Projet de conception

- Troisième partie -

- ÉPISODE III -

Le grand rassemblement

http://www.culut.com/free-star-wars-ships-wallpapersstar-backgrounds/

Objectif :

Amener l'étudiant à développer un programme graphique interactif de complexité moyenne comportant :

- le vaisseau/objet que vous avez conçu précédemment
- un environment

Résultat attendu :

Un programme donnant l'impression que l'observateur se déplace dans un environnement semblable au système solaire (les distances et les proportions n'auront pas à être respectées).
Le vaisseau (ou objet équivalent) devra apparaître dans votre scène à l'endroit de votre choix.

Contraintes :

1- Lors d'un déplacement dans votre système solaire, on devra pouvoir observer au moins trois planètes connues, dont la Terre.

Pour obtenir des textures de planète :

http://planetpixelemporium.com/earth.html

2- L'usager ne devra pas pouvoir utiliser la souris pour interagir avec votre programme. On devra plutôt utiliser les flèches du clavier pour se déplacer dans votre scène (avance/recule et rotation à droite/à gauche). Ainsi, à chaque fois qu'on appuiera sur une des flèches, une matrice incrémentale (de translation ou de rotation) sera multipliée par la gauche à la matrice de modélisation s'appliquant à toute la scène

Par exemple, si on appuie sur la flèche pemettant d'avancer, alors on exécutera m = mult(translate(0,0,1), m);
Si on appuie sur la flèche permettant de tourner à droite, on exécutera m = mult(rotate(5.0,0,1,0), m);
Vous devrez modifier les valeurs de déplacement et d'angle de rotation en fonction de l'effet visuel obtenu.

3- La lune devra être localisée à proximité de la Terre (note: la lune n'est pas une planète).

4- Les planètes devront tourner très lentement sur elles-mêmes et la lune devra tourner autour de la Terre. Il n'est pas nécessaire de faire tourner les planètes autour du soleil (elles peuvent donc être fixes dans l'espace).

5- Le vaisseau (ou l'objet) développé dans le travail #5 devra être présent dans votre système solaire.

6- Vous devrez insérer vos éléments dans un "skybox" (http://en.wikipedia.org/wiki/Skybox_(video_games)) recréant un environnement spatial.

Vous pouvez utiliser ou vous inspirer de l'exemple suivant : "space-skybox.html"
- Pour obtenir les images de cet exemple, cliquez ici.

Dans un véritable "skybox", on doit faire en sorte que ce dernier suive la position de l'observateur lorsque ce dernier se déplace. En d'autres termes, si vous appliquez une translation aux objets de votre scène pour donner l'impression que l'observateur avance ou recule, cette dernière ne doit pas être appliquée au "skybox". Ceci donnera un effet de grande distance entre la position de l'observateur et les étoiles d'arrière-plan. **Il est important de noter, cependant**, que les transformations de rotation associées à la progression de l'observateur doivent, quant à elles, s'appliquer au "skybox".

7- Lors du lancement de votre programme, un cube translucide (tournant sur lui-même) devra être visible à proximité de votre vaisseau. Vous devrez "signer" votre animation en insérant votre nom sur l'image qui sera appliquée en texture sur ce cube.

8- Toujours lors du lancement de votre programme, un cube réfléchissant l'environnement (tournant sur lui-même) devra aussi être visible à proximité de votre vaisseau. Il est recommandé d'utiliser des images claires (comme, par exemple, celles du fichier suivant: ciel-nuages.zip) pour créer le "cube map" de cet objet.

Partie facultative (cette partie ne vous méritera aucun point supplémentaire)

Vous pouvez inclure le vaisseau ARC-170 en utilisant la fonction "createModelFromObjFile()" présente dans le programme Javascript inclus dans le fichier suivant :

=> Cliquez ici pour accéder à la page HTML montrant le vaisseau ARC-170

Ce vaisseau a été créé à l'aide d'un logiciel de modélisation et sauvegardé dans le format OBJ (information).

Évaluation :

Comme pour le travail #5, ce dernier sera évalué selon les éléments suivants:

Critère #1: qualité du code (organisation, structures de données, documentation ...)
Critère #2: qualité du résultat graphique obtenu