



ÉCOLE  
**CENTRALE**LYON

# ÉCOLE CENTRALE LYON

MOS 2.2  
RAPPORT

## Informatique Graphique

*Elève :*  
Julien BRONNER

*Enseignant :*  
Nicolas BONNEEL

## Table des matières

<b>Introduction</b>	<b>2</b>
<b>1 Création des sphères</b>	<b>2</b>
1.1 Sphère seule . . . . .	2
1.2 Plusieurs sphères . . . . .	2
1.3 Murs . . . . .	3
<b>2 Correction Gamma</b>	<b>3</b>
<b>3 Ombres portées</b>	<b>4</b>
<b>4 Surfaces non diffuses</b>	<b>4</b>
4.1 Réflexion . . . . .	5
4.2 Sphères transparentes et creuses . . . . .	5
<b>5 Lumière indirecte</b>	<b>6</b>
<b>6 Parallélisme</b>	<b>6</b>
<b>7 Anti-Aliasing</b>	<b>6</b>
<b>8 Surface émissive - Lumière avec une taille</b>	<b>7</b>
<b>9 Flou - Modéliser la caméra</b>	<b>7</b>
<b>10 Ajout modèle 3D</b>	<b>8</b>
10.1 Import simple . . . . .	8
10.2 Bounding Box . . . . .	9
10.3 Bounding Box Récursive . . . . .	9
10.4 Prise en compte des normales du modèle . . . . .	9
10.5 Texture . . . . .	9
<b>11 Mouvement de caméra</b>	<b>11</b>
<b>Conclusion et retours</b>	<b>12</b>
<b>Acronymes</b>	<b>12</b>

## Introduction

Durant ce module de cours à l'Ecole Centrale de Lyon (ECL), nous avons construit pas à pas un Raytracer, c'est-à-dire un programme permettant de créer les images que l'on souhaite. Un Raytracer permet d'avoir un comportement de la lumière réaliste, notamment lors de réflexion ou de réfraction. Pour simplifier, on trace tous les rayons de lumière entre la (ou les) source de lumière et le capteur. Il faut donc prendre en compte tous les trajets possibles pour avoir un rendu le plus réaliste possible.

En pratique, les Raytracers sont utilisés dans l'industrie du cinéma, où un long temps de calcul n'est pas un problème, comparé aux jeux-vidéos par exemple, où un rendu en temps réel est nécessaire.

Nous développerons dans ce rapport, chaque étape réalisée, dans un ordre chronologique, en commençant par la création de sphères simples, jusqu'à l'implémentation de modèle 3D en passant par des surfaces réfléchissantes.

## 1 Crédit des sphères

Nous avons tout d'abord créé des sphères, puisque c'est un objet simple à paramétriser.

Nous avons commencé par tracer un simple disque blanc sur fond noir, mais je n'en ai pas gardé de trace.

Dans les scènes présentées ici, et sauf mention du contraire, la lumière est placée en haut à droite, et jusqu'à l'implémentation des surfaces émissives, la lumière est ponctuelle.

### 1.1 Sphère seule

On a donc affiché une sphère seule, rouge sur fond noir, que l'on discerne bien sphérique puisque la lumière se reflète différemment suivant l'angle d'incidence sur la sphère.

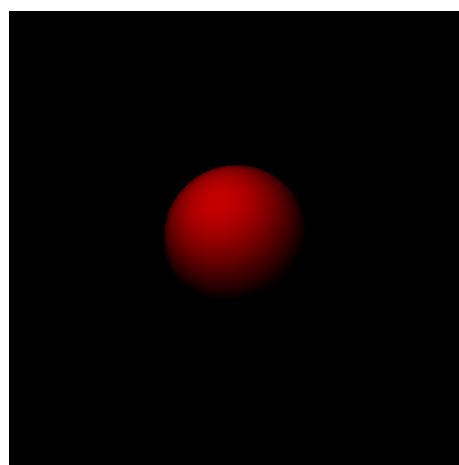


FIGURE 1 – Sphère rouge sur fond noir

### 1.2 Plusieurs sphères

Nous avons ensuite intégré un classe *Sphère* au code, permettant d'intégrer plusieurs sphères dans la scène.

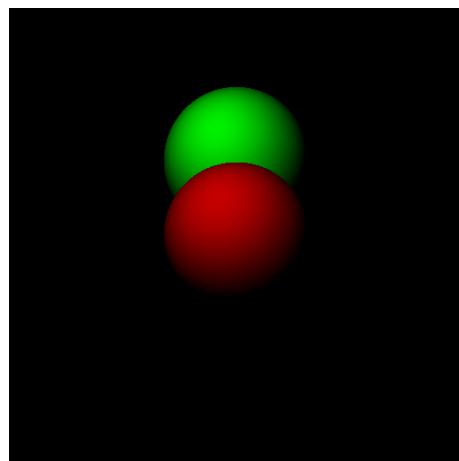
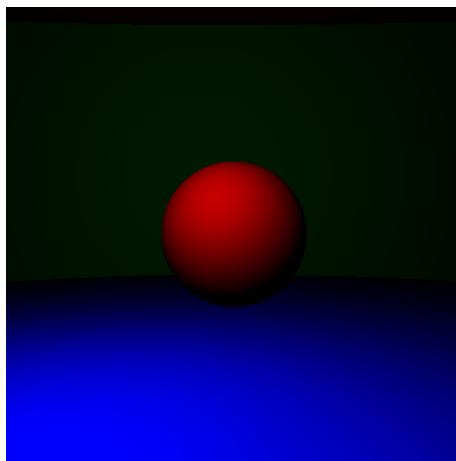


FIGURE 2 – Deux sphères sur fond noir

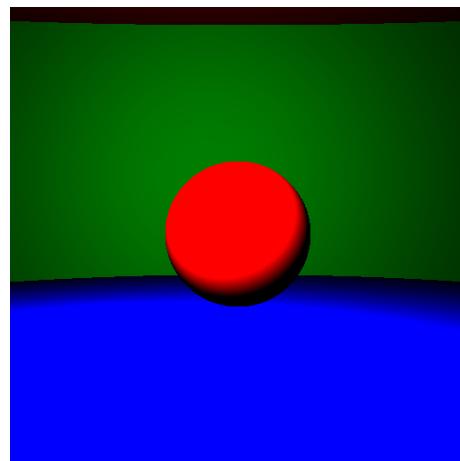
### 1.3 Murs

Ensuite, pour avoir des murs de manière simple, nous avons placé des sphères loin et de grand diamètre, ce qui fait qu'à l'échelle de la sphère centrale et de la caméra, les murs sont quasiment plats. Dans la figure 3a, la luminosité habituelle n'était pas suffisante pour bien voir tous les murs, elle a donc été augmentée pour faire la figure 3b.

Jusqu'à maintenant, le temps de calcul est très rapide, et reste inférieur à **une seconde**.



(a) Image avec la même luminosité que précédemment



(b) Augmentation de la luminosité pour mieux distinguer les murs

FIGURE 3 – Ajout des murs, du toit et du sol

## 2 Correction Gamma

Par ailleurs, lors de l'affichage sur un écran d'ordinateur, il y a un facteur Gamma qui est appliqué, il faut donc le corriger. On applique pour cela une puissance Gamma (qui vaut 0.45 dans notre cas) pour ainsi augmenter la luminosité comme on le voit sur la figure 4.

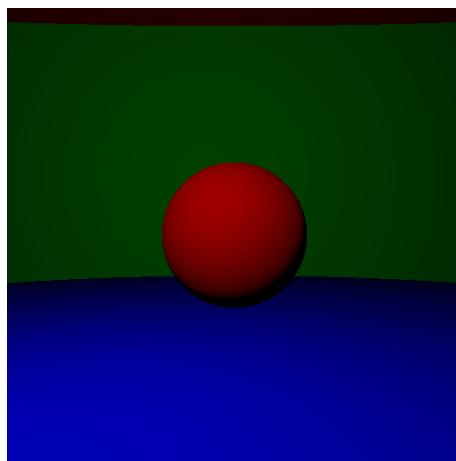
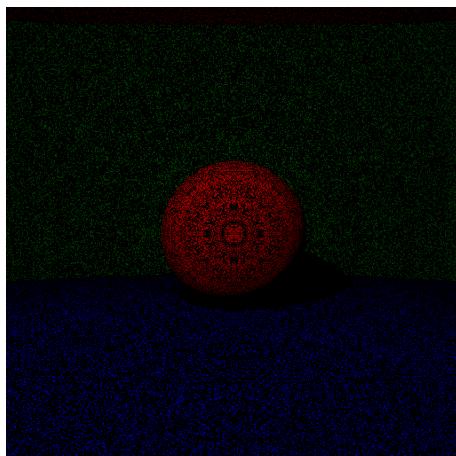


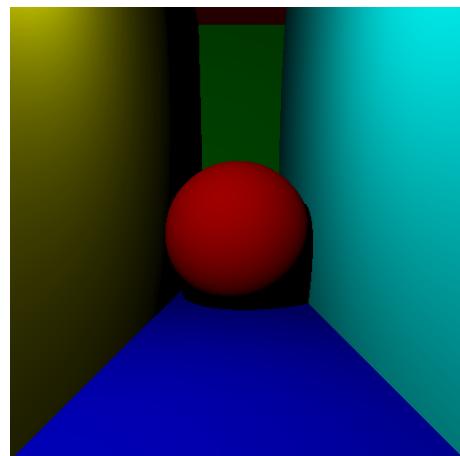
FIGURE 4 – Ajout de la correction Gamma

### 3 Ombres portées

Pour rendre les images plus réalistes, il faut ajouter une ombre projetée par les objets, pour cela on regarde s'il y a un objet sur la trajectoire entre le point d'où l'on part et la lumière. On voit que si l'on fait ça simplement, on obtient l'image 5a, où il y a un problème car les pixels ne sont pas forcément exactement à la surface, et donc parfois on vise un point légèrement sous la surface qui est donc noir puisqu'il est masqué par la sphère. On rajoute donc un léger facteur Epsilon en direction de l'extérieur, qui permet d'obtenir l'image 5b, où l'on voit bien l'ombre portée sur les murs.



(a) Ajout de l'ombre naïve



(b) Ajout d'un facteur Epsilon

FIGURE 5 – Ajout de l'ombre

### 4 Surfaces non diffuses

Nous avons ensuite codé des surfaces non diffuses, il y a donc des sphères réfléchissantes ou transparentes.

## 4.1 Réflexion

Il y a tout d'abord eu des surfaces réfléchissantes, que l'on calcule en prenant la réflexion par rapport à la normale de la surface. On peut donc voir les murs de couleurs se réfléchir sur la sphère, être déformés comme sur l'image 6b et même voir de multiples réflexions sur l'image 6c. Pour ne pas avoir des réflexions à l'infini, nous avons décidé arbitrairement qu'au bout de cinq "rebonds" la lumière renvoyée est noire.

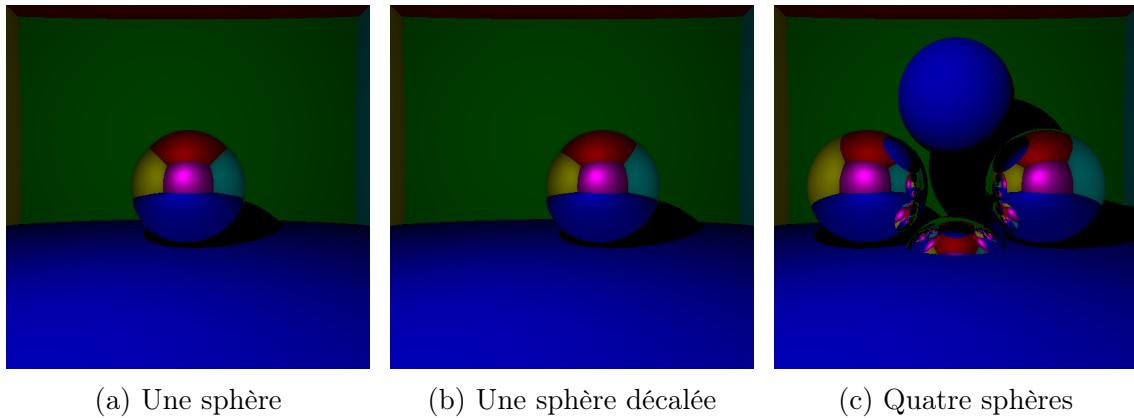


FIGURE 6 – Sphères réfléchissantes

## 4.2 Sphères transparentes et creuses

Nous avons ensuite fait des sphères transparentes, qui diffractent la lumière qui les traversent. On voit donc l'ombre inversée sur la sphère du milieu de l'image 7, puisqu'elle est réfléchie dans la sphère transparente. On donc donc aussi facilement faire une sphère creuse, en mettant dans une première sphère transparente une autre sphère plus petite et dont les normales sont inversées, ce qui donne la sphère de droite.

Pour générer cette image, il a fallu environ 8 secondes, ce qui commence à être long pour générer une image de taille 512 par 512.

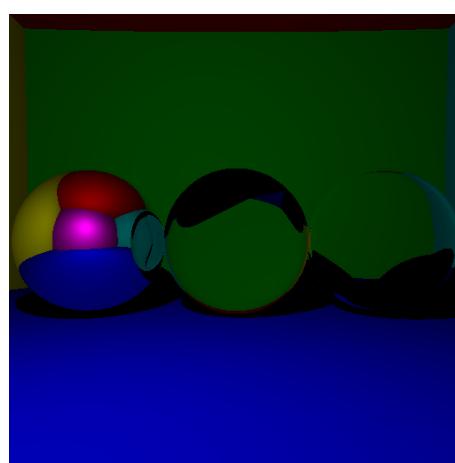


FIGURE 7 – Dans l'ordre de gauche à droite : sphères réfléchissante, transparente et creuse

## 5 Lumière indirecte

L'ombre précédente étant faite uniquement de noir pur, elle n'est pas très réaliste. Pour la rendre plus réaliste, nous pouvons prendre en compte la lumière indirecte, c'est-à-dire des rayons s'étant réfléchis sur d'autres surfaces. On va donc tirer plusieurs rayons pour un seul pixel, et leur faire faire des rebonds. Comme dans le cas des réflections et transmissions de lumière, on limite le nombre de rebonds à cinq.

Dans l'image 8, il y a 100 rayons de "tirés". Cela a pris 193 secondes, donc un peu plus de trois minutes, ce qui est beaucoup plus long que précédemment (environ 25 fois plus long).

On remarque toutefois l'efficacité de cette méthode, puisque l'ombre est beaucoup plus douce. Elle est plus transparente, comme c'est le cas dans la réalité.

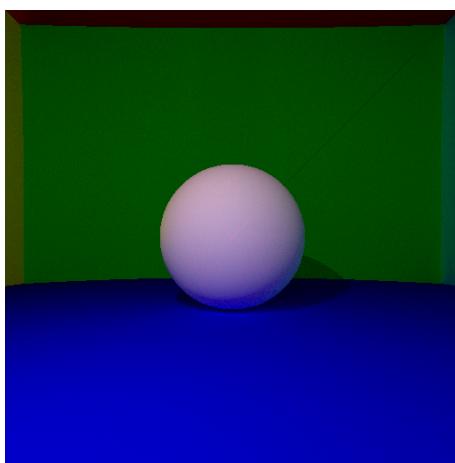


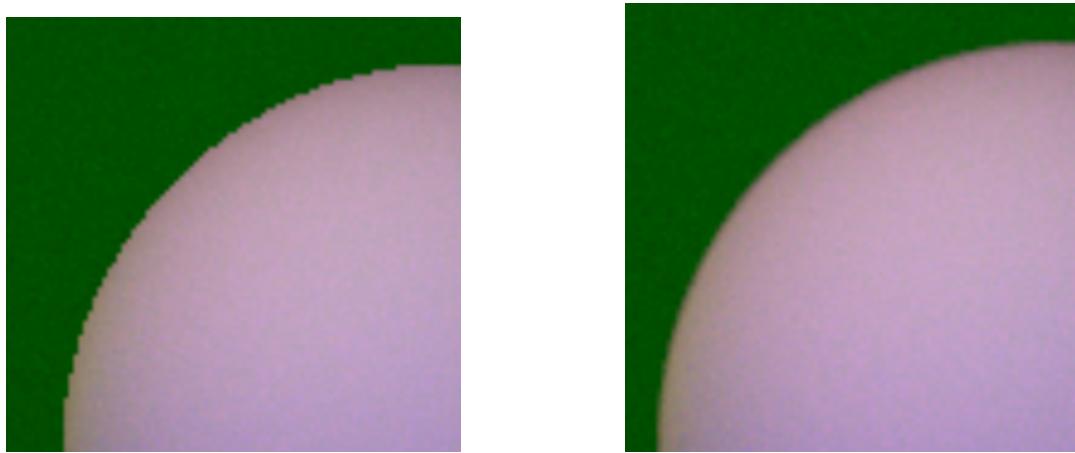
FIGURE 8 – Lumière indirecte

## 6 Parallélisme

Puisque l'ajout de multiple rayons démultiplie le temps de calcul considérablement, il est intéressant de paralléliser les calculs. Pour cela il suffit de rajouter une simple ligne de code à un endroit. On obtient ainsi la même image, mais en seulement 53 secondes, ce qui a réduit le temps par un facteur entre trois et quatre.

## 7 Anti-Aliasing

Pour avoir des sphères dont la courbe est moins "en escalier" comme on peut le voir sur l'image 9a, il est possible d'appliquer de l'anti-aliasing. Pour cela, on tire plusieurs rayons mais pas exactement toujours au même point, mais dans une zone proche autour. Cela permet de lisser le rendu, comme on peut le voir sur l'image 9b. Cela n'augmente pas le temps de rendu puisque l'on utilisait déjà plusieurs rayons avec la lumière indirecte. Il y a également 100 rayons de tirés ici.



(a) Zoom sur l'image sans anti-aliasing

(b) Zoom sur l'image avec anti-aliasing

FIGURE 9 – Comparaison d'images avec et sans l'anti-aliasing

## 8 Surface émissive - Lumière avec une taille

Jusqu'à maintenant, la lumière est générée par une source ponctuelle, dans la vraie vie, les lumières ont une dimension. Il peut donc être intéressant de créer une sphère émettrice de lumière. Pour cela, on applique ce que l'on a fait avec la lumière ponctuelle à toute la surface de la sphère. Cela donne l'image 10, qui a été calculée en 104 secondes. L'ajout de cette lumière à donc quasiment doublé le temps de calcul.

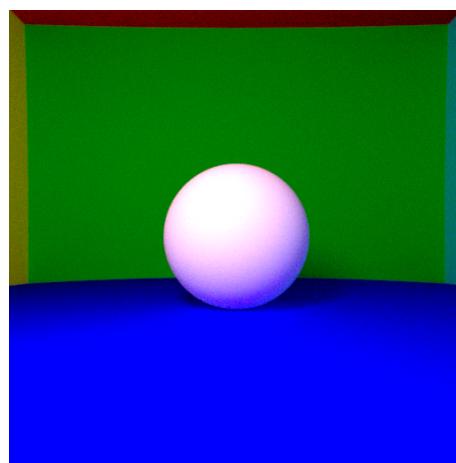


FIGURE 10 – Sphère émissive

## 9 Flou - Modéliser la caméra

Jusqu'à maintenant, nous considérons une caméra (un capteur) avec une profondeur de champs infinie, c'est-à-dire que l'image est nette partout. Dans la réalité, ce n'est pas le cas. De plus, il peut y avoir un aspect artistique à avoir un fond flou, pour détacher le sujet.

Pour créer cette profondeur de champs, on considère que le capteur à une certaine taille, et que donc chaque rayon tiré peut arriver sur une zone différente de ce capteur.

On voit donc dans la figure 11, un flou déjà présent mais faible à gauche, et un flou plus important à droite. Les boules représentées sont toutes de la même taille, et placées à une profondeur différente. Ceci ne rallonge pas le temps de calcul pour un même nombre de rayons (comme sur les images 11) et la même taille d'image.

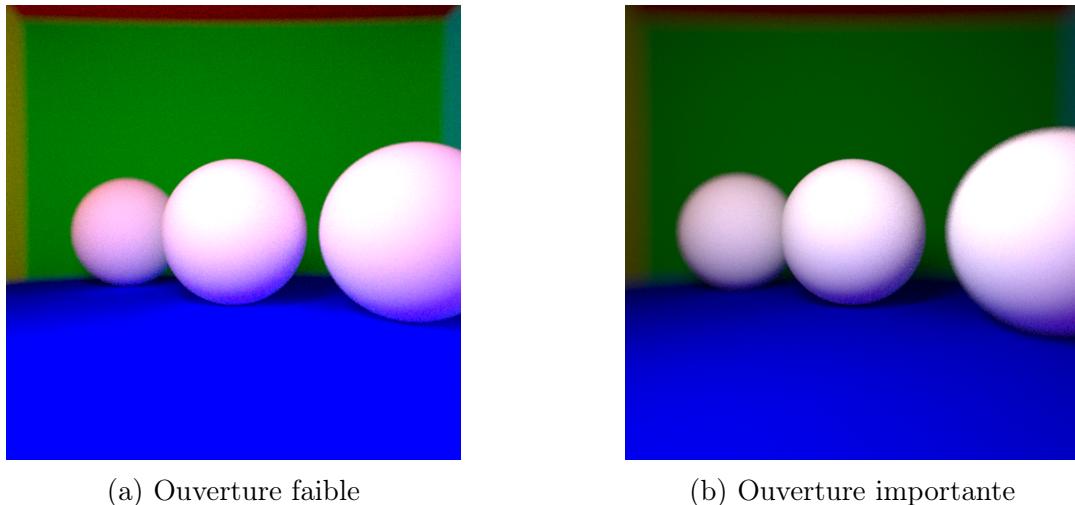


FIGURE 11 – Prise en compte de l'ouverture de l'obturateur

## 10 Ajout modèle 3D

### 10.1 Import simple

Pour l'instant, nous ne travaillons qu'avec des objets simples, des sphères, mais nous pouvons vouloir avoir des formes géométriques complexes. Nous avons donc ajouté la possibilité d'intégrer un objet 3D. Ceux-ci sont modélisés par des triangles (parfois des quadrilatères) de petites tailles, permettant de reproduire la forme désirée. On peut voir sur l'image 12 un chien, de dos, et tout noir. La couleur est dû à un problème de normale qui n'était pas encore résolu. La qualité est moins bonne que d'habitude (ici 128 par 128), et il n'y a qu'un rayon de tiré, mais le temps de calcul est quand même de 170 secondes.

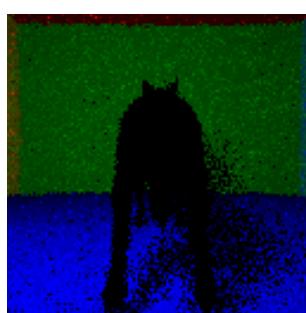


FIGURE 12 – Modèle 3D du chien

## 10.2 Bounding Box

Pour améliorer le temps de calcul, on peut utiliser une boîte englobante de l'objet 3D. Ainsi, si le rayon ne vise pas cette boîte, on sait qu'il ne visera pas non plus l'objet 3D, cela permet d'éviter des temps de calcul en testant moins d'intersection avec les triangles du modèle, puisque sinon on teste l'intersection avec chacun des triangles. On passe donc à un temps de calcul d'un peu moins de 20 secondes, on a donc divisé par 8 environ le calcul pour le même nombre de rayon et la même taille d'image. Le chien est également blanc après avoir résolu le problème, et regarde maintenant la caméra.

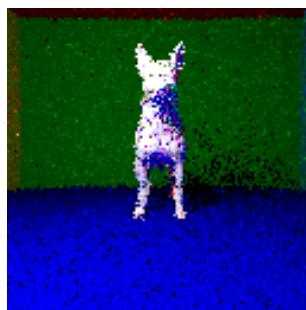


FIGURE 13 – Modèle 3D du chien, calculé avec une Bounding Box (BB)

## 10.3 Bounding Box Récursive

Comme la BB précédente, il est possible d'appliquer ce principe de boîte englobante à des groupes de triangles composant le modèle. On va donc tester les triangles faisant parti de ces BB à chaque étape, ce qui permet d'en éliminer de plus en plus. On s'arrête lorsqu'une BB ne contient plus que quelques triangles (arbitrairement 5 par exemple). Ainsi on passe à un temps de calcul de 676ms pour la même image que précédemment, on divise donc le temps de calcul par environ 29.

Pour une image de meilleure résolution (512x512) et 100 rayons, comme l'image 14b, on a un temps de calcul de 23 secondes environ, ce qui est semblable à celui de la partie précédente pour une image plus petite et plus bruitée. Il y a donc une accélération extrêmement importante.

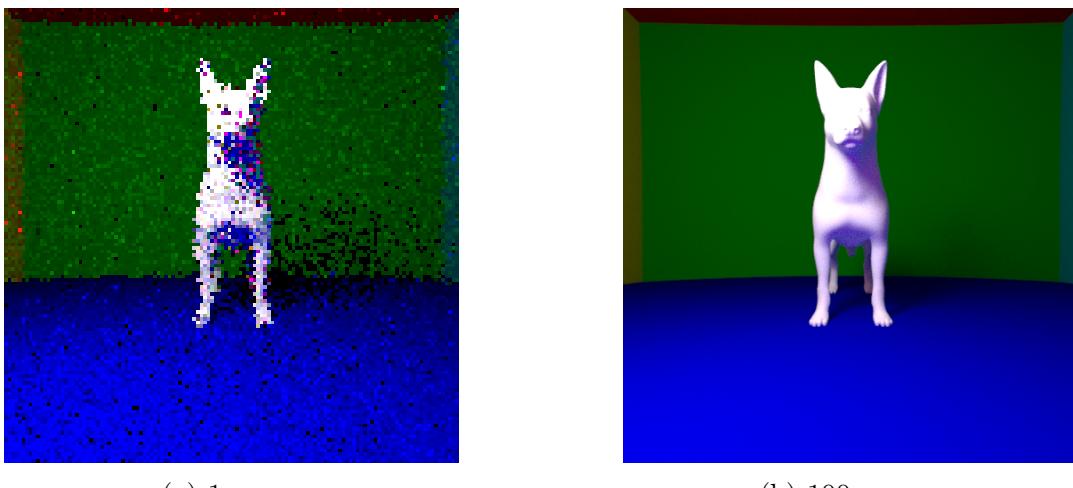
## 10.4 Prise en compte des normales du modèle

Lors de la création d'un modèle 3D, les artistes peuvent également donner une normale à leur surface, qui ne sera donc pas forcément celle géométrique, que l'on utilisait jusqu'à maintenant. Cela permet de donner des variations dans la manière dont se reflète la lumière sur l'objet, et de créer des ombres particulières. Cela ne change pas le temps de calcul.

De plus, pour varier un peu, j'ai fait certains essais avec un modèle 3D du personnage Link, de la série de jeu *The Legend of Zelda*, comme on le voit sur l'image 15b.

## 10.5 Texture

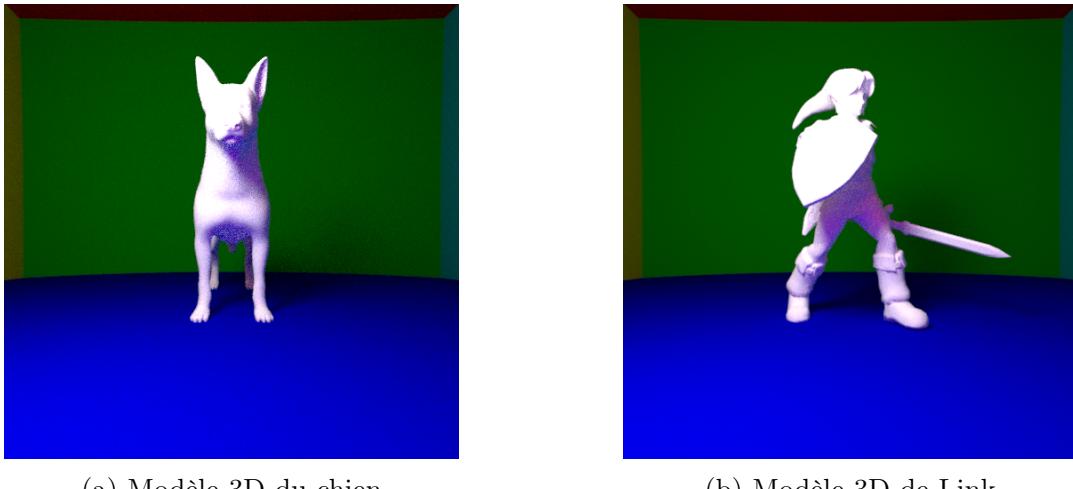
Enfin, lors de la création du modèle, beaucoup d'artistes ajoutent une texture, pour que le modèle soit plus réaliste, il s'agit donc de prendre en compte celle-ci. Dans l'image



(a) 1 rayon

(b) 100 rayons

FIGURE 14 – BB Récursive

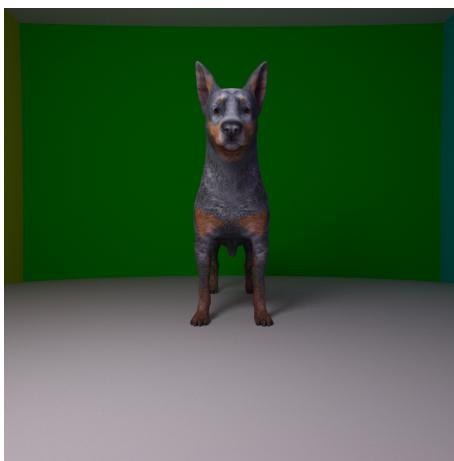


(a) Modèle 3D du chien

(b) Modèle 3D de Link

FIGURE 15 – Utilisation des normales du modèle

16a, de résolution 1024\*1024 et avec 100 rayons, cela a pris 1013 secondes pour le calcul, ce qui est long mais très peu étonnant vu la résolution et le peu de bruitage de l'image. On peut voir que l'image de droite a pris 1375 secondes pour le calcul pour les mêmes propriétés. Cela est dû au fait qu'il y ait plus de triangles dans le modèle, et que les textures sont plus complexes car il y en a plusieurs assemblées.



(a) Modèle 3D du chien



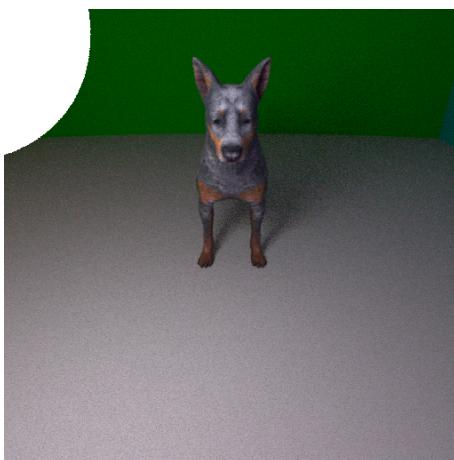
(b) Modèle 3D de Link

FIGURE 16 – Ajout des textures

## 11 Mouvement de caméra

Il est également intéressant de pouvoir bouger la caméra. Nous pouvions déjà changer son origine, mais pas sa direction de regard. C'est ce qui est fait pour les images de la figure 17, où sur celle de gauche on a monté la caméra et baissé son angle de regard, pour toujours voir le chien en entier. En ce qui concerne l'image de droite, elle a également été déplacée sur la droite, et son axe de regard vers la gauche.

Dans les deux cas, le temps de calcul n'a pas changé par rapport à avant, et on voit maintenant la sphère lumineuse dans le champ, c'est le morceau de sphère visible en haut à gauche.



(a) Mouvement de caméra vertical



(b) Mouvement de caméra oblique

FIGURE 17 – Mouvement de caméra

## Conclusion et retours

Ce cours permet une bonne découverte, et même un apprentissage assez poussé du Raytracing, ce qui est intéressant, notamment car nous sommes plus souvent confrontés aux affichages temps réels des jeux-vidéos, plutôt qu'à nous poser la question de comment sont fait les films, et quels sont les techniques utilisées pour une image photo-réaliste. Nous le savons maintenant.

J'ai trouvé que ce cours était intéressant car totalement interactif, ce n'était jamais juste un cours magistral, nous devions en permanence suivre, donc même à distance, je ne m'endormais pas devant mon écran.

Toutefois, pour faire des retours constructifs :

- Comme je ne connaissais pas le C++, ni même aucun langage compilé, j'ai appris sur le tas le langage et la logique du code, ce qui rend parfois compliqué l'avancement du projet en tant que tel, puisque l'on bute souvent sur des problèmes liés au langage et non sur un problème lié au Raytracing. Par exemple, je ne sais toujours pas bien utilisé les pointeurs, alors que c'est un fondamental du C++. Il pourrait donc être intéressant de donner, quelques semaines avant le cours, un tutoriel à faire pour être plus à l'aise dès la première séance. D'autant plus que par rapport à Python (langage souvent le plus utilisé à Centrale), l'installation peut être plus compliquée puisqu'il faut un IDE et un compilateur. J'ai moi-même eu des difficultés au début pour l'utilisation. De même, je ne comprends toujours pas certaines erreurs (ou certains Warning de l'IDE), qui sont pourtant sûrement simple, les expliquer permettrait d'avoir un code plus propre.
- Par ailleurs, j'ai vu qu'un élève avait fait le choix de ne pas coder en C++, car vous le permettiez, mais je pense que c'est plus un frein à son apprentissage, et il pourrait donc être intéressant pour vous d'obliger le C++, car c'était loin d'être le seul à ne pas connaître le langage.
- Nous n'avions pas de vision d'ensemble de ce que nous allions faire, il était donc parfois compliqué de bien structurer notre code et de savoir si ce que nous faisions allait être compatible avec la suite.
- En me renseignant sur le C++, je me suis rendu compte qu'il faut normalement séparer son code en plusieurs fichiers, et importer les classes et les fonctions. Je pense que cela permettrait de se retrouver plus facilement dans les fichiers, vous pourriez donc nous montrer comment le faire, et nous aurions ainsi les bonnes pratiques du langage.

J'ai également une remarque, si l'année prochaine des élèves ont des images qui ne se créent pas sans message d'erreur, c'est potentiellement dû à l'anti-virus (pour ma part Avast) qui ne signale rien mais empêche la création de l'image. Il est possible de mettre une exception sur le dossier concerné pour ne plus avoir de problème.

Toutefois, merci beaucoup pour ce cours, qui a été très intéressant et très enrichissant, et ce, tout au long des séances.

## Acronymes

**BB** Bounding Box. 9, 10

**ECL** Ecole Centrale de Lyon. 2