

**CENTRALE
LYON**

ÉCOLE CENTRALE LYON

MOS 8.5
INFORMATIQUE GRAPHIQUE
RAPPORT

Rapport final - Moteur 3D

Élèves :
Rémi DE MAISTRE

Enseignant :
Nicolas BONNEEL

Table des matières

1	Ray tracing avec des sphères	2
2	Ajout des maillages	2
2.1	Implémentation naïve	2
2.2	Bounding box	3
2.3	Bounding volume hierarchy	4
3	Autres fonctionnalités du moteur	5
4	Rendu final	7

1 Ray tracing avec des sphères

Dans le but de construire un moteur de rendu 3D réaliste, la première étape a été d'implémenter l'intersection d'un rayon avec une sphère. En résolvant l'équation du second degré, on arrive à un rendu d'une pièce, avec des sphères de grand rayon pour faire des murs. On peut aussi changer la BRDF des surfaces pour avoir des sphères réfléchissantes et transparentes, et l'albedo des sphères peut être dépendant de la position du point d'impact pour donner une texture aux sphères. De plus, on met une source de lumière sphérique, et on prend en compte le fait que les objets reçoivent de la lumière aussi des objets environnant, pour un rendu réaliste avec des ombres douces.

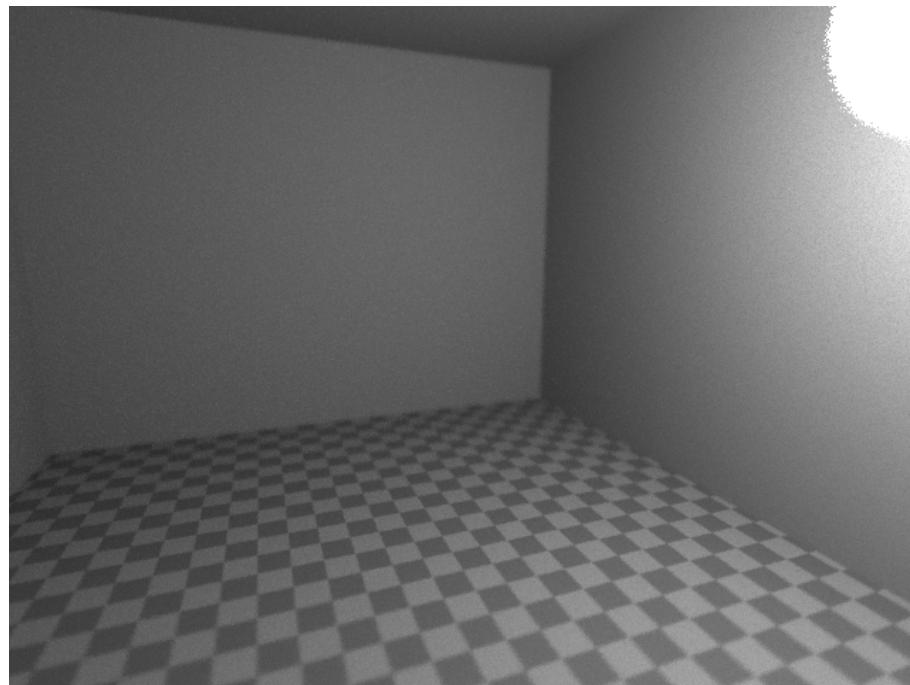


FIGURE 1 – Image d'une pièce avec des sphères.
 $N = 640 \times 480, n_r = 32, n_b = 5, t = 17.1 \text{ s}, t_r = 6.95 \mu\text{s}$

Les paramètres utilisés ici sont une résolution de 640×480 , un nombre de rayons par pixel $n_r = 32$, et un nombre de rebonds maximum $n_b = 5$. Augmenter le nombre de rayons par pixel permet d'avoir plus de précision pour chaque pixel, car la couleur sera moyennée sur plus de rayons, mais le calcul prendra d'autant de fois plus de temps. Finalement, le rendu a pris ici un total de $t = 17.1 \text{ s}$, avec un temps par rayon de $t_r = 6.95 \mu\text{s}$. Le temps par rayon est le temps moyen pris pour un rayon, avec la relation $t = \frac{Nn_r}{N_{threads}} \times t_r$.

2 Ajout des maillages

2.1 Implémentation naïve

Une fois l'intersection avec des sphères mise en place, on peut rajouter l'intersection avec des maillages. Un maillage est défini comme un ensemble de triangles qui ont chacun leurs coordonnées de sommets, leur coordonnées de texture, et leurs normales. En implé-

mentant l'intersection d'un triangle avec un rayon, il est possible de procéder de la même manière qu'avec les sphères. Cependant, on s'aperçoit très vite d'un problème.



FIGURE 2 – Des Pokémons sont apparus, avec des petites balles.

$$N = 640 \times 480, n_r = 1, n_b = 5, t = 9 \text{ min } 53.4 \text{ s}, t_r = 7.71 \text{ ms}$$

Ici, la qualité a été grandement réduite, avec une variance très forte entre les pixels, car le paramètre a été réduit à 1 rayon par pixel. Mais même avec ce changement, l'image a pris presque 10 minutes à se générer, car les rayons prennent en moyenne 1000 fois plus longtemps qu'avant ! Cependant, on peut quand même voir les deux maillages de la scène, un Cobaltium et un Diancie, qui ont bien leur textures (les maillages ont été trouvés sur internet, sur le site free3d.com), avec les normales et les textures interpolées pour un meilleur rendu, ainsi que trois sphères, une diffuse, une en verre ($n = 1.5$), et une réfléchissante dans le coin.

Le problème est que pour chaque rayon, on teste si il y a une intersection avec les 6713 triangles du Cobaltium et avec les 6188 triangles du Diancie, ce qui fait plus de 12000 intersections avec des objets à tester par rayon, au lieu de juste 7 sphères avant ça.

2.2 Bounding box

On peut donc améliorer ça en disant que si les rayons n'intersectent pas avec un pavé englobant chaque maillage, ils n'intersectent aucun des triangles du maillage. Cela permet d'éviter de calculer énormément d'intersections qui ne servent à rien, et on observe un gain de temps conséquent.

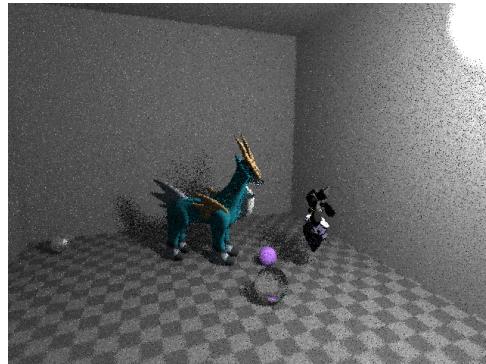


FIGURE 3 – Rien n'a bougé, sauf le temps.
 $N = 640 \times 480, n_r = 1, n_b = 5, t = 41.1\text{ s}, t_r = 534\text{ }\mu\text{s}$

Entre les deux images, les seules différences sont le bruit, qui est quand même assez présent étant donné le nombre très faible de rayons par pixel. Mais le fait qu'à part ça rien n'ait bougé est rassurant, car la seule modification entre les deux images est qu'on a arrêté de calculer des intersections qui n'aboutissaient pas. Néanmoins, rien qu'avec ça, on a déjà un calcul plus de 14 fois plus rapide qu'avec la méthode naïve sur cette scène.

2.3 Bounding volume hierarchy

Il est toutefois possible de faire encore mieux, en répétant le processus de la bounding box. On coupe à chaque fois le volume en deux, et cela permet d'éliminer jusqu'à la moitié des triangles avec simplement deux intersections avec des boîtes, ce qui permet d'arriver à un équilibre entre le nombre d'intersections à calculer avec des boîtes et avec des triangles.

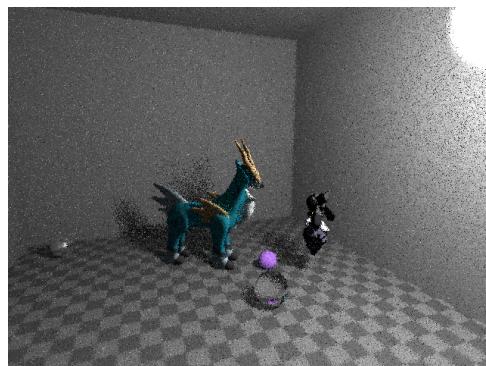


FIGURE 4 – Encore une fois, rien n'a bougé, sauf le temps.
 $N = 640 \times 480, n_r = 1, n_b = 5, t = 1.2\text{ s}, t_r = 15.1\text{ }\mu\text{s}$

On a encore une fois la même image, mais cette fois ci, on est revenus à des temps du même ordre de grandeur que la scène du début sans maillage, avec un temps par rayon seulement 2.5 fois plus long. Cela représente une réduction du temps mis à générer la scène d'un rapport presque 500, par rapport à la méthode naïve. Cela permet donc de faire des images d'une bien meilleure qualité, dans un temps tout à fait raisonnable.



FIGURE 5 – La même image, mais avec moins de bruit.
 $N = 640 \times 480, n_r = 32, n_b = 5, t = 38.4 \text{ s}, t_r = 15.6 \mu\text{s}$

3 Autres fonctionnalités du moteur

Pour avoir un rendu plus réaliste, on a implémenté des fonctionnalités comme par exemple la profondeur de champ. Afin de simuler un rendu par une vraie caméra, on a supposé l'existence d'un plan focal, par lequel tous les rayons passent, et la caméra est ensuite déplacée légèrement pour avoir un flou sur les objets loin du plan focal. Si on désactive cette fonctionnalité, on obtient la figure ci-dessous, on peut notamment voir que le mur du fond et le carrelage deviennent complètement nets.

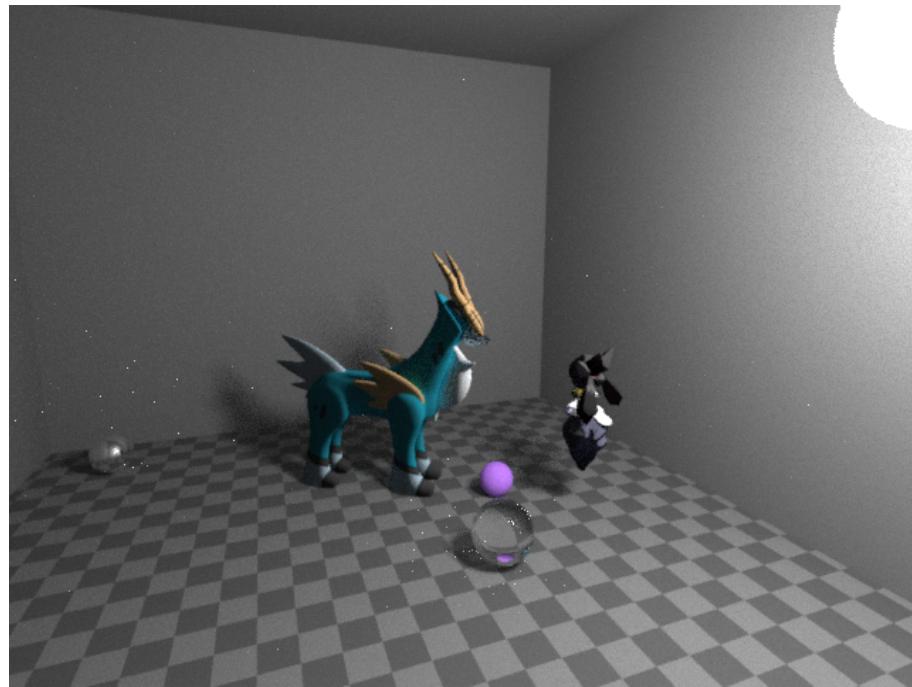


FIGURE 6 – La même image mais sans l'effet de profondeur de champ

De plus, pour avoir des images plus douces, on utilise de l'antialiasing, qui consiste à moyenner la couleur sur la surface entière de chaque pixel au lieu de viser un seul point, qui permet d'avoir des contours moins durs, car ils sont à moitié sur plusieurs pixels. Si on le désactive aussi, on peut voir des créneaux, par exemple sur le mur du fond.

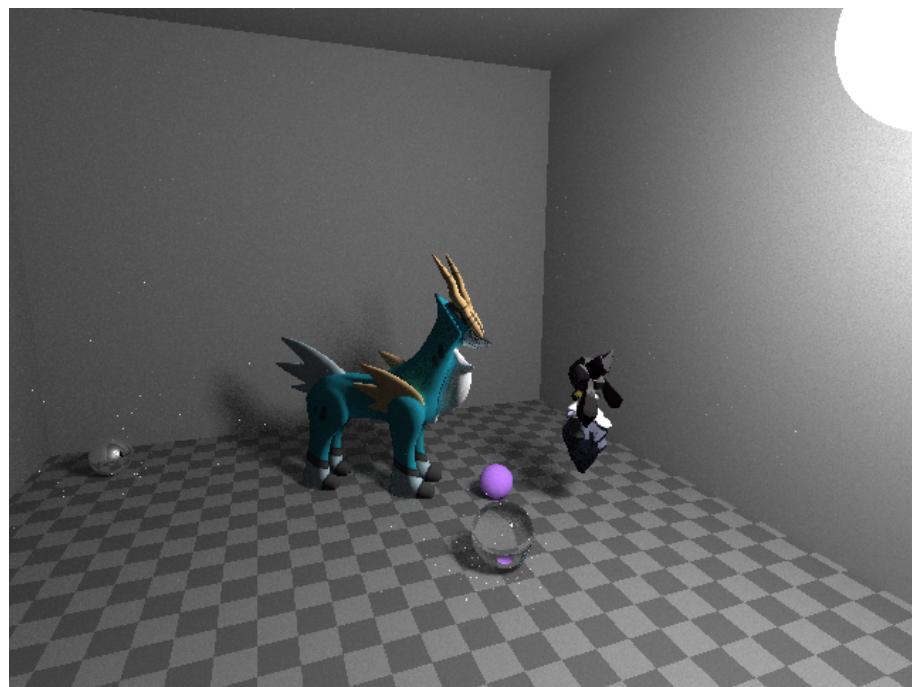


FIGURE 7 – La même image que précédemment mais sans antialiasing

Finalement, afin d'avoir un rendu des maillages bien plus réaliste, les normales et les textures sont interpolées (linéairement), c'est à dire que la normale et la texture n'est pas

constante dans un triangle, elle est fonction des valeurs aux sommets et des coordonnées barycentriques. Si on désactive ça, l'éclairage des faces devient beaucoup plus abrupt.

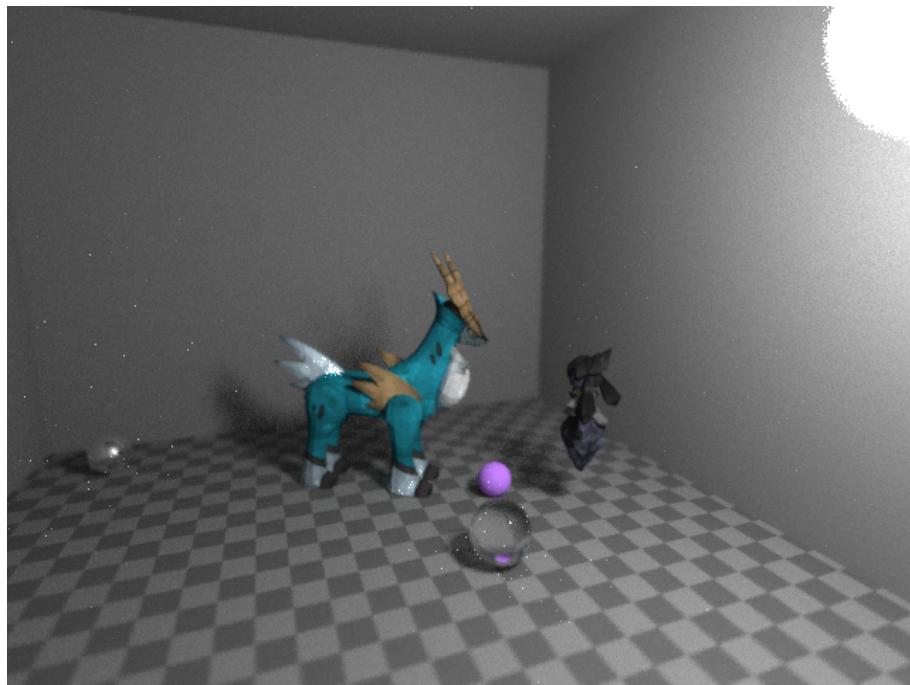


FIGURE 8 – La scène, mais sans interpolation des normales

4 Rendu final

Pour finir, voici une image avec des paramètres plus élevés, pour une qualité bien meilleure. Cette image a une résolution de 1920×1080 , et 1024 rayons par pixel, pour un temps de calcul d'environ une heure.

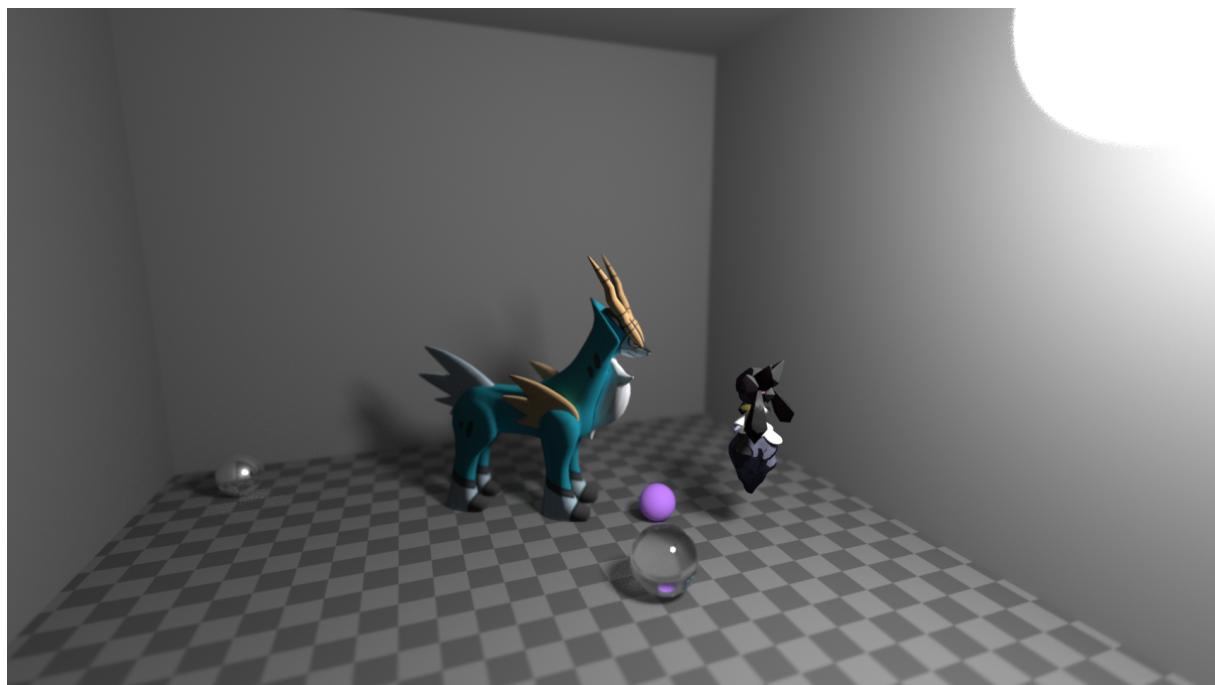


FIGURE 9 – L'image finale