Ecole Nationale Supérieure d'Informatique pour l'Industrie et l'Entreprise

Rapport du projet PAP

RAY TRACING

Réalisé par :
DE GANI Guillaume
MOUMMOU Nisrine

Encadré par :
TORRI Vincent

Année universitaire 2019-2020

Contents

1. Présentation du sujet du projet

1.1. Le Ray Tracing

Le ray tracing ou « lancer de rayons » est une technique de calcul d'optique par ordinateur, utilisée pour le rendu en synthèse d'image ou pour des études de systèmes optiques. Elle consiste à simuler le parcours inverse de la lumière : on calcule les éclairages de la caméra vers les objets puis vers les lumières, alors que dans la réalité la lumière va de la scène vers l'œil. Pour faire court, le Ray Tracing tient à reproduire de la manière la plus fidèle possible le trajet de la lumière dans un environnement en 3D.

Le concept du Ray Tracing date de 1968, et le premier algorithme récursif a été présenté en 1980 par Whitted. Cela a été la première méthode permettant d'obtenir des images de synthèse photo-réalistes, prenant en compte la réflexion, la réfraction, l'élimination des parties cachées, et d'autres effets optiques. Bien que cette technique donne des résultats de haute qualité, elle est assez peu utilisée en pratique, car nécessite un temps de calcul et de l'espace mémoire très importants.

1.2. Objectif du projet :

En ce qui concerne notre projet, nous étions amenés à implémenter l'algorithme général du tracé de rayon qui consiste en :

- La construction de la scène avec les objets, la source de lumière et la caméra (i.e. l'œuil).
- Le calcul de l'image.
- La sauvegarde de l'image.

3. Les étapes et l'avancement du projet

3.1. Les étapes du projet et le raisonnement suivi :

Dans le sujet proposé, nous devions implémenter les 3 étapes de l'algorithme du Ray Tracing citées précedemment. Pour celà nous avons commencé tout d'abord par :

- Définir une scène constituée d'objets dans un espace en 3 dimensions ; les 5 quadrilatères.
- Ensuite il fallait tracer un rayon partant du point d'observation et qui passe par chacun des points de l'écran à tour de rôle, ce qui nous permet par la suite de déterminer la définition de notre image.
- Puis, il faut tester si le rayon obtenu rencontre un objet de notre scène.
  - Si ce rayon a effectivement un point d'intersection avec un unique objet, on passera à l'étape suivante,
  - si par contre il rencontre plusieurs objets, il faudra tout d'abord déterminer lequel de ces différents objets le rayon croise en premier avant de passer à l'étape suivante.
  - Et par ailleurs, si le rayon ne touche aucun objet, alors le pixel de l'écran par lequel passe ce rayon aura une couleur définie ; la couleur de fond de l'image.
- L'étape suivante alors consiste à, une fois le point de l'objet croisé soit déterminé, on lui attribue une couleur.
- Aussi, il faudra prendre le soin de vérifier que notre point de l'objet n'est pas caché d'une source lumineuse par un autre objet de la scène, car cela voudra dire qu'il n'est pas éclairé. Ainsi nous obtiendrons une variation de luminosité en fonction des sources lumineuses ce qui permettra à notre image d'être en 3D.

- Pour finir, on va aussi ajouter les ombres correspondants aux objets éclairés ainsi que leur reflets.

3.2. Les classes importantes implémentées :

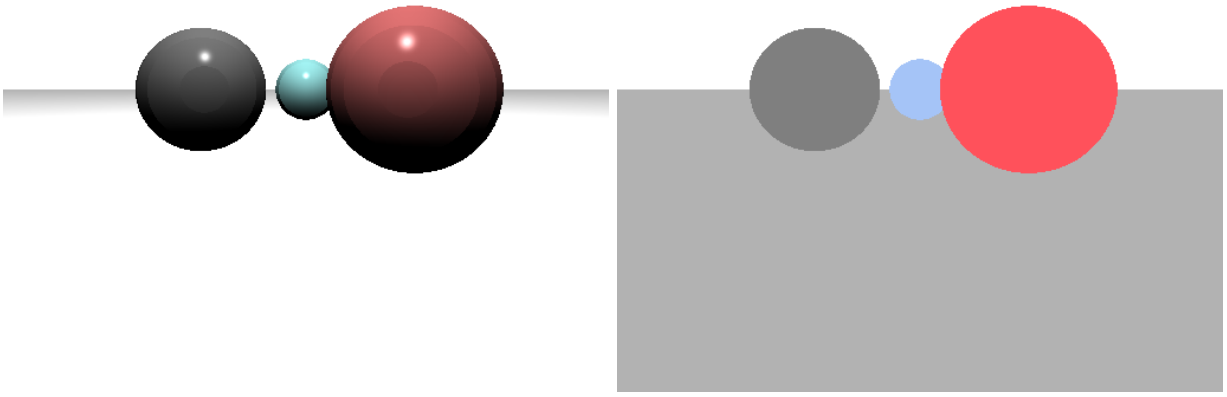
Pour mener à bien notre projet, la création des classes suivantes était indispensable :

- **Vector** : Classe utilisée pour faire des manipulations basiques sur les vecteurs.
- **Ray** : Classe qui implémente l'origine, la direction et le point d'intersection entre le rayon et l'objet. C'est la classe la plus importante du projet vu qu'il y dépend entièrement.
- **Light , Material** : Deux classes qui représentent l'objet.
- **Shape** : Classe qui implémente des méthodes virtuelles nécessaires pour appliquer le Raytracing à différentes formes :
  - **Cube** : Classe fille de la classe Shape.
  - **Sphere** : Classe fille de la classe Shape.
  - **Plane** : Classe fille de la classe Shape.

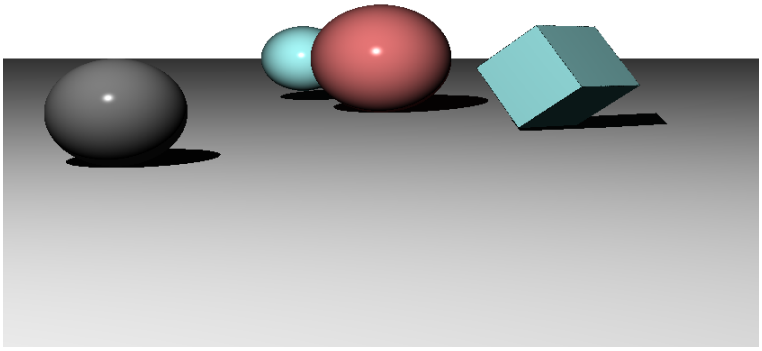
Bien évidemment il fallait aussi implémenter une classe Screen afin de pouvoir présenter les objets.

3.3. L'avancement du projet en image :

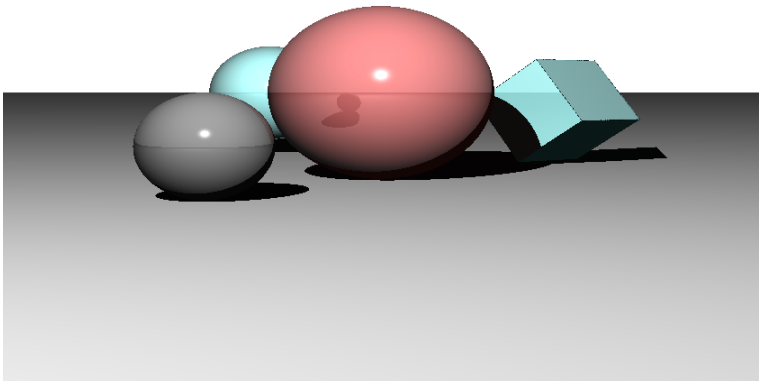
La création de la scène ainsi que des objets éclairés en 3D :



L'ajout des ombres des objets et de la nouvelle forme ; Cube :



L'ajout du reflet des objets sur les autres objets :



4. Les problèmes rencontrés et les solutions trouvées

Durant l'implémentation du code de ce projet, nous avons affronté plusieurs problèmes dont on cite :

- **Problème de calcul d'intersection** : Le calcul d'intersection pour un cube non aligné aux axes est extrêmement compliqué.

=> *La solution fut de calculer l'intersection avec chaque face du cube en les considérant comme des plans et d'ensuite détérminer lequel des plans était touché en premier.*

- **Problème de couleur des objets** : La détermination de la couleur en un point en fonction de sa distance par rapport à la lumière ne fonctionne pas.

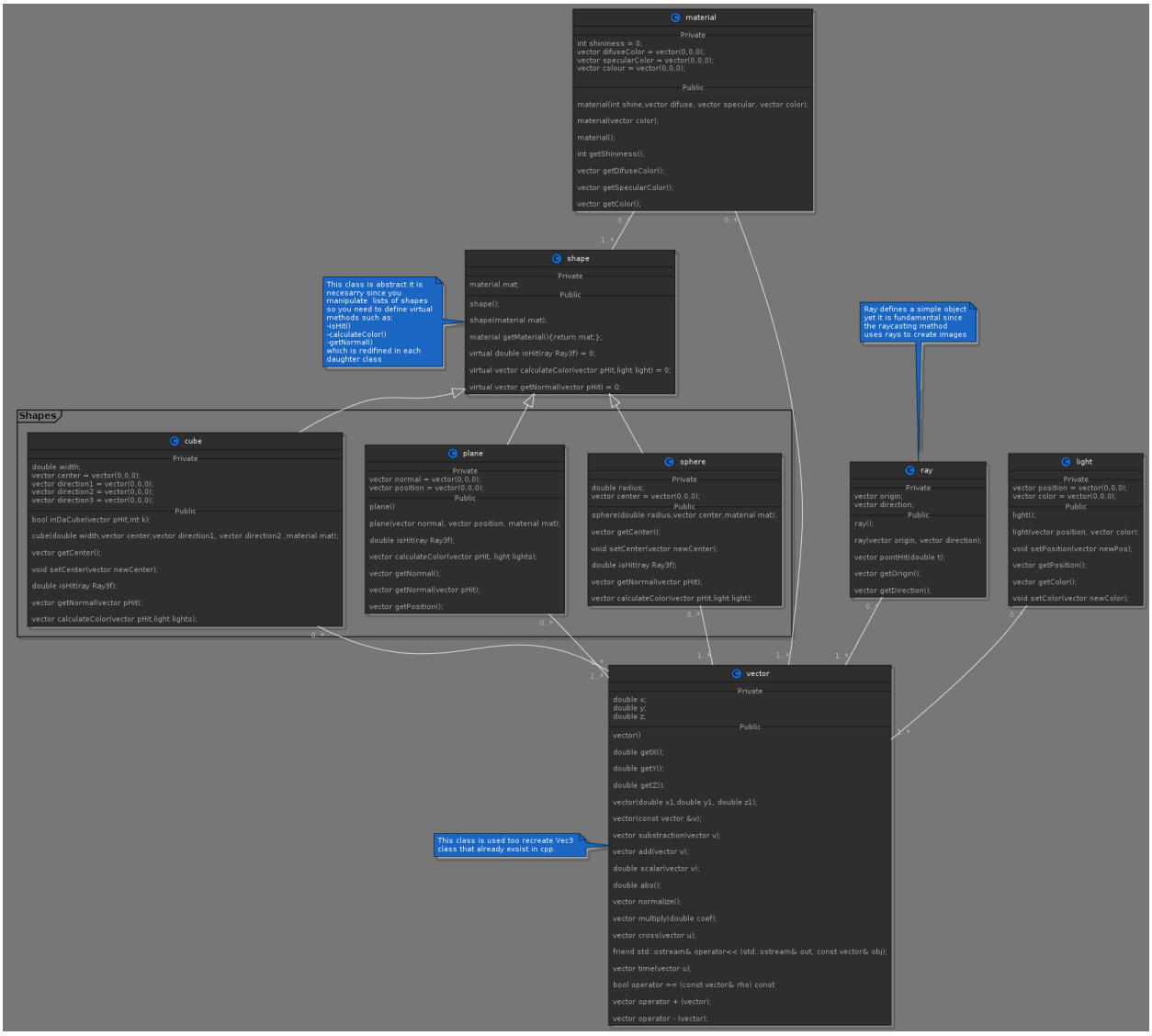
=>*Le calcul de la couleur en 1 point en applicant le modèle de Phong.*

- **Problème de réflexion** : L'implémentation de la réflexion était problématique et complexe.

=> *L'implémentation de la réflexion en utilisant un appel récursif de la fonction raytrace.*

5. Le graphe UML

Le graphe UML de notre projet est le suivant :



6. Conclusion

Ce projet s'est révélé très enrichissant dans la mesure où il a consisté en une approche concrète de la programmation en langage C++ et par ailleurs du métier d'ingénieur en informatique. En effet, la prise d'initiative, le respect des délais et le travail en équipe seront des aspects essentiels de notre avenir.

De plus, il nous a permis d'appliquer nos connaissances en physique et en programmation avancée tout en cherchant à les développer davantage.

Toutefois, si un tel sujet devait être complété, il serait judicieux de chercher à optimiser le code afin de minimiser le coût de calcul. À part celà, nous avons réussi à atteindre tous les objectifs assignés concernant ce projet.