

CENTRALE
LYON

Informatique Graphique

Rapport final

Élève :
Baptiste GRIGNON

Enseignant :
Nicolas BONNEEL

17 mars 2025

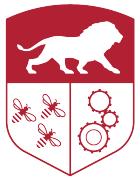


Table des matières

1	Introduction	2
2	Travail effectué & résultats	3
2.1	Sphères	3
2.2	Maillage	4

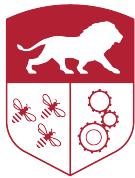


1 Introduction

Ce rapport présente les résultats obtenus dans le cours d'Informatique Graphique donné à l'École Centrale de Lyon par M. Nicolas Bonneel (<https://perso.liris.cnrs.fr/nicolas.bonneel/teaching.html>). Ce cours consiste en la création d'un moteur de rendu basé sur la physique de type "ray-tracer". Le principe est de tirer des rayons du centre de la caméra vers le plan image, puis de simuler leur propagation dans la scène.

Le code de mon projet est disponible sur le dépôt github suivant : <https://github.com/ITYL-dev/MyGraphicsEngine>. Par ailleurs, toutes les images générées présentes dans ce rapport et sur le dépôt sont de résolution 512 x 512.

J'ai rédigé presque entièrement le code, à part les morceaux donnés (lecture fichier OBJ, écriture image en .png, ...) et la fonction *BoundingBox :: intersect*, qui a directement été copiée du tableau lors du cours, car je n'en comprenais pas le principe au moment où nous avons traité cette partie en cours (je l'ai depuis comprise, mais le code est resté inchangé). ChatGPT n'a été utilisé qu'occasionnellement comme documentation rapide pour les fonctionnalités de C++, mais jamais pour générer du code. Je tiens néanmoins à préciser qu'ayant été voisins pendant la quasi-totalité des séances du cours, Gulliver Larsonneur et moi-même nous sommes souvent entraînés, pour la résolution de bugs notamment. Ainsi, bien que je n'aie pas copié de parties de son code, il n'est pas impossible que certaines similarités apparaissent. Par exemple, il me semble que nous utilisons tous les deux la macro `_OPENMP` pour conditionner certaines parties du code afin qu'il soit compilable avec et sans OpenMP.



2 Travail effectué & résultats

L'objectif est de démontrer les capacités du moteur de rendu écrit pendant ce cours à travers des exemples de rendus, mettant en évidence les différentes fonctionnalités.

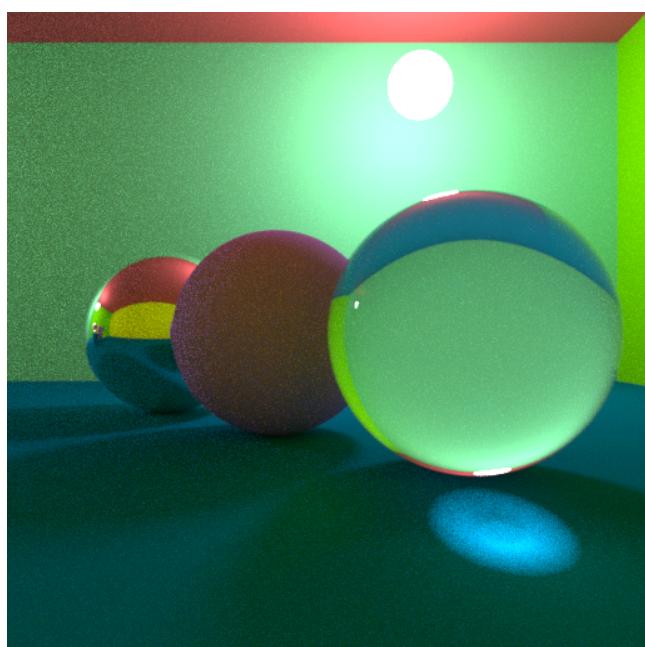
2.1 Sphères

Sauf indication contraire, les images de cette partie ont été générées avec 1024 rayons et 8 rebonds (512 x 512). Le temps de génération est donné entre parenthèses pour chaque image.

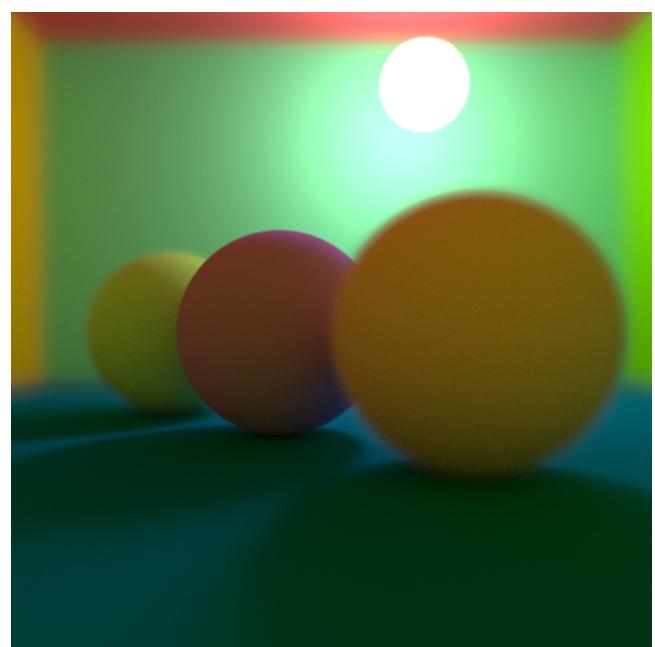
Dans notre moteur de rendu, nous avons commencé par développer des fonctionnalités pour des sphères. Les 2 images suivantes présentent l'étendue des fonctionnalités de notre moteur. La source de lumière est sphérique et la scène est illuminée par éclairage direct et indirect. Les murs de la scène sont des sphères de rayons très grands et de centres très lointains. 3 sphères sont présentes au centre de la scène.

Sur l'image (1a), le mur de gauche et la sphère la plus éloignée ont les propriétés de réflexions d'un miroir, la sphère du milieu est une sphère diffuse et la sphère la plus proche est transparente. On peut par ailleurs constater l'apparition d'un caustique très simple.

Sur l'image (1b), de la profondeur de champ a été appliquée, ce qui rend la sphère au 1er plan et la sphère en arrière-plan floues, tandis que la sphère médiane est nette.

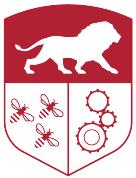


(a) Sphères diffuses, miroirs et transparentes avec éclairage indirect (262 secs)



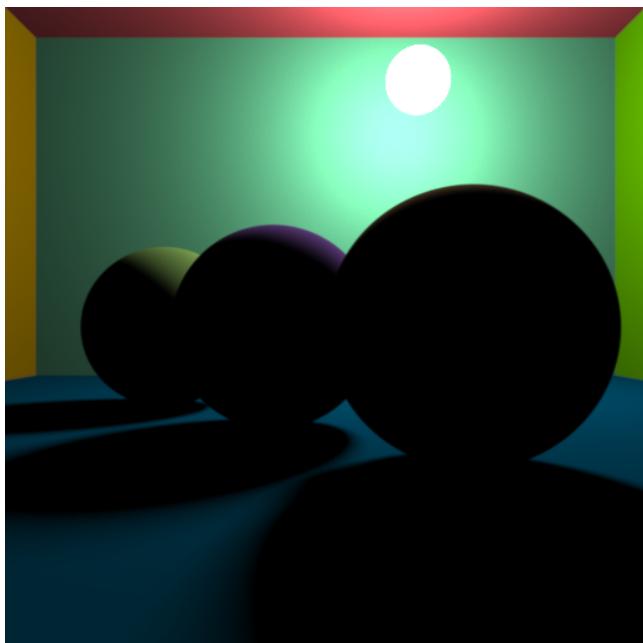
(b) Sphères diffuses avec éclairage indirect et profondeur de champs (324 secs)

FIGURE 1

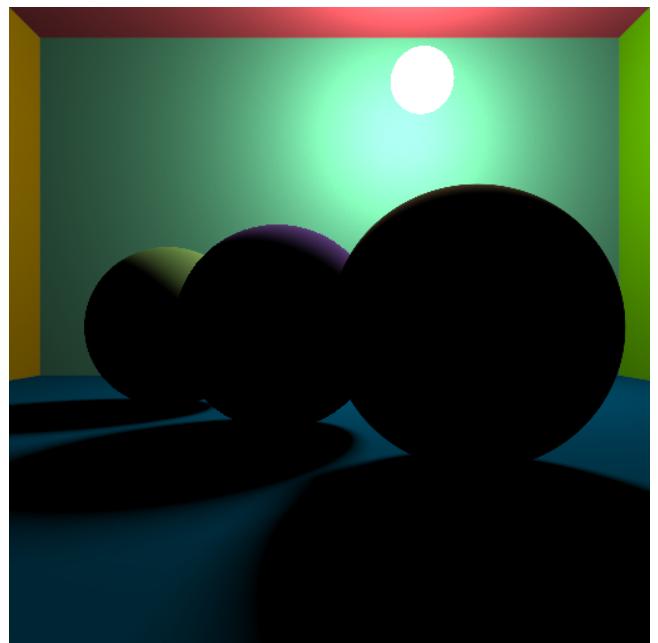


Les 2 images suivantes présentent la même scène que (1b), mais sans l'éclairage indirect et la profondeur de champs. On remarque sur la figure (2a) que la scène est beaucoup plus sombre et que les ombres sont totalement noirs. On remarque également que les ombres ne sont pas tout à fait nette, car la source de lumière est sphérique.

Sur la figure (2b), l'antialiasing (qui est autrement activé pour toutes les images) a été désactivé pour mettre en évidence les bords des sphères sont pixelisés.



(a) Sphères diffuses avec lumière directe uniquement (43 secs)



(b) Idem, sans antialiasing (45 secs)

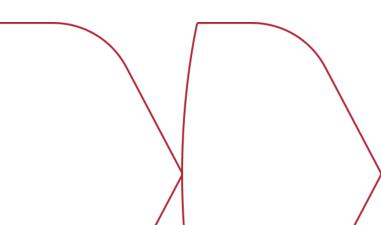
FIGURE 2

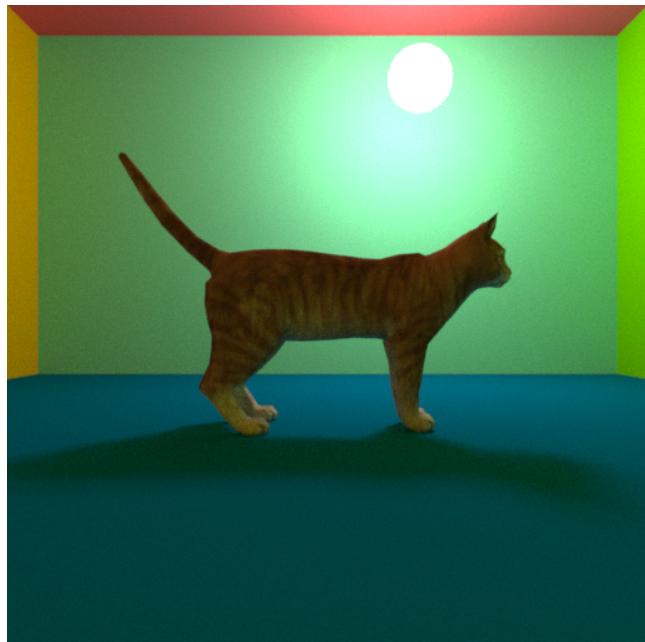
2.2 Maillage

Sauf indication contraire, les images de cette partie ont été générées avec 128 rayons et 8 rebonds (512 x 512). Le temps de génération est donné entre parenthèses pour chaque image.

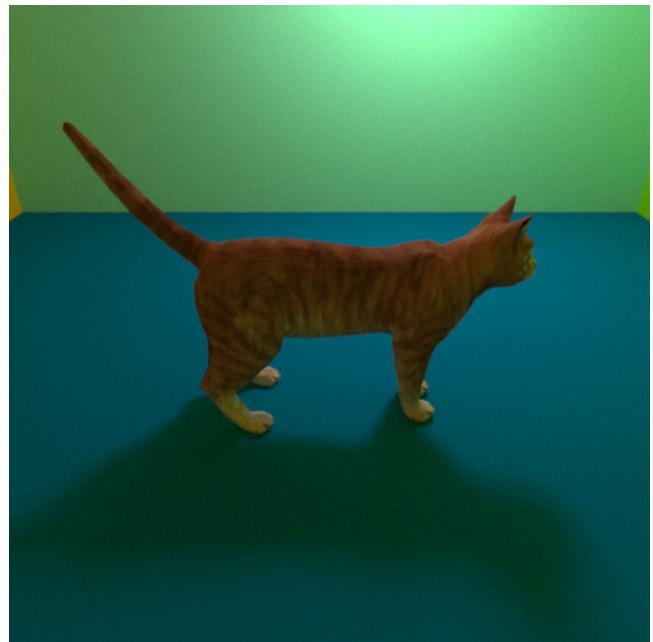
Une fois notre moteur fonctionnel avec des sphères, nous sommes intéressé au rendu de maillages. Les images suivantes présentent maillage texturé de chat dans une scène similaire à celle de la partie précédente.

Sur l'image (3a), on observe un modèle 3D de chat et sur l'image (3b), on observe la même scène d'un point de vue différent, ce qui a été permis par la transformation géométrique de la caméra. De plus, dans le dossier "images" présents sur le dépôt github, une animation au format gif présente une rotation de la caméra de 30° vers le bas. Celle-ci a été générée en faisant le rendu de 31 images (de manière similaire à (3b)) avec des angles de caméras différents. Par ailleurs, toutes les images présentes dans ce rapport sont également présentes dans ce dossier.





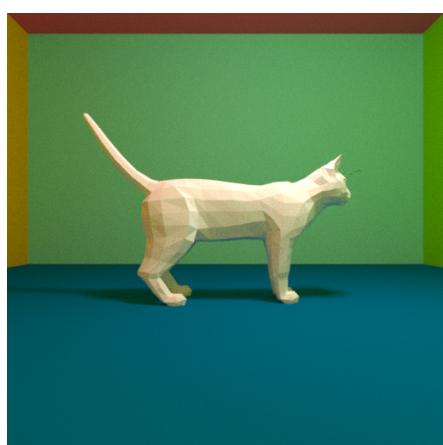
(a) Maillage texturé avec interpolation des normales (106 secs)



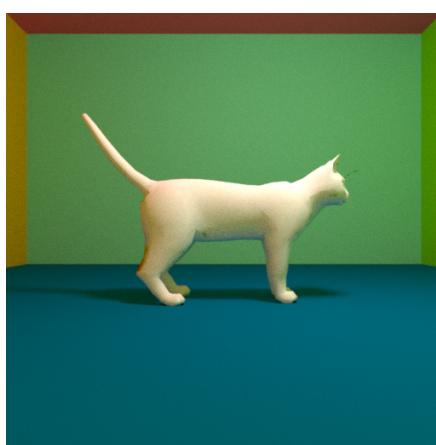
(b) Vue plongeante sur le chat (130 secs)

FIGURE 3

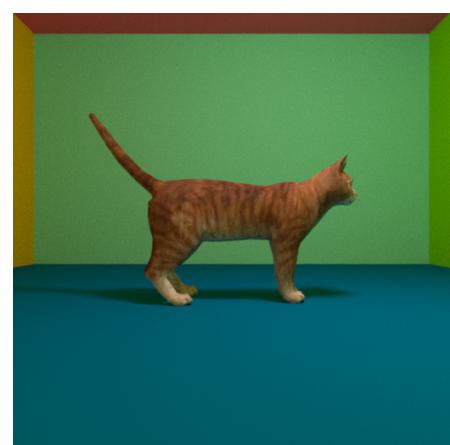
Cependant, nous ne sommes pas arrivés tout de suite au maillage texturé et lisse de l'image (3a), nous sommes passés par les 3 étapes suivantes (la lumière a été déplacée pour mieux voir les différences) :



(a) Maillage brut (94 secs)



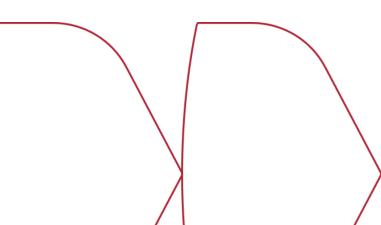
(b) Interpolation des normales (96 secs)



(c) Ajout de la texture (97 secs)

FIGURE 4

Tout d'abord, on a implémenté le rendu d'un maillage brut par calcul de l'intersection entre les rayons lancés et les triangles qui composent le modèle 3D (image (4a)).

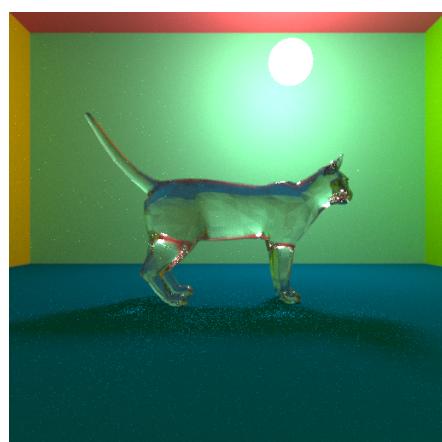




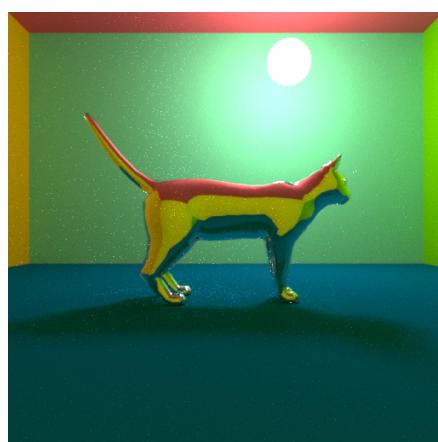
Ensuite, on implémenté l'interpolation de Phlong (image (4b)). En effet, le fichier OBJ qui présente les données du modèle 3D contient non seulement les positions des sommets des triangles, mais également des valeurs de normal en ces sommets. En considérant la normale de chaque triangle comme l'interpolation de la normale en ses 3 sommets, on peut "lisser" le rendu du modèle.

Enfin, on ajoute une texture au modèle en utilisant les coordonnées des sommets dans l'espace UV (qui est donnée par le fichier OBJ) comme sur l'image (4c). En utilisant les coordonnées barycentrique d'un point dans un triangle et les coordonnées UV des sommets, on peut mapper ce point à une coordonnée UV, et lire dans l'image de texture la valeur de couleur correspondante.

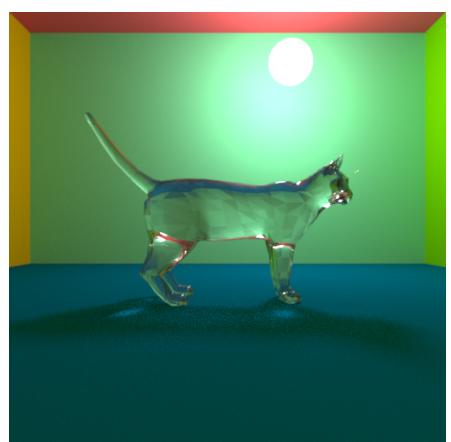
Sur les images suivantes, on montre qu'on peut rendre le modèle 3D transparent (images (5a) et (5c)) ou comme un miroir (image (5b)) de la même manière que les sphères :



(a) Maillage transparent (133 secs)



(b) Maillage miroir (94 secs)

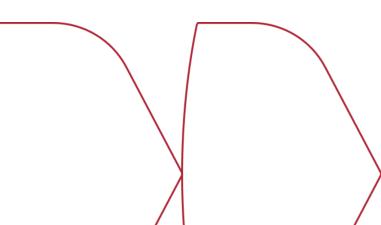


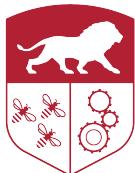
(c) Maillage transparent (1024 rayons, 8 rebonds, 1149 secs)

FIGURE 5

On peut également générer des textures de manière procédurale comme dans ce cas simple où la couleur des sphères dépend du modulo des coordonnées du point avec une valeur arbitraire, ce qui permet de réaliser un damier (images (6a) et (6b)). La valeur arbitraire (que l'on va appeler "fréquence") détermine la taille des carreaux du damier. Ainsi, si le modulo d'une coordonnée est sous la moitié de la fréquence, on renvoie du noir et sinon on renvoie la couleur originale de la sphère. La difficulté réside dans le fait qu'on a 3 coordonnées et qu'il faut combiner le résultat des tests pour les 3 coordonnées en utilisant des "ou exclusifs". La formule précise que je n'arrivais pas à trouver m'a été donnée par Gulliver Larsonneur.

On remarque dans (6a) qu'il y a une symétrie du damier autour des axes, ce qui permet de les visualiser et de voir où est l'origine du repère, mais ce qui est aussi peu esthétique. En ajoutant un grand offset aux coordonnées avant le calcul du modulo, on peut déplacer ce phénomène très loin de sorte à avoir un résultat plus jolie, comme en (6b).





(a) Damier (116 secs)



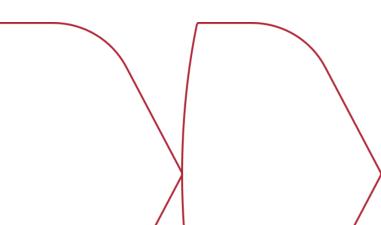
(b) Damier sans la symétrie autour des axes (108 secs)

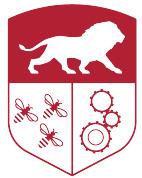
FIGURE 6

Enfin, on s'intéresse à l'accélération du rendu de maillage permise par l'utilisation de "Bounding Box" et de "Bounding Volume Hierarchy" (BVH). Pour cela, on a rendu 3 fois la même image (7) avec les mêmes paramètres (16 rayons, 4 rebonds) mais avec des méthodes différentes. Les résultats sont présentés dans le tableau suivant :

Méthode	Temps de rendu
Parcours itératif de tous les triangles	583 secondes
Parcours itératif de tous les triangles si intersection avec la Bounding Box	109 secondes
Bounding Volume Hierarchy	8 secondes

Ainsi, l'utilisation d'une Bounding Box améliore la vitesse de rendu par 5 et l'utilisation d'une BVH par presque 75.





CENTRALE
LYON

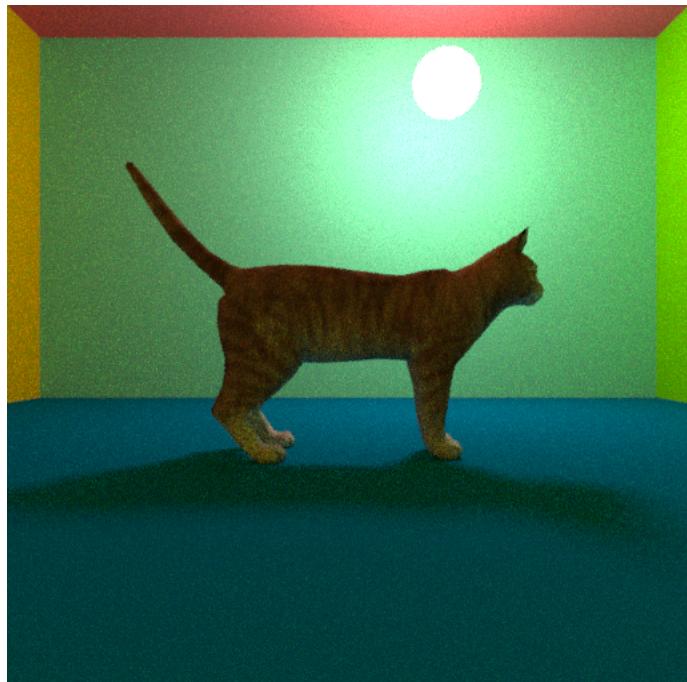


FIGURE 7 – Maillage texturé (16 rayons, 4 rebonds)