Travaux Pratiques Graphique 3D

TP noté du 6 janvier 2010

—EFREI, Master 1, IRV—

Programmation 3D

Dans ce TP, nous allons reprogrammer une (toute petite) partie des algorithmes utilisés par OpenGL : l'algorithme de gestion des faces cachées dit du *Zbuffer* et le calcul de l'éclairement des facettes.

▸ **Exercice 1.** ***(Étape préliminaire)*** *Récupérez les fichiers nécessaires au TP, compilez le projet et exécutez le. Déplacez la caméra à gauche et à droite en cliquant avec les boutons gauche et droite de la souris. Que remarquez vous ? Pour mieux voir, vous pouvez aussi appuyer sur la touche* 'c' *pour commuter l'affichage des faces des cubes avec des couleurs différentes.*

Le problème est que les facettes sont dessinées dans un ordre qui fait que certaines facettes sont dessinées par dessus d'autres qui devraient se trouver devant. Plusieurs algorithmes existent pour pallier ce problème, parmis lesquels l'algorithme du Zbuffer.

▸ **Exercice 2.** ***(L'algorithme du* Zbuffer*)***

Cet algorithme consiste, pour chaque pixel dessiné de mémoriser sa profondeur. Lorsqu'on doit dessiner un nouveau pixel, on vérifie d'abord si un pixel ayant une profondeur plus faible n'a pas déjà été dessiné. Si ce n'est pas le cas, le nouveau pixel est dessiné. Dans le cas contraire, il est ignoré.

Le but de cet exercice est de modifier la fonction display_pixels(), *dans le fichier* tp_note.c, *pour y intégrer l'algorithme du* Zbuffer.

Ce fichier contient l'équivalent du TP 3 dans lequel on a remplacé toutes les fonctions 3D d'OpenGL utilisées par leur équivalent dans la bibliothèque fournie. Cette bibliothèque contient, de plus, une fonction generate_pixels(), *permettant de générer tous les pixels à dessiner pour représenter une facette à l'écran. Une fois que cette fonction a été appelée pour une facette, la fonction* display_pixels() *est appelée pour les dessiner à l'écran, en utilisant les primitives graphiques 2D d'openGL.*

▸ **Exercice 3.** ***(Éclairement des facettes)***

Maintenant que nous avons un dessin potable de nos objets 3D, nous allons les rendre un peu plus réalistes, en leur ajoutant des effets d'éclairement.

Calculez le facteur d'éclairement de chaque face du cube par une source ponctuelle que vous aurez définie. Pour cela, la fonction transformePoint() *pourra vous être utile. Enfin, prenez en compte cet éclairement dans l'affichage.*

NOTE : Il faudra modifier la gestion des couleurs pour cela.