Documentatie Proiect de Semestru la Prelucrare Grafica

Scena prezentata: Toys Army Scene



Autor: HÎRȚESCU CIPRIAN GABRIEL

Grupa: 30231

Summary:

1. Prezentarea temei de lucru………………………………….
2. Scenariul urmat……………………………………………..
3. Detalii legate de implementare……………………………..
4. Prezentarea interfetei grafice si a manualilui de utilizare…..
5. Bibliografie………………………………………………….

Capitolul 1: Prezentarea temei de lucru

Principala ideea a acestui proiect este de a crea o scena de vizualizare ce contine obiecte in format 3D, incercand sa simuleze intr-o maniera cat mai realista o imagine de vizualizare din lumea reala. Tema aleasa a plecat de la crearea unei aproximatii ale unei scene din jocul: “Army Men: RTS”, astfel am ajuns la scena cu obiecte detaliata in capitolele care urmeaza. Se mentioneaza faptul ca exista posibilitatea navigarii prin scene in formatul “ first person ”, iar textura obiectelor din scena de lucru este infuentata de unii stimuli externi ca si lumini, umbre etc.

Capitolul 2: Scenariul urmat

**2.1 Descrierea scenei si obiectelor:**

Scena reprezinta o pajiste pe care sunt plasate mai multe jucarii de plastic, dintre acestea se mentioneaza: turnurile de vizualizare, bazele unde se ascund soldatii , masinute avioane, un steag, grenade si pistoale.Texturile obiectelor sunt clasice,simple specifice unor jucarii, desigur se observa ca aceastea ies in evidenta si se completeaza una pe cealalta intregind scena din toate punctele de vedere. Toata scena este inconjurata de un cub ce simuleaza un cer.

**2.2 Functionalitati:**

Obiectele beneficiaza pe de o parte de o sursa de lumina directionala plasata la inceput in centrul scenei, dar pe parcurs fiind posibila mutarea ei din tastele “Z & X” pentru (+,-) pe axa z, “U & I” pentru axa x si respectiv “O & P” pentru axa y. Pe de alta parte steagul din proximitatea centrului scenei constituie de asemenea o sursa de lumina, de aceasta data fiind una punctiforma, care se poate porni si stinge de pe tastele “2 & 3”.

Prezenta luminii directionale a implicat crearea umbrelor astfel incat scena sa fie cat mai realista, asa ca limina directionala influenteaza direct umbrele si forma acestora pe planul de baza.

O data ce utilizatorul se indeparteaza de scena de obiecte, ceata iti face aparitia.

Soldatii de plastic desi au o textura verde, au parte si de o transparenta vizibila in cadrul proiectului.

Se poate face comutarea intre diferite moduri de vizualizare: solid, punctiform si wireframe prin tastele “V, B, N” iar pozitia avionului de deasupra scenei se modifica la apasarea tastei “H” , iar cand acesta se departeaza prea mult, se poate repozitiona prin apasarea tastei “1”.

Rotirea scenei se poate face de pe tastele “Q & E”,iar rotirea luminii directionale de pe testele “J & K”.

Capitolul 3: Detalii de implementare

**3.1 Functii si algoritmi:**

1. Adaugarea obiectelor in scena:

Functia target: initObjects()

scene.LoadModel("objects/space/scena.obj");

avion.LoadModel("objects/space/avion.obj");

lightCube.LoadModel("objects/cube/cube.obj");

screenQuad.LoadModel("objects/quad/quad.obj");

* datorita faptului ca exista bind de la varfuri la texturi realizat prin intermediul fisierului .mtl, aplicarea texturilor de mana este eliminata din start, clasa Model se ocupa in spate de acest lucru.

1. Miscarea prin scena a camerei:

Clasa target: Camera.cpp

Descrierea clasei:

* Pozitia camerei in world space : descrisa printr-un vector cu 3 componente : glm::vec3 cameraPosition
* Punctul spre care priveste, de asemenea descris ca un vector cu 3 componente: glm::vec3 cameraTarget

Obs. Scazand din target, pozitia => vectorul directie

1. Animatia avionului:

Are loc printr-o simpla operatie de translatie (folosind o matrice de trnslatie) a obiectului de la pozitia curenta la pozitia urmatoare, calculata ca si o incrementare a coordonatei y astfel incat sa se creeze o perceptie de miscare curenta a avionului pe ecran.

1. Lumina punctiforma: iluminarea steagului

Pentru a genera o lumina punctiforma, care sa se aprinda pe un buton si sa se stinga de pe alt buton s-a plecat de la ideea unui enable. Enable-ul este controlat de tastele 2 si 3 si are ca si scop atribuirea de valoare a intensitatii luminii punctiforme care va influenta obiectele situate in proximitate.

Fragmentul de cod care descrie acest eveniment este:

if (glfwGetKey(glWindow, GLFW\_KEY\_2))

{

en = 1;

}

if (glfwGetKey(glWindow, GLFW\_KEY\_3))

{

en = 0;

}

if (en == 0)

{

intens = 0.0f;

}

else if (en == 1)

{

intens = 10.0f;

}

Mai departe variabila intensitate este trimisa la fragment shader sub forma unui uniform pentru calculul intensitatii luminii punctiforme.

glUniform1f(glGetUniformLocation(myCustomShader.shaderProgram, "intens"), intens);

glUniform3fv(glGetUniformLocation(myCustomShader.shaderProgram, "flagPos"), 1, glm::value\_ptr(glm::vec3(view \* model \* glm::vec4(flagPos, 1.0f))));

flag-ul in acest caz este de fapt un vector de 3 coordonate care specifica pozitia flag-ului adica a obiectului de la care se va pleca sursa de lumina punctiforma; acesta este hardcodat la valoarea:

glm::vec3(3.41f, 2.76f, 14.58f);

1. Generarea Umbrelor:

Pentru generarea de umbre s-a plecat de la un depth buffer care stocheaza depth-ul relativ la sursa de lumina directionala in cazul acesta, la o prima trecere prin algoritm se observa ca are la baza ideea de z-buffer. Valoarea din buffer se actualizeaza doar daca punctul referit este mai aproate de sursa de lumina decat ce a fost inainte in buffer. Si atunci orice punct care se afla in z-buffer este iliminat, restul sunt umbrite. De pe tasta M se poate vedea depth-map-ul.

**3.2 Motivarea solutiei alese:**

Functionalitatile detaliate in capitolul anterior au fost implementate astfel incat sa completeze scena din punct de vedere vizual, astfel incat utilizatoul sa aiba parte de o experienta placuta, jucandu-se cu aplicatia si privind obiectele cu tot cu ceea ce reprezinta limini, umbre si movement. Pe de alta parte venind vorba de unele imbunatatiri, la nivelul functionalitatilor, s-ar putea aduce mai ales pe partea de animare a obiectelor sau chiar alte modele de iluminare.

**3.3 Modelul grafic:**

Obiectele prezente in scena sunt de tipul 3D preluate de pe diferite platforme astfel incat sa constituie o scena completa. Aceste au fost plasate pe un plan, modelat in Blender, pentru a adauga diferite asemanari de dealuri, sau gropi iar deasupra lui au fost puse obiectele, astfel fiind scutiti de a mai efectua operatii de translatie, rotatie si scalare la nivelul aplicatiiei pentru a le aduce in pozitia dorita.

In final, obiectele a caror pozitie nu se va modifica niciodata au fost exportate ca un singur .obj iar obiectul avion care este singurul ce are parte de animatie ramane fix.

Scena si avionul mai apoi au fost exportate in cadrul aplicatiei din OpenGL din ide-ul Visual Studio.

**3.4 Structuri de date:**

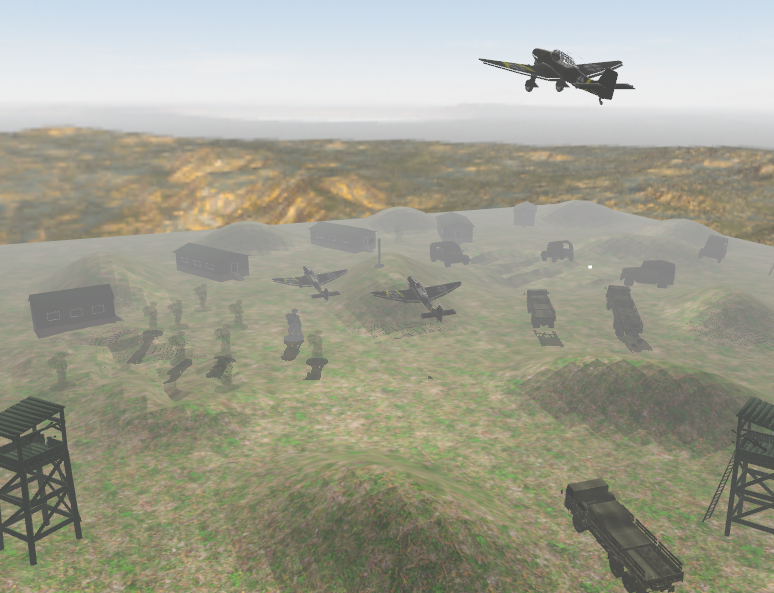
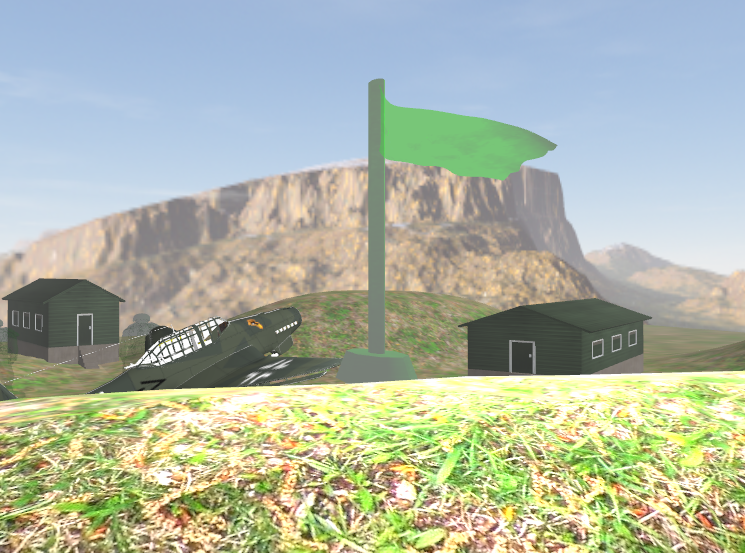
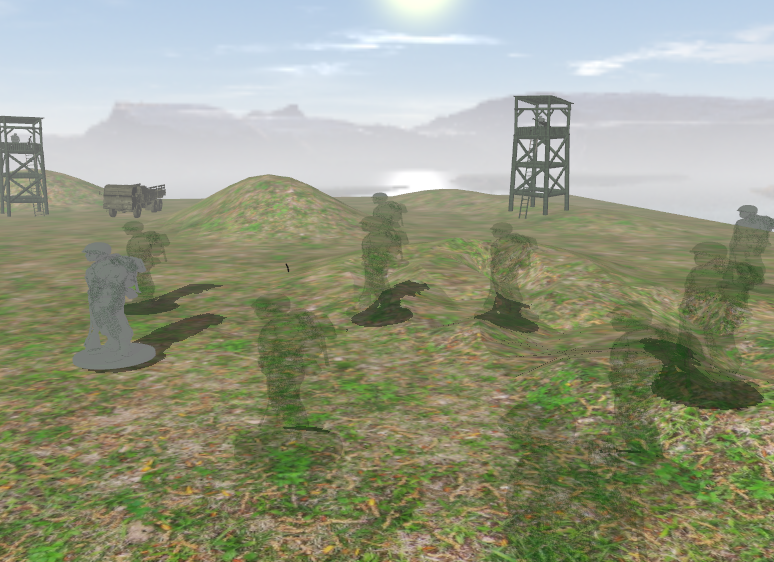
Din punctul de vedere al structurilor de date utilizare la nivelul aplicatiei se remarca vectorii, matricile pentru operatii de clacul si chiar a umbrei si luminilor.

**3.5 Ierarhii de clase:**

Ierarhia de clase se bazeaza pe mai multe librarii printre care: Camera.cpp este clasa care intruneste toate operatiile la nivelul camerei, adica ceea ce va vedea utilizatorul si cum va vedea din scena de obiecte, pe langa aceasta o alta clasa care se remarca este Shader.cpp care are drept scop legarea dintre programul principal din OpenGL cu shaderele aplicatiei, adica acele programe de mici dimensiuni scrise in GLSL (apropiat de C) cu ajutorul caruia se GPU-ul poate fi folosit la capacitate maxima. Ultimele clase folosite sunt Mesh.cpp si Model.cpp a caror utilitate este de de lega fisierele de tip .obj de cele de tip .mtl astfel ca aplicarea texturii prin instructiuni de binding si altele de acest tip sa fie evitate.

**4. Prezentarea interfetei grafice si a manualului de utilizare:**





**O imagine care conține verde, iarbă, exterior, pod

Descriere generată automat**

**O imagine care conține iarbă, cer, exterior, munte

Descriere generată automat**

Manual de utilizare:

1. Controlul camerei:
   1. Taste: W, A, S, D
   2. Mouse movement, Front Scroll : zoom + , Back Scroll: zoom –
2. Controlul animatiei :

Taste: H miscare pe directia inainte , 1 pentru resetarea pozitiei

1. Controlul luminii directionale: light cube

Taste:

* 1. Z, X miscarea pe axa Z
  2. U,I miscarea pe axa X
  3. O,P miscarea pe axa Y
  4. J,L pentru rotire

1. Controlul luminii punctiforme: green flag

Taste: 2, 3 pentru ON/OFF

1. **Bibliografie:**

* **Skybox:** [**https://www.humus.name/index.php?page=Textures**](https://www.humus.name/index.php?page=Textures)
* **Obiecte 3D:**
  + [**https://free3d.com/**](https://free3d.com/)
  + [**https://www.turbosquid.com/,**](https://www.turbosquid.com/,)