ÉCOLE
CENTRALE LYON

Rapport de projet d'informatique graphique :

Raytracing

Professeur
Nicolas Bonneel

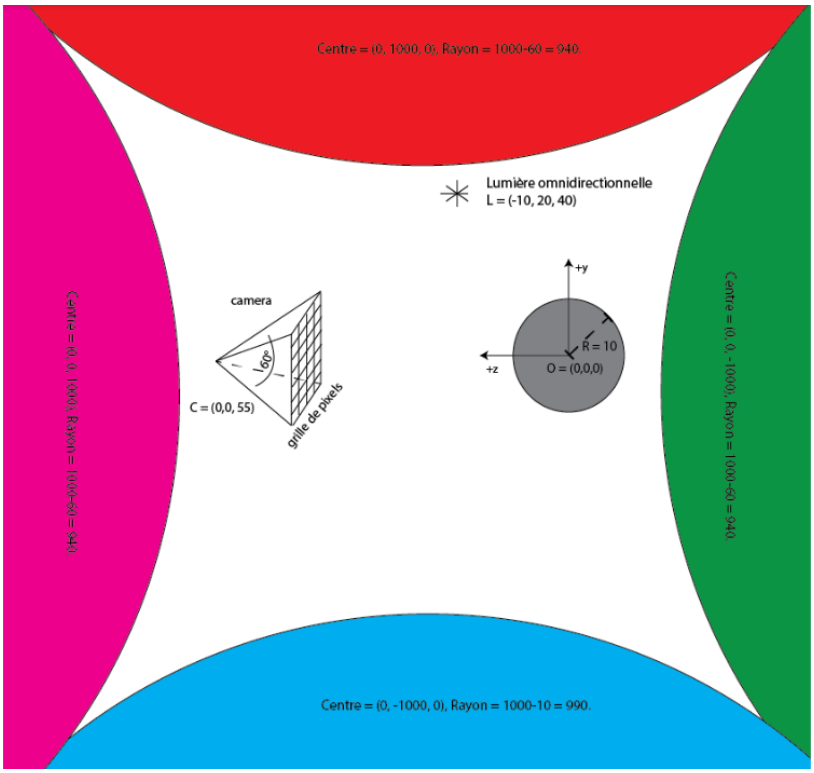
Étudiant
Fazil Mouhamad

16/03/2020

Table des matières

Introduction

Ce projet d'informatique graphique s'intéresse au Raytracing, une technique de calcul d'optique utilisée pour le rendu d'images. Elle consiste à simuler le parcours inverse de la lumière : on calcule les éclairages de la caméra vers les objets puis vers les sources de lumières contrairement à la réalité où la lumière. L'objectif de ce projet est la création d'une scène où figure une caméra depuis laquelle on envoie des rayons vers différents éléments telles que des sphères, plans, triangles, etc, puis vers les sources de lumières. Nous avons implémenté un raytracer basée sur une scène simple (en s'inspirant de l'image ci-dessous) à laquelle on a ajouté au fur et à mesure des éléments supplémentaires.



Ce rapport présente les résultats obtenus après l'implémentation de ces différents éléments. Le code du projet se trouve à l'adresse suivante :
https://github.com/fmouha/Informatique-Graphique

1 Création de la scène élémentaire

Pour commencer, nous avons établit une scène noire avec simplement une sphère blanche telle que les points d'intersection entre les rayons qui sont envoyées depuis la caméra et la sphère apparaissent en blanc.

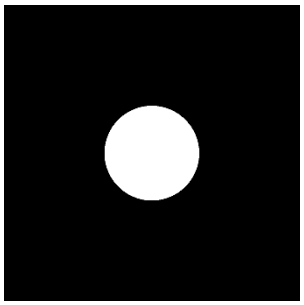


Figure 1 – Sphère simple blanche

Ce rendu manque de relief, c'est pourquoi nous ajoutons une source de lumière et en une couleur à la sphère en lui ajoutant une propriété d'albedo :

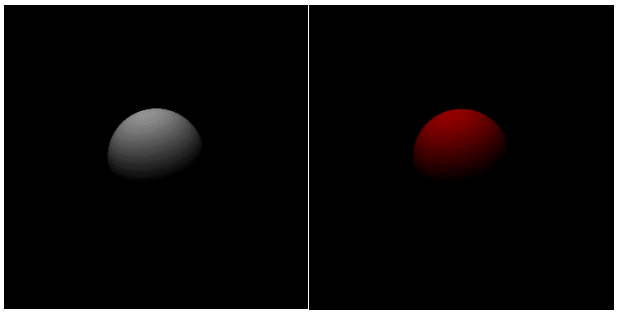


Figure 2 – Sphères blanche et rouge avec source de lumière

On peut finalement ajouter d'autres sphères de manière à créer un décor. On prend en effet des sphères de rayons très grands, ce qui les fera apparaître sur l'image comme étant des murs, sol et plafond plans :

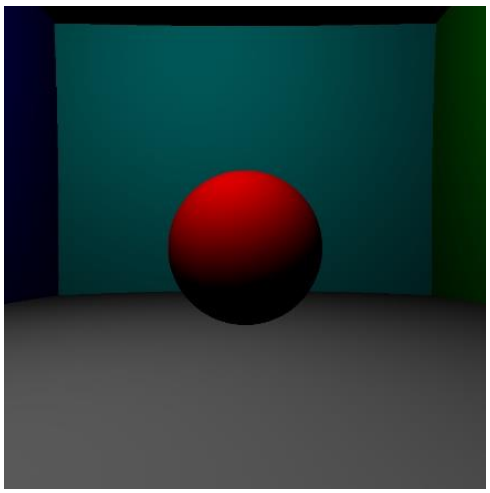


Figure 3 – Scène (multiples sphères)

2 Les surfaces

On dipose maintenant de notre sphère dans une scène, mais pour avoir un rendu pour réaliste, on implémente les ombres portées en générant des rayons secondaires partant des points d'intersection des objets vers la source de lumière et en regardant s'il y a de nouvelles intersections. Par un simple calcul de distance, on détermine si ces points sont ombrés ou non, on obtient ainsi (la source de lumière a été placée en haut à droite) :

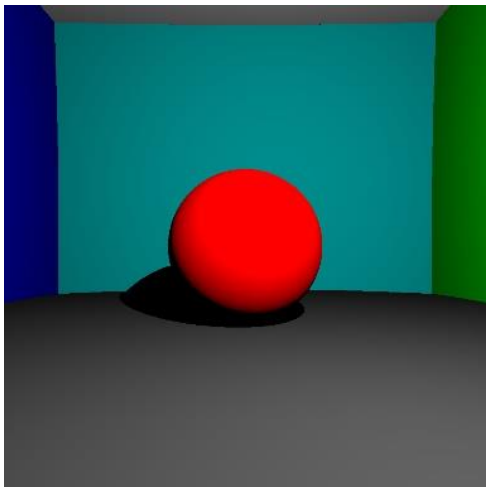


Figure 4 – Sphère avec ombre

On peut maintenant gérer différentes surfaces : les surfaces spéculaires (miroirs) et les surfaces transparentes qui reposent sur des principes de calcul similaires mais avec de la récursion.
On applique également une correction gamma sur l'intensité des pixels avec une puissance 0.45 pour avoir plus de précision sur les pixels sombres. Le rendu final est le suivant avec à gauche une sphère transparente et à droite une sphère miroir :

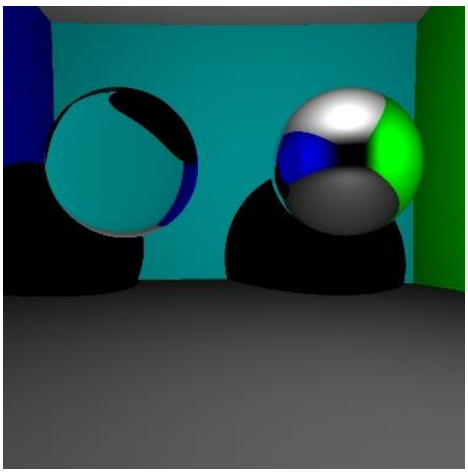


Figure 5 – Sphère transparente (Nverre = 1.3) et sphère miroir

3 L’éclairage indirect

On va maintenant simuler des effets d’éclairage indirect, c’est à dire lorsque les objets sur lesquels la source principale de lumière est réfléchie agissent comme des sources secondaires. On utilise pour cela l’équation du rendu qui implique le calcul d’une fonction de réflectance (BRDF) et un échantillonage par importance. L’image qu’on obtient est initialement très bruitée, mais en augmentant le nombre de rayons par pixel (ici 8), on peut arriver à un rendu convenable :

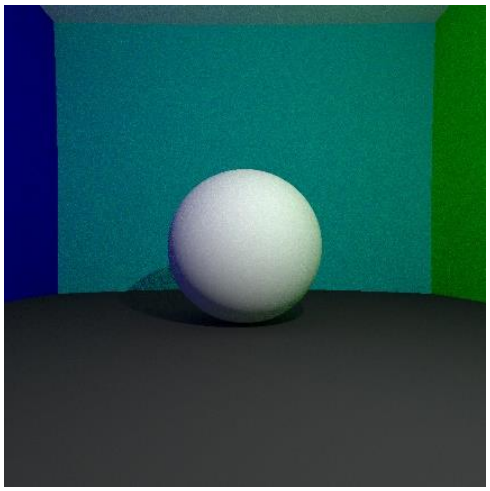


Figure 6 – Rendu avec éclairage indirect

On remarque si l’on zoome sur cette image, un effet de crénelage des pixels sur les bords de la sphère. Afin d’y remédier, on a recourt à l’anti-aliasing :

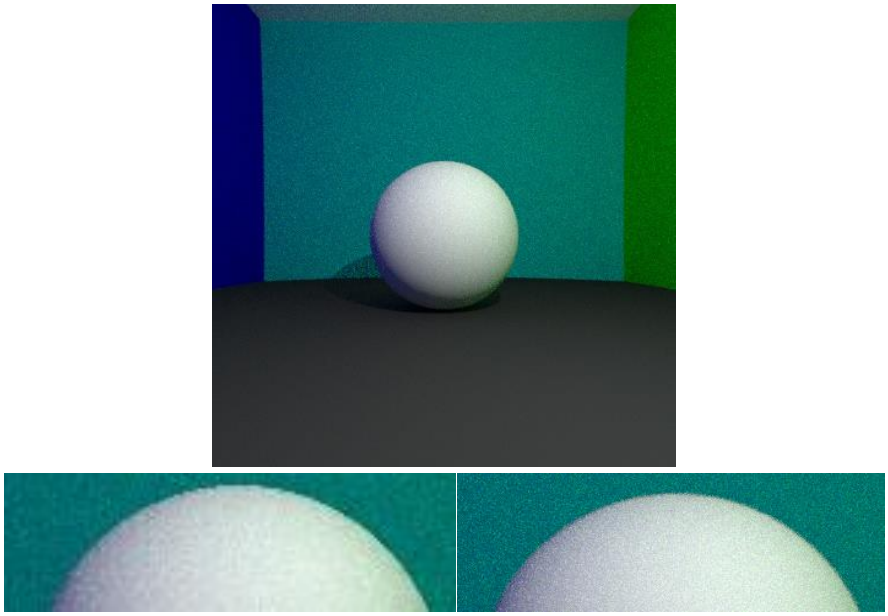


Figure 7 – Rendu avec anti-aliasing (haut)
Comparaison avant (gauche) et après (droite) anti-aliasing

On crée maintenant une source de lumière étendue au lieu d’une lumière ponctuelle, c’est-à-dire une source qui a une certaine surface : en l’occurence une surface sphérique. Pour ce

faire, on considère cette source comme une surface émissive et lorsqu'un rayon arrive sur cette lumière, on lui assigne son émissivité avec une BRDF nulle. Cela permet d'avoir des ombres douces (la sphère a été grossie afin d'observer plus clairement les ombres) :

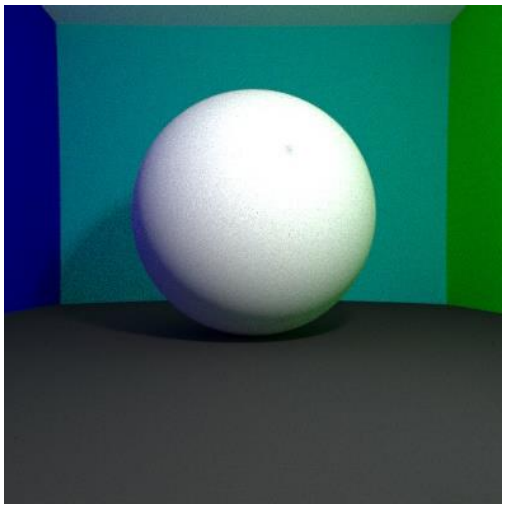


Figure 8 – Visualisation des ombres douces

Enfin, nous implémentons des effets de profondeur de champ comme on peut le voir dans l'image suivante.

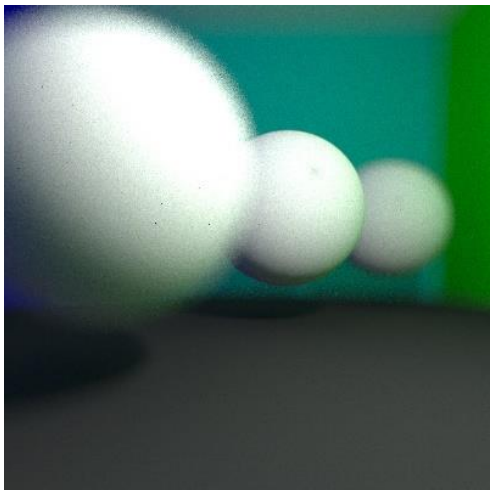


Figure 9 – Plusieurs sphères avec effets de perspective

On implémente également la réflexion de Fresnel sur les objets transparents :

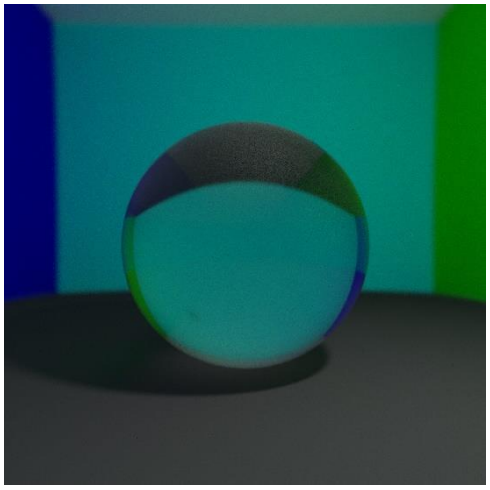


Figure 10 – Sphère transparente avec réflexion de Fresnel (Nverre = 1.45)

4 Les maillages

Jusqu'à présent, notre scène était uniquement constituée de sphères. Afin d'implémenter d'autres géométries, on a créée une classe abstraite *Object* dont la classe Sphere et d'autres vont hériter. On peut par exemple créer des triangles en appliquant le principe d'intersection entre rayons et plan :

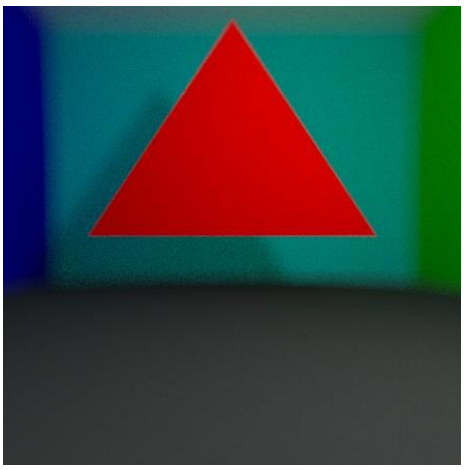


Figure 11 – Triangle

On peut maintenant implémenter des maillages, des structures géométriuqes plus complexes. Pour cela, un maillage est divisé en petits triangles et pour accélérer les calculs, on représente une boîte englobante autour du maillage : si les rayons ne croisent pas cette boîte, il n'y a pas d'intersection. On divise la boîte englobante principale du maillage en sous-boîtes de façon récursive et on calcule ainsi un arbre (BVH) qui permet d'avoir des calculs d'intersection plus précis, de telle sorte à accélérer le rendu :

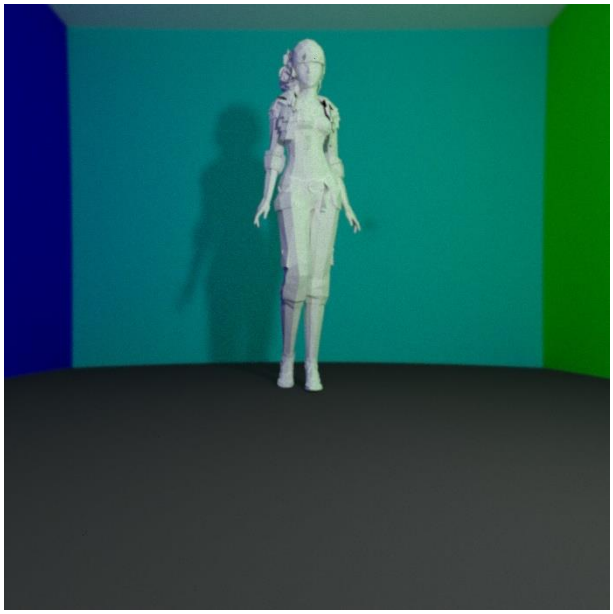


Figure 12 – Maillage

On peut aussi améliorer le rendu et avoir des surfaces plus lisses grâce aux normales recalculées :

Figure 13 – Maillage avec normales recalculées

Enfin, on peut rajouter les textures sur le maillage :

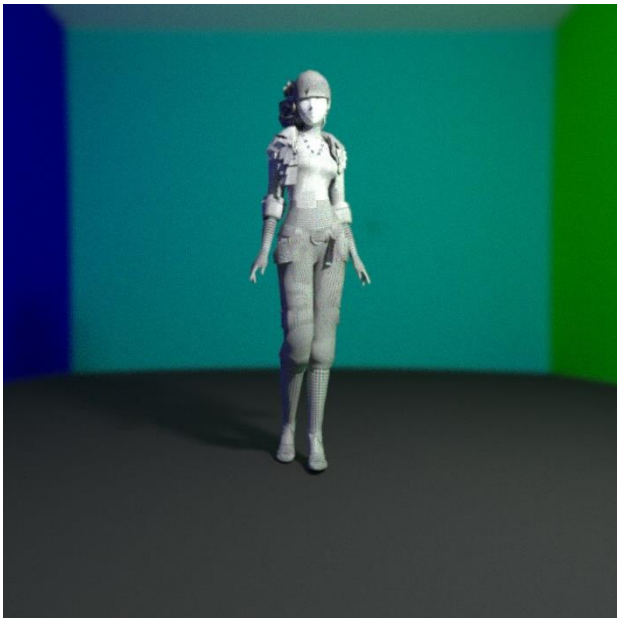


Figure 14 – Maillage avec textures

5 Autres rendus intéressants

Pour finir, en exploitant les différents fonctions implémentées, on obtient :

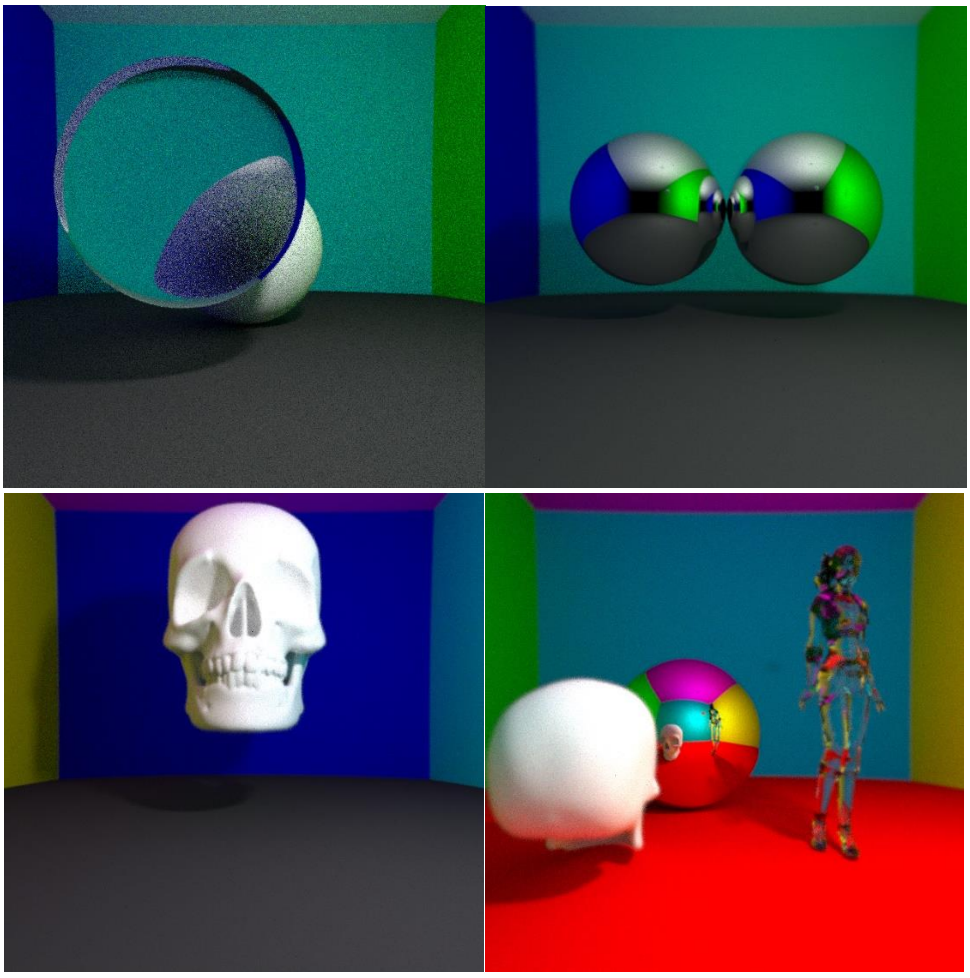


Figure 15 – Divers rendus

Avis sur le cours

J’ai trouvé ce cours sur le Raytracing très intéressant. N’ayant aucune connaissance antérieure sur le sujet mais étant simplement intéressé par tout ce qui est animations, jeux-vidéos, etc, le fait de pouvoir obtenir des rendus assez réalistes avec du code relativement simple m’a beaucoup intéressé. J’ai trouvé que les explications en cours sur chaque implémentation étaient claires et que la progression du cours en difficulté était également bien adaptée. Enfin, le fait également d’avoir des vidéos sur internet m’a été très utile, surtout quand on n’a pas tout compris pendant le cours, elles sont également bien expliquées.