TP2 : Chargement d'un fichier Obj

Djihane BABAHENINI

07/07/2019
1.0

Légende

 Entrée du glossaire

 Abréviation

 Référence Bibliographique

 Référence générale

Table des matières

Objectifs

À la fin de ce deuxième TP, l'étudiant sera capable de:

- Connaître la création de la matrice de transformation.
- Savoir programmer avec la bibliothèque GLM.
- Comprendre les étapes de chargement d'un fichier obj.

Pré-requis :

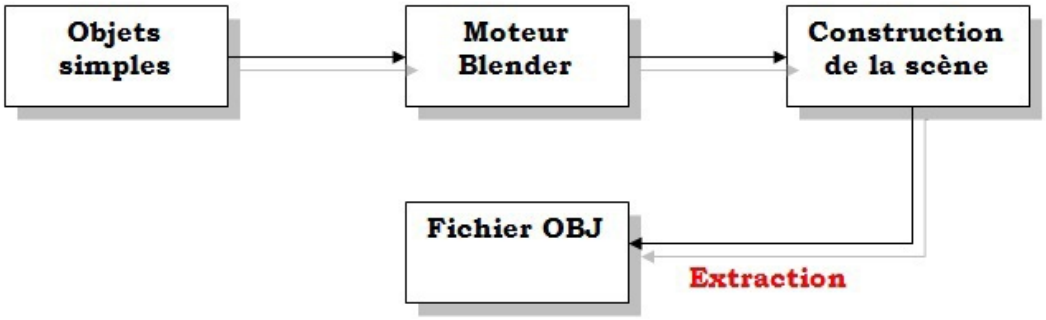
Pour pouvoir suivre ce TP avec succès il faut au préalable savoir:

- Des notions de base sur les matrices.
- Des notions de base sur les transformations géométriques.

Introduction

La modélisation de la scène se fait indépendamment des environnements de programmation.

Pour modéliser une scène 3D, on utilise le modeleur Blenderp.22 p.20 , qui à partir des objets simples (sphere, cube, plan, surphase,...), on peut créer un objet complexe et qui sera ensuite exporter en un fichier obj qu'on peut utiliser dans notre application de rendu.



Modélisation d'une scène 3D.

Matrice de transformation

La matrice de transformation OpenGL appelé aussi matrice de modèle, vue et projection (MVP)p.21 est le résultat de produit de trois matrices : matrice de moèle, matrice de vue et matrice de projection.

Elle a utilisé pour transformer tous les objets de ses propre espace vers l'espace écran.

Pour manipuler ces trois matrices, on utilise la bibliothèque GLMp.20 p.21 . On la déclare dans notre application OpenGL :

```
1 #include <glm/glm.hpp>
2 #include <glm/gtc/matrix_transform.hpp>
3 #include <glm/gtc/type_ptr.hpp>
```

Complément : Matrice de modèle

Elle permet de transformer les coordonnées locales de l'objet à l'espace global :

```
1 //La matrice de modèle est la matrice d'identité
2 glm::mat4 Model = mat4(1.0f);
```

Complément : Matrice de vue

Elle permet de transformer les coordonnées de l'espace global vers l'espace de la caméra :

```
1 //Caméra
2   glm::mat4 View       = lookAt(
3     glm::vec3(5,4,4), // Caméra à la position (4,3,3)
4     glm::vec3(0,0,0), // et se voit à l'origine
5     glm::vec3(0,1,0)  // selon l'axe y
6     );
```

Complément : Matrice de projection

Elle permet de transformer les coordonnées de l'espace caméra vers l'écran :

```
1 //Projection est de type perspective
2 mat4 Projection = perspective(45.0f, width / height, 0.1f, 100.0f);
```

Texte légal

La matrice MVP a défini comme :

```
1 glm::mat4 MVP = Projection * View * Model;
```

Exercice

[solution n°1 p.18]

Répondez par vrai ou faux :

0. Exercice

La matrice de transformation est une matrice qui permet de placer un objet dans l'espace global

- Vrai
- Faux

0. Exercice

La matrice de transformation est le produit de trois matrice : modèle, vue, projection :

- ○ Vrai
- ○ Faux

0. Exercice

La bibliothèque utilisé pour créer MVP est la bibliothèque SFML

- Vrai
- Faux

0. Exercice

La projection dans MVP est de type parallèle

- Vrai
- Faux

0. Exercice

La matrice modèle est une matrice d'identité

- Vrai
- Faux

Modélisation de la scène

VIII

1. Structure d'un fichier Obj

Un Obj est un format d'un fichier texte contenant la description d'un objet ou d'une scène 3D, il existe plusieurs logiciels qui exploitent les fichiers de format obj tel que Blender, Maya, 3D

Studio Max, la forme de base dans un fichier obj est le polygone, donc un objet est constitué par un ensemble de polygones.

La structure d'un fichier obj est comme suit :

- Les commentaires dans un fichier obj sont commencés par le caractère #.
- Un sommet est défini par : v 0.0 1.0 1.0, où v désigne un vertex, c'est à dire un sommet et 0.0 1.0 1.0 sont les coordonnées de ce sommet en 3D.
- Une coordonnée de texture est définie par : vt 1.0 0.0. Les coordonnées de texture sont des veteurs 2D qui donne la position d'un texel dans une image de texture.
- Une normale est un vecteur de trois composantes décrit par : vn 0.0 1.0 0.0
- Chaque face est définie par un ensemble d'indices faisant référence aux coordonnées des points, de texture et des normales.Par exemple : f v1/vt1/vn1 v2/vt2/vn2 v3/vt3/vn3 c'est un triangle constitué des sommets d'indices v1, v2 et v3 dans la liste des sommets v. Chacun de ces sommets possède une coordonnée de texture identifiée par son indice dans la liste des coordonnées de texture vt et une normale identifiée dans la liste des normales vn.
- Lorsque la scène contient plusieurs objets, il faut identifier chacun de la manière suivante : g [nom_objet]
- Des matériaux peuvent être référencés en faisant une importation des fichiers .mtl de la manière suivante : usemtl [nom_materiau]

Exemple

Exemple d'un fichier Obj représentant un cube généré par Blender:

```
# Blender v2.69 (sub 0) OBJ File: ''
# www.blender.org
mtllib cube.mtl
o Cube
v 1.000000 -1.000000 -1.000000
v 1.000000 -1.000000 1.000000
v -1.000000 -1.000000 1.000000
v -1.000000 -1.000000 -1.000000
v 1.000000 1.000000 -0.999999
v 0.999999 1.000000 1.000001
v -1.000000 1.000000 1.000000
v -1.000000 1.000000 -1.000000
usemtl Material
s off
f 1 2 3 4
f 5 8 7 6
f 1 5 6 2
f 2 6 7 3
f 3 7 8 4
f 5 1 4 8
```

2. Utilisation de fichier Obj

On crée un petit analyseur qui permet de charger un modèle obj dans l'application OpenGL :

Méthode

- On définit des variables temporaires dans lesquelles on stocke les vertex, normale, texture (ou bien couleur) du fichier .obj :

```
std::vector< unsigned int > vertexIndices, uvIndices, normalIndices;
std::vector< glm::vec3 > temp_vertices;
std::vector< glm::vec2 > temp_uvs;
std::vector< glm::vec3 > temp_normals;
```

- On ouvre le fichier obj, on lecture et on le lit ligne par ligne jusqu'à la fin :

```
FILE * file = fopen(path, "r");
if( file == NULL ){
    printf("Impossible to open the file !\n");
    return false;
}
while( 1 ){

    char lineHeader[128];
    // lit le premier mot de la ligne
    int res = fscanf(file, "%s", lineHeader);
    if (res == EOF)
        break;
}
```

- Ensuite, on charge le contenu de fichier obj dans les vecteurs, temp_vertices, temp_uvs,

temp_normals, par exemple pour les sommets du modèle :

```
if ( strcmp( lineHeader, "v" ) == 0 ){
    glm::vec3 vertex;
    fscanf(file, "%f %f %f\n", &vertex.x, &vertex.y, &vertex.z );
    temp_vertices.push_back(vertex);
}
```

- Maintenant, on doit retourner l'indice des sommets, normales et uvs, pour qu'OpenGL puisse utiliser ces vecteurs, voici un exemple pour les sommets :

```
// Pour chaque sommet de chaque triangle
    for( unsigned int i=0; i<vertexIndices.size(); i++ ){
       unsigned int vertexIndex = vertexIndices[i];
       glm::vec3 vertex = temp_vertices[ vertexIndex-1 ];
       out_vertices.push_back(vertex);
    }
```

3. Exercice

[solution n°2 p.19]

3.1. Exercice

Dans un fichier obj les nouveau objets sont déterminés par la lettre :

3.2. Exercice

Quel est l'utilité d'un fichier obj ?

3.3. Exercice

Avec quel logiciel on génère un fichier obj?

TP2

IX

1. Énoncé de TP

Dans l'environnement Visual Studio 2008, créer un projet de type "application console permettant de :

1. Créer la matrice MVP avec les paramètres :
   - Matrice modèle : matrice d'identité.
   - Matrice de vue : qui a ces caractéristiques :
     - Espace global dans : 4.0, 3.0, 3.0.
     - L'origine : 0.0, 0.0, 0.0
     - direction : selon l'axe y.
2. Écrire un programme OpenGL permettant de charger un modèle 3D de votre choix.

Conclusion

IX

La matrice MVP permet de placer les objets de la scène 3D dans la fenêtre de l'écran pour qu'on puisse calculer le rendu finale.

Dans ce deuxième TP, nous avons vu :

1. L'utilité de la bibliothèque GLM pour faciliter les fonction de manipulation des vecteurs et des matrices dans une application OpenGL.
2. La structure d'un fichier obj.
3. Comment utiliser et exploiter un fichier obj dans une application OpenGL.

Solutions des exercices

> **Solution** n°1

Exercice p. 7

Exercice

La matrice de transformation est une matrice qui permet de placer un objet dans l'espace global

- ○ Vrai
- ◉ Faux

Exercice

La matrice de transformation est le produit de trois matrice : modèle, vue, projection :

- ◉ Vrai
- ○ Faux

Exercice

La bibliothèque utilisé pour créer MVP est la bibliothèque SFML

- ○ Vrai
- ◉ Faux

Exercice

La projection dans MVP est de type parallèle

- ○ Vrai
- ◉ Faux

Exercice

La matrice modèle est une matrice d'identité

- ◉ Vrai

- Faux

> **Solution** n°2

Exercice p. 15

Exercice

Dans un fichier obj les nouveau objets sont déterminés par la lettre :

o

Exercice

Quel est l'utilité d'un fichier obj ?

décrire la géométrie d'un modèle 3D

Exercice

Avec quel logiciel on génère un fichier obj?

Blender

Glossaire

Blender

Logiciel gratuit de modélisation et de rendu 3D.

GLM

Bibliothèque d'OpenGL qui offre des classes et des fonctions pour la manipulations des données mathématique : les vecteurs, les matrices, ...

Abréviations

GLM : openGL Mathematics

MVP : Model View Projection

Bibliographie

IX

Antoine Veyrat, *Débutez dans la 3D avec Blender,* Vol 1, Eyrolles