UNIVERSITATEA ”ALEXANDRU-IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

LUCRARE DE LICENȚĂ

Motor Fizic 3D

propusă de

Mihaila Andrei

Sesiunea: Iulie, 2019

Coordonator științific

Asist. dr. Eugen Croitoru

UNIVERSITATEA ”ALEXANDRU-IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

Motor Fizic 3D

Mihaila Andrei

Sesiunea: Iulie, 2019

Coordonator științific

Asist. dr. Eugen Croitoru

Avizat,
Îndrumător lucrare de licență,
Asist. dr. Eugen Croitoru.
Data: ..... Semnătura: .....

DECLARAŢIE PRIVIND ORIGINALITATE ŞI RESPECTAREA DREPTURILOR DE AUTOR

Prin prezenta declar că Lucrarea de licenţă cu titlul **Motor Fizic 3D** este scrisă de mine şi nu a mai fost prezentată niciodată la o altă facultate sau instituţie de învăţământ superior din ţară sau străinătate. De asemenea, declar că toate sursele utilizate, inclusiv cele preluate depe Internet, sunt indicate în lucrare, cu respectarea regulilor de evitare a plagiatului:

- toate fragmentele de text reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi deţin referinţa precisă a sursei;
- reformularea în cuvinte proprii a textelor scrise de către alţi autori deţine referinţa precisă;
- codul sursă, imagini etc. preluate din proiecte open-source sau alte surse sunt utilizate cu respectarea drepturilor de autor şi deţin referinţe precise;
- rezumarea ideilor altor autori precizează referinţa precisă la textul original.

Data: ..... Semnătura: .....

Declarație de consimțământ

Prin prezenta declar că sunt de acord ca lucrarea de licență cu titlul **Motor Fizic 3D**, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test, etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de informatică.

De asemenea, sunt de acord ca Facultatea de informatică de la Universitatea ”Alexandru-Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Absolvent **Mihaila Andrei**

Data: ..... Semnătura: .....

Cuprins

Motivație

motivatie

Introducere

Prin creearea unui motor fizic 3D, avem ca scop simularea aproximata a unor actiuni fizice, ce au loc in lumea reala.

Capitolul 1

Chapter 1

intro

1.1 Section 1

section

Capitolul 2

Tehnologii Folosite

intro

2.1 C++

Mediul de lucru este C++17 [1]sub Visual Studio 2019. C++ a fost ales in principal datorita performantei si a experientei anterioare. Prin intermediul OOP au fost create structuri pentru o implementare cat mai naturala a motorului fizic.

2.2 OpenGL

Reprezentarea grafica se realizeaza cu ajutorul librariei OpenGL 3.3[2]. Motivatia librariei OpenGL a fost atat introducerea ei in Anul 3 cat si permitarea implementarii cat mai primitiva a motorului fizic.

2.3 OpenGL Mathematics (GLM)

Fiind un motor fizic, este necesar ca toate calculele matematice sa fie realizate cat mai corect si rapid, astfel libraria GLM [3] a fost aleasa pentru viteza si compatibilitatea ei cu OpenGL. Principalele structuri folosite din GLM au fost:

- glm::vec2/glm::vec3/glm::vec4 reprezentand vectori.

[1] http://www.cplusplus.com/
[2] https://www.opengl.org/
[3] https://glm.g-truc.net/0.9.9/index.html/

- glm::quat, (glm quaternions implementati sub forma w,x,y,z). Folositi in principal pentru reprezentarea orientarii obiectelor. Motivatia quaternionilor in favoarea matricelor de rotatie a fost performanta si evitarea problemelor unghiurilor euler, cum ar fi Gimbal Lock [4]

- glm::mat3/glm::mat4, Matrici patratice, in principal pentru matricile MVP.

Pe langa structuri, metodele esentiale:

- glm::rotate, folosit pentru a roti quaternioni dupa o axa de rotatie cu un anumit unghi

- glm::translate, pentru translatea matricii de translatie

- glm::dot/glm::cos

- glm::mat4_cast pentru conversia din quaternioni in matrici de rotatie.

Alternative au fost: Eigen [5] si CML [6].Toate librariile fiind testate in `https://github.com/mfoo/Math-Library-Test`.

2.4 GLFW

Pentru initializarea ferestrei OpenGL a fost folosit [7] GLFW. Aceasta librarie ofera in mod simplistic un context OpenGL si mai multe optiuni cum ar fi VSync sau DoubleBuffering. De asemenea, keyboard si mouse events sunt capturate prin GLFW.

Alternative au fost : GLUT [8]. GLFW a fost ales in favoarea GLUT deoarece pe masina unde a fost dezvoltata aplicatia, GLUT nu reusea sa intanstieze OpenGL 3.3+.

2.5 Concluzii

C++ impreuna cu OpenGL si GLM ofera un mediu de lucru cat mai aproape de procesor si in acelasi timp un set de structuri si unelte pentru dezvoltarea naturala a Motorului Fizic.

[4] `https://en.wikipedia.org/wiki/Gimbal_lock/`
[5] `http://eigen.tuxfamily.org/index.php?title=Main_Page`
[6] `https://github.com/demianmnave/CML/wiki/The-Configurable-Math-Library`
[7] `https://www.glfw.org/`
[8] `http://freeglut.sourceforge.net/`

Capitolul 3

Configurare

Pentru a putea fi folosit in cat mai multe moduri, motorul fizic poate fi setat sa ruleze in moduri diferite, iar obiectele sale pot avea mai multe proprietati, descrise mai jos.

3.1 Variables

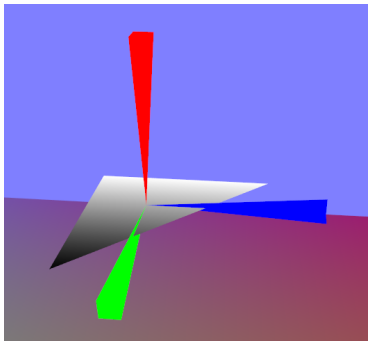
Variabilele ce pot fi schimbate sunt de 2 tipuri:

- variabile de sistem, in fisierul $envrules.h$

```
constexpr auto DT = 0.0166666f;
constexpr auto gravForce = 9.8f;
constexpr auto pi = 3.14159265359f;
```

unde DT reprezinta timpul per frame, $gravForce$ forta gravitationala si PI

- variabile ale obiectelor
  - $gravInvul$, atunci gravitatia nu mai este aplicata obiectelor
  - $showDirection$



In cazul in care $showDirection$ este $true$ atunci obiectul va avea 3 axe

* $Red - Y$
* $Green - Z$
* $Blue - X$

ce reprezenta orientarea obiectului, in caz contrar acestea nu vor aparea.

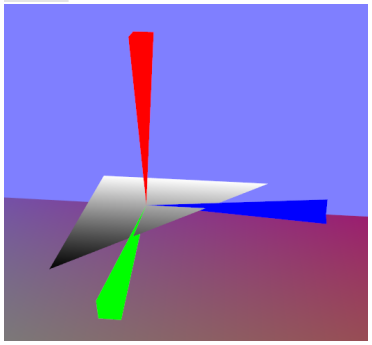
3.2 Objects

Un obiect fizic este un obiect derivat in $oglVertexObject$ si poate exista in simulator doar daca are un fisier txt in care se regasesc proprietatile si regulile urmatoare:

- rotatii dupa o axa oarecare intre $DT1$ si $DT2$ unde $DT2 > DT1$. (DT[1])
- pozitia x,y,z in world space
- factor de elasticitate
- un array de puncte ce reprezinta fiecare varf, aflate oriunde in object space [2].
- un array de culori ce sunt asignate fiecarui punct

De exemplu, pentru urmatorul fisier txt apare un singur triunghi:

```
1  0
2  3.0 3.0 0.0 0.0
3  v 0.0 0.0 0.0
4  v 0.0 0.0 1.0
5  v 1.0 0.0 0.0
6  c 1 1 1
7  c 0 0 0
8  c 1 1 1
```



Conform fisierului txt, acesta are 0 rotatii, este plasat in punctul $C(3, 3, 0)$ in world space, are o elasticitate 0, iar varfurile sale reprezinta un triunghi unde varful $V(0, 0, 1)$ va avea culoarea $RGB(0, 0, 0)$.

[1] Delta time, timp scurs de inceperea executiei.

[2] puncte in referinta fata de C(0,0,0) unde va fi centrul obiectului.

3.3 Head

section

3.4 Headless

section

Capitolul 4

Arhitectura

intro

4.1 Simulator Core

section

4.2 OpenGL Container

section

4.3 Scenes

section

4.4 oglVertexObject

section

Concluzii

conclusions

Bibliografie

- i1