14.01.2025

Bogdan Sipos

grupa 30235

Proiect

Prelucrare grafica

1. **Cuprins**

 1. **Prezentarea temei**  
O introducere în scopul și obiectivele proiectului, prezentând utilizarea bibliotecilor OpenGL, GLFW și GLM pentru crearea unei scene 3D complexe. Sunt descrise funcționalitățile interactive și realismul implementat prin iluminare și texturare.

 2. **Scenariul**  
**2.1. Scena de obiecte**  
Detalii despre obiectele 3D utilizate în scenă, inclusiv modele texturate și mediul specific al scenei.  
**2.2. Funcționalități**  
Enumerarea funcționalităților interactive care permit explorarea dinamică a scenei.

 3. **Detalii de implementare**  
**3.1. Funcții și algoritmi**  
Explicații despre funcțiile principale din codul sursă și algoritmii importanți, precum rotația camerei, iluminarea Blinn-Phong și generarea umbrelor.  
**3.2. Modele grafice**  
Prezentarea pipeline-ului grafic și utilizarea shaderelor pentru transformarea și rasterizarea obiectelor.

 4. **Structuri de date și ierarhia de clase**  
Organizarea proiectului în directoare, rolul fiecărei componente și structurile de date folosite pentru eficiență.

 5. **Prezentarea interfeței grafice utilizator / Manual de utilizare**  
Instrucțiuni detaliate pentru utilizator, descriind navigarea în scenă, activarea efect

1. **Prezentarea temei**

Proiectul constă în crearea și afișarea unei scene 3D complexe utilizând Visual Studio și bibliotecile OpenGL, care interacționează direct cu placa grafică. Utilizatorul poate explora dinamic scena folosind mouse-ul și tastatura. Obiectele din scenă sunt texturate, iar iluminarea este realizată cu ajutorul mai multor surse de lumină poziționale. Scena este plasată pe o plaja, la mare, si este completata cu diferite vietuitoare si cu o asezare veche.

1. **Scenariul**

* **O imagine care conține palmier, copac, bazin de înot, cer

  Descriere generată automat**O imagine care conține în aer liber, cer, sol, plajă

  Descriere generată automat**Scena de obiecte**: In proiectul meu am folosit diferite obiecte precum: o lanterna care poate lumina, o minge care se rostogoleste neincetat si o plaja realizata in blender cu diferite animale (crabi, pelicani, soparle, rechini, delfini, pescarusi, leu de mare, testoase), doua feluri de palmieri, un copac cazut, o casa din piatra veche, o scoica, mai multe butoaie, o barca scufundata in nisip, o barca normala, un arc de piatra, balize si un lightCube care are rolul unui soare. Palmieri descriu o mica padurice in jurul casei.
* **Functionalitati**: Aplicația oferă o gamă variată de funcționalități care permit utilizatorului să exploreze scena 3D într-un mod dinamic și interactiv. Navigarea se realizează cu ajutorul tastelor W, A, S, D, pentru deplasarea înainte, înapoi, la stânga și la dreapta, respectiv R și F pentru deplasarea în sus și în jos. Utilizatorul poate controla modul de afișare al scenei prin trecerea între modurile **solid** (implicit) folosind tasta Z, **wireframe** cu X și **poligonal** cu C. Ceața, implementată pentru a oferi un efect realist de adâncime, poate fi activată sau dezactivată utilizând tastele V și B. Sistemul de iluminare include mai multe surse de lumină, cum ar fi lumina punctiformă, care poate fi activată și dezactivată cu tastele H și G, și lumina direcțională/spotlight controlată prin tastele T și Y. De asemenea, utilizatorul poate iniția un **tur automat** al scenei, apăsând tasta 0, camera deplasându-se de-a lungul unui traseu prestabilit. Rotirea liberă a camerei se realizează cu ajutorul mouse-ului, în timp ce tastele Q și E permit rotirea manuală a luminii(soarelui). În plus, aplicația include funcționalitatea de scalare globală a scenei, controlată prin tastele 3 (pentru mărire) și 4 (pentru micșorare). În cazul în care utilizatorul dorește să închidă aplicația, poate apăsa tasta ESC. Aceste funcționalități contribuie la o experiență captivantă și personalizabilă de explorare a scenei 3D.
* Partea superioară a formularului

Partea inferioară a formularului

1. **Detalii de implementare**

**Functii si algoritmi:**

* Funcția keyboardCallback gestionează intrările de la tastatură, permițând închiderea aplicației prin apăsarea tastei ESC și actualizând starea tastelor în array-ul pressedKeys, care este folosit ulterior în funcția processMovement. Funcția mouseCallback permite rotația camerei pe baza mișcărilor mouse-ului, calculând offset-ul de rotație în funcție de poziția curentă și anterioară a mouse-ului. Aceasta actualizează unghiurile yaw și pitch ale camerei, constrângând pitch-ul între -89° și 89° pentru a evita blocarea gimbal.
* **Funcții de procesare a mișcărilor și interacțiunilor:**  
  Funcția processMovement gestionează mișcările camerei și alte interacțiuni bazate pe taste. Aceasta permite deplasarea camerei înainte, înapoi, stânga, dreapta, sus și jos, folosind tastele W, A, S, D, R, F. De asemenea, utilizatorul poate modifica rotația și scala scenei cu tastele 1, 2, 3, 4, controla modurile de randare (solid, point, wireframe) cu tastele Z, X, C și activa sau dezactiva efecte precum ceața, luminile punctiforme și spot prin tastele V, B, H, G, T și Y. Tasta 0 inițiază un tur automat al camerei, utilizând traseul definit în cameraPath.
* **Funcții de inițializare:**  
  Funcția initOpenGLWindow creează fereastra aplicației OpenGL, configurând dimensiunea și viewport-ul acesteia. Funcția setWindowCallbacks asociază funcțiile callback pentru gestionarea evenimentelor de redimensionare a ferestrei, tastatură și mouse. initOpenGLState configurează stările globale OpenGL, activând testul de adâncime pentru afișarea corectă a obiectelor 3D și eliminarea fațetelor ascunse. Modelele 3D și skybox-ul sunt încărcate în initModels, iar shaderele sunt încărcate și compilate în initShaders. Funcția initUniforms configurează locațiile uniformelor necesare în shadere, incluzând matricile de transformare și proprietățile luminii. În final, initFBOs configurează framebuffer-ul pentru generarea hărții umbrelor, folosind o textură de adâncime.
* **Funcții de randare:**  
  Funcția drawObjects este responsabilă pentru randarea obiectelor din scenă, aplicând transformări precum rotații, scalări și translații asupra modelelor și trimițând datele necesare la shadere. Umbrele sunt generate de funcția showShadow, care creează o hartă de adâncime prin randarea scenei din perspectiva luminii. Funcția principală de randare, renderScene, integrează toate aceste componente, utilizând harta umbrelor pentru a aplica umbrele dinamice și renderizând scena principală.
* **Funcții auxiliare:**  
  Funcția computeLightSpaceTrMatrix calculează matricea de transformare pentru spațiul luminii, combinând proiecția ortografică și matricea de vizualizare din perspectiva luminii. Funcția updateDelta actualizează poziția obiectelor dinamice, cum ar fi mingea, în funcție de timpul scurs.
* **Iluminarea directă** este calculată folosind funcția computeDirLight, care determină componentele de lumină ambientală, difuză și speculară. Lumina difuză este determinată prin unghiul dintre vectorul normal al fragmentului și direcția luminii, în timp ce lumina speculară este calculată folosind reflexia și poziția observatorului.
* **Spotlight-ul** este gestionat de funcția computeSpotlight. Acesta utilizează poziția și direcția luminii, împreună cu unghiurile de tăiere interioară și exterioară, pentru a genera un efect de lumină focalizată. Atenuarea este calculată în funcție de distanța față de sursa de lumină, iar intensitatea luminii este ajustată în funcție de unghiurile de tăiere.
* **Luminile punctiforme** sunt implementate în funcția computePunctiformLight. Acestea iau în considerare distanța dintre fragment și fiecare sursă de lumină punctiformă pentru a calcula o atenuare realistă. Similar cu spotlight-ul, se determină componentele ambientală, difuză și speculară, care sunt apoi combinate cu culorile texturilor.
* **Umbrele** sunt calculate de funcția computeShadow, care folosește o hartă de adâncime generată anterior pentru a verifica dacă fragmentul este acoperit de alt obiect în raport cu sursa de lumină. Bias-ul este adăugat pentru a evita artefactele vizuale precum acneea umbrelor.
* **Ceața** este implementată în funcția computeFog, care utilizează un model de ceață exponențială pentru a determina factorul de amestec între culoarea fragmentului și culoarea ceții, în funcție de distanța fragmentului față de cameră.
* **Funcția principală:**  
  Funcția main este punctul de intrare al aplicației. Aceasta inițializează toate componentele necesare, precum ferestrele, modelele, shaderele și framebuffer-ul, și rulează bucla principală. În buclă, funcția gestionează intrările utilizatorului, randează scena și actualizează camera în modul tur automat, dacă este activat.
* **Algoritmi Importanți:**  
  Pentru rotația camerei, se utilizează formula Euler pentru yaw și pitch, actualizând direcția camerei în mod dinamic. Interpolarea liniară este folosită pentru a crea un traseu fluid al camerei, iar generarea umbrelor se bazează pe o hartă de adâncime, care identifică zonele umbrite prin sampling în shader. Acești algoritmi contribuie semnificativ la realismul și dinamismul aplicației.

**Modelul grafic:**

Proiectul are ca obiectiv realizarea unei prezentări fotorealiste a unor scene 3D, utilizând librăriile OpenGL, GLFW și GLM. Aplicația respectă pipeline-ul grafic clasic pentru rasterizarea scenelor, asigurând o procesare eficientă și realistă a obiectelor 3D.

Obiectele definite în coordonate globale sunt procesate inițial în vertex shader, unde sunt aplicate transformări intermediare, cum ar fi transformările de vizualizare și proiecție. Ulterior, acestea trec prin fragment shader, unde se calculează culorile pentru fiecare fragment. În final, pixelii corespunzători sunt rasterizați pe ecran cu culorile generate pentru fiecare fragment.

Proiectul utilizează modelul de iluminare Blinn-Phong pentru a simula sursele de lumină. Acest model presupune calculul componentelor de iluminare ambientală, difuză și speculară pentru fiecare fragment ce urmează a fi rasterizat. Modelul Blinn-Phong este optimizat pentru a lucra cu surse de lumină poziționale, fiecare având o culoare specifică. Această abordare contribuie la crearea unui mediu vizual realist și captivant.

**Structuri de date si ierarhia de clase:**

Proiectul este organizat în mai multe foldere, fiecare având un rol specific. Folderul principal conține codul sursă C++ al aplicației, iar folderele *models*, *shaders* și *skybox* sunt dedicate stocării resurselor necesare pentru rularea proiectului.

Folderul *glm* include biblioteca GLM, esențială pentru funcțiile utilizate în codul principal al aplicației.

Resursele din folderul *skybox* constau în imagini utilizate pentru maparea unui fundal panoramic în jurul scenei de obiecte 3D. Folderul *shaders* conține fișierele .vert și .frag, care definesc operațiile aplicate asupra vârfurilor și fragmentelor fiecărui obiect din scenă în procesul de rasterizare.

Folderul *models* stochează toate modelele 3D utilizate în scenă, fiecare fiind reprezentat de un fișier .obj. Aceste fișiere sunt asociate cu fișiere .mtl, care specifică materialele și texturile necesare pentru încărcarea corectă a obiectelor. Deși obiectele *cube* și *quad* nu sunt esențiale pentru afișarea scenei, acestea au roluri funcționale.

O imagine care conține text, meniu, captură de ecran

Descriere generată automat

O imagine care conține text, captură de ecran, Font, număr

Descriere generată automat

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

Aplicația oferă utilizatorului o experiență interactivă completă pentru explorarea unei scene 3D, punând la dispoziție o gamă variată de funcționalități accesibile prin tastatură și mouse. Navigarea în scenă se realizează cu ajutorul tastelor W, A, S, D, care permit deplasarea înainte, înapoi, la stânga și la dreapta, respectiv R și F pentru deplasarea în sus și în jos. De asemenea, utilizatorul poate roti liber camera utilizând mișcările mouse-ului, oferind astfel o vedere panoramică a întregii scene.

Pentru o explorare detaliată, aplicația oferă posibilitatea de a schimba modul de afișare a scenei. Astfel, apăsarea tastei Z afișează scena în modul solid, implicit, tasta X activează modul wireframe, iar tasta C permite vizualizarea în modul poligonal. Aceste opțiuni sunt utile pentru utilizatorii care doresc să examineze structura geometrică a obiectelor sau modul în care acestea sunt randate.

Un efect special de ceață este inclus pentru a îmbunătăți realismul scenei prin simularea adâncimii. Utilizatorul poate activa sau dezactiva acest efect folosind tastele V și B. Ceața face ca obiectele îndepărtate să apară mai estompate, oferind o senzație realistă de perspectivă.

Sistemul de iluminare al aplicației este extrem de flexibil, incluzând mai multe surse de lumină controlabile. Luminile punctiforme, care simulează surse de lumină localizate, pot fi activate și dezactivate utilizând tastele H și G. În plus, spotlight-ul, o lumină direcțională, poate fi controlat prin tastele T și Y. Aceste opțiuni permit utilizatorului să experimenteze diferitele tipuri de iluminare și efectele lor asupra scenei.

Pentru o experiență mai relaxată, aplicația include funcționalitatea de tur automat al scenei. Prin apăsarea tastei 0, camera începe să se deplaseze de-a lungul unui traseu prestabilit, oferind utilizatorului o privire generală asupra punctelor de interes. În plus, utilizatorul poate controla manual rotirea luminii (soarelui) folosind tastele Q și E, ceea ce adaugă un nivel suplimentar de personalizare a experienței.

Aplicația permite și scalarea globală a scenei pentru a se potrivi nevoilor utilizatorului. Aceasta se realizează prin apăsarea tastei 3 pentru mărire sau 4 pentru micșorare. În cazul în care utilizatorul dorește să închidă aplicația, acest lucru se poate face rapid prin apăsarea tastei ESC.

Aceste funcționalități variate contribuie la crearea unei experiențe captivante și personalizabile, permițând utilizatorului să exploreze scena 3D într-un mod dinamic și interactiv. Fie că este vorba despre examinarea detaliilor obiectelor, ajustarea modului de afișare sau explorarea diferitelor tipuri de iluminare, aplicația oferă toate uneltele necesare pentru o interacțiune completă și intuitivă.

1. **Concluzii si dezvoltari ulterioare**

Astfel, realizarea acestui proiect evidențiază clar superioritatea procesorului GPU față de procesorul CPU în ceea ce privește rasterizarea imaginilor fotorealiste. Utilizarea bibliotecilor OpenGL, în combinație cu mecanismele de accelerare hardware, demonstrează avantajele remarcabile ale procesorului grafic în comparație cu metodele tradiționale de rasterizare. GPU-ul permite procesarea paralelă a fragmentelor, ceea ce face posibilă analiza rapidă și eficientă a unui volum mare de date.

* În plus, utilizarea funcțiilor predefinite și a structurilor de date din OpenGL simplifică semnificativ calculele matematice și execuția pipeline-ului grafic. Funcții precum glCullFace, dot, cross sau structurile de date precum vec3, vec4 și mat4 reduc complexitatea implementării, accelerând procesul de dezvoltare și sporind eficiența generală a aplicației.

Ca dezvoltari ulterioare se pot implementa algoritmi de reflexie, animatie de ploaie, animarea diferitelor obiecte dar si un algoritm de eliminare a fragmentelor.

1. **Referinte:**

* Laboratorul 12/PG - Fog and fragment discarding.
* Laboratorul 6/PG - Texturi si animatii simple.
* Laboratorul 8/PG - Modelul de iluminare Blinn-Phong.
* Laboratorul 9/PG - Shadow Mapping.
* <https://learnopengl.com/Lighting/Multiple-lights>