TP3 : Illumination Locale.

Motivations

La simulation de l'éclairage est omniprésente dans les environnements virtuels. En fonction du contexte d'utilisation (films, jeux vidéos, dessins animés, ...) et des souhaits, les techniques d'illumination sont radicalement différentes.

Le rendu photoréaliste est utilisé dans certains jeux, films et création d'images réalistes. L'objectif est de reproduire de la manière la plus visuellement fidèle le comportement de la lumière et des matériaux. On peut le faire en utilisant des techniques **d'illumination locale** où l'on va utiliser de nombreuses sources de lumières, les combiner entre elles pour obtenir un résultat convaincant. Ces techniques sont généralement économiques et permettent leur utilisation dans des contextes interactifs. Ou alors on utilise des techniques **d'illumination globale** qui cherchent à se rapprocher au plus du comportement de la lumière (**raytracing**, **physically based rendering**). Le temps pour obtenir une image peut alors devenir très long en fonction de la complexité de la scène (plusieurs jours).

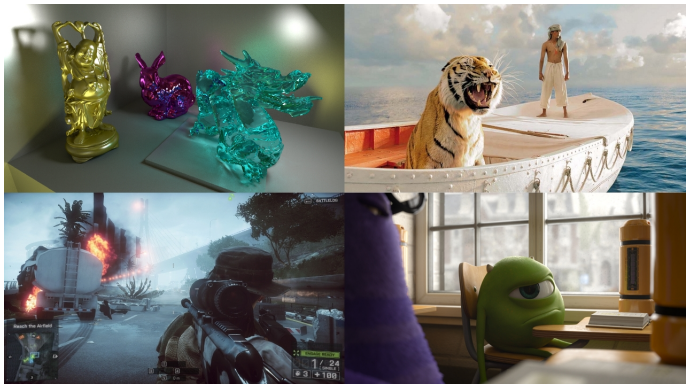


Figure 1: Différents exemples de rendu photoréalistes dans différents domaines : création d'image, films et jeux vidéos.

Le rendu non photoréaliste (NPR : non photo realistic rendering) va dans un autre sens. Il permet de reproduire des effets obtenus par des média autres que la lumière (crayon de papier, pinceau, ...) et de reproduire l'expressivité plus que la réalité. Ce domaine est très utilisé dans les dessins animés, les jeux vidéos mais également le design et la peinture virtuelle.

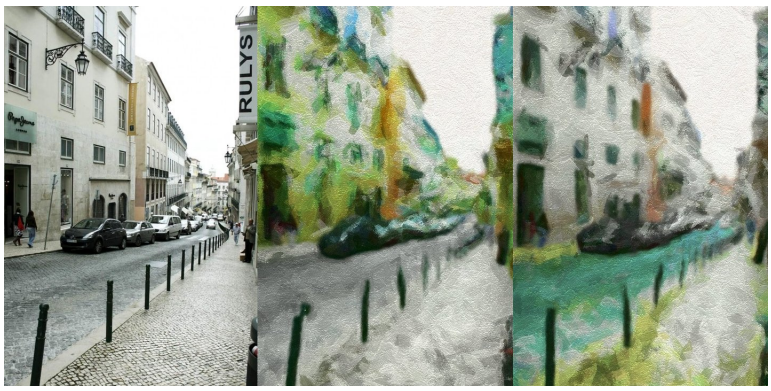


Figure 2: Le rendu non photoréaliste permet de reproduire des médias connus de tous comme l'aquarelle.

Dans ce TP on s'intéressera aux bases de l'illumination locale.

Une introduction aux modèles d'illumination

Les modèles suivant permettent de déterminer la radiance d'un point sur une surface. Plus grossièrement ils permettent de déterminer quelle couleur attribuer à un point d'une surface.

Rendering Equation [Kajiya 1986]

$$L(\mathbf{p} \rightarrow \mathbf{e}) = L_e(\mathbf{p} \rightarrow \mathbf{e}) + \int_{\Omega_n} \rho(\mathbf{p}, \mathbf{e}, \mathbf{l})(\mathbf{n}.\mathbf{l})L(\mathbf{p} \leftarrow \mathbf{l})d\mathbf{l}$$

- $\mathbf{e}$ est la direction du regard
- $\mathbf{n}$ est la normale au point $\mathbf{p}$
- $\mathbf{l}$ est la direction de la lumière sur l'hémisphère Ω_n
- $L(\mathbf{p} \rightarrow \mathbf{e})$ est la radiance sortante. Combien d'énergie arrive à l'oeil.
- $L_e(\mathbf{p} \rightarrow \mathbf{e})$ est la radiance emmise. Combien d'énergie produit la surface.
- $L(\mathbf{p} \leftarrow \mathbf{l})$ est la radiance entrante. Toute l'illumination venant de $\mathbf{l}$ et arrivant au point $\mathbf{p}$.
- $(\mathbf{n}.\mathbf{l})$ est l'orientation de la surface, le produit scalaire entre la normale au point et la direction de la lumière.
- $\rho(\mathbf{p}, \mathbf{e}, \mathbf{l})$ sont les propriétés matérielles. Combien d'énergie la surface réfléchit vers l'oeil étant donnée une lumière incidente $\mathbf{l}$. On les appelle **BRDF** (Bidirectional Reflectance Distribution Function).

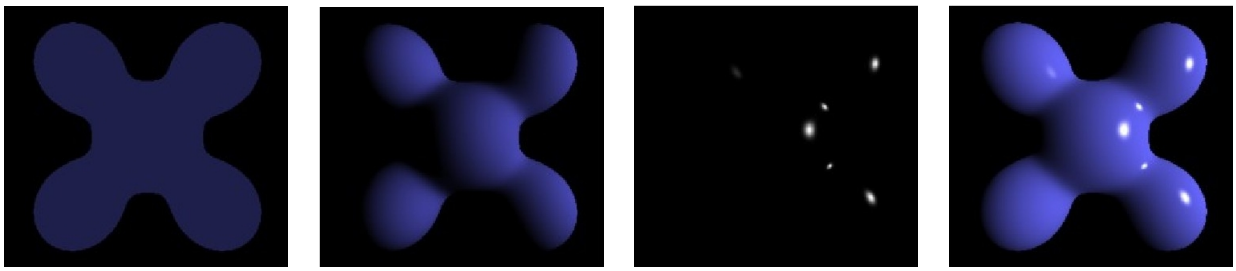
Modèle Générale d'Illumination Locale

$$L(\mathbf{p} \rightarrow \mathbf{e}) = \rho_a L_a + \sum_k \rho(\mathbf{p}, \mathbf{e}, \mathbf{l_k})(\mathbf{n}.\mathbf{l_k})L(\mathbf{p} \leftarrow \mathbf{l_k})$$

1. k est le nombre de lumière dans la scène.
2. $\rho_a L_a$ est la lumière ambiante (approximation d'une lumière indirecte).

Modèle de Phong

Le modèle de Phong est une somme pondérée de terme ambiant, diffus et spéculaire.



Ambient Diffus Spéculaire Total

Figure 3: Différentes composantes du modèle d'illumination de Phong

$$L(\mathbf{p} \rightarrow \mathbf{e}) = \rho_a L_a + \sum_k \rho_{kd} L_{kd}(\mathbf{n}.\mathbf{l_k}) + \rho_{ks} L_{ks} max(\mathbf{r}.\mathbf{e}, 0)^s$$

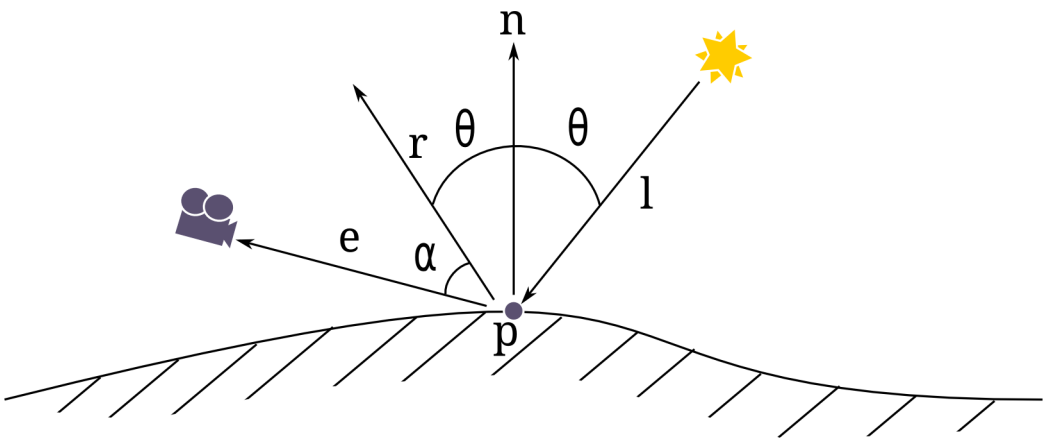
Pour une seule lumière :

$$L(\mathbf{p} \rightarrow \mathbf{e}) = \rho_a L_a + \rho_d L_d(\mathbf{n}.\mathbf{l}) + \rho_s L_s max(\mathbf{r}.\mathbf{e}, 0)^s$$

$\mathbf{r}$ est la réflexion de la lumière sur la surface.
$\rho_a, \rho_d, \rho s$ sont les propriétés matérielles de la surface pour les termes ambiants, diffus et spéculaire.
L_a, L_d, L_s sont les couleurs lumières pour les termes ambiant, diffus et spéculaire. s est la brillance de la surface.

Travail demandé

Exercice 1 : Gouraud Shading



La technique du Gouraud Shading revient à implémenter le modèle de Phong dans le vertex shader. Une couleur est attribué à chaque sommet. Cette couleur est une somme pondéré des propriétés de la lumière et du matériau.

L'intensité du terme spéculaire ou du terme diffus est contenu dans le produit scalaire. Un alignement des vecteurs du produit scalaire donne une valeur maximale et donc une intensité maximale.

1. Dans le vertex shader initialiser les propriétés de la lumière et du matériau.
2. Dans quel espace doivent être exprimés les différents vecteurs et positions ?
3. Calculer la radiance sortante du modèle de Phong dans le vertex shader et transmetter la au fragment shader. Conseil : Procéder composante par composante (ambiant, diffus puis spéculaire).
4. Tester les différents paramètres.

Les nouvelles fonctions GLSL utilisées :

1. normalize(...);
2. dot(... , ...);
3. reflect(..., ...);
4. pow(..., exponent);
5. max(..., ...);

Attention : Les vecteurs doivent être normalisés.

Au cas où : Si l'objet que l'on manipule subit un changement d'échelle alors il faut recalculer les normales car celles ci ne sont pas préservées. On peut gérer ce changement d'échelle via une opération matricielle pour replacer correctement les normales dans l'espace de la caméra :

```
normal matrix = transpose(inverse(upperleft3x3(modelview matrix)));
```

Dans ce TP cette modification n'est pas nécessaire mais il reste de bon de le savoir.

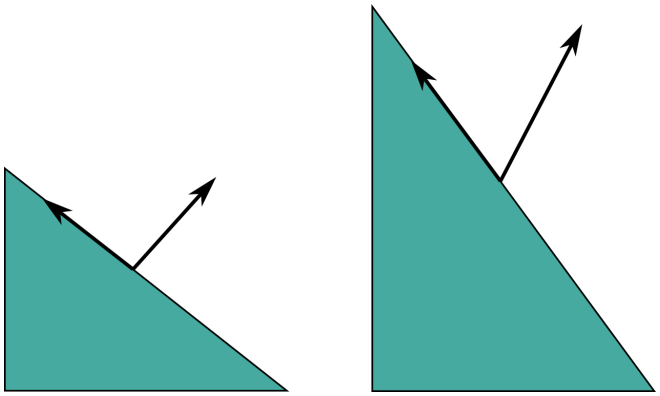


Figure 4: A gauche après une mise à l'échelle uniforme les normales sont préservées. A droite après un scaling vertical les normales ne sont plus préservées.

Exercice 2 : Phong Shading

La technique du Phong Shading est l'implémentation du modèle de Phong dans le fragment shader. Une couleur est attribuée à chaque fragment.

1. Implémenter le **Phong shading**.
2. Qu'observez vous ? Quelle est la différence ?
3. Visualisez séparément l'éclairage ambiant, l'éclairage diffus puis l'éclairage spéculaire.
4. Dans le **Gouraud Shading**, utilisez le mot clé **flat** pour l'attribut de couleur que vous transmettez au fragment shader. Que se passe t-il ? Pourquoi ?

Exercice 3 : Blinn-Phong Model

Soit $\mathbf{h} = \frac{\mathbf{l}+\mathbf{e}}{\| \mathbf{l}+\mathbf{e} \|}$, le modèle de Blinn-Phong s'écrit :

$$L(\mathbf{p} \rightarrow \mathbf{e}) = \rho_a L_a + \rho_d L_d(\mathbf{n}.\mathbf{l}) + \rho_s L_s max\left(\mathbf{h}.\mathbf{n}, 0\right)^s$$

1. Implémenter le modèle de Blinn-Phong dans le fragment shader.
2. Qu'observez vous ?
3. Multipliez la brillance par 2. Qu'observez vous ?
4. Quelle est l'avantage de ce modèle sur le modèle de Phong ?

Exercice 4 : Toon Shading

Le **Toon Shading** est un modèle générant des images ressemblant aux personnages de cartoons. L'idée clé est de seuiller les valeurs d'éclairage, afin d'obtenir des intervalles de couleurs uniformes. Implémenter un Toon Shading simple ne prenant pas en compte la composante spéculaire. (Bonus : Implémenter un toon shading prenant en compte la composante spéculaire).

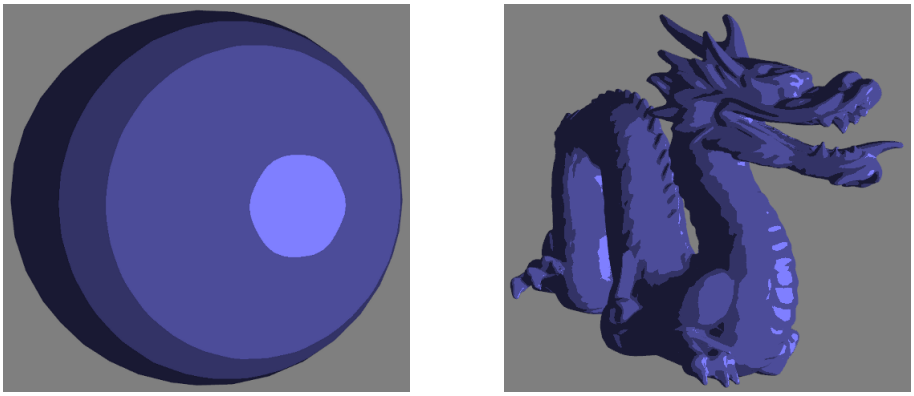
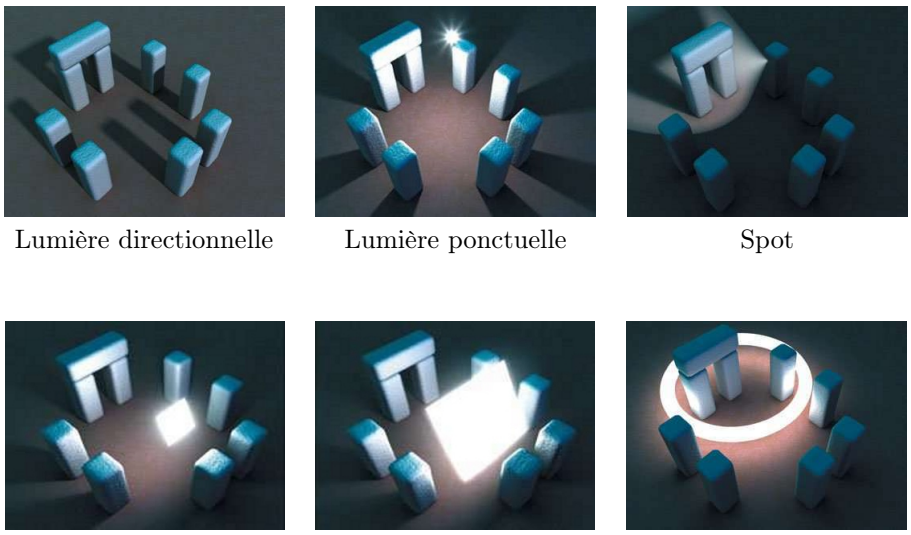


Figure 5: Exemple de Toon Shading

Exercice 5 : Lumière directionnelle et Spot



Lumière directionnelle Lumière ponctuelle Spot

Figure 6: Lumières définies sur différentes régions.

Il exite deux familles de lumières. Les lumières **infinitésimal** où l'on retrouve les lumières ponctuelles, directionnelles et spots. Les lumières définies par **région**. Les premières sont simples à implémentées tandis que les secondes sont très chers à calculer.

1. Implémenter une lumière directionnelle.
2. Implémenter un spot.

La lumière directionnelle est uniquement définie par une direction. On retrouve ce type de lumière dans les jeux vidéos pour faire office de lumières distantes (sun).

Le spot lumineux est définie par une direction, une valeur *cut off* qui définie le cône du spot et un exposant qui précise si la concentration/l'atténuation de la lumière.

Exercice 6 : Génie Logiciel

Les lumières sont directement intégrées dans le shader. Cela permet une expérimentation rapide minimisant le risque d'erreur. Néanmoins on s'est limité à une seule lumière et à une lumière fixe. Dans n'importe quel moteur graphique ces deux limitations sont levées par un génie logiciel légèrement plus poussé.

1. Encapsuler le travail sur les sources lumineuses dans des classes C++.

2. Gérer l'envoi des données aux shaders.
3. Connecter la position de la lumières et ses différents paramètres aux touches clavier pour pouvoir interagir.
4. Implémenter un soleil se levant et se couchant périodiquement.
5. Comment géreriez vous plusieurs lumières ?

Organisation

Rendre une archive *nom1_ nom2.zip* contenant :

- Le code commenté prêt à être compilé/executé et générant les sorties demandées.
- Un rapport contenant vos résultats (réponses, images, commentaires, ...).

Webographie

- `www.shadertoy.com` - Expérimentation/Partage en ligne de shader.
- `www.lighthouse3d.com` - Différents tutoriaux.
- `http://digital-lighting.150m.com/ch02lev1sec2.html` - Pour en savoir un peu plus sur l'éclairage.
- `http://gurneyjourney.blogspot.fr/2010/01/color-constancy.html` - Sur la consistance de la couleur.