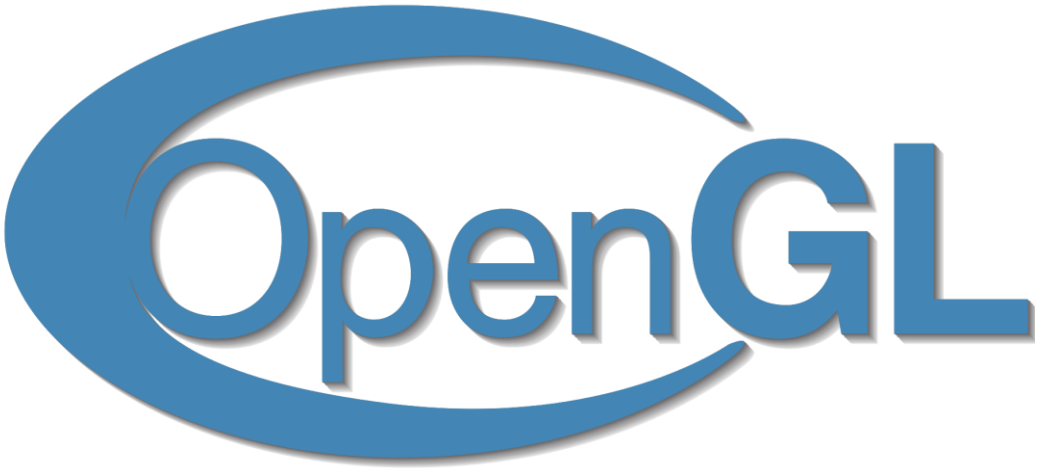
SIMULATION DE FLUIDE



31/05/2018

Rapport de projet OpenGL

Professeur : *M.Lurkin*

Etudiant : *Emine TAS, 16362*

Simulation de fluide

RAPPORT DE PROJET OPENGL

Table des matières

INTRODUCTION

Dans le cadre du cours de Synthèse d'images, nous devions faire un projet de simulation de fluide avec OpenGL. Un code de base a été fournis par le professeur. Le but était de reprendre ce code est de faire le projet demandé.

Le code fourni par le professeur était composé d'un certain nombre de fichiers mais dans ce rapport nous allons voir les fichiers que j'ai dû modifier pour ce projet.

PROJET

Le but du projet était de créer une zone contenant des particules et d'avoir un effet de vague.

Gravity.comp

Un fichier shaders est présent pour que nos particules aient une gravité. Dans ce fichier j'ai mis les limites de parois sous forme de condition pour les coordonnées des particules qui ont 3 coordonnées : x, y et z. L'idée est d'avoir un rectangle en 3D pour contenir nos particules de vague.

```
    if(p.y < 0) {
        p.y=0;
        v.y=-v.y;
    }

    if(p.y > 5) {
        p.y=5;
        v.y=v.y;
    }

    if(p.x < 0) {
        p.x=0;
        v.x=-v.x;
    }

     if(p.x > 10) {
        p.x=10;
        v.x=-3;
    }
```

Simulator.cpp

```
        int i=0;
        int j=0;
        int k=0;
        int l=0;

        Particule pp[nbPoints];
        for(i=0; i<5; i++){
            for(j=0; j<5; j++){
                for(k=0; k<5; k++){
                    pp[l] = Particule(vec3(i, j,
k), vec3(1,0,0));
                    l++;
                }
            }
        }
```

Ce fichier est le main. Ce bout de code permet de dessiner les particules sous forme d'un cube.

```
    Particule()
    {
        position = vec4(0.0f, 0.0f, 0.0f, 0.0f);
        velocity = vec4(0.0f, 0.0f, 0.0f, 0.0f);
    }
```

Suite à une simple erreur de constructeur j'avais dû mettre un constructeur de particule sans paramètres.

```
float ll[] = {
        0,-0.1,-0.1,
        0,5,0,

         0,-0.1,-0.1,
        10,-0.1,-0.1,

        0,5,0,
        10,5,0,

        10,-0.1,-0.1,
        10,5,0,

        10,0,-0.1,
        10,0,5,

        0,0,5,//
        10,0,5,

        10,0,5,
        10,6,5,

        10,0,5,
        10,0,0,

        0,0,5,//
        0,-0.1,-0.1,

        10,5,0,
        10,6,5,

        0,0,5,//
        0,6,5,//

        10,6,5,
        0,6,5,//

        0,5,0,
        0,6,5,//

    };
```

J'ai modifié le code pour ajouter l'image du rectangle 2D.

Le bout de code à gauche permet d'avoir les points nécessaires pour pouvoir tracer les lignes du rectangle 3D dans un tableau.

Ensuite il fallait ajouter un nouveau shader pour cela et refaire un setVertexData.

Pour cela j'ai dû ajouter une primitive permettant le traçage de ligne dans le fichier Renderer.h :

```
    PRIMITIVE_LINES = GL_LINES
```

```
    lignes = new Buffer(ll, sizeof(ll));

    program = new Program();
    program->addShader(Shader::fromFile("shaders/perspective.vert"));
    program->addShader(Shader::fromFile("shaders/black.frag"));
    program->link();

    renderer = program->createRenderer();
    traceRenderer = program->createRenderer();

    renderer->setVertexData("vertex", particules, TYPE_FLOAT, 0, 3,
sizeof(Particule));
    traceRenderer->setVertexData("vertex", lignes, TYPE_FLOAT, 0, 3,
sizeof(float)*3);
```

CONCLUSION

Le projet n'est pas terminé. L'effet de vague est manquant. Ce projet m'a permis de me familiariser avec le monde d'OpenGL.

Voici ce que cela donne :