Université de Limoges

master ISICG limoges

Informatique Graphique
Moteurs 3D temps réels
TP2 : Illumination

G.Gilet

29 septembre 2017

Au cours de ce TP, nous allons réaliser un *shader* permettant de calculer l'illumination d'un objet suivant le modèle de Phong, afin de produire des scènes plus réalistes.

1 Modèle de Phong - Rappels

Nous allons ici utiliser la méthode (empirique) de Phong afin de calculer une valeur de lumière pour chacun de sommets de notre scène. Cette valeur approxime à la quantité de lumière réfléchie par l'élément de surface au sommet atteignant l'oeil de l'utilisateur (ici la caméra). Dans ce modèle, ce terme est définit en fonction de trois composantes :

La composante ambiante : Ce terme représente une approximation de la lumière réfléchie dans l'ensemble de la scène. Elle à une valeur constante.

La composante diffuse : Ce terme décrit la lumière provenant d'une source de lumière et réfléchie dans toutes les directions. La quantité de lumière réfléchie dépende de l'incidence de la lumière par rapport à la surface.

La composante spéculaire : Ce terme décrit les réflexions parfaites. La lumière est réfléchie dans une direction privilégiée. Ici, l'intensité lumineuse est maximale dans la direction de réflexion et décroît au fur et à mesure que l'on s'en éloigne. La contribution de cette composante spéculaire dépend donc ici du point de vue (c'est-à-dire de la position de l'observateur par rapport au sommet considéré).

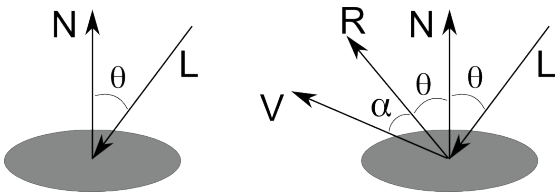

FIGURE 1 – Illumination diffuse et spéculaire.

Le modèle d ?éclairage permettant de calculer la couleurs réfléchie I d'un sommet s est :

$$I(s) = K_a * I_a + K_d * I_d + K_s * I_s \tag{1}$$

avec K_a, K_d et K_s les coefficients ambiant, diffus et spéculaire du matériau de l'objet considéré. Ces trois coefficients permettent de pondérer les facteurs calculés pour chaque élément de surface et sont un reflet de la nature du matériau de l'objet. K_s permet de déterminer si le matériau est fortement spéculaire ou non tandis que K_d permet de déterminer le caractère lambertien de l'objet. En règle générale ($K_a + K_d + K_s = 1.0$).
Les composantes I_a, I_d et I_s sont calculées en fonction des couleurs ambiantes, diffuses et spéculaires des matériaux (C_a, C_d et C_s). Ce modèle nous permet d'associer une couleur spécifique pour chaque composante des objets (il est ainsi possible de définir un objet qui à une couleur diffuse rouge mais une couleur spéculaire bleue)
Les composantes se déduisent d'après les équations suivantes (:

— $I_a = C_a$
— $I_d = C_d * max(cos(\theta),0)$
avec θ l'angle entre la normale N à un sommet et la direction de la lumière $-L$ (cf figure 1 à gauche)
— $I_s = C_s * max(cos(\alpha),0)^s$
avec α l'angle entre la direction de réflexion de la lumière R et le vecteur de vue V (cf figure 1 à droite). R peut être calculé en utilisant la fonction glsl *reflect*.

Il est également possible de multiplier chaque composantes par LC afin de prendre en compte la couleur LC de la lumière.

2 Exercices

2.1 Eclairage par sommet - Eclairage Diffus

Nous allons commencer par réaliser un éclairage de Phong en effectuant le calcul pour chaque sommet d'un maillage. Nous commencerons par calculer les termes diffus (et ambiants) uniquement.

2.1.1 Premier Shader

Afin de commencer, déclarez dans le VP une variable $posLum$ à la position $(0,30,0)$ et calculez le vecteur correspondant, pour chaque sommet, à la direction de la lumière ($-L$ sur la figure 1). Calculez ensuite le terme diffus (en fonction de la couleur de l'objet) et la valeur de couleur finale de l'objet et transmettez cette valeur au *fragment program* (qui devra bien entendu l'assigner à chaque fragment).

2.1.2 Indépendance Lumière-objet

L'exercice précédent nous permet d'avoir un éclairage diffus. Toutefois, la lumière est définie dans le shader et reste surtout attachée au lapin. Il est donc impossible en l'état de faire tourner notre objet sans faire tourner également la lumière. Nous allons maintenant séparer ces deux éléments, c'est a dire définir la position de la lumière dans le repère de la scène, indépendamment du repère du lapin. Nous considérons pour la suite que la lumière se trouve aux coordonnées $(0,30,0)$ dans le repère de la scène.

— Déplacer dans le shader la déclaration de la variable PosLum vers le bloc des variables uniform

— Créez dans la classe BaseMaterial une GPUvariable et recuperez l'adresse GPU de la variable PosLum
— Il va maintenant falloir effectuer un changement de repère, c'est à dire trouver l'expression du point $(0,30,0)$ du repère de la scène dans le repère de l'objet. Exprimez , dans la fonction update(), la position de la lumière dans le repère de la scène et envoyez le tout au shader (pour cela, regardez les fonctions de la classe Frame). Note : Vous pouvez accéder au repère de la scène en utilisant $Scene::getInstance()->getScene()->frame()$.
— Déplacez maintenant le lapin indépendamment de la lumière (Pensez a sélectionner le bon noeud a manipuler)

2.2 Eclairage par sommet - Eclairage Spéculaire

Ajoutez maintenant le terme spéculaire du modèle de Phong. Pour cela vous aurez besoin de la position de la caméra par rapport à l'objet : le point $(0.0,0.0,0.0)$ du repère de la caméra, à transformer dans l'espace adéquat. Transmettez cette valeur au shader et ajoutez a votre calcul précédent le terme spéculaire de l'ombrage de Phong (avec $C_s = (1.0,1.0,1.0)$ et $s = 100$).

2.3 Eclairage par fragment

Nous avons, dans les exercices précédents, calculé une valeur d'illumination pour chaque sommet et assigné cette valeur interpolée à chaque fragment lors du *fragment program*. Cette méthode peut toutefois présenter des artefacts visuels, notamment dans le cas de modèles à faible tesselation.
Il s'agit maintenant d'effectuer ce calcul d'éclairage au sein du *fragment program* en utilisant la valeur interpolée (automatiquement) des normales pour chaque fragment. Les vecteurs de direction de la lumière et de l'observateur s'interpolant linéairement le long d'un triangle, il est suffisant de calculer ces valeurs au sein du *vertex program* et de les transmettre à la suite du pipeline afin d'utiliser une interpolation de ces valeurs. Réalisez donc le calcul d'éclairage dans le *fragment program* en utilisant les valeurs interpolées adéquates (pensez toutefois a re-normaliser les vecteurs interpolés).