Gamagora - Université Lyon 2
2020/2021

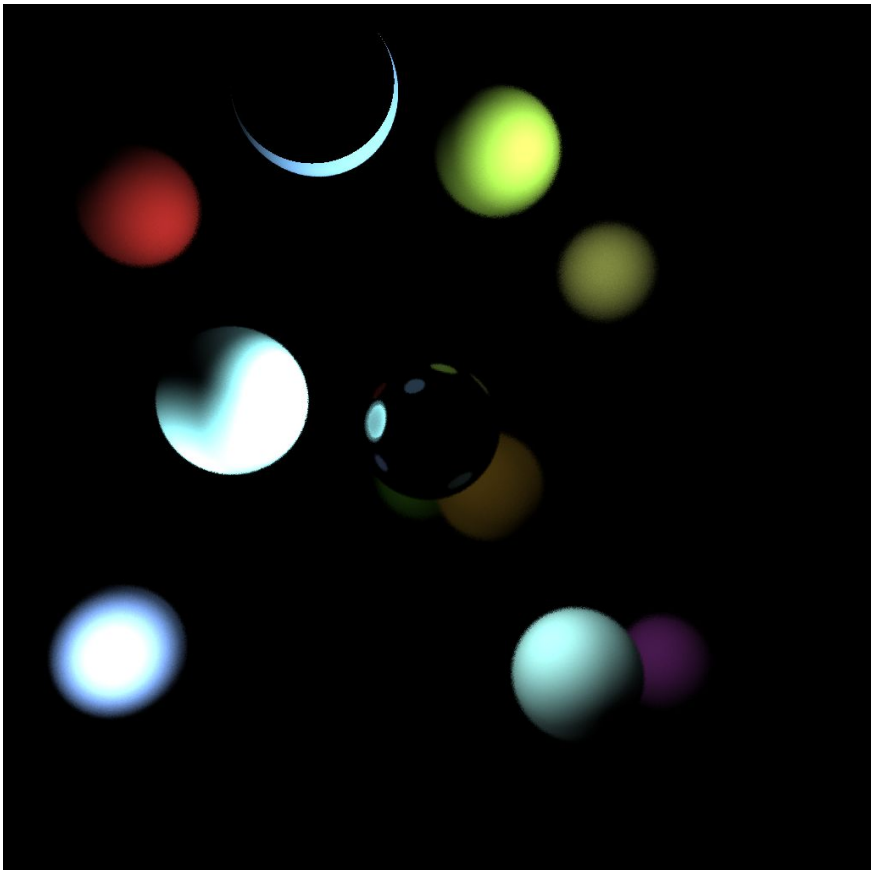
Synthèse d'images

Lorenzo MARNAT

Lien Github: https://github.com/LorenzoMarnat/SimulationRayTracing

Description du projet

L'objectif de ce projet est de calculer la lumière en fonction de plusieurs lampes disposées arbitrairement dans la scène. Pour cela, des rayons sont tirés en direction de la scène depuis un point "caméra". Lorsqu'une sphère est touchée, un rayon est tiré en direction de chacune des lampes afin de savoir si des lampes éclairent la sphère et, le cas échéant, à quelle distance elles se trouvent . Le pixel de l'image correspondant au point d'intersection entre le premier rayon et la sphère est coloré en fonction de la couleur de la sphère et de l'intensité de la lumière. Les sphères miroirs reflètent les sphères voisines et font rebondir les rayons qui les atteignent. De plus, l'utilisation de la perspective permet de retranscrire la profondeur sur l'image finale.



Afin d'optimiser les calculs, les sphères entourées par des boîtes englobantes. Ces dernières sont elles-mêmes disposées dans une structure de données récursive: les boîtes

englobantes sont contenues dans une boîte plus grande, elle-même dans une boîte plus grande etc, jusqu'à parvenir à une boîte contenant toute la scène. Ainsi, plutôt que de tester l'intersection d'un rayon avec toutes les sphères pour savoir lesquelles se trouvent sur son chemin, il est possible de tester récursivement l'intersection avec les boîtes de plus en plus petites. Cela permet d'éliminer d'office toutes les sphères contenues par des boîtes ne se trouvant pas sur le chemin du rayon et de gagner un temps énorme d'exécution.

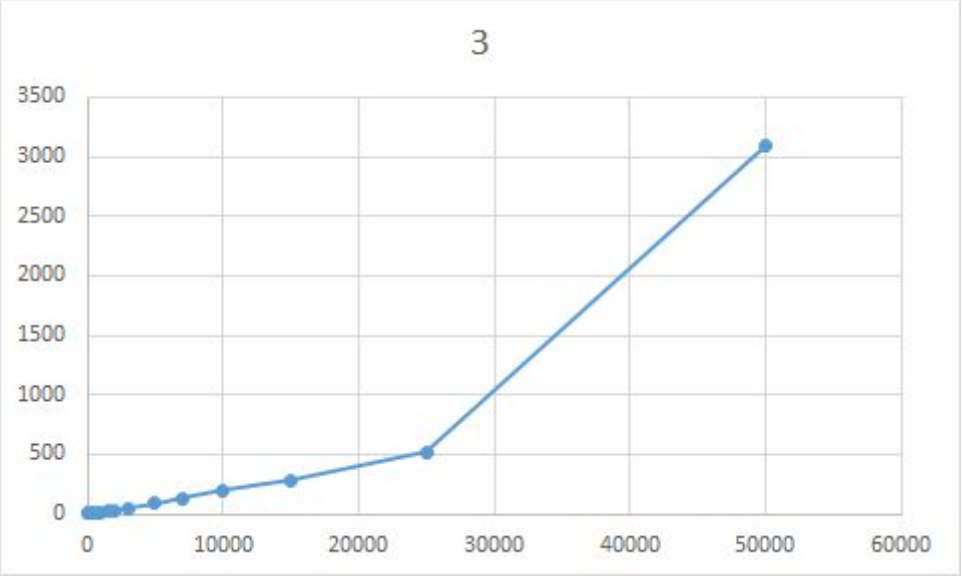
Résultats

Dans cette section, nous allons analyser l'efficacité de la structure de données. Pour gagner du temps, les tests sont effectués avec un seul rayon tiré par pixel de l'image. De même, un unique rayon est tiré en direction de chaque lampe.

La structure de données a été testé avec plusieurs profondeurs, allant de 3 à 15. La profondeur correspond au nombre de divisions en sous-boîtes de la scène.

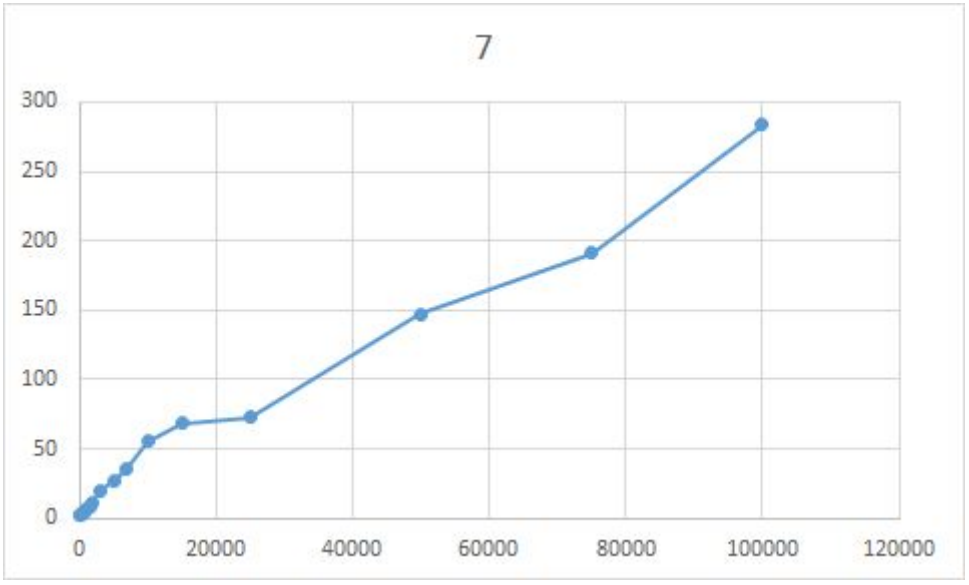
En ordonnée: temps d'exécution en secondes
En abscisse: nombre de sphères dans la scène

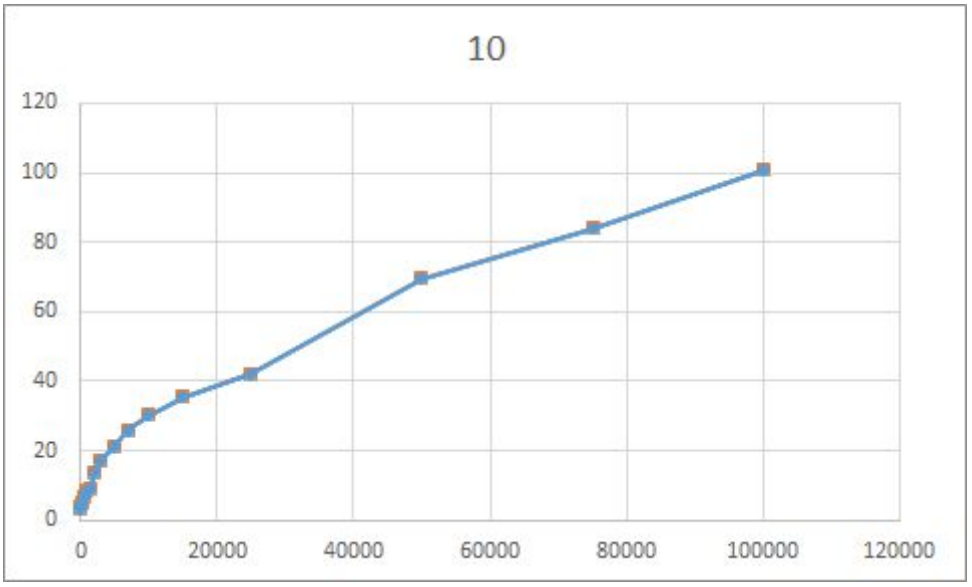
Profondeur 3



A profondeur 3, le temps de calcul explose au-delà de plus de 100 sphères et semble suivre une augmentation exponentielle. Le nombre de sous-boîtes est beaucoup trop faible et chacune contient trop de sphères.
Cependant, cette profondeur obtient les meilleurs résultats pour un faible nombre de sphères (moins de 100).

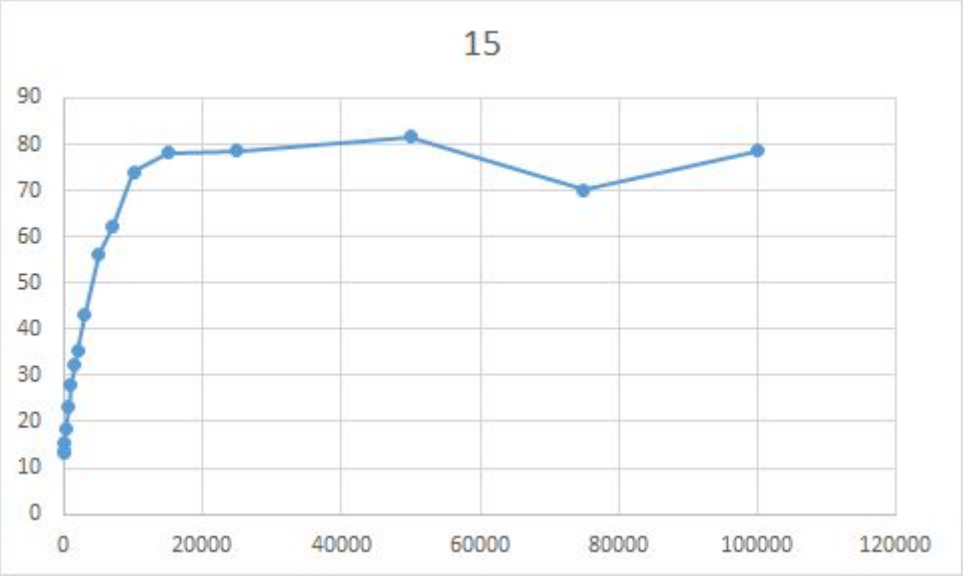
Profondeurs 7 et 10





Plus efficaces que la profondeur 3, les profondeurs 7 et 10 semblent suivre une progression plus linéaire.

Profondeur 15



La profondeur 15 est totalement inefficace pour un faible nombre de sphères. Cependant, elle est très efficace au-delà de 10 000 sphères. Le temps de calcul n'augmente pratiquement pas entre 10 000 et 100 000 sphères.

Conclusion

La structure des boîtes permet de gagner énormément de temps de calcul. Cependant, il faut utiliser une profondeur adaptée: une profondeur faible est très adéquate pour un faible nombre de sphères mais quasiment inutile au-delà d'une centaine de sphères. Inversement, une profondeur élevée est très efficace pour des très grands nombres de sphères mais fait perdre beaucoup de temps pour des nombres très faibles.