Écrasement de photons pour l'illumination globale

F. Lavignotte, M. Paulin

IRIT
Université Paul Sabatier
118, route de Narbonne
31062 Toulouse Cedex
lavignot,paulin@irit.fr

Résumé : *Dans cet article, une méthode basée image est présentée pour accélérer le calcul de l'éclairement global dans une scène virtuelle. L'éclairement est reconstruit après une phase de lancer de photons. Le principe est de reconstruire l'éclairement en écrasant les photons sur la surface à laquelle ils appartiennent. Cet écrasement est réalisé dans l'espace image. Par conséquent, la phase de reconstruction est indépendante de la complexité de la scène. Cette reconstruction par écrasement est accélérée par le matériel graphique ce qui permet d'avoir des temps de calcul réduit même pour un grand nombre de photons. Diverses solutions sont apportées pour que la reconstruction reste précise malgré les limites du matériel graphique. Enfin, nous présentons un lancer de rayon modifié pour prendre en compte notre méthode et qui permet de calculer efficacement une image de l'éclairement complet.*

Mots-clés : Illumination globale, Rendu basé image, Rendu accéléré

1 Introduction

Pour obtenir des images photo réalistes à partir de la description virtuelle d'une scène, il est nécessaire d'utiliser un modèle d'éclairage physiquement réaliste. De nombreux travaux ont porté sur ce domaine en commençant par les travaux de Goral et al. présentés dès 1984 [GTGB84], limités aux transferts diffus. Kajiya a posé les bases du modèle en proposant l'équation du rendu qui définit la luminance en un point [Kaj86]. Il a aussi proposé une méthode de résolution par des méthodes de type Monte Carlo qui consiste à échantillonner les chemins de transfert lumineux partant de l'oeil ou des lumières. Un échantillonnage adapté de ces chemins permet de prendre en compte de nombreux effets complexes de l'éclairage comme les caustiques, les ombres douces, les réflections sur une surface rugueuse, et l'éclairage diffus indirect [Laf96], [VG95] [VG97]. Cependant, les méthodes de type Monte Carlo, bien que très élégantes, ne sont pas encore sorties des applications de recherche du fait de leur très lente convergence et donc de la nécessité de tracer un grand nombre de chemins pour éliminer le bruit sur l'image.

Des méthodes multi-passes ont donc été proposées pour résoudre le problème d'efficacité des méthodes pures. L'équation du transfert lumineux est découpée alors en différentes parties, et chacune de ces parties est résolue avec différent algorithmes. L'éclairage direct et spéculaire est généralement pris en compte par un lancer de rayon classique, la radiosité peut être utilisée pour prendre en compte l'éclairage diffus indirect [CRMT91] ou le lancer de photons pour prendre en compte les caustiques [Hec90]. Le problème de ces travaux est de stocker leur contribution à l'éclairage sur des textures ou un maillage, et donc d'être dépendant de la complexité de la scène. D'autres travaux récents ont été proposés comme les cartes de photons [Jen96], ou les vecteurs d'éclairement [SP01] pour l'éclairage indirect. Ces travaux opèrent directement dans l'espace image et s'adapte donc mieux à des scènes complexes.

Dans cet article, nous proposons une méthode pour calculer directement une image de l'éclairement qui prend en compte l'éclairage diffus et les caustiques. Cette méthode est décomposé en deux passes, la première consiste en un lancer de photons, et la seconde en une reconstruction sur l'image de l'éclairement basé sur la technique d'écrasement de photons proposé initialement par Sturzlinger et Bastos [SB97]. Nous étendons cette technique à des scènes complexes et essayons de résoudre le problème de précision de la méthode limitée par les contraintes du matériel graphique. Dans un premier temps, nous posons le problème de manière à l'adapter au manque de précision des calculs quand on travaille directement sur le tampon de couleur. Puis, nous proposons une méthode pour réduire un artefact visuel important tel que le biais sur les bords des surfaces. Enfin, les différents détails de mise en oeuvre sur la génération actuelle de cartes graphiques sont présentés. Un des intérêts de cette méthode est de pouvoir gérer facilement un grand nombre de photons puisque l'accès aux photons est linéaire durant la reconstruction, on peut facilement le cacher sur le disque dur. L'autre intérêt est d'être de s'adapter plus facilement

aux scènes complexes. Finalement, nous présentons un algorithme basé sur un lancer de rayon adapté à l'utilisation de notre méthode de calcul de l'éclairement diffus basé image.

2 Écrasement de photons

2.1 Estimation de densité

Dans ce paragraphe, nous allons décrire rapidement le principe de reconstruction de l'éclairement à partir d'un ensemble de photons et de l'estimation de densité. La reconstruction est basée sur l'estimation de densité et cette théorie est utilisé dans de nombreux travaux : radiosité [Wal98], textures d'éclairement indirects [Mys97], textures de caustiques [Hec90] [GDW00], et cartes de photons [Jen96].

Le lancer de photons consiste à tracer des chemins lumineux aléatoires à partir des sources lumineuses. Le nom de photon est un peu trompeur puisque le but n'est pas de simuler le comportement corpusculaire de la lumière. Les photons sont en fait les sommets des chemins aléatoires générés depuis les sources lumineuses. Au final, une position ,un poids ou énergie et une direction sont associés à chaque photon.

Une fois le lancer de photons accompli, un ensemble de photons est obtenu. L'éclairement est alors reconstruit à partir de cet ensemble. Heckbert le premier a noté que c'était équivalent à un problème d'estimation de densité [Hec90]. En effet, la distribution des photons suit une densité de probabilité proportionnelle à l'éclairement :

$$p(x,\lambda) = \frac{E(x,\lambda)}{\int_S E(y,\lambda)dy} \tag{2.1}$$

L'intégration de l'éclairement peut être remplacée par $n\phi$ avec n le nombre total de photons et ϕ le poids transporté par un photon. Cette densité de probabilité est totalement inconnue, donc une estimation non paramétrique est utilisée comme l'estimateur de densité par noyau [Sil86].

La méthode d'estimation de densité par noyau construit une densité estimée $\hat{p}$ à partir d'un ensemble de n points $X_1, ..., X_n$ générés par p :

$$\hat{p}(x) = \frac{1}{nh^d}\sum_{i=1}^{n} K(\frac{x - X_i}{h}) \tag{2.2}$$

où d est la dimension de x, $K(x)$ est une fonction noyau unitaire, symétrique et généralement égale à zéro si $|x| > 1$, h est le paramètre de lissage.

Une propriété très intéressante de l'estimateur à noyau est le compromis entre la variance et le biais contrôlé par le paramètre de lissage. En effet, le biais est proportionnel à h^2 et la variance proportionnelle à nh^{-1} [Sil86].

Pour notre problème, les équations (2.1) et (2.2) sont combinés pour nous donner un estimateur pour l'éclairement à chaque point :

$$E(x,\lambda) = \frac{\phi}{h^2}\sum_{i=1}^{n} K(\frac{x - X_i}{h}) \tag{2.3}$$

2.2 Principe

Notre but est de reconstruire efficacement l'éclairement à chaque pixel en utilisant l'estimateur (2.3). Cet estimateur peut être vu comme la somme de la contribution de chaque photon X, la contribution d'un photon étant égale à :

$$c(X) = \frac{\phi}{h^2}K(\frac{x - X}{h}) \tag{2.4}$$

En fait, la contribution d'un photon est nulle si sa distance au point d'estimation est supérieure à h. La contribution non nulle d'un photon aux points d'une surface forme alors un disque de rayon h centré sur sa position X. L'estimation dans l'espace image revient donc à ajouter la contribution du photon à chaque pixel recouvert par la projection de son disque sur l'image. Cette projection de la contribution du photon est exacte si la surface est plane, pour une surface quelconque cela reste une approximation. Dans ce cas, la contribution est en quelque sorte

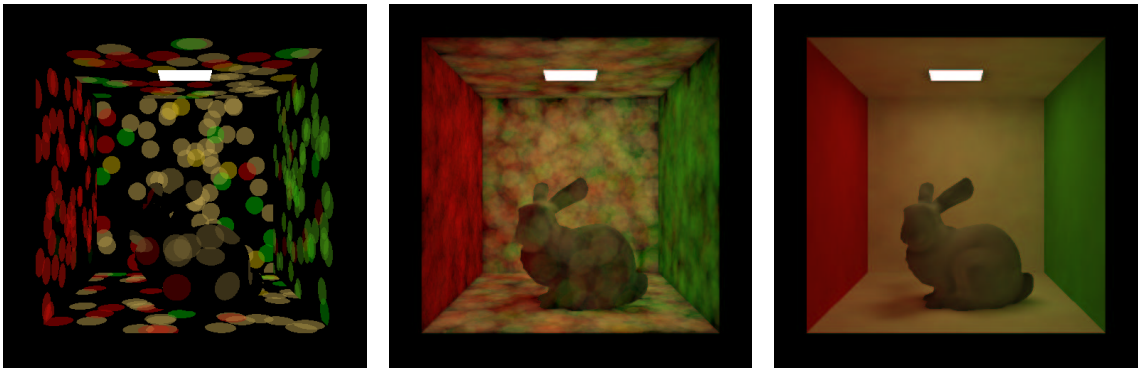

FIG. 1 – Écrasements de photons à 0.05% (Gauche), 1% (Milieu) et 100% (Droite).

écrasée sur la surface. Cette approximation est correcte si la courbure de la surface varie peu et si la taille du disque est peu importante, un exemple d'écrasement de photons, arrêté à différentes étapes, est montré sur la figure 1.

De plus, dans une scène avec plusieurs surfaces, la contribution du photon ne doit pas être ajouté à tous les pixels recouvert par le disque. En effet, la surface visible d'un pixel peut appartenir à une surface différente de la surface d'intersection du photon, donc la contribution est ajouté seulement si la surface visible du pixel correspond à la surface intersectée par le photon.

Le fait d'ajouter la contribution du photon au pixel revient donc à projeter un disque sur l'image, cela correspond à un processus de rastérisation et peut être effectué avec une libraire graphique comme OpenGL. La contribution d'un photon est alors rendue avec un quadrilatère texturé et coloré. La texture correspond à la fonction noyau discrétisée (qui a la forme d'un disque) et la couleur est égale au poids du photon divisé par h^2. Le résultat de leur modulation est alors égale à la contribution du photon. Le quadrilatère est rendu avec le mélange activé pour ajouter la contribution à chaque pixel recouvert par le quadrilatère. Il faut aussi éviter d'ajouter cette contribution aux pixels n'ayant pas la même surface visible, la mise en oeuvre de ce processus est décrite dans le paragraphe (3.2).

2.3 Précision des calculs

Le principe de l'écrasement de photons est assez simple mais une mise en oeuvre efficace sur une carte graphique accélérée pose un certain nombre de problèmes. Le tampon de couleur est utilisé pour ajouter la contribution des photons. Malheureusement, la précision du tampon de couleur est limité à huit bits par composante sur les cartes graphiques courantes. Les précédents travaux [SB97] travaillaient avec un tampon de couleur douze bits et des problèmes de quantification étaient présents sur les images. En effet, l'écrasement de photons nécessite une précision très importante, le nombre de contributions par pixel est de l'ordre de cinq cents à mille sur les scènes que nous avons testé.

Pour pouvoir utiliser des tampons avec une précision limitée un certain nombre de choix doivent être fait. Tout d'abord, nous allons choisir un noyau constant $K(x) = \frac{1}{\pi}$. Cette limitation n'est pas aussi réductrice qu'il n'y parait. Il a été démontré [Sil86] que l'efficacité de ce noyau par rapport au noyau optimal est de 0.93. C'est à dire que pour avoir une estimation de qualité comparable avec n points et le noyau optimal, il faut $n/0.93$ points avec un noyau constant.

Si on considère que le paramètre de lissage h et le poids ou l'énergie ϕ de chaque photon est constant, il nous suffit pour estimer l'éclairement du nombre de contributions N_{ij} en chaque pixel (i, j). L'éclairement est donc calculé par :

$$E(x_{ij}) = N_{ij} \frac{\phi}{\pi h^2} \tag{2.5}$$

Cette formule exige que le poids et le paramètre de lissage soit constant par photon. En fait puisque la reconstruction est effectuée par surface, le paramètre de lissage est choisi constant par surface. Il est choisi avec cette

FIG. 2 – Comparaison de deux images rendues sans réduction du biais sur les bords (Gauche) et avec (Droite) Noter les régions plus sombres sur les bords et les petites surfaces

heuristique souvent utilisé [SWH+95] [SB97] :

$$h = C\sqrt{\frac{A}{N}}$$

où A est l'aire de la surface, N le nombre de photons sur la surface et C une constante définie par l'utilisateur généralement comprise entre 20 et 30.

2.4 Biais sur les bords

L'estimateur de densité par noyau assume que le support de la probabilité de densité est infini. Dans notre cas, le support correspond à une surface qui peut être fermée. Du coup, des fuites d'énergies apparaissent du fait que l'on sous estime l'éclairement sur les bords des surfaces ouvertes. Cette sous estimation est aussi très apparente sur les surfaces dont l'aire est inférieure au support du noyau, voir Figure 2. Ce problème peut être résolu en divisant l'estimateur en x par $A_x/\pi h^2$ où A_x est l'aire de l'intersection entre le support de l'estimateur en x et la surface visible en x. Le support de l'estimateur est un disque de rayon h centré en x. La surface est projetée sur ce disque et l'aire d'intersection est alors calculée.

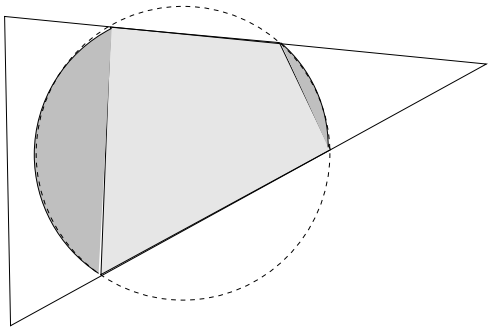

FIG. 3 – Calcul de l'aire d'intersection entre un triangle et un disque

Nous allons présenter ici le calcul de l'aire de la zone d'intersection entre en triangle et un disque. Le triangle est clippé par rapport au cercle. Le résultat est un polygone à n sommets. Si $n = 0$, soit le triangle est en dehors du cercle, soit il englobe le cercle. On vérifie alors si le centre du disque est dans le triangle, si c'est le cas alors l'aire d'intersection est égale à l'aire du disque, sinon elle est égale à zéro. Si $n = 2$, l'aire est alors l'aire d'intersection du demi plan formé par le segment. Si $n > 2$, l'aire d'intersection est alors l'aire de ce polygone (en gris clair sur la Figure 3) plus l'aire d'intersection des demi plans formés par les nouveaux segments (en gris foncé sur la

Figure 3). On appelle nouveaux segments les segments qui ne sont pas colinéaire aux trois segments initiaux du triangle. Ces nouveaux segments définissent chacun un demi plan.

A partir de l'aire d'intersection entre un triangle et un disque, on peut calculer l'aire d'intersection pour une surface représentée par un maillage triangulaire connecté. Il suffit de récupérer le triangle qui contient le point d'estimation, dans notre cas les pixels. L'aire d'intersection avec ce triangle est alors calculée, et en même temps pour chaque arête du triangle on calcule son intersection avec le disque. Il suffit après d'ajouter l'aire d'intersection des triangles qui partagent les arêtes qui intersectent le disque. Pour éviter les cycles sur le maillage, il faut marquer les triangles déjà visités.

Un autre intérêt de ce calcul est d'obtenir une meilleure approximation pour les surfaces non planes. En effet, le fait de représenter la contribution d'un photon reste une approximation pour ce genre de surface. Ainsi, l'aire du disque projeté sur la surface ne correspond pas réellement à πh^2. En calculant précisément l'aire couverte par le noyau pour réduire le biais sur les bords, l'erreur est réduite pour les surfaces non planes.

3 Mise en oeuvre

3.1 Poids constant

Pour l'instant, nous avons considéré que le poids de chaque photon est constant. Pour pouvoir réaliser cela, plusieurs conditions doivent être satisfaites. Tout d'abord, nous allons rappeler les équations qui permettent de calculer le poids ϕ_λ d'un photon généré d'une source lumineuse dans une direction ω pour une longueur d'onde donné :

$$\phi_\lambda = \frac{L_e(x,\omega,\lambda)\cos\theta}{Np_e(x,\omega)} \tag{3.1}$$

avec p_e la fonction de densité de probabilité utilisée pour sélectionner x et ω. Cette fonction doit être choisi égale à $\frac{L_e(x,\omega,\lambda)\cos\theta}{\Phi_\lambda^T}$, avec Φ_λ^T la puissance totale des sources lumineuses pour la longueur d'onde λ. Ainsi, le poids de chaque photon généré d'une source lumineuse est constant.

Puis, le photon est tracé dans la scène et à chaque intersection avec une surface, le photon est soit réfléchi, soit absorbé. S'il est réfléchi, son nouveau poids ϕ'_λ est :

$$\phi'_\lambda = \frac{f(y,\omega,\lambda)\cos\theta'}{p_r(\omega')}\phi_\lambda$$

avec f la FDRB en y, ω la direction du photon, ω' la nouvelle direction du photon sélectionnée avec la densité de probabilité p_r et θ' l'angle entre ω et ω'. Si f est réversible, p_r peut être choisi proportionnelle à $f(y,\omega,\lambda)\cos\theta'$ et le poids du photon reste constant après réflection. Il existe des modèles de FDRB réversible tel que le modèle de Phong modifié [LW94] ou le modèle de Ward [War92]. Donc en utilisant ces modèles de FDRB, un poids constant par longueur d'onde est obtenu pour chaque photon.

Il est donc possible d'obtenir un poids constant pour chaque longueur d'onde. Dans notre cas, nous travaillons en tri-chromatique RVB, il nous faudra alors générer les photons et reconstruire l'éclairement pour les trois composantes. Le fait de générer les photons de manière totalement indépendante pour chaque composante pose des problèmes. En effet, les trois composantes sont corrélées et donc les séparer artificiellement entraîne une plus grande variance.

La méthode classique est donc de choisir d'abord une composante, puis une source lumineuse par rapport à cette composante etc... La méthode que nous avons mise en oeuvre est différente, chaque photon va transporter un triplé au lieu d'un seul poids. Pour réaliser cela, une source lumineuse est choisie pour chaque composante mais à partir de la même variable aléatoire. Deux ou trois des sources lumineuses sélectionnées peuvent être identiques. Le photon sera alors lancé des sources lumineuses avec un triplé $(c0, c1, c2)$, les ci pouvant prendre comme valeur 0 ou 1. Si par exemple, le tirage des sources lumineuses nous a donné trois fois la même source lumineuse, le photon sera tracé de cette source avec le triplé (1,1,1). Évidemment, les chances d'avoir des sources lumineuses identiques est d'autant plus grande que la distribution d'émission des sources est identique. Après avoir rencontré une surface, le photon va être absorbé ou réfléchi selon la valeur de la FDRB en ce point. La probabilité d'absorption ou de réflexion est faite pour chaque composante mais toujours avec la même variable aléatoire, le triplé étant modifié

selon le résultat du tirage. Donc, au final, le poids du photon est représenté par un triplé binaire, pour avoir son poids il suffit multiplier ce masque avec le poids constant ϕ. Cette solution permet d'améliorer grandement la qualité de l'image et l'efficacité de la méthode sur les scènes testées où il est vrai les types de luminaires différents sont peu nombreux.

3.2 Rendu accéléré

Le rendu du photon doit être restreint à la partie visible de sa surface. Nous présentons ici la mise en oeuvre de ce comportement avec la librairie graphique OpenGL.

Tout d'abord, il nous faut obtenir un tampon d'identificateur, c'est à dire avoir pour chaque pixel de l'image un identificateur de la surface visible en ce pixel. Cela peut être obtenu en rendant chaque surface de la scène avec une couleur égale à son identificateur et en activant l'élimination des parties cachées avec le tampon de profondeur.

Une fois obtenu le tampon d'identificateur, il faut maintenant pouvoir réaliser une opération de comparaison entre l'identificateur de surface du photon et celui des pixels. Cette opération doit se dérouler lors des opérations sur les fragments, c'est à dire par pixel. En effet quand OpenGL rastérise une primitive géométrique, celle ci est décomposé en fragments. Un fragment correspond à un pixel recouvert par la projection de la primitive sur l'image. Une série d'opérations est réalisée sur ces fragments avant d'arriver au tampon de couleur. En particulier, le test de stencil peut être utilisé pour comparer une valeur référence avec la valeur stencil du pixel. Malheureusement, le tampon de stencil est limité à huit bits et permet donc d'effectuer une comparaison pour seulement 255 identificateurs de surfaces différents.

Une autre possibilité consiste à utiliser les opérations programmables sur les fragments. Différentes instructions sont fournies aux programmeurs pour modifier la couleur RVBA du fragment à partir de différentes entrées, telles que la couleur primaire interpolée ou la couleur correspondant à une unité de texture. En particulier, nous allons modifier la composante alpha pour utiliser le test alpha pour restreindre le rendu à la surface visible. Notre but est donc de générer un alpha égal à 1 si la surface visible et la surface du photon sont égale, et une valeur différente de 1 sinon.

Pour réaliser cela, il faut en entrée du programme sur les fragments l'identificateur de surface visible des pixels. Le tampon d'identificateur est donc chargé comme une texture. On veut pour chaque fragment l'identificateur du pixel correspondant, il faut donc que les coordonnées du texel soit les mêmes que les coordonnées du pixel. Pour cela, les coordonnées de texture pour chaque sommet du quadrilatère sont les mêmes que la position de ces sommets, et la matrice de transformation du repère scène vers l'image est chargée comme matrice de texture. En OpenGL, le repère image est défini entre -1 et 1, il faut donc appliquer une dernière transformation pour passer au repère de texture entre 0 et 1. L'identificateur de surface du photon est passé comme couleur des sommets. An niveau, du programme de fragments, on a donc en entrée l'identificateur de photon et du pixel. La comparaison de ces deux valeurs se fait en utilisant des instructions conditionnelles qui permettent de modifier la valeur d'un registre par rapport à un test sur un autre registre.

3.3 Dépassement de capacité

Le problème de reconstruction a été simplifié de manière à se ramener à un simple comptage de la projection des photons. Un problème se pose encore, ce nombre peut dépasser la précision du tampon de couleur. En effet, le tampon de couleur est limité sur les cartes graphiques courantes à huit bits par composante. Une première solution est d'ajouter périodiquement le contenu du tampon de couleur à un tampon plus précis. En OpenGL, le tampon d'accumulation permet effectivement de réaliser cela. Si le tampon d'accumulation n'est pas accéléré, il est toujours possible d'émuler son fonctionnement en lisant le tampon de couleur et en ajoutant son contenu à un tableau géré par l'application. Le problème reste à savoir quand il est nécessaire d'ajouter le contenu de tampon de couleur, et il n'est pas évident de définir une heuristique pour réaliser cela.

Une deuxième solution reste possible, utiliser le test et les opérations sur le stencil pour avoir au final 16 bits de précision. En OpenGL, le tampon de stencil est modifié par des opérations qui dépendent du résultat du test de stencil. Il existe différentes opérations et plus particulièrement une opération qui incrémente de un la valeur dans le tampon de stencil, et une autre qui met à zéro cette valeur. Ces opérations peuvent donc être configuré de manière à faire fonctionner les opérations de stencil comme un compteur des 8 bits de poids faible avec les 8 bits de poids fort dans le tampon de couleur :

– Test de stencil : la valeur du tampon de stencil doit être égal à 255
– Si le test rate : on incrémente la valeur du stencil
– Si le test réussit : on met à zéro la valeur

Si le test réussit, le fragment arrive jusqu'au tampon de couleur. Le tampon de couleur est alors incrémenté de 1, puisque le mélange est activé et que la couleur du fragment est 1. Au final après avoir rendu tous les photons, il suffit de récupérer dans deux tableaux le tampon de stencil et le tampon de couleur pour reconstruire le nombre de contributions à ce pixel. Le désavantage de cette méthode est de devoir rendre les photons séparément. En effet, il n'est plus possible de compter les trois composantes en même temps, seulement une valeur peut être compté. En utilisant la technique présentée dans le paragraphe 3.1, nous comptons d'abord les photons avec un triplé (1,1,1), puis (1,1,0), etc.. Le nombre de contributions est alors ajouté en prenant compte le triplé.

4 Lancer de rayon basé image

Une image de l'éclairement diffus est obtenue par la méthode présentée précédemment. A partir de l'éclairement, la luminance diffuse peut être obtenu facilement en combinant avec la partie diffuse de la fdrb. En utilisant la notation d'Heckbert [Hec90], l'image obtenu représente donc les chemins de type $L(D|S)^*DE$. Pour calculer tous les chemins lumineux possibles, nous devons ajouter à notre méthode un algorithme qui calcule les chemins $L(D|S)^*SE$.

En fait, nous allons calculé la luminance en trois parties :

1. L'éclairage direct $L(D|S)E$ est calculé avec un lancer de rayon stochastique [SWZ96]

2. L'éclairage indirect diffus $L(D|S)^+DE$ est calculé avec notre méthode d'écrasement de photons, en n'écrasant que les photons qui ont au moins un rebond.

3. L'éclairage indirect spéculaire $L(D|S)^+SE$ est calculé en échantillonnant les chemins de type $L(D|S)^*DS^+E$ avec un lancer de rayon basé image.

Les chemins de type $L(D|S)^*DS^+E$ sont échantillonnés en traçant un chemin spéculaire à partir de l'oeil S^+E, à chaque intersection avec une surface on récupère l'éclairement diffus $L(D|S)^*D$. L'éclairement diffus est calculé en écrasant les photons. Le problème est que nous avons besoin dans ce cas de l'éclairage diffus en divers points de la scène. Notre méthode ne peut calculer qu'une image ce qui ne couvre pas tous les points de la scène.

Nous avons décidé d'utiliser des images de profondeur multi couches communément appelées LDI [SGHS98] pour stocker l'éclairement. Un pixel d'une LDI est une liste d'échantillons de la scène correspondant aux point d'intersection des surfaces rencontrés par un rayon tracé à partir du centre du pixel. Comme dans [LR], trois LDI sont utilisées correspondant à une vue orthographique des trois directions orthogonales de la scène. Cette représentation est complète dans le sens que nous sommes assurés d'avoir un échantillon pour chaque surface de la scène quelque soit son orientation à condition que la taille de la surface soit supérieure à la résolution spatiale des LDI.

L'objectif est d'utiliser l'écrasement de photons pour reconstruire l'éclairement dans une LDI. Une première méthode est tout simplement d'écraser les photons pour chaque couche des LDI. En effet, une couche de LDI représente une image. Malheureusement, cette méthode n'est pas du tout efficace pour des scènes complexes. Il peut en effet y avoir jusqu'à une cinquantaine de couches de profondeur. De plus, ces couches consistent parfois en très peu de pixels, l'image résultante est donc pleine de trous.

Une méthode différente est donc proposée. Dans un premier temps, les LDI sont construites mais sans calculer l'éclairement, avec uniquement les informations géométriques telles que la profondeur. On marque tous les pixels du LDI comme étant non initialisé au niveau de l'éclairement. Lorsque le lancer de rayon interroge les LDI pour récupérer une valeur en un point de l'éclairement indirect, on regarde si les pixels correspondants aux points sont initialisés. Si ce n'est pas le cas, une image d'éclairement est calculé dans la direction incidente de l'interrogation. Cette image est alors reprojeté sur les LDI pour remplir la valeur d'éclairement des pixels du LDI. Ainsi, la cohérence spatiale des directions spéculaires est pris en compte et permet de diminuer grandement le coût par rapport à un calcul direct du LDI.

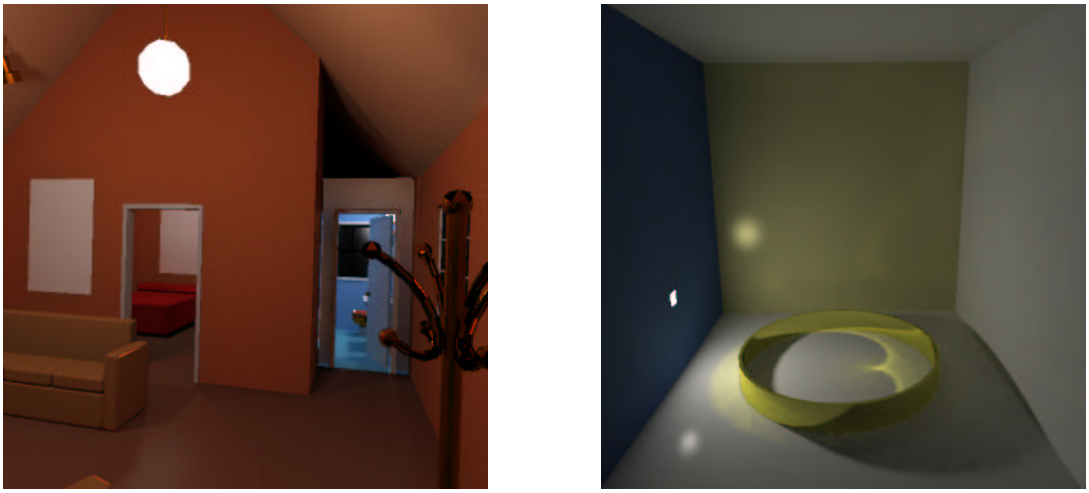

FIG. 4 – Ecrasement de photons combinés avec le lancer de rayon. Noter la réflection de la caustique sur l'anneau.

5 Résultats

Dans ce paragraphe, quelques résultats avec différentes scènes MGF sont présentées. Tous les résultats ont été obtenues sous Linux sur un AtlhonXP 1,6 GHz avec 512 Mo de RAM et une carte graphique GeForce4 Ti 4600.

– *Caustics* : une scène très simple avec des caustiques
– *Conference* : une salle de conférence, géométriquement complexe (environ 100 000 triangles) et avec un grand nombre de sources lumineuses étendues.
– *Cabin* : un chalet avec plusieurs pièces (environ 45 000 triangles)

Scène	LP	NP	EP1	EP2	EP3	RI
Caustics	7.02s	1830901	1.14s	4.44s	9.5s	0.6s
Conference	41.89s	1601516	0.96s	3.25s	6.61s	1.5s
Cabin	22.27s	1882184	0.75s	2.15s	4.52s	1.2s

TAB. 1 – Résultats sur trois scènes, LP : lancer de photons, NP : nombre total de photons, EP1, EP2, EP3 : écrasement de photons respectivement pour une image 256x256, 512x512 et 1024x1024 , RI : reconstruction de l'image 512x512

Le tableau 1 présente les temps de reconstruction de l'éclairement $L(D|S)^*DE$, pour un million de photons générés des sources lumineuses. Plusieurs observations peuvent être faites sur ces résulats. Tout d'abord, la phase de reconstruction d'image correspond à construction de l'irradiance à partir de la formule 2.5, et surtout de la correction du biais sur les bords. C'est cette correction qui prend le plus de temps puisque cela implique des calculs géométriques pour chaque pixel de l'image. En effet, la cohérence spatiale au niveau de l'image n'est actuellement pas pris en compte, c'est pour cette raison que les temps n'ont été montrés que pour une taille d'image puisque la complexité de cette phase varie linéairement avec le nombre de pixels. Enfin, la phase d'écrasement de photons est indépendante de la complexité de la scène et dépendante de la taille de l'image ce qui est un résultat tout à fait logique. Les différences au niveau des trois scènes s'expliquent par le fait que sur la scène simple l'image engloble toute la scène, donc tous les photons sont effectivement écrasés ce qui accroit le travail de la carte graphique. Globalement, la complexité de la phase d'écrasement de photons est proportionnelle aux nombres de photons fois la taille moyenne sur l'image de leur écrasement.

Ainsi sur le tableau 2, nous avons représenté les différents temps pour cent mille et un million de photons sur la même scène mais avec différentes valeurs de la constante C qui contrôle le paramêtre de lissage (voir paragraphe 2.3) donc la taille de l'écrasement des photons. En fait, la constante C est choisie généralement entre 20 et 30, en dessous l'image est trop bruitée et au dessus le résultat est trop lisse.

Enfin, dans le tableau 3, nous présentons quelques résultats pour le lancer de rayon basé image. La taille des LDIs est de 256x256 pour les trois scènes. Les LDI sont construites en utilisant un tracé de rayon modifié. La première

Constante	15	30	60
Temps (1e5)	0.77s	2.34s	8.47s
Temps (1e6)	2.04s	4.67s	10.65s

TAB. 2 – Temps pour différentes valeurs de la constante C

scène a été calculé en lancer de rayon avec 9 rayons pour les ombres et les réflections, les deux autres scènes avec 36 rayons pour les ombres et les réflections.

Scène	EP	EL	LR
Caustics	10.8s	18s	21.7s
Conference	39s	35s	95.8s
Cabin	23.76s	83s	325.73s

TAB. 3 – Résultat sur trois scènes avec le lancer de rayon basé image, EP : lancer de photons et écrasement de photons pour l'image de l'irradiance indirect, EL : écrasement de photons pour remplir le LDI, LR : lancer de rayon

6 Conclusion

Nous venons de présenter une nouvelle méthode basée image pour l'éclairement global. Le principal intérêt de notre méthode est de tirer parti de la puissance des cartes graphiques sans sacrifier la précision de la simulation. Pour l'instant, nous sommes encore partiellement dépendant de la complexité de la scène pour la résolution du biais sur les bords. Mais avec les nouvelles possibilités des prochaines cartes graphiques, une méthode qui travaille entièrement dans l'espace image est envisageable et souhaitable. Il sera aussi important de tirer parti du gain en flexibilité pour améliorer la qualité de la reconstruction en utilisant des méthodes adaptatives d'estimation de densité. Un autre point à approfondir est l'utilisation de LDI. Pour l'instant , nous nous servons de LDI uniquement pour stocker l'éclairement dans la scène. Il est aussi possible de tracer un rayon au sein d'une LDI pour récupérer un échantillon de la scène correspondant à l'intersection avec ce rayon. Cela pourrait accélérer le calcul des réflections brillantes qui ont besoin de moins de précision que les réflections de type mirroir.

Références

[CRMT91] Shenchang Eric Chen, Holly E. Rushmeier, Gavin Miller, and Douglass Turner. A Progressive Multi-Pass Method for Global Illumination. In *Computer Graphics (ACM SIGGRAPH '91 Proceedings)*, volume 25, pages 164–174, July 1991.

[GDW00] Xavier Granier, George Drettakis, and Bruce Walter. Fast global illumination including specular effects. In B. Peroche and H. Rushmeier, editors, *Rendering Techniques 2000 (Proceedings of the Eleventh Eurographics Workshop on Rendering)*, pages 47–58, New York, NY, 2000. Springer Wien.

[GTGB84] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modelling the Interaction of Light Between Diffuse Surfaces. In *Computer Graphics (ACM SIGGRAPH '84 Proceedings)*, volume 18, pages 212–222, July 1984.

[Hec90] Paul Heckbert. Adaptive Radiosity Textures for Bidirectional Ray Tracing. In *Computer Graphics (ACM SIGGRAPH '90 Proceedings)*, volume 24, pages 145–154, August 1990.

[Jen96] Henrik Wann Jensen. Global Illumination Using Photon Maps. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 21–30, New York, NY, 1996. Springer-Verlag/Wien.

[Kaj86] James T. Kajiya. The Rendering Equation. In *Computer Graphics (ACM SIGGRAPH '86 Proceedings)*, volume 20, pages 143–150, August 1986.

[Laf96] Eric Lafortune. *Mathematical Models and Monte Carlo Algorithms for Physically Based Rendering*. Ph.D. thesis, Leuven, Belgium, February 1996.

[LR] Dani Lischinski and Ari Rappoport. Image-based rendering for non-diffuse synthetic scenes. In *Rendering Techniques '98 (Proceedings of the Ninth Eurographics Workshop on Rendering)*, pages 301–314.

[LW94] Eric P. Lafortune and Yves D. Willems. Using the Modified Phong BRDF for Physically Based Rendering. Technical Report CW197, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium, November 1994.

[Mys97] Karol Myszkowski. Lighting reconstruction using fast and adaptive density estimation techniques. In Julie Dorsey and Philipp Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 251–262, New York, NY, 1997. Springer Wien. ISBN 3-211-83001-4.

[SB97] Wolfgang Sturzlinger and Rui Bastos. Interactive rendering of globally illuminated glossy scenes. In Julie Dorsey and Philipp Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 93–102, New York, NY, 1997. Springer Wien. ISBN 3-211-83001-4.

[SGHS98] Jonathan W. Shade, Steven J. Gortler, Li-Wei He, and Richard Szeliski. Layered depth images. *Computer Graphics*, 32(Annual Conference Series) :231–242, 1998.

[Sil86] B.W. Silverman. *Density estimation for statistics and data analysis*. Chapman and Hall, New York NY, 1986.

[SP01] X. Serpaggi and B. Peroche. An adaptive method for indirect illumination using light vectors. In *Computer Graphics Forum (Proceedings of Eurographics 2001)*, volume 20, pages C–278–C–287, September 2001.

[SWH+95] Peter Shirley, Bretton Wade, Philip M. Hubbard, David Zareski, Bruce Walter, and Donald P. Greenberg. Global Illumination via Density Estimation. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 219–230, New York, NY, 1995. Springer-Verlag.

[SWZ96] Peter Shirley, Changyaw Wang, and Kurt Zimmerman. Monte Carlo Techniques for Direct Lighting Calculations. *ACM Transactions on Graphics*, 15(1) :1–36, January 1996.

[VG95] Eric Veach and Leonidas J. Guibas. Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *Computer Graphics Proceedings, Annual Conference Series, 1995 (ACM SIGGRAPH '95 Proceedings)*, pages 419–428, 1995.

[VG97] Eric Veach and Leonidas J. Guibas. Metropolis light transport. In *Computer Graphics (ACM SIGGRAPH '97 Proceedings)*, volume 31, pages 65–76, 1997.

[Wal98] Bruce Johnathan Walter. *Density Estimation Techniques for Global Illumination*. PhD thesis, Program of Computer Graphics, Cornell University, Ithaca, NY, August 1998.

[War92] Gregory J. Ward. Measuring and Modeling Anisotropic Reflection. In *Computer Graphics (ACM SIGGRAPH '92 Proceedings)*, volume 26, pages 265–272, July 1992.