TD de Synthèse d'images Avancée n°1

Cours d'infographie
Master 2 : Science informatique option ISI

Harmoniques Sphérique et IBL

Calcul de l'illumination d'une scène par une texture d'environnement. Application sous OpenGL.

Le but de ce travail est de calculer l'illumination que reçoivent un ensemble d'objet 3D d'une scène lumineuse. Plus précisément, nous cherchons à calculer l'éclairage induit par une texture d'environnement. Cette technique, qui utilise des images (réelles, dessinées ou de synthèse) pour calculer l'éclairage, s'appelle, en anglais l'*Image Based Lighting*. Dans ce cadre, on utilise plus de source de lumière virtuelle mais bien un cube d'image représentant l'environnement entourant les objets. Cette technique a les mêmes contraintes que celles de l'*environnement mapping*.

La lumière incidente L issue de l'environnement, ainsi que l'irradiance E en un point d'une surface, peuvent être décomposés sur la base de fonctions des harmoniques sphériques.

$$L(\theta,\phi)=\sum_{l=0}^{\infty}\sum_{m=-l}^{l} L_{l,m}Y_{l,m}(\theta,\phi)$$

$$E(\theta,\phi)=\sum_{l=0}^{\infty}\sum_{m=-l}^{l} E_{l,m}Y_{l,m}(\theta,\phi)$$

Les définitions des fonctions d'harmonique sphérique sont données dans l'article « An Efficient Representation for Irradiance Environment Maps » de Ravi Ramamoorthi et Pat Hanrahan. Si on ignore l'ombrage, et que l'on considère des surfaces purement diffuses, alors on obtient la relation liant l'irradiance d'une surface à la luminance reçue :

$$E(n)=\int_{\Omega(n)} L(\omega)(\vec{n}.\vec{\omega})\,d\omega$$

Ressource : Le programme de base

Vous disposez d'un programme non fini en C/C++, patchwork de choses commencées et non finies. Votre objectif est d'exploiter ce « terreau » pour implémenter les techniques citées précédemment. Rien de bien sophistiqué concernant le C++ n'apparaît dans ces classes. Ces fichiers étant un patchwork d'ancien code d'*environment mapping* vous y trouverez des fonctions peu utiles, et probablement du code GL version 2.X.

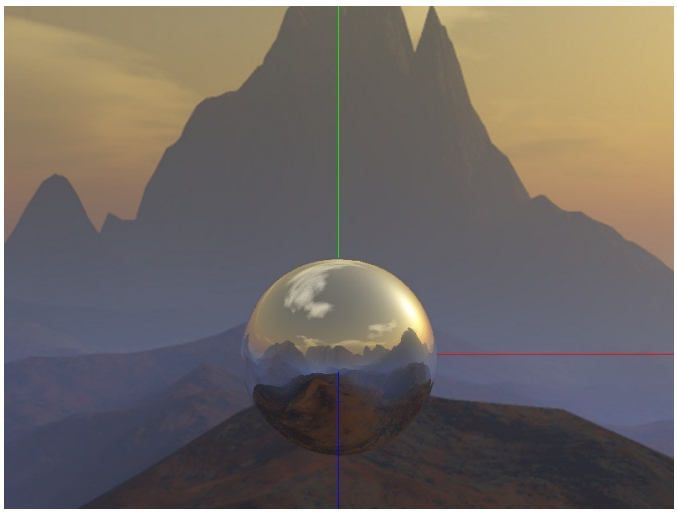
Les fichiers .h/.cpp situés dans le répertoire tools vous permettront d'avoir des fonctions mathématiques et utilitaires utiles pour la mise en œuvre du programme. Les classes les plus utiles sont surtout vector3d et 4d (un vecteur 3D & 4D), texture (chargement d'image pour les textures) et shaders (gestion de shaders), ainsi que camera... Cet ensemble de fichier est robuste et conforme à OpenGL 3.X+

Les fichiers situés dans le répertoire src forment l'application et sont les + disparates. Les fichiers lightp ne seront pas ou peu utilisés. Les fichiers interface forment le cœur de l'IHM, alors que les fichiers display s'occupent plus particulièrement de l'affichage OpenGL. Les fichiers obj3ds gère en fait les maillages en provenance de fichier .3ds. Les fichiers scenedes stockent la scène complète. Enfin, le plus important consiste en les fichiers skybox qui vont contenir les images de l'environnement.

Pour le moment l'application se contente de n'afficher en OpenGL que deux surfaces (un cube et un cylindre) avec 2 shaders très simples. Vous pouvez activer les différents modes de visualisation sur les touche F1,F2,F3 et F4. La touche F11 vous permet de voir directement les points lights. Les flèches ainsi que les touches 'Page Up' et 'Page Down' vous permettrons de vous déplacer au sein de la scène. Les touches 'a', 'q', 'w' et 'x' permettent de s'orienter.

Exercice 1 : Implémentation de l'environment mapping

1. Téléchargez sur le site Internet http://www-igm.univ-mlv.fr/~biri/ section Enseignement, puis MII2, l'archive. Dézippez, détarrez et compilez. Vous risquez d'avoir besoin de la lib3ds que vous devrez installer localement.
2. Étudiez chacune des classes et fichiers du répertoire `src` et plus particulièrement `skybox`, `interface` et `display`.
3. Afin de vous familiariser avec l'application, essayer d'afficher le modèle 3D du lézard (il est préchargé dans l'application). Appliquez lui des transformations afin qu'il ait une taille & une orientation normale. Vous pouvez exploiter, pour ce faire, la pile de matrice implémentée dans `matrix_stack.hpp` du répertoire `tools`.
4. Affichez une skybox. Pour ce faire, vous devez créer les éléments géométriques nécessaires dans les fichiers `skybox.*` et vous pouvez essayer d'exploiter le cube défini dans `basic_mesh.hpp` (du répertoire `tools`). Vous afficherez sur la skybox les textures préchargées. Pour ce faire vous devrez probablement modifier les shaders.
5. Mise en œuvre du shader envmap : Vous devez implémenter le shader permettant de faire de l'*environment mapping*. Une cube map est une texture spéciale composée de six images qui se répartissent sur les faces d'un cube et dont les coordonnées de textures sont tridimensionnelles. Ces dernières représentent un vecteur dans l'espace objet. D'une manière générale, le but de l'exercice est de calculer en chaque point de la surface de l'objet le vecteur issu de la réflexion du vecteur provenant de l'œil. Ces calculs, effectués dans un shader, nécessitent certains arrangements car les calculs doivent être effectués dans le repère de l'objet. Ainsi une fois les vecteurs obtenus dans le repère de l'œil, il faut le transformer pour l'obtenir dans le repère de l'objet. Un moyen pour effectuer cette opération consiste à récupérer la matrice ModelView dans le programme OpenGL, de calculer son inverse et de la transmettre au shader. Récupérer la matrice ModelView peut être effectué via la fonction `glGetFloatv(GL_MODELVIEW_MATRIX, mat)`.
6. Faites en sorte de pouvoir changer, à la volée, la texture d'environnement (quelques exemples sont données avec l'archive mais vous pouvez aussi en télécharger sur Internet facilement).



Exercice 2 : Image Based Lighting

En vous inspirant de l'article mentionné ci-dessus, vous implémentez un éclairage basé image. Pour ce faire :

1. Créez un ensemble de fonction permettant de calculer chaque harmonique sphérique pour n'importe quel vecteur (x,y,z) donné.
2. Calculez les 9 coefficients $L_{l,m}$ à partir de votre skybox. Attention, n'oubliez pas que nous avons des images couleurs !
3. Calculez les coefficients de la matrice *M* à utiliser dans le shader
4. Exploitez cette matrice *M* dans le shader, et implémentez le IBL
5. Faites en sorte que l'on puisse changer, dynamiquement, l'environnement.
6. Faites en sorte de pouvoir effectuer une rotation de l'environnement.

Exercice 3 : Aller plus loin

1. Réalisez une visualisation de la fonction E. Pour ce faire, vous pouvez, par exemple, colorier une sphère unité avec des couleurs relatives à l'intensité de E dans chaque direction.
2. Dans cet exercice, nous n'avons considéré que des surfaces purement diffuses. Quelles seraient les modifications à apporter pour passer à des BRDF plus complexe. Essayez avec une BRDF anisotrope.