Rapport du TP5 - Lancer de rayons et intégration numérique

Loïs Paulin

Février 2018

Résumé

Dans ce rapport nous présentons le travail effectué pour le TP5 de rendu avancé. Au cours de ce TP nous avons implémenté un lanceur de rayon permettant de rendre l'occlusion ambiante de scènes importées depuis un fichier wavefront. Nous proposons plusieurs méthodes d'échantillonage dont la spirale de Fibonacci perturbée qui donne la convergence la plus rapide. Afin de pouvoir calculer le rendu de scènes contenant un nombre plus important de triangles en un temps décent nous avons rajouté un BVH permettant une accélération considérable du calcul d'intersection rayon-scène.

1 Comparaison des méthodes d'échantillonage

Afin de rendre l'occlusion ambiante nous avons intégré la lumière directe atteignant chaque point visible avec le ciel une source de lumière d'intensité 1. Nous avons implémenté quatre stratégies d'échantillonage :

— Selon une loi de répartition uniforme
— Selon une loi de répartition cosinusoïdale
— Avec une spirale de Fibonacci
— Avec une spirale de Fibonacci perturbée

Dans cette partie nous allons comparer les résultats des différentes méthodes.

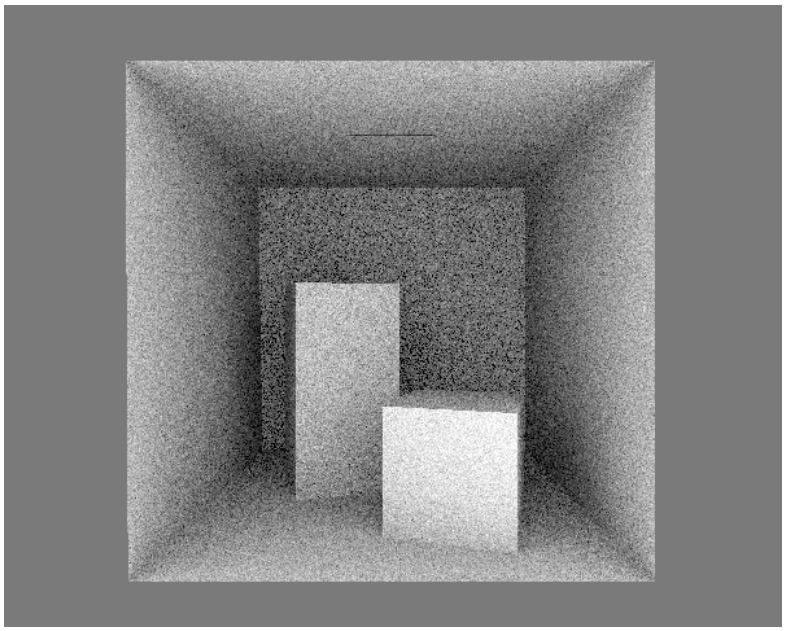

FIGURE 1 – Loi de répartition uniforme (32 samples par pixel)

L'échantillonnage par spirale de Fibonacci non perturbée donne l'effet attendu de lignes d'ombre dûes à un échantillonage dans les mêmes direction entre un point et son voisin. Ensuite nous observons que pour un nombre d'échantillons équivalent la spirale de Fibonacci donne un résultat moins bruité. L'échantillonnage avec une répartition cosinusoïdale semble donner un résultat légèrement moins bruité que l'échantillonnage uniforme.

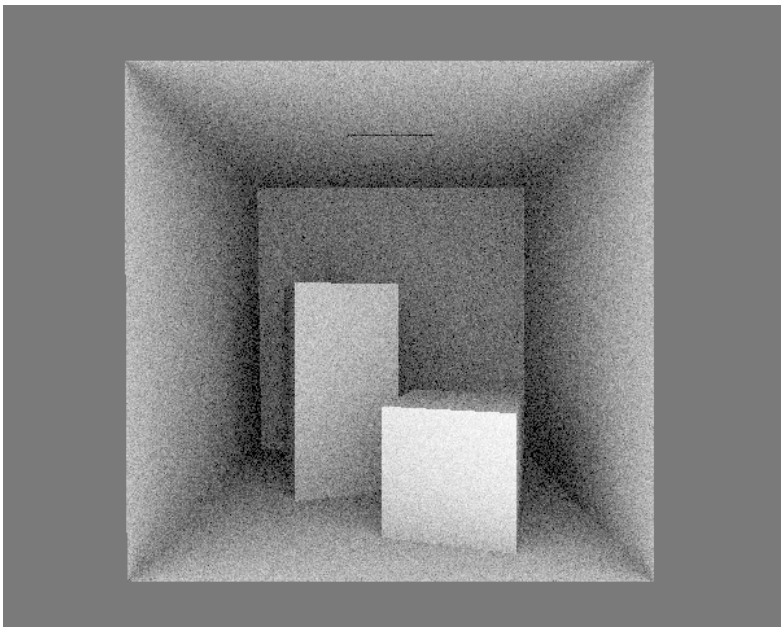

FIGURE 2 – Loi de répartition cosinusoïdale (32 samples par pixel)

Lors des tests nous avons observé que la spirale de Fibonacci intègre la fonction de cosinus à 0.5 sur l'hémisphère. Nous corrigeons donc le résultat de l'intégration par un facteur 2 afin d'avoir les mêmes résultats qu'avec l'échantillonage uniforme.

2 Structure d'accélération

Afin de pouvoir calculer plus rapidement les intersections rayon / scène nous avons implémenté une structure de BVH sur les meshs. Le BVH permet de séparer la scène en une hiérarchie arborescente de boîtes englobante. Lors de l'exploration pour tester l'intersection à la scène nous mettons testons en premier la sous-boite la plus proche de l'origine du rayon afin d'améliorer les chances de tomber sur l'intersection la plus proche tôt et donc diminuer le temps d'exploration de l'arbre.

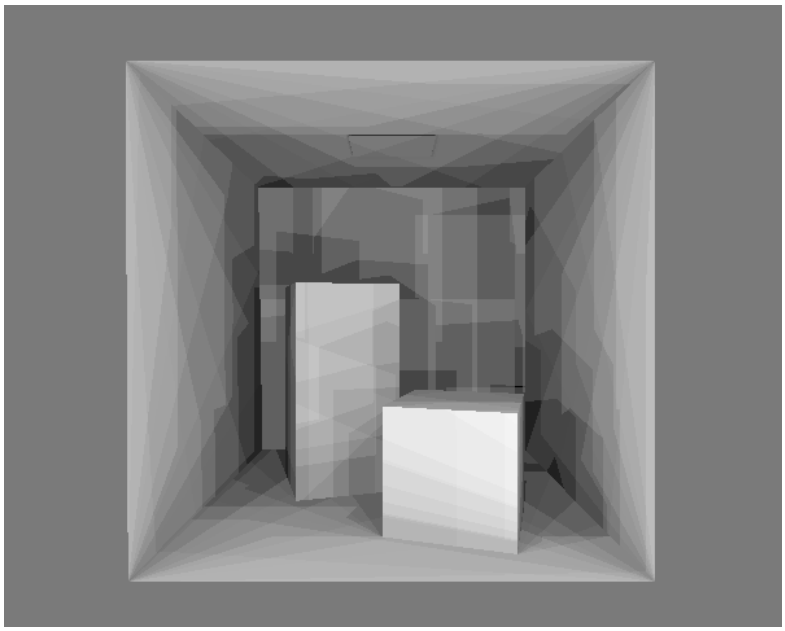

FIGURE 3 – Spirale de Fibonacci (32 samples par pixel)

3 Informations

Plusieurs méthodes ont été implémentées pour le calcul d'intersection Rayon - Triangle, pour la méthode de choix de normale (interpolée ou par face) et pour la méthode d'échantillonage. Des constantes préprocesseur peuvent être définie dans Mesh.cpp et Scene.cpp pour choisir parmi les possibilités.

Le projet n'a pas été réalisé à l'aide de gkit, l'importation des fichiers wavefront ayant été faite pour correspondre aux fichiers donnés dans le sujet, la présence de deux '/' dans la description des indices de sommets de face est nécessaire même si le fichier n'est pas obligé de renseigner les normales ni les coordonnées de texture.

Le projet a été réalisé au sein d'un moteur de rendu légèrement plus étendu permettant de rendre aussi des scènes ayant des matériaux simple comme des matériaux diffus, purement spéculaires et transparents, ainsi que des objets tels que des sphères. Cependant le BVH est pour l'instant implémenté dans la classe Mesh qui ne contient que des triangles. Pour

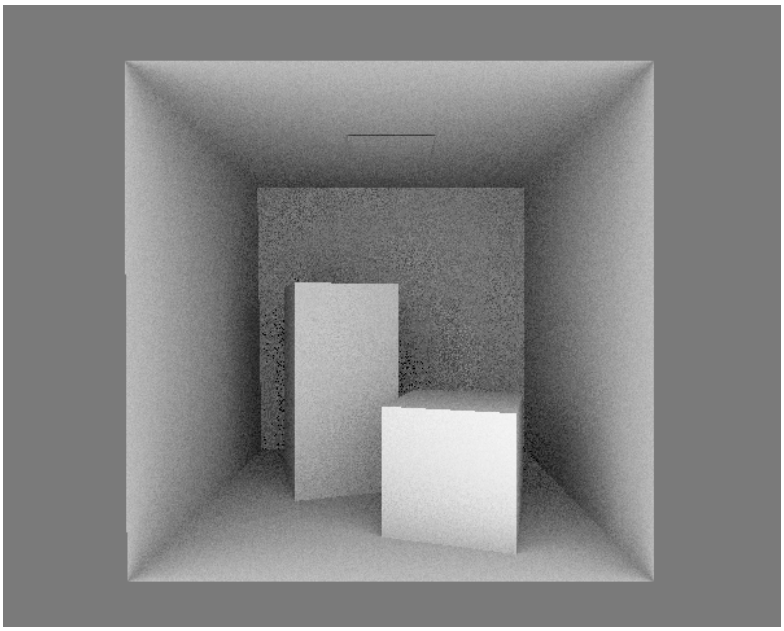

FIGURE 4 – Spirale de Fibonacci perturbée (32 samples par pixel)

adapter le BVH à des scènes contenant plus de types d'objets, on pourrait doter chaque objet d'une fonction calculant la boîte englobante de l'objet et utiliser cela pour définir les primitives du BVH.

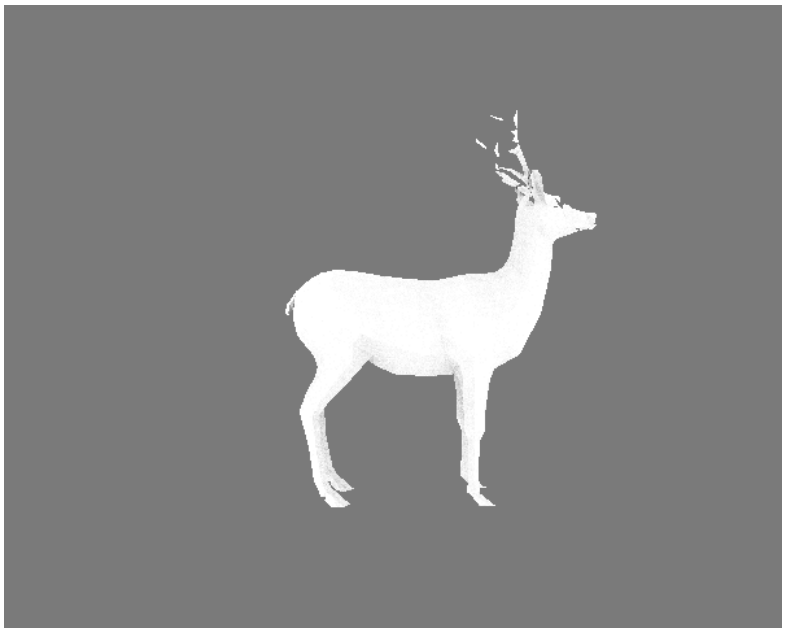

FIGURE 5 – Occlusion Ambiante d'un mesh à 800 triangles (1.3s avec BVH, 45.3s sans BVH)