HAI819I - Moteurs de jeux – TP1

Prise en main de GLFW

Noura Faraj *noura.faraj@umontpellier.fr*

Objectif

Le but du premier TP est de réaliser une application de rendu 3D en utilisant GLFW/OpenGL. Dans ce travail, chaque étudiant aura pour but de réaliser différentes tâches :

- Afficher une scène simple (une caméra, un objet)
- Apprendre à gérer les évènements (clavier, souris)
- Créer une surface et afficher ses triangles
- Contrôler les déplacements de caméra
- Avoir « commité » sur le serveur son projet en fin de séance

Décompresser le fichier, dans le dossier résultant :
mkdir build
cd build
cmake ..
make -j
./launch-TP1.sh

Framework

Pour nos différents TP, nous allons utiliser GLFW. Nous nous servirons principalement de ce Framework afin de gérer les IO de notre application.

Gestionnaire de version

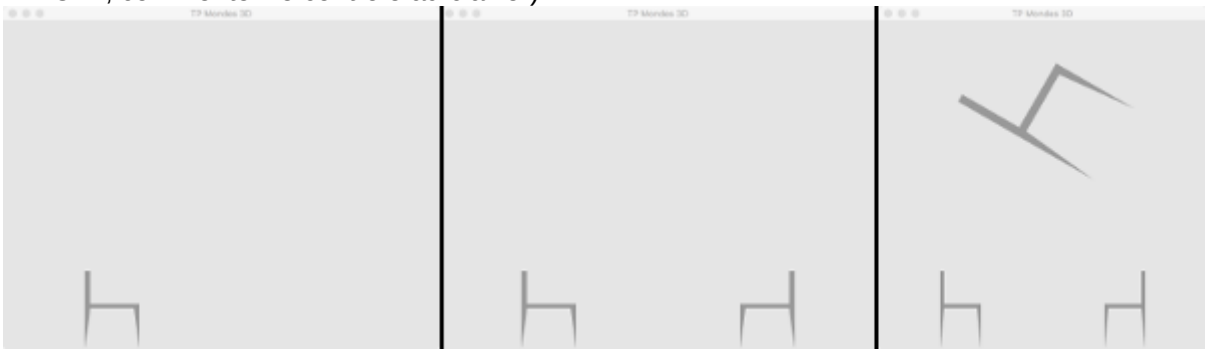
La première étape à réaliser est de sélectionner son groupe de travail. Lorsque votre espace GIT est choisi, alors merci de m'envoyer par mail votre : nom, prénom, groupe sélectionné.
Vous devrez rendre un compte rendu du TP1, ainsi que votre code source (moodle et git).

Question 1

1. Ajoutez une variable uniforme de type mat4 au vertex shader représentant la matrice de transformation à appliquer aux sommets. Construire une matrice identité.

2. Faites-en sorte que les effets de zoom et de translation fonctionnent via cette matrice de transformation.

3. Pour cette petite série d'exercices, modifiez les valeurs de la matrice à la main (dans le code C++, commentez le contrôle au clavier).



a. Réduisez la dimension de la chaise par 2 et la poser sur le sol et légèrement à gauche.
b. Rajoutez une chaise en face.
c. Rajoutez une troisième chaise dont la seule transformation sera une rotation contrôlée au clavier.
d. Faites-en sorte que la chaise tourne autour de son centre de gravité en (0,0.5,0). Vous pourrez construire plusieurs transformations que vous combinerez dans le bon ordre.

4. Changer le model chargé par suzanne.off

   a. Appliquez-lui une rotation 3D d'un angle contrôlé au clavier, et autour d'un axe quelconque via `glm::rotate(matrice, angle_in_degrees,rotation_axis)`.
   b. Calculez et appliquez une rotation de telle sorte que l'axe vertical du personnage (0,1,0) soit aligné avec le vecteur (1,1,1) du repère monde.

Question 3

1. Pour passer d'un espace 2D à une "vraie" vue 3D, il nous faut tout d'abord définir une **matrice de projection** et l'exploiter dans nos shaders. Pour cela, créer une matrice Vous pouvez contrôler la pyramide de vision avec la méthode `glm::perspective(...)`.
2. La seconde étape consiste à contrôler la position et orientation de la caméra via une **matrice de vue**. Pour cela implémentez la méthode `glm::lookAt(...)` dont le but est de définir la matrice `ViewMatrix` de transformation du repère monde au repère de la caméra à partir d'une position d'observation `position`, une cible à viser `target` et une orientation verticale `up` (utiliser les variables `camera_position, camera_target, camera_up`). Pensez à mettre à jour vos shaders pour exploiter cette matrice supplémentaire (vos shaders doivent maintenant prendre en compte 3 matrices : la matrice *objet*, la matrice de *vue*, et la matrice de *perspective*). Testez en créant une vue de 3/4.
3. Vous pouvez zommer et dé-zoomer en utilisant les touches Z et S. Inspirez-vous de ce code pour ajouter les déplacements latéraux.

Question 4

En vous inspirant des méthodes présentes, écrivez les méthodes permettant d'initialiser et d'afficher une surface plane (16*16 sommets) composée de triangles.

Pour cela, vous devrez écrire deux nouvelles méthodes permettant de générer la géométrie du plan et de le dessiner à l'aide de shaders.

Conseil - dessinez les triangles dans un même plan (z=0). Modifier ensuite la caméra pour toujours garder la surface visible.

Ensuite créer une fonction pour calculer les 16*16 sommets et faces. Plaquez une texture sur la surface.

Question 5

Modifier l'altitude (z) des sommets pour réaliser un relief.