Chef d'oeuvre - Spécifications
Approximation d'éclairage indirect en temps réel

Blaise Cardonne Gauthier Bouyjou Valentin Camus
Rihab Elrifai Sylvain Durand

Encadrant/Client : François Desrichard

Janvier
2020

Contents

1 Contexte

Ce projet s'inscrit dans le cadre de l'UE chef d'oeuvre du M2 IGAI de l'université Toulouse III - Paul Sabatier. Le client et encadrant de notre groupe François Desrichard, nous a proposé de travailler sur l'approximation de l'éclairage indirect en temps réel à partir de méthodes existantes basées sur la radiosité instantanée de Alexander Keller[1].

2 Description du projet

2.1 Objectif global

Le système final doit permettre de visualiser une scène avec de l'éclairage direct et indirect, le tout en temps réel.

L'implantation s'appuiera sur les deux articles de recherche *Virtual spherical gaussian lights for real-time glossy indirect illumination*[2] et *Reflective shadow maps*[3] ainsi que le blog du développeur MJP *The Danger Zone. SG series, 2016*[4].

Une exigence facultative est d'implémenter les *Imperfect Shadow Maps*[5] en plus des *Reflective shadow maps*[3] qui permettraient de ne pas négliger la visibilité entre un point de la scène et les VPLs.

Le système devra fournir un moteur de rendu minimaliste permettant la visualisation des différentes méthodes d'éclairages indirectes. L'importation de scène aux formats standards permettrait de comparer nos résultats avec un rendu physiquement réaliste tel que le logiciel *PBRT* peut fournir. Nous aurons également l'opportunité de comparer nos résultats avec les références fournies dans certains articles (notamment celui sur les gaussiennes sphériques) du moment que nous sommes en mesure de récupérer les scènes correspondantes.

3 Cahier des charges

3.1 Définition du besoin

Comme énoncé précédemment, l'objectif du projet est la simulation d'un rebond d'éclairage indirect en temps réel à partir d'un ensemble de méthodes définies.

3.2 Exigences fonctionnelles

Le produit fini devra offrir l'ensemble des fonctionnalités suivantes (fonctionnalités optionnelles incluses):

- Le chargement d'une scène 3D
- L'affichage d'une scène 3D via une BRDF physiquement réaliste

- La gestion de caméra dynamique
- La gestion de sources lumineuses (éventuellement dynamiques)
- Le changement de mode de rendu (notamment pour comparer les différentes méthodes)
- La gestion de l'éclairage indirect selon les méthodes suivantes:
  - *Reflective shadow maps*[3]
  - *Reflective shadow maps*[3] et les *gaussiennes sphériques*[2]
  - *Reflective shadow maps*[3], les *gaussiennes sphériques*[2] et les *Imperfect Shadow Maps*[5]
- La sauvegarde d'une image rendue pour comparaison avec une image de référence.

3.3 Niveau de priorité des exigences

FP = Fonction Principale
FO = Fonction Optionnelle

Fonction	Description	Priorité
FP1	Charger une scène 3D.	Forte
FP2	Dessiner la scène avec éclairage direct.	Forte
FP3	Déplacer la caméra.	Forte
FO1	Déplacer les lumières.	Faible
FP4	Choisir quelle méthode d'éclairage indirect utiliser.	Forte
FP5	Générer de l'éclairage indirect par avec la méthode *Reflective Shadow Maps*.	Forte
FP6	Générer de l'éclairage indirect avec les méthodes *RSM* et les gaussiennes sphériques.	Moyenne
FO2	Générer de l'éclairage indirect avec les méthodes *RSM*, les gaussiennes sphériques et *Imperfect Shadow Maps*.	Faible
FO3	Sauvegarder une image rendue sur disque.	Faible

Figure 1: Niveau de priorité des exigences

3.4 Validation des fonctionnalités

Fonction	Objectif à remplir pour validation
FP1	L'utilisateur pourra importer des scènes 3D à partir du logiciel.
FP2	L'utilisateur après avoir importé une scène pourra la visualiser en utilisant l'éclairage direct avec calcul de visibilité.
FP3	L'utilisateur pourra déplacer la caméra au clavier.
FO1	L'utilisateur peut sélectionner une source de lumière (et optionnellement la déplacer).
FP4	L'utilisateur peut choisir quel mode de rendu utiliser parmis les méthodes proposées.
FP5	L'utilisateur peut visualiser le résultat avec les *RSMs* seules.
FP6	L'utilisateur peut visualiser le résultat avec les *RSMs* et les gaussiennes sphériques.
FO2	L'utilisateur peut visualiser le résultat avec les *RSMs*, les gaussiennes sphériques et les *ISMs*.
FO3	L'utilisateur peut sauvegarder une image.

Figure 2: Validation des exigences

3.5 Validation de l'éclairage indirect

La validation du résultat visuel se fera de plusieurs manières. Dans un premier temps, on pourra se contenter d'effectuer des comparaisons visuelles pour valider l'effet des différentes méthodes implanter.

Pour une validation plus robuste et formelle, on pourra effectuer des calculs de distances sur les images produites par notre projet. Dans les deux cas, nous utiliserons un jeu d'images de références, soit fournies dans les articles, soit générées sur un logiciel comme *PBRT*.

3.6 Exigence non fonctionnelle

Nous avons une exigence non fonctionnelle fondamentale dans le cadre de ce projet, qui est en réalité une contrainte: le fonctionnement en temps réel. En effet, au moins pour la méthode *Reflective shadow maps*[3] et les *gaussiennes sphériques*[2], nous aurons besoin d'obtenir un résultat fonctionnant à un taux d'images par seconde satisfaisant (supérieur à 60 images par seconde).

3.7 Contraintes

Notre implantation est soumise à un ensemble de contraintes de versions et de compatibilité:

- GLSL 4.10
- OpenGL 4.1 (pour la compatibilité MacOS, cela deviendra un obstacle si on doit utiliser des compute shaders)

- Idéalement compilable pour Linux, Windows et MacOS
- Code C++

3.8 Délais de rendu

Le rendu du logiciel se fera avec le client lors de la recette qui aura lieu le 25 février 2020. Un manuel d'utilisation sera fourni avec le logiciel.

4 Vue générale du système

Le moteur de rendu permettra de visualiser une scène de l'utilisateur en utilisant les différentes méthodes d'éclairages indirectes implantées et permettra de faire des mesures de performance et de qualité visuelle en sauvegardant une image de rendu pour comparaison avec une image de référence (pré-calculée par exemple avec *PBRT*).

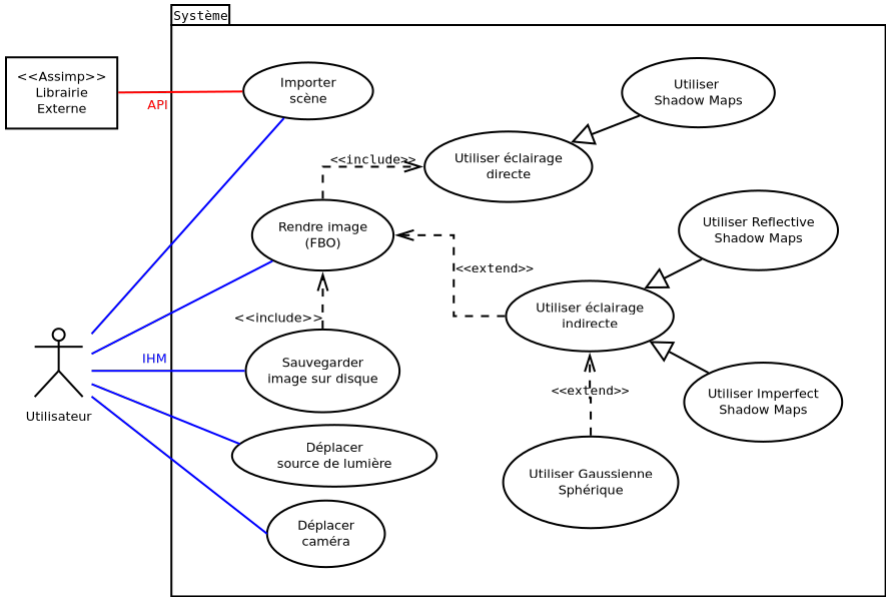

Figure 3: Diagramme des cas d'utilisation

Le système propose cinq cas d'utilisation :

- Importer une scène: demande à l'utilisateur quel fichier importer
- Visualiser la scène à l'écran: une fenêtre de rendu doit occuper la plus grande partie de l'écran

- Sauvegarder une image sur disque: saisie d'un nom de fichier au clavier et d'un chemin pour l'enregistrement
- Déplacer la caméra: l'utilisateur modifie le point de vue de la scène via une entrée clavier
- Déplacer une source de lumière: l'utilisateur modifie la position d'une source via un clic et un "drag" de gizmo

5 Modules du système

Le système peut être divisé en quatre modules:

- Module Core: la base du moteur de rendu
- Module GUI: l'interface et les événements associés
- Module IO: l'import de scène
- Module Lighting: système de shaders pour les calculs d'éclairage

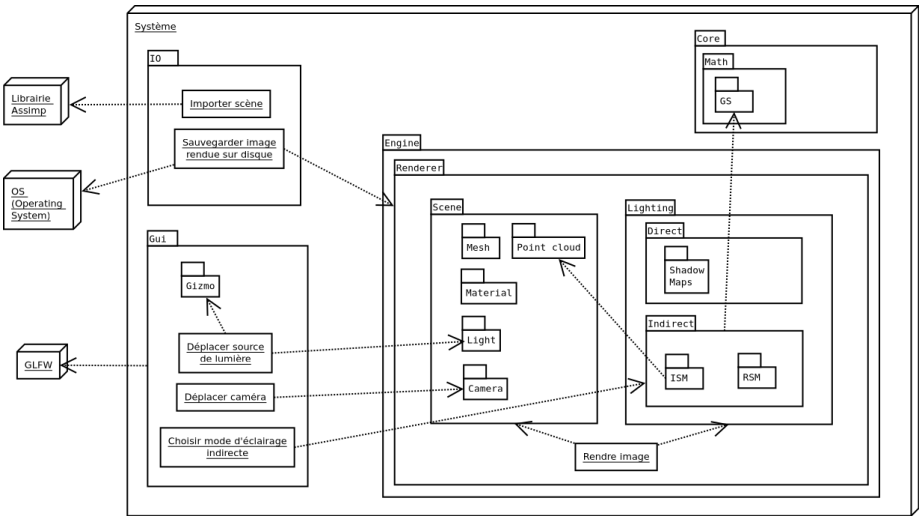

Figure 4: Découpage en modules

6 Planning prévisionnel et délais

6.1 Détail des modules

6.1.1 Module Core

Tâche	Description	Jours / Homme	Nombre de développeurs	Difficulté
Boucle de rendu	Ecriture de notre boucle de rendu de base.	1	1	
Algorithme de rendu	Implémentation de l'algorithme du *deferred shading*.	2	1	
Shader program	Création de shader program : vertex shader prenant en compte les diverses transformations liés aux objets de la scène, fragment shader simpliste utilisant l'albédo du matériau.	2	1	
Gestion des matériaux	Matériau simpliste : modèle de BRDF dans le fragment shader (rugosité, albedo, diffus, spéculaire).	3	1	
Gestion de textures	Gestion et importation de textures avec *stb_image*.	2	1	
Gestion des primitives d'éclairage	Structure de données pour les lumières : ponctuelles, directionnelles, spot, ...	3	1	
Gestion d'un nuage de points	Structure de donnée "nuage de point" pour l'échantillonnage des VPLs pour l'ISM.	3	1	
Gestion caméra	Gestion de caméras.	2	1	

Figure 5: Planification pour le module Core

6.1.2 Module GUI

Tâche	Description	Jours / Homme	Nombre de développeurs	Difficulté
Création fenêtre de rendu	Création de la fenêtre de rendu avec GLFW.	2	1	
Gestion entrées clavier	Création d'"event handler" lors d'entrées clavier typiquement pour les déplacements de caméras.	1	1	
Gestion entrées souris	Sur la fenêtre de rendu (changement de direction caméra, ...).	1	1	
Selection mode d'éclairage	Création d'une interface utilisateur pour la sélection des différents mode de rendu.	2	1	
Transformation objet	Permettre le déplacement ou la rotation d'objets (lumière ou mesh).	6	1	
Selection d'objet	Permet la sélection d'un objet pour modification des matrices de transformation.	5	1	
Sauvegarde sur disque	Permettre à l'utilisateur de sauvegarder l'image courante.	2	1	
Import de fichier scène	Importer un fichier de scène .	2	1	
Éditeur de lumière	Permettre d'éditer la couleur des lumières et la puissance pour mieux visualiser le résultat obtenu lors de comparaison de techniques d'éclairage indirect.	5	1	

Figure 6: Planification pour le module GUI

6.1.3 Module IO

Tâche	Description	Jours / Homme	Nombre de développeurs	Difficulté
Importer scène	Utilisation de la librairie externe Assimp pour importer tous types de fichier compatibles vers notre structure de scène.	5	1	
Sauvegarde image sur disque	Sauvegarde de l'image courante sur disque. Utilisation de write de *stb_image*.	5	1	

Figure 7: Planification pour le module IO

6.1.4 Module Lighting

Tâche	Description	Jours / Homme	Nombre de développeurs	Difficulté
"G-Buffer"	Utilisation d'un "G-Buffer" pour les cartes de profondeurs de chaque source lumineuse.	3	1	
Contribution	Évaluation de l'éclairage pour un fragment de l'image.	4	1	
Interpolation	Interpolation en espace écran.	4	1	
Opérations arithmétiques	Opérateurs arithmétiques pour les gaussiennes.	2	1	
Échantillonnage spatial	Phase de pré-calcul, création d'un nuage de points de la scène.	4	1	
Atlas de VPLs	Création d'une texture regroupant toutes les cartes de profondeurs des VPLs.	5	1	
Contribution	Calcul de la contribution des VPLs avec les gaussiennes sphériques.	5	1	

Figure 8: Planification pour le module Lighting

6.2 Diagramme de Gantt prévisionnel

Module	Core Engine	Gui	IO	Lighting
Couleur				

Figure 9: Code couleur pour le diagramme de Gantt ci-dessous

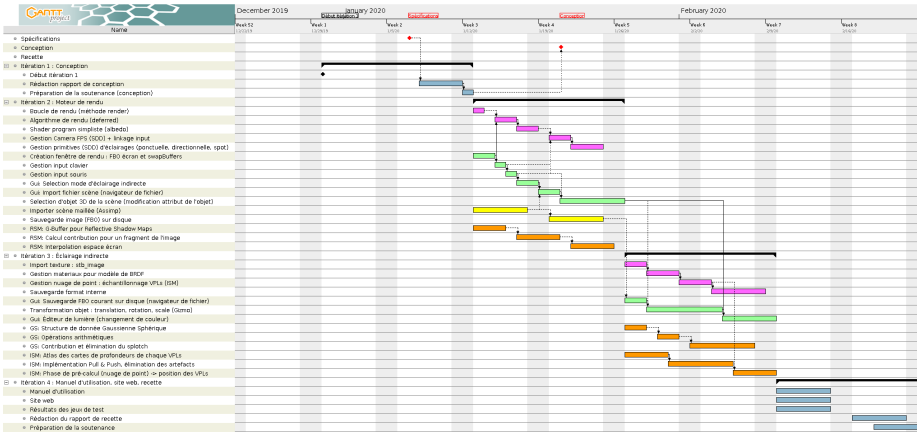

Figure 10: Diagramme de Gantt prévisionnel, n'hésitez pas à zoomer, beaucoup

7 Analyse des risques

Risque	Probabilité d'apparition	Impact	Prévention	Solution
Indisponibilité/abandon d'un membre de l'équipe au cours du projet	10%	Fort	Faire en sorte que sur le planning des tâches il y ait des tâches faites en binôme.	Les tâches en binôme deviendront des tâches pour une seule personne.
Base moteur instable	40%	Moyen	Prendre exemple sur moteur existant (code source libre) et tester rigoureusement le code produit.	On n'aura d'autres choix que de corriger les problèmes qui apparaissent à la volée.
Perte de temps sur le moteur alors que le coeur du sujet du chef d'oeuvre est l'éclairage indirecte	50%	Moyen	L'objectif est de faire une base moteur minimaliste afin de mieux se concentrer sur la partie "éclairage".	Faire une très bonne prévision des tâches et une conception détaillée précise évitant les questions lors des sprints.
Performance décevantes	90%	Faible	Utiliser des techniques d'optimisations.	Se référer aux implémentations existantes et utiliser des outils de mesures de performances et de "profiling".
Compatibilité multi OS	80%	Faible	Il faut que tous les membres de l'équipe puissent compiler sur son OS pour développer sereinement.	Remplacer les morceaux de code non portable.
Rendu réaliste non performant (éclairage indirect)	50%	Faible		Se référer aux implémentations existantes et utiliser des outils de mesures de performances et de "profiling".
Dépassement des délais	60%	Fort	Effectuer une planification temporelle assez large.	Abandonner des fonctionnalités optionnelles.
Sprint inefficace si on doit réviser des contrôles pour d'autres UEs	40%	Fort	Connaître les dates de contrôles à l'avance, ce qui n'est pas le cas ici.	Revoir la prévision afin de caler les itérations pour que les sprints de développement ne se trouve pas dans une phase de révision d'un CT.
Mauvaise estimation des délais (trop court)	20%	Moyen	Prévoir une marge dans l'estimation des délais dès le départ.	Reprendre le planification des délais. Impossibilité de prévoir des marges car on doit tout développer en peu d'itérations.
Manque de communication dans le groupe	10%	Moyen	Utiliser un bon outil de communication dès le départ.	Utiliser un autre logiciel de communication.

Figure 11: Analyse des risques

References

[1] Alexander Keller. Instant radiosity. Technical Report 287, Fachbereich Informatik, 1997.

[2] Yusuke Tokuyoshi. Virtual spherical gaussian lights for real-time glossy indirect illumination. *Computer Graphics Forum*, 34(7):89–98, 2015.

[3] Carsten Dachsbacher and Marc Stamminger. Reflective shadow maps. In *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*, I3D '05, pages 203–231, New York, NY, USA, 2005. ACM.

[4] MJP The Danger Zone. SG series, 2016.

[5] T. Ritschel, T. Grosch, M. H. Kim, H.-P. Seidel, C. Dachsbacher, and J. Kautz. Imperfect shadow maps for efficient computation of indirect illumination. *ACM Trans. Graph.*, 27(5):129:1–129:8, December 2008.