Projet OpenGL – ESIEE IT Computer Graphics

malek.bengougam@esiee.fr

Projet à rendre

En groupe de 3 (voire 2)

OpenGL 2.x ou OpenGL 3.x au choix

Objectif : être capable d'afficher une scène 3D composée de plusieurs objets et pouvoir naviguer dans la scène.

Partie 1 : Affichage

- Plusieurs objets différents, avec plusieurs shaders différents
    - Par exemple, un objet avec une couleur simple, un autre avec une texture, un autre avec une env. map etc…
    - Note : attention à bien gérer le sRGB !
- Etre capable de charger des objets au format OBJ en utilisant la bibliothèque **TinyOBJLoader**, en gérant les matériaux (couleurs ambiantes, diffuses, speculaires) https://github.com/tinyobjloader/tinyobjloader
- Avoir de l'illumination avec l'eq. de Phong ou Blinn-Phong
    - On peut aussi avoir le choix avec des objets qui ont juste du Lambert, d'autres Phong etc…

Partie 2 : Navigation

- Les objets doivent être placés à des positions différentes dans la scene (avec des translations, rotations, scale etc… différents)
- Si possible utiliser un UBO pour projection+camera et un autre pour la *transform*.
- Le plus important est d'avoir une caméra qui peut se déplacer dans la scene et/ou orbiter autour d'un objet

Partie 3 : Options

- Implémenter une classe mat4 en C++ qui inclus la multiplication de matrice
    - Par exemple dans le but de passer la WorldMatrix en uniform (ou via UBO) à la place des trois matrices Translation, Rotation et Scale
- Implémenter les FBO et faire un post-traitement
- Implémenter une interface graphique à l'aide **d'IMGUI** https://github.com/ocornut/imgui

Informations complémentaires

Affichage des objets 3D

La plupart du temps ces données sont fournies directement par le logiciel de modélisation.
On va utiliser la librairie tiny_obj_loader afin de charger des fichiers de type obj.

https://github.com/tinyobjloader/tinyobjloader

Il suffit d'ajouter les fichiers tiny_obj_loader.h et tiny_obj_loader.cc à votre projet.

La structure définissant la variable mesh contient les informations de position, coordonnées de texture et normales (si existantes) ainsi que la liste des indices pour la forme actuelle.

La particularité du format OBJ réside dans le format de description des faces : 1 face est composé de triplets d'indices : en effet les positions, normales et coordonnées de textures sont stockées dans des tableaux différents, et pour des raisons d'optimisation, les tableaux peuvent avoir une taille différente !
On a donc 3 indices, 1 pour les positions, 1 pour les coordonnées de texture et 1 pour les normales.
Comme OpenGL n'est pas capable de gérer 3 tableaux d'indices il faut donc recréer les vertices et éventuellement les indices pour être compatible avec OpenGL.
Voir l'exemple de récupération des indices à partir des « shapes » TinyObjLoader dans le README.md.

Note : il est important de s'assurer que votre modèle 3D est bien triangularisé (composé de triangles et pas de quads ou de polygones). Alternativement, TinyObjLoader offre une option pour forcer la triangularisation.

La bibliothèque gère également les fichiers MTL qui contiennent les informations de matériaux : en premier lieu on a les textures à utiliser pour les composantes ambiantes, diffuses et spéculaires.
Le format supporte également des valeurs de couleurs pour chacune de ces composantes.

Camera 3D et sRGB

Voir le document « Préparation au projet » pour plus de détail

Framebuffer Objects (FBO)

Voir le document « TP Framebuffer object » pour plus de détail.