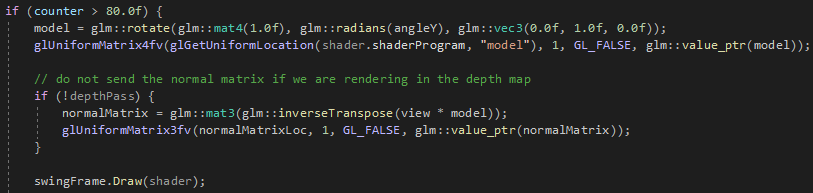
1. **Scena**



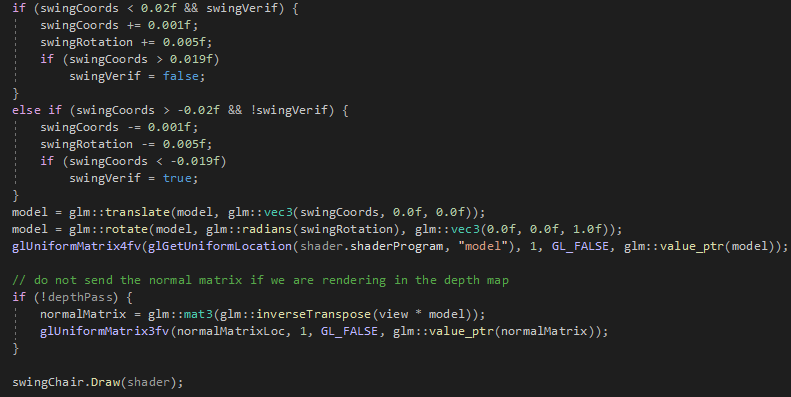
Scena se va desena simplu, aplicand o rotatie asupra unei matrici unitate si apoi se va trimite matricea model in shader.

1. **Leaganul**

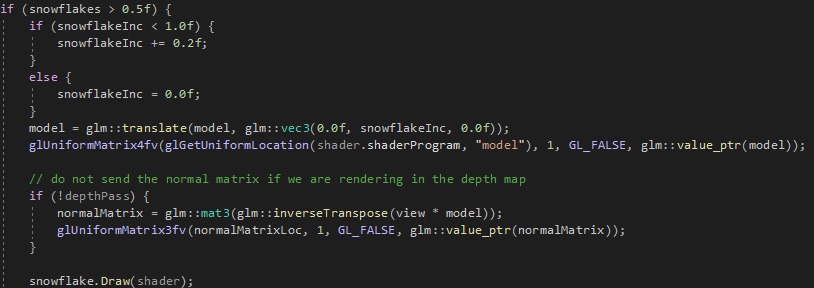
Leaganul este desenat cu ajutorul a doua obiecte : rama acestuia si scaunul. De precizat este faptul ca pentru a genera acest obiect in mod dinamic am initializat la inceputul lui renderScene() un counter care sa ia valoarea actuala a timpului de la rularea aplicatiei. In cazul in care s-a ajuns la 80 se va desena leaganul. Mai intai se va desena componenta statica a acestuia si anume rama.



Dupa care, se va desena scaunul. Pentru ca obiectul sa se miste in permanenta, vom incrementa / decrementa valoarea coordonatelor pentru rotatia si translatia acestuia. In cazul in care se ajunge la limita dorita de miscare, vom folosi o variabila de tip bool pentru a opri legeanul din a mai merge in directia impusa, si pentru a schimba sensul.

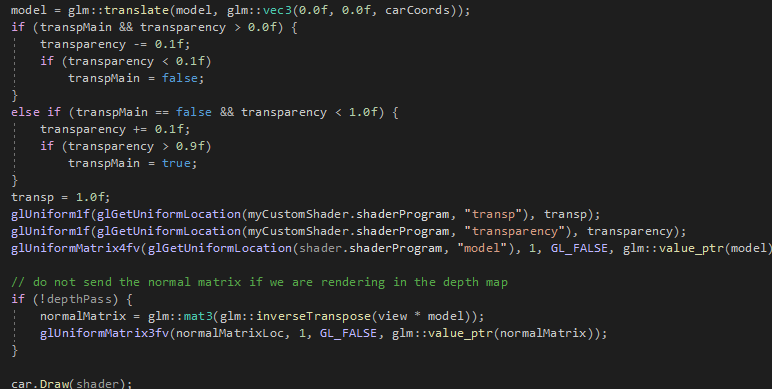


1. **Ninsoarea**



Pentru animatia de ninsoare vom avea o variabila care sa verifice daca butonul corespunzator a fost apasat. In cazul in care a fost apasat, obiectul va face o translatie pentru a simula animatia de cadere a fulgilor de zapada.

1. **Masina**

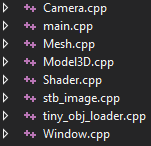


Masina va fi obiectul care se va translata prin apasarea tastelor. Asadar, translatia se va face la valoarea CarCoords. De asemenea, am ales ca efectul de transparenta sa fie implementat pe masina, asadar vom avea ca si la animatie leaganului, un doua if-uri prin care sa verificam daca vrem ca obiectul sa fie transparent si cat de transparent sa fie. Avand in vedere ca vorbim despre transparenta, aceasta va scala intre 0 si 1. In momentul in care ajunge intr-un capat, va porni catre valoarea opusa. Valorile de adevar si de transparenta se vor transmite catre shader si la final, se va face 0 si se va transmite iar, pentru a evita posibilitatea ca celelalte obiecte sa fie transparente. Dupa cum a fost precizat si in laborator, pentru ca ordinea de rasterizare este foarte importanta masina a fost ultimul obiect desenat.

# **4.3. Structuri de date**

Ca si structuri de date in mare parte au fost folosite obiecte de tipurile prezentate in ierarhie, si vec3, vec4 si matrici de diverse dimensiuni, in functie de necesitate si utilizare.

# **4.4. Ierarhia de clase**

Ca si clase ale proiectului avem urmatoarele clase :

* Camera.cpp care se ocupa de toate actiunile camerei.
* Main.cpp unde avem aplicatia principala.
* Window.cpp care se ocupa cu fereastra pentru deschiderea aplicatiei.
* Shader.cpp unde sunt realizate functii pentru toate operatiile care se pot executa asupra unui shader.
* Mesh.cpp care incarca texturile pentru fiecare obiect din cadrul scenei.
* Model3D.cpp care executa actiunile de incarcare, citire si de incarcare a texturii pentru fiecare obiect .obj situat in scena noastra.

# **5. Prezentarea interfetei grafica utilizator / manual de utilizare**

Dupa cum am precizat si la inceput, scena poate fi vizualizata in 3 moduri :

* Prin apasarea tastei 1 scena se poate viziona in modul wireframe;
* Prin apasarea tastei 2 scena se poate viziona in modul punctiform;
* Prin apasarea tastei 3 scena se poate viziona in modul fill.

Utilizatorul se poate deplasa prin cadrul scenei fie cu ajutorul mouse-ului sau cu ajutorul tastaturii.

Cu ajutorul mouse-ului, directia mouse-ului va reprezenta directia de privire a camerei iar marginile au fost fixate in asa fel incat in urma rotirii camerei, camera sa nu poata sa se dea peste cap sau sa nu poata face rotiri mai mari de un unghi de 90 de grade, simuland astfel capacitatea de privire a unui cap uman.

Cu ajutorul tastelor, vom avea urmatoarele posibilitati :

* W – pentru miscarea camerei pe directia inainte;
* S – pentru miscarea camerei pe directia inapoi;
* A – pentru miscarea camerei catre stanga;
* D – pentru miscarea camerei spre dreapta;
* Q – pentru rotirea camerei catre stanga;
* E – pentru rotirea camerei catre dreapta;
* Sageata sus – pentru miscarea camerei pe orizontala, in sus;
* Sageata jos – pentru miscarea camerei pe orizontala, in jos;

Pentru a simula coliziunea cu pamantul, camera nu poate cobori sub nivelul acestuia.

Pentru miscarea masinii, utilizatorul poate apasa tastele :

* Sageata stanga – pentru miscarea masinii in fata;
* Sageata dreapta – pentru miscarea masinii in spate.

Prin apasarea acestei taste se va incrementa / decrementa pozitia de translatie a masinii, dar nu la infinit, limitandu-se miscarea masinii in spate si in fata pentru a nu iesi de pe asfalt sau pentru a nu se izbi in garaj.

Pentru miscarea sursei de lumina punctiforme si studierea umbrelor din diverse unghiuri s-a folosit :

* J – pentru miscarea sursei catre stanga;
* L – pentru miscarea sursei catre dreapta.

Sursa de lumina poate initializa o miscare de 360 de grade in jurul casei.

Pentru realizarea altor operatii au fost folosite tastele urmatoare :

* P – pentru pornirea / oprirea ninsorii;
* O – pentru pentru pornirea animatiei de prezentare;
* G – pentru alternarea surselor de lumina;
* F – pentru pornirea / oprirea cetii.

# **6. Concluzii si dezvoltari ulterioare**

In urma realizarii acestui proiect pot afirma faptul ca am invatat notiunile de baza despre animarea obiectelor in scena si alte efecte precum ceata, umbre, transparenta si pozitionarea luminilor intr-o scena. Consider ca aceste lucruri sunt esentiale pentru orice persoana care doreste sa lucreze pe aceasta ramura a realizarii de scene / jocuri, avand in vedere toti acesti algoritmi prezentati si beneficiile lor si as putea spune ca a fost chiar o placere sa lucrez la acest proiect, fiind o persoana pasionata de astfel de subiecte.

Ca in orice proiect, mereu este loc de mai bine asadar, in cadrul acestuia consider ca se pot aduce urmatoarele imbunatatiri :

* Realizarea mai multor lumini (ex. felinare) pentru a aduce acest proiect cat mai aproape de realitate;
* Conturarea mai bine a umbrelor, avand in vedere ca acestea se afla in stadiul in care apar foarte multe colturi;
* Realizarea coliziunilor pentru mai multe obiecte(momentan doar pamantul are o astfel de mecanica, avand in vedere ca in casa vrem sa intram sa vedem ce se afla prin ea cu usurinta);
* Realizarea animatiilor pentru mai multe obiecte din scena(usi, garaj, geamuri, horn);
* Realizarea transparentei geamurilor(aceasta problema nu este dificila de rezolvat, insa, in cadrul acestui proiect casa a fost exportata cu tot cu geamuri ca si obiect si nu au putut fi prelucrate);

# **7. Referinte**

*Indrumatorul de laborator*

[*Newsfeed - Sketchfab*](https://sketchfab.com/feed)

[*Free 3D Models download - Free3D*](https://free3d.com/3d-models/)

[*Learn OpenGL, extensive tutorial resource for learning Modern OpenGL*](https://learnopengl.com/)

[*Stack Overflow - Where Developers Learn, Share, & Build Careers*](https://stackoverflow.com/)