Université Mohamed Khider-Biskra
Faculté des Sciences Exactes et Sciences de la Nature et de la Vie
Département d'informatique

Master 1 IVA **Module: ISI**
Durée: 02 séance. **2019/2020**

TP N°01

Découverte d'OpenGL et modélisation d'un personnage (description hiérarchique d'une scène)

Dans ce TP, nous allons découvrir et manipuler les fonctions de base d'OpenGL. Nous utiliserons ensuite des primitives de haut-niveau (sphère, cylindre, cube et cône) afin de modéliser un personnage (chacun pouvant ajouter sa "`touche artistique"'). Ce personnage nous servira dans les TPs suivants.

1. Prise en main et tests

Récupérez le code fourni sur la page web facebook de votre grpoupe TPs : http://www-evasion.imag.fr/Membres/Estelle.Duveau/SIA.html, compilez puis exécutez (tp1).

Nous allons dans la suite examinée progressivement ce programme et modifier quelques fonctionnalités afin de nous rendre compte de leur impact sur le rendu final.

1.1 Manipulation de la caméra

Le contrôle de la caméra se fait par les flèches du clavier (left, right, up, down, page up, page down). Testez leur comportement. Ces touches sont considérées comme spéciales par GLUT. C'est pourquoi elles sont gérées par la fonction glutSpecialFunc(specialKey) et non par glutKeyboardFunc(commonKey) (cf: main). Inspectez le code de la fonction specialKey() qui gère ces touches spéciales et ses conséquences dans setCamera(). Quels sont les degrés de liberté liés à chaque touche?

1.2 Manipulation de l'affichage

Par défaut, la gestion de l'éclairage est désactivée dans ce programme, ce qui nous permettra de mieux comprendre l'influence des différents paramètres. Modifiez les décisions prises dans initScene() (couleur de fond, modèle de shading, taille des primitives, . . .) et observez à chaque fois le résultat dans les différents modes d'affichage des polygones.

1.3 Manipulation de l'éclairage

Activez la gestion de l'éclairage en décommettant glEnable(GL_LIGHTING) dans l'initialisation de la scène. Le contrôle de la position de la lumière se fait par les flèches du clavier (F1, F2). Testez leur comportement.

1.4 Manipulation de la scène

Quelle est la fonction appelée lors du rendu de la scène?

La fonction drawPrimitives() se charge de positionner et de dessiner les primitives que nous allons utiliser : cube, sph_ere, cylindre et cône. Examinez les fonctions qui permettent de créer ces quatre primitives. Quel est, brièvement, le processus de création de chacune des primitives?

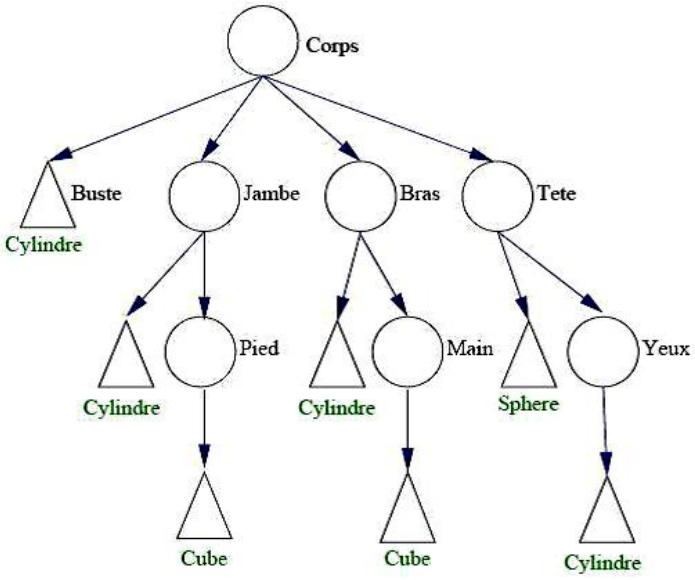
Modifiez les valeurs des paramètres de translation dans drawPrimitives() et observez les résultats. Que pouvez-vous en déduire sur l'enchainement des transformations?

2. Modélisation hiérarchique

Nous allons maintenant assembler ces primitives afin d'obtenir un robot. Nous voulons donc créer (au minimum) son buste, sa tête, ses bras et avant-bras, ses jambes et ses pieds. Vous pouvez bien sûr personnaliser votre création mais il est préférable de commencer par un personnage simple quitte à le raffiner ensuite.

2.1 Hiérarchie des repères

La bonne approche pour modéliser un personnage est d'utiliser une hiérarchie de repères (donc de transformations en OpenGL). On choisit tout d'abord le positionnement du corps tout entier (que l'on choisit traditionnellement comme étant défini par le placement du buste). On positionne ensuite les bras, les jambes et la tête de manière relative au buste. De la sorte, lorsque l'on déplace le corps, tous les membres suivent : ils restent
à la même position relative.
Avant d'implémenter toute hiérarchie en OpenGL, on la représente tout d'abord par un arbre. Dans notre cas, la hiérarchie de base est représentée par l'arbre ci-contre où les disques correspondent aux nœuds de la hiérarchie et où les triangles représentent les formes géométriques dessinées.



Définition

- Un modèle hiérarchique permet de décrire facilement des objets complexes composés d'objets simples. La scène est organisée dans un arbre tel que les objets ne sont plus définis par leur transformation absolue par rapport au repère de toute la scène, mais par leur transformation relative dans cet arbre.
- Comme un même objet peut être inclus plusieurs fois dans la hiérarchie, la structure de données est un Graphe Orienté Acyclique (DAG)
- A chaque noeud est associé un repère. Le repère associé à la racine est le repère de la scène.
- A chaque arc est associée une transformation géométrique qui positionne l'objet fils dans le repère de son père.

OpenGL utilise les coordonnées homogènes pour manipuler ses objets (Math). Il maintient trois matrices 4x4 distinctes pour contenir les différentes transformations.
Pour coder l'arbre de description de la scène, il faut utiliser la pile de transformation, en empilant la matrice de transformation courante (sauvegarde des caractéristiques du repère local associé) avant de descendre dans chaque noeud de l'arbre, et en dépilant la matrice en remontant (récupération du repère local associé).

- **glPushMatrix()** Empile la matrice courante pour sauvegarder la transformation courante.

  glPopMatrix() Dépile la matrice courante (La matrice du haut de la pile est supprimée de la pile, et elle devient la matrice courante).