Nightmare Before Christmas

Assante Di Capillo Christophe

Costantini Theo

Classes

Engine : gère l'ensemble du programme. C'est le controleur de façade du programme. Il gère les événements souris et clavier et renvoie le rendu en fonction.

Scene : contient toutes les informations d'une scène (meshs, shader, lumières, caméra.)

Mesh : contient un objet 3D (et ses sous-objets) et toutes les informations (position, scale, rotation, matériaux, etc.)

Shader : objet shader utilisé pour le rendu.

Vertex : représente un point (position, normale, coordonnées des textures.)

Light / AmbientLight / DirectionalLight / PointLight : lumières des scènes. Grçace au polymorphise on peut définir autant de type de lumière que l'on veut. Il faut cependant les implémenter dans le shader ensuite.

TrackballCamera : caméra pour les scènes.

Architecture

l'Engine gère l'ensemble de l'application. Au lancement, il va initialiser tous les composants (SDL, OpenGL, le Shader, les Scene.)

Pour les scènes, l'Engine va charger le fichier *assets/scenes/scenesList.json* qui liste l'ensemble des scènes, ainsi que le fichier json associé à chaque scène. Après cela, l'Engine créé toutes les scènes.

Chaque scène est composée d'un Shader, d'un ou plusieurs Mesh, d'une TrackBallCamera et d'une ou plusieurs Light, dont les informations sont stockées dans les .json de chaque scène. Chaque Mesh est composé de plusieurs MeshEntry, stockées dans un Vector. Cela permet de gérer les objets 3D complexes.

La classe Mesh va s'occuper de charger le fichier de l'objet 3D, créer les MeshEntry et charger les textures.

Lors de l'affichage, toutes les MeshEntry de toutes les Mesh qui composent une scène sont affichés les unes à la suite des autres. Les calculs des matrices des positions sont faites à ce moment la. Chaque Mesh ayant des propriétés différentes, pour chaque Mesh rendu les valeurs du Shader utilisées pour le rendu sont mises à jour.

L'Engine gère aussi les événements clavier (passer d'une scène à une autre, quitter l'application) et souris (bouger la caméra) à chaque tour dans la boucle d'affichage.

Problèmes

L'intégration d'Assimp à la structure d'application que nous avions envisagé a été problématique. Certains éléments d'Assimp ne s'accordant pas avec ce que nous avions préparé, il nous a fallu modifier quelque peu la structure afin de pouvoir y intégrer Assimp. Il nous a aussi été tout simplement difficile d'apprendre à utiliser Assimp, qui est bien plus complexe que ce que nous avions vu jusqu'à présent en TD, et qui nous a poussé à faire un programme qui ne ressemble très peu à ce que nous avions appris à faire durant les TD.

Pour le futur

Par manque de temps, nous n'avons pas eu l'occasion de nous pencher sur certains points que nous aurions aimé intégrer à notre application. Voici les idées que nous avions :

- Une navigation par click sur des éléments de la scène plutôt que par les flèches du clavier.

- L'affichage d'ombres portées sur les objets des scènes.

- L'intégration de normal map.

- L'intégration de Specular Map

- Une structure de classes plus claire, légère et fonctionnelle, afin d'optimiser au maximum le fonctionnement de l'application et son potentiel d'expansion pour de futurs projets.