Projet Ray Tracing

Université de Montpellier

Rapport Finale

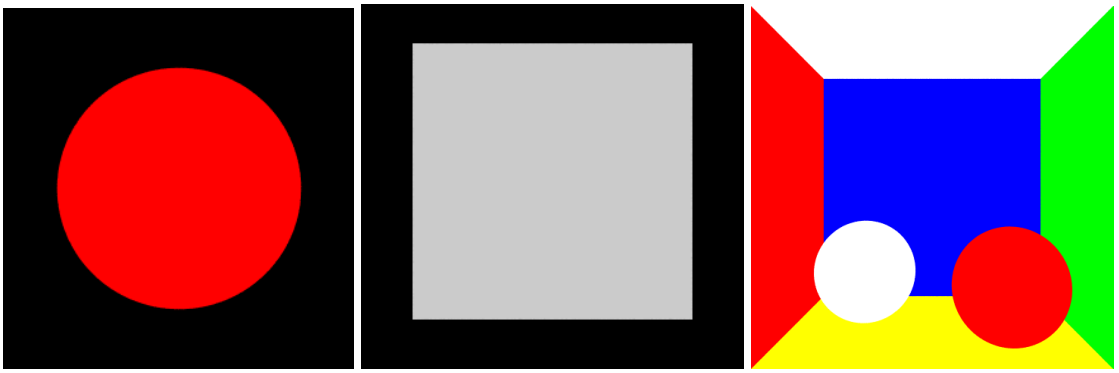
Alexandre Fournier

Introduction

Lors de ce projet, nous allons apprendre à utiliser la méthode de ray tracing pour afficher une scène, ses effets et sa luminosité. Pour ce second rapport, les parties sur la phase 1 et 2 n'ont pas changé par rapport au premier rapport.
(Toutes les images sont dans img/Resultat_rendu)

Phase 1 :

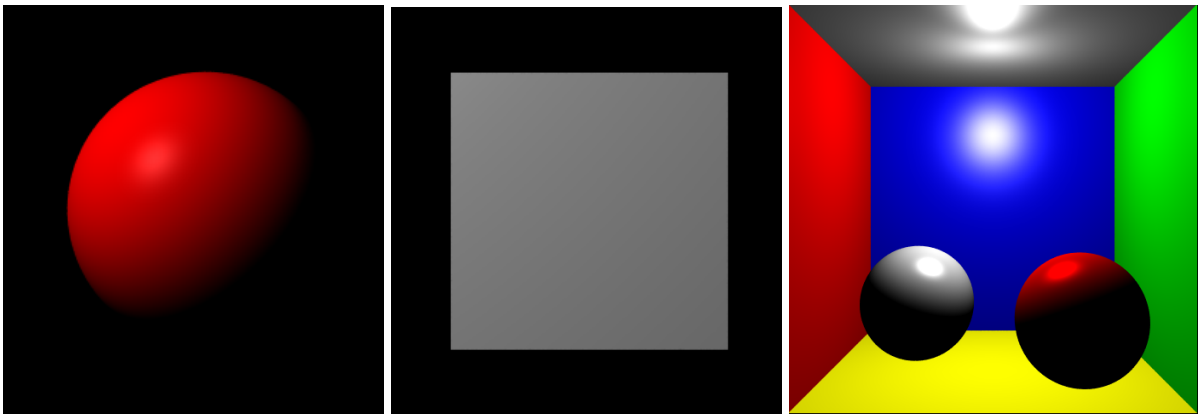
Pour la partie 1, tout est fonctionnel même si actuellement, je n'enregistre toujours pas 2 des paramètres pour sphère (thêta et phi) et squares (u et v). Même s'ils sont nécessaires pour l'uv mapping ne le faisant pas, je ne les ai pas ajouter. Le reste fonctionne parfaitement.
Vous pouvez voir ci-dessous les résultats de la partie 1 sur les 3 scènes.



Phase 2 :

Modèle Phong

Le modèle Phong est fonctionnel, j'utilise les méthodes vues en cours.
Vous pouvez voir ci-dessous les résultats du modèle phong sur les 3 scènes.



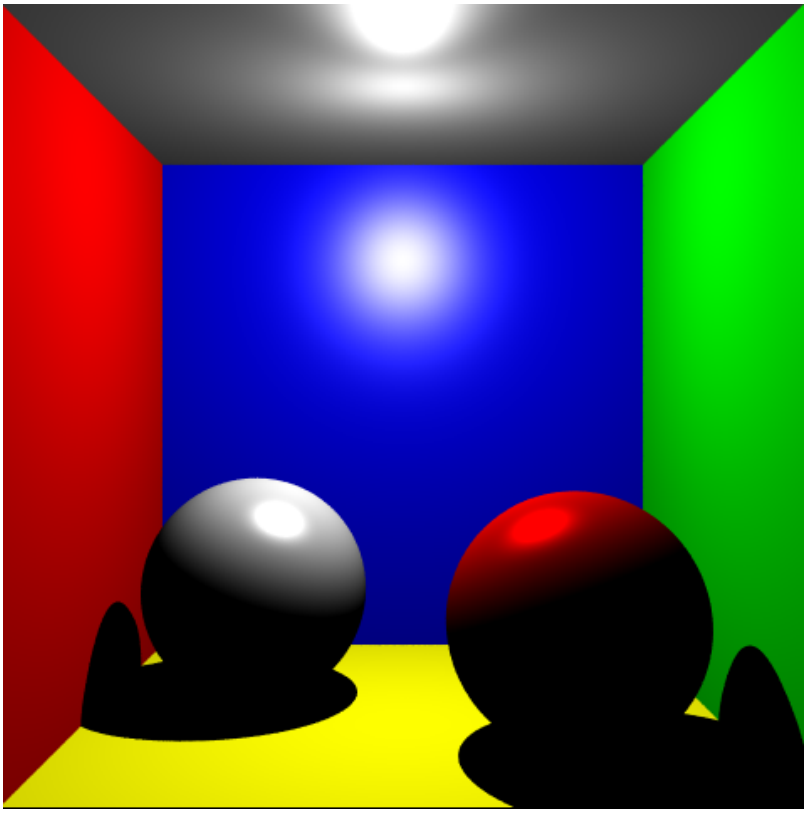
Ombre

Les ombres fonctionnent chez moi.

J'utilise dans ma fonction computeInstersectionShadow(), un offset que j'appelle epsilon pour être un peu permissif sur la position du rayon au cas où il aurait été détecté à l'intérieur de la sphère. Et la variable bias sert pour les ombres projetées trop loin. Dans cette fonction, d'ailleurs, je parcours d'abord les sphères et si une intersection valide y est trouvée, je ne parcours pas les squares. Vous pouvez afficher le résultat des ombres de bases en dé commentant cette partie du code :

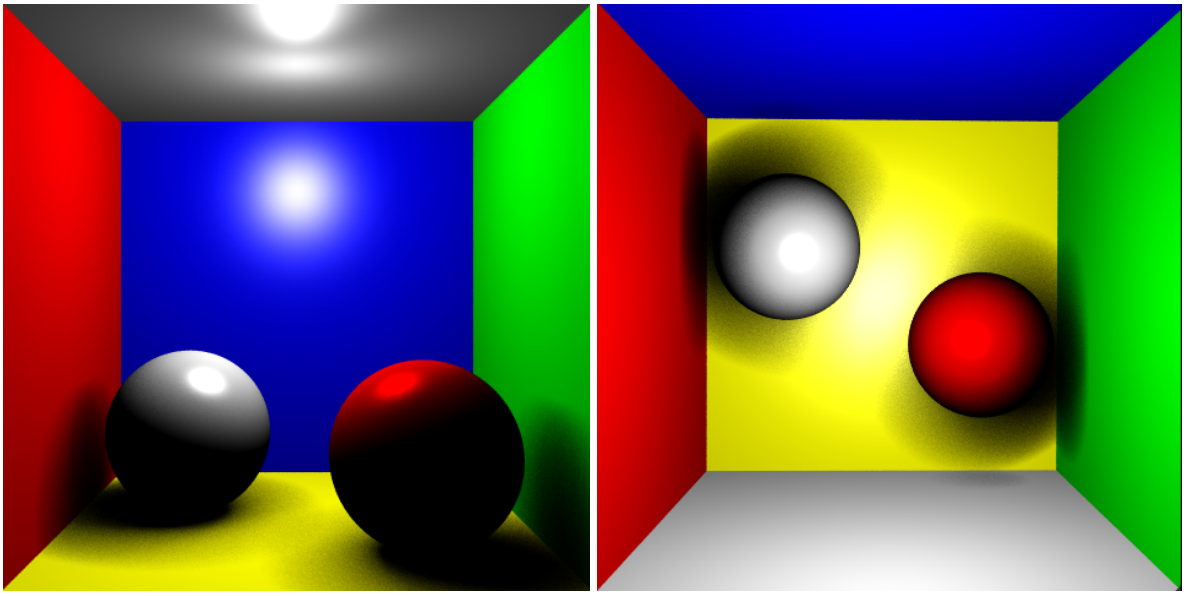
```
//SHADOW
/*//Hard
//Ray shadow = Ray(objectIntersect, L);
//float percentShade = isShadowed(shadow);
*/
```

Et en commentant celle pour les softs shadow.
Dans la fonction isShadowed() se trouve en commentaire une ébauche de code pour suivre la méthode proposé par la phase pour les ombres douces, mais celle-ci n'est pas terminée et j'ai donc utilisé une autre méthode.
Vous pouvez voir ci-dessous les résultats des ombres.



Ombre douce

Les Ombres douces fonctionnent dans mon projet. Et pour celle-ci, j'utilise donc une méthode qui envoie bien plusieurs rayons, elles aussi, pour ensuite permettre un pourcentage de ceux qui passent et ne passent pas. Ce pourcentage sera multiplier aux valeurs de couleur obtenue.
Contrairement à la méthode proposée, je n'utilise pas un quad, mais je fais une sphère à laquelle j'envoie des rayons à des positions aléatoires sur celle-ci. Comme cette fonction renvoie plusieurs rayons pour chaque point, elle est bien plus lente que celle des ombres de base.
Vous pouvez voir ci-dessous les résultats des ombres douces.

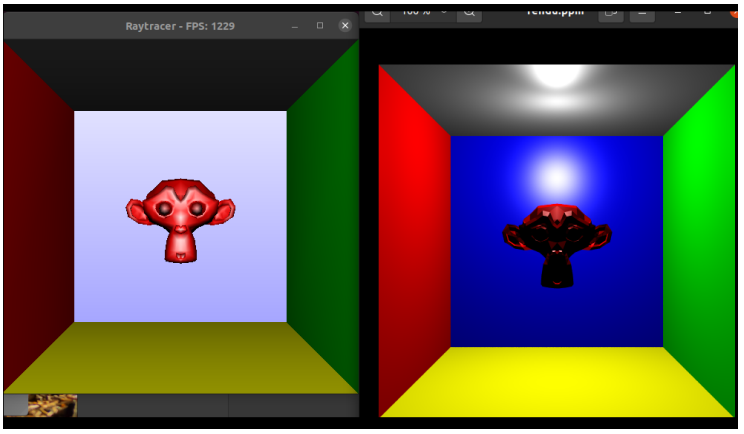


Phase 3 :

Mesh

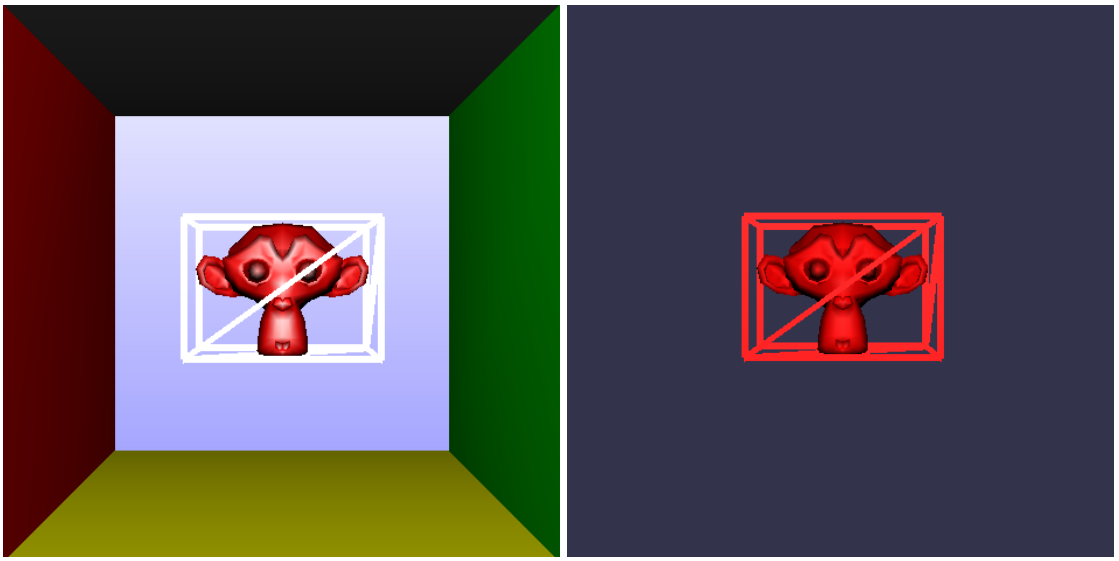
Pour cette phase, j'ai rajouté 2 scènes après les 3 premières. J'ai eu aussi quelques problèmes, mais j'ai réussi à obtenir un résultat. J'arrive à charger et à afficher un maillage 3D que ça soit seul ou dans la boîte de cornell. J'arrive aussi à appliquer le phong sur celui-ci et ainsi une couleur / ombrage.

Mesh seul avec phong



Mesh dans la boîte de cornell, affichage console/rendu

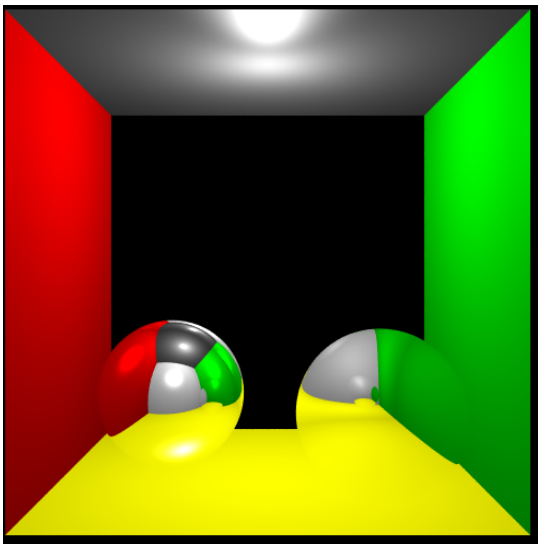
Néanmoins, le rendu pour ces deux scènes était particulièrement lent. J'ai donc rajouté du code pour faire les boîtes englobantes pour les meshes uniquement et accélérer le calcul de rendu de la scène. Pour montrer que je les ai, je les ai rajoutées dans draw. Si vous voulez les voir, décommenter la partie correspondante.
Le code des boundingBox se trouve dans le fichier correspondant, et celle-ci est ajoutée à chaque ajout de mesh. Après avoir testé sur la scène mesh seul, avec (~33 secondes) et sans bounding box (~4min32), j'ai constaté une très nette amélioration du temps de calcul.
Les bounding box étant aussi obligatoire pour le KD-tree, leur ajout est double à mon code. Cependant, je n'ai pas mis ce que j'avais fait pour le KD-tree, cela ne fonctionnait pas du tout et posait des soucis au reste de mon code.

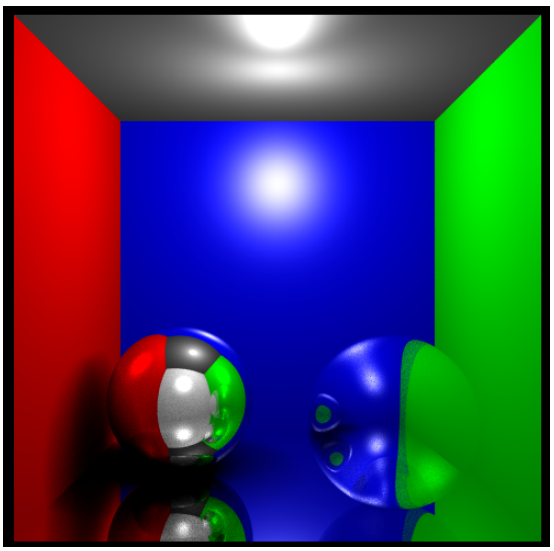


Comme vous pouvez aussi le constater sur l'image du mesh dans la boîte de cornell, ma méthode de shadow ne s'applique pas pour le mesh. Il rend aussi le rendu beaucoup, beaucoup plus lent, je recommande de commenter la partie concernant les ombres pour tester les scènes de mesh si vous le faites.

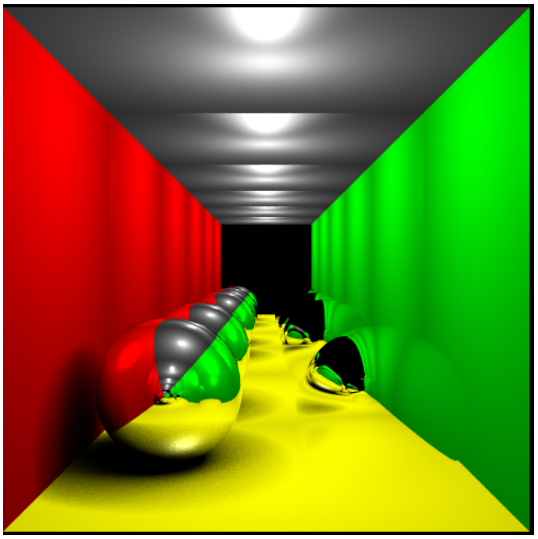
Phase Finale :

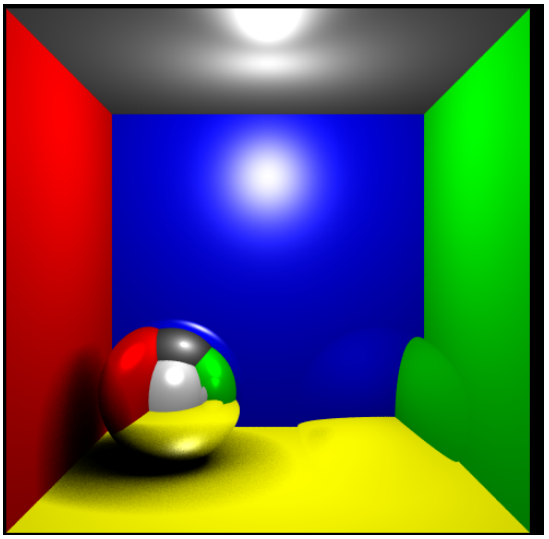
Pour cette phase, j'ai eu quelques problèmes, mais j'ai réussi à obtenir un résultat. J'ai eu des problèmes d'abord avec mes fichiers sphère et square, que j'ai corrigés. Mais j'avais aussi un problème avec un miroir posé sur le mur du fond qui s'affichait noir. La raison de ce problème était mon znear quand je rappelais rayTraceRecursive. Celui-ci était trop bas par rapport à l'endroit où était détectée l'intersection et donc quand le ray retour qui devait partir du miroir, il partait finalement de derrière celui-ci. Causant l'affichage noir ci-dessous.


Problème du miroir à cause de znear


Exemple problème de sphère

Le rendu final se voit sur les deux images suivantes et comme vous pouvez le constater, les problèmes ont été résolus.


Problème du miroir au fond corrigé


Version de base, sans problème sphère/square

Phase Optionnel :

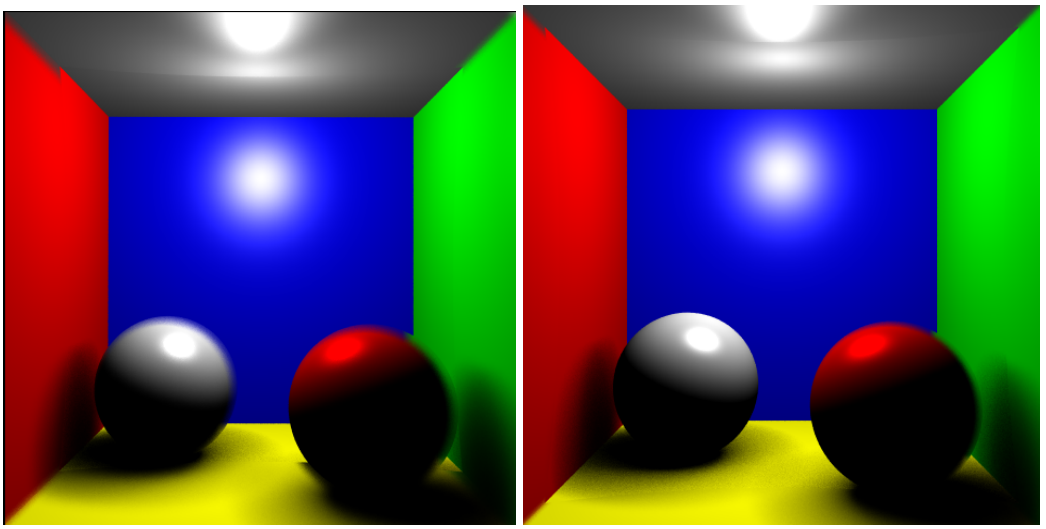
KD-Tree

Je n'ai pas réussi à faire fonctionner le KD-Tree.

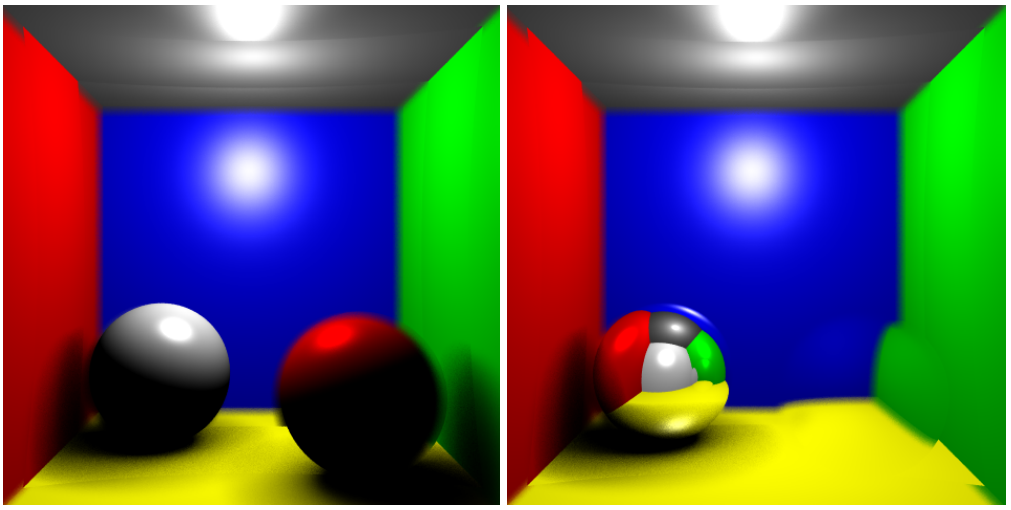
Profondeur de champ ajustable

Pour la profondeur de champ ajustable, j'ai créé une scène spécifiquement pour. La numéro 5. On peut adapter à quelle distance commence le focal, et à partir de quelle distance il se fait, et la distance à laquelle il peut rester nette.
Pour faire cette profondeur de champ ajustable, j'envoie d'abord un rayon pour récupérer la

couleur au point. puis j'envoie un second rayon qui va retourner uniquement la position de ce point. Et je vérifie si celui-ci se trouve ou non dans ma zone floue. Le souci de cette méthode, c'est qu'elle va être plus lente quand il y a des meshes (plus de rayons). Mais aussi parce que je coupe et redémarre mon flou brutalement, au lieu de le faire progressivement. Cela se voit nettement sur les images qui vont suivre. Une fois que je sais que le point est dans ma zone de flou, je vais renvoyer 8 rayons avec des directions légèrement modifiées aléatoirement pour effectuer un flou moyenneur. Et je vais faire la moyenne des couleurs obtenue par ces rayons pour obtenir la nouvelle couleur à ce point. Donnant donc un effet de flou.
Résultat de l'option ci-dessous :
Profondeur de champ uniquement à l'avant :



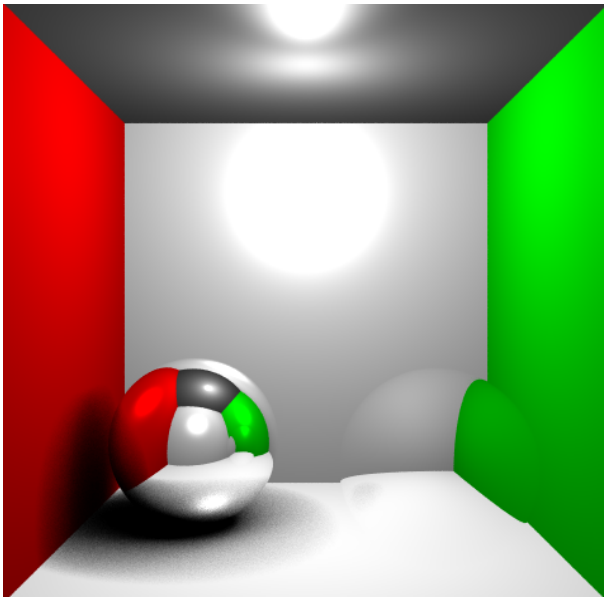
Profondeur de champ à l'avant et arrière :



Conclusion

Pour conclure ce rapport, je vais rajouter 2 captures d'écran, une avec la scène de cornell box donné par défaut au début du projet et une plus jolie en poussant un peu des éléments que j'ai développés.

Par défaut :



Plus poussée :