Vecteurs d'éclairement pour un observateur mobile

Light Vectors for a Moving Observer

R.Crespin B.Péroche

LIRIS, Laboratoire d'InfoRmatique en Image et Systèmes d'information
CNRS / INSA de Lyon / Université Lyon 1 / Université Lyon 2 / Ecole Centrale de Lyon

Bâtiment Nautibus
Université Claude Bernard Lyon 1
43, Boulevard du 11 novembre 1918
69622 Villeurbanne Cedex
rcrespin@liris.cnrs.fr, bperoche@liris.cnrs.fr

June 20, 2003

Résumé

Lors de la création d'images de synthèse, le rendu interactif est généralement réalisé avec des modèles d'illuminations simples, le rendu d'images réalistes étant trop lent pour être interactif. Dans cet article, nous proposons une méthode, basée sur le lancer de rayons, qui permet d'obtenir des images réalistes de façon interactive avec un éclairement global. Notre méthode allie le rendu à base d'images (le render cache) à un cache qui résume l'éclairement incident en une source virtuelle (méthode des vecteurs d'éclairement).

Mots Clef

Synthèse d'image, rendu, illumination globale.

Abstract

Interactive rendering is usually made with simple illumination models. High quality rendering is too slow to be interactive. In this paper, we extend the notion of light vector by computing it in five dimension and we try to obtain an interactive high quality rendering with global illumination by merging the notion of light vector and the render cache approach.

1 Introduction

En informatique graphique, le rendu est l'étape finale qui consiste à calculer la couleur des pixels à afficher. Cette étape est étudiée depuis la fin des années soixante, et on peut distinguer plusieurs types de rendu, suivant par exemple le type d'application visée :

- Le rendu *interactif*, le plus connu et utilisé, qui s'appuie le plus souvent sur les capacités des cartes graphiques et sur une représentation polygonale de la scène pour permettre un rendu en temps réel, avec des images qui peuvent être de qualité grâce à l'utilisation de textures diverses et variées (placage [17], perturbation de normales [16, 4], *environnement mapping* [3], ...). Cependant, depuis quelques années, des solutions basées sur le lancer de rayons permettent aussi le temps interactif sur des machines classiques avec des bases de données de plusieurs millions de polygones [24].

- Le rendu *photo-réaliste*, qui cherche à simuler la lumière et le comportement des matériaux vis-à-vis de celle-ci. Cette simulation permet d'obtenir certains effets comme la réfraction, les ombres portées, les inter-réflexions, les caustiques, les phénomènes spectraux, ...
  Pour obtenir ce genre d'images, de nombreuses solutions ont été proposées depuis une quinzaine d'années : le lancer de rayons [27], le *path tracing* [10], la carte de photons [9], le lancer de rayons bidirectionnel [22, 11], les vecteurs d'éclairement [28], la radiosité [18], ... Il faut noter que le point commun de toutes ces méthodes est la longueur des temps de calcul (de plusieurs minutes à plusieurs heures pour le calcul d'une scène un peu complexe).

- Le rendu *non photo-réaliste*, apparu plus récemment, qui s'attache à l'expressivité artistique et qui cherche à simuler des techniques artistiques telles que la gravure [15], la peinture (impressionnisme [14],

lavis [5],...) ou des réalisations de rendu technique [20].

L'objectif du travail présenté ici est de produire des images photo-réalistes, mais en se rapprochant du temps interactif. Pour cela, nous combinons deux approches assez différentes : celle du *render cache* et celle des *vecteurs d'éclairement*.
Cependant, le concept de *vecteur d'éclairement* a initialement été introduit pour des scènes statiques avec un point de vue fixe. Afin de passer outre cette limitation, nous proposons d'étendre la notion de *vecteur d'éclairement* au $5D$, en ajoutant un cache directionnel au cache spatial défini dans le modèle initial.
Dans la suite de ce document, nous présenterons les travaux antérieurs sur ce type de problème, en insistant sur les concepts sur lesquels repose notre travail, dans la section 2. Nous décrirons ensuite notre extension de la méthode des *vecteurs d'éclairement* dans la section 3 et nous expliquerons, dans la section 4, comment déterminer la validité des *vecteurs d'éclairement* pour la phase d'interpolation. Nous présenterons quelques résultats dans la section 5 avant de conclure dans la section 6.

2 Travaux antérieurs

Dans cette section, nous allons d'abord dresser un bref historique du rendu interactif photo-réaliste, puis nous présenterons les deux concepts sur lesquels notre travail s'appuie : le *render cache* et les *vecteurs d'éclairement*.

2.1 Historique

Les différents types de rendu ne sont pas exclusifs. En effet, le rendu interactif tend à devenir de plus en plus réaliste avec des méthodes utilisant la puissance des cartes graphiques, comme les multi-textures [17], les moteurs programmables [13] ou les *pixel shaders* [17]. En rendu photo-réaliste, différentes propositions [6] ont été faites pour accélérer les calculs, par exemple en utilisant des méthodes à base de cache [12, 1, 23] ou en utilisant des optimisations liées au matériel [24].

2.2 Le ¨render cache¨

Le *render cache* [23] est une méthode dérivée du rendu à base d'images (ou Image Based Rendering [19]) qui utilise l'image comme cache de luminance. À part la première, chaque image est déterminée en utilisant l'image calculée à l'itération précédente par une méthode de reprojection.
Le *render cache* peut être interfacé à n'importe quel moteur graphique capable de calculer séparément des pixels. Dans notre cas, nous utilisons un logiciel de lancer de rayons, comme dans la méthode originelle. L'architecture du *render cache*, montrée sur la Figure 1, permet des interactions asynchrones entre le *render cache* et un logiciel de lancer de rayons.

Lors d'une édition de la scène, l'interface envoie les ordres au *render cache*. Par exemple, si l'utilisateur change

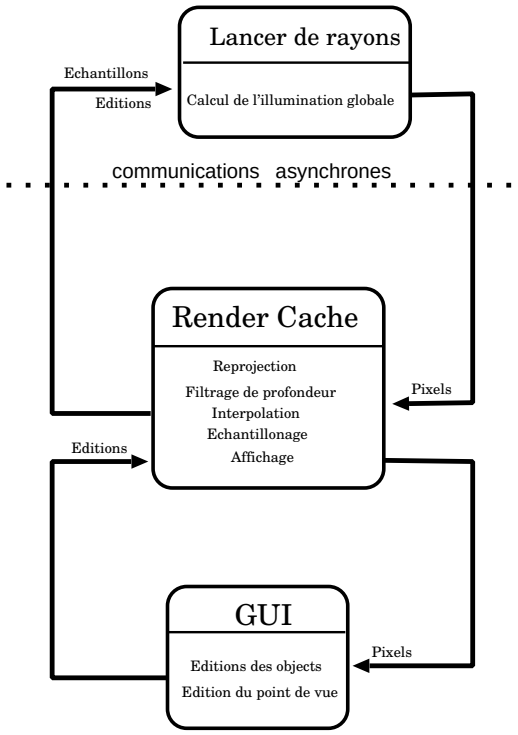

Figure 1: Structure et interface du *render cache*

le point de vue, l'interface graphique (notée GUI sur la Figure 1) envoie les modifications du point de vue au *render cache* qui utilise l'image précédemment calculée comme base pour calculer la nouvelle image.
L'image de l'itération précédente est d'abord reprojetée selon le nouveau point de vue ; ensuite un filtre de profondeur, ou *depth culling*, est appliqué pour éliminer les points doublement projetés et les points cachés.
Simultanément, une *image de priorités* est construite. Après un seuillage et une diffusion d'erreur, l'image de priorités permet de choisir les échantillons que le lancer de rayons recalculera. Cela permet de choisir les pixels les plus pertinents à calculer, étant donné que l'on veut en calculer un nombre limité.
Ensuite le *render cache* continue de boucler en attendant de nouvelles éditions ; s'il n'y en a aucune, l'image sera progressivement recalculée intégralement. On peut noter que la technique du *render cache* privilégie l'interactivité au détriment de la qualité des images. En effet, le nombre de pixels qui peuvent être recalculés entre deux affichages successifs dépend de la puissance de la machine utilisée ; il est en général relativement faible. La nouvelle image obtenue par reprojection et calcul d'un nombre limité de pixels peut posséder des artefacts.
Pour de plus amples informations le lecteur peut se reporter aux articles de Walter *et al.* [23, 25].

Dans ce travail, nous allons utiliser l'approche du *render cache* pour obtenir des images réalistes à une vitesse interactive. Outre l'envoi de requêtes pour chaque pixel, nous

utiliserons les *vecteurs d'éclairement* sous forme hiérarchique. Si certaines conditions sont remplies, on peut arriver à calculer une zone de plusieurs pixels de l'image grâce à une simple interpolation vectorielle, ce qui permet de réduire le temps nécessaire au calcul des pixels ou encore d'en calculer d'avantage.

2.3 Définition des vecteurs d'éclairement

Le *vecteur d'éclairement* (VE) est un concept introduit par Zaninetti *et al.* [28] qui a pour base les travaux de Ward et Heckbert [26]. L'idée principale est de stocker les luminances dans un tampon. Ainsi, au lieu de calculer la luminance en tout point, les *vecteurs d'éclairement* déjà calculés peuvent être interpolés pour produire un nouveau *vecteur d'éclairement*, quand certains critères sont respectés.
En fait, en chaque point, cela revient à remplacer l'éclairement global provenant de la scène par une source lumineuse virtuelle ; ainsi un *vecteur d'éclairement* est défini par une direction, calculée grâce à l'équation (2.2), et une magnitude, calculée grâce à l'équation (2.3).

L'équation (2.1), dérivée de celle de Kajiya [10], indique que la luminance réfléchie en un point x de la scène dans une direction $\overrightarrow{\omega_r}$ dépend des luminances incidentes L_i selon toutes les directions possibles $\overrightarrow{\omega_i}$, de la $BRDF$ f_r associée aux directions $\overrightarrow{\omega_i}$ et $\overrightarrow{\omega_r}$, et de la luminance émise L_e lorsque l'objet est une source lumineuse.

$$L_r(x,\overrightarrow{\omega_r}) = L_e(x,\overrightarrow{\omega_r}) + \int_{\Omega_i} f_r(x,\overrightarrow{\omega_i} \rightarrow \overrightarrow{\omega_r}) L_i(x,\overrightarrow{\omega_i}) \cos\theta_i d\omega_i \quad (2.1)$$

En partant de l'équation (2.1), on peut définir les composantes du *vecteur d'éclairement*.

$$\overrightarrow{D} = \int_{\Omega_i} L_i(x,\overrightarrow{\omega_i}) \overrightarrow{\omega_i} d\sigma(\overrightarrow{\omega_i}) \quad (2.2)$$

$$M = \frac{\int_{\Omega_i} f_r(x,\overrightarrow{\omega_i} \rightarrow \overrightarrow{\omega_r}) L_i(x,\overrightarrow{\omega_i}) \cos\theta_i d\omega_i}{f_r(x,\overrightarrow{D} \rightarrow \overrightarrow{\omega_r}) \cos(\overrightarrow{n},\overrightarrow{D})} \quad (2.3)$$

où $d\sigma(\overrightarrow{\omega_i})$ est la valeur de l'angle solide associé à la direction incidente $\overrightarrow{\omega_i}$.
On remarquera le lien avec la $BRDF$, dans l'équation (2.3), qui induit une dépendance par rapport au point de vue.

L'équation (2.1) peut être décomposée en cinq composantes indépendantes :

$$L_r(x,\overrightarrow{\omega_r}) = L_e(x,\overrightarrow{\omega_r}) + L_{spec}(x,\overrightarrow{\omega_r}) + L_{dir}(x,\overrightarrow{\omega_r}) + L_{ind}(x,\overrightarrow{\omega_r}) + L_{caust}(x,\overrightarrow{\omega_r}) \quad (2.4)$$

où $L_e(x,\overrightarrow{\omega_r})$ est la luminance émise, $L_{spec}(x,\overrightarrow{\omega_r})$ la composante spéculaire, $L_{dir}(x,\overrightarrow{\omega_r})$ la composante directe, $L_{ind}(x,\overrightarrow{\omega_r})$ la composante indirecte et $L_{caust}(x,\overrightarrow{\omega_r})$ la composante caustique.

Cette décomposition permet de traiter spécifiquement chaque composante. En effet, chaque composante ne se comporte pas de la même façon : par exemple, l'éclairement indirect varie peu alors que l'éclairement direct a des variations plus importantes, voire des discontinuités. On peut donc optimiser le traitement pour chaque cas.

Le terme $L_e(x,\overrightarrow{\omega_r})$, s'il existe, est défini par les propriétés de l'objet. Le terme $L_{spec}(x,\overrightarrow{\omega_r})$ présente de grandes variations et on n'aurait aucun avantage à essayer d'interpoler ses valeurs ; il est donc calculé par un lancer de rayons classique.
Il reste donc trois composantes lumineuses, qui conduisent à trois sortes de *vecteurs d'éclairement* :

- les *vecteurs d'éclairement directs (VED)*, pour l'éclairement arrivant directement des sources lumineuses;
- les *vecteurs d'éclairement indirects (VEI)*, pour l'éclairement arrivant en un point après au moins une réflexion sur un objet non spéculaire;
- les *vecteurs d'éclairement caustiques (VEC)*, pour l'éclairement arrivant sur les objets non spéculaires après avoir eu au moins une interaction avec des surfaces non lambertiennes.

Cette décomposition est illustrée sur la Figure 2.

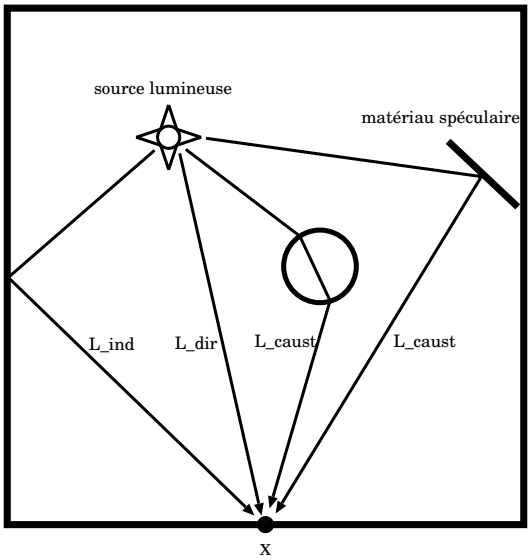

Figure 2: Différentes composantes lumineuses

Le calcul de l'éclairement indirect en un point est très coûteux (c'est en général une méthode de Monte-Carlo

qui est utilisée). Ainsi, pour bien tester notre idée, nous avons décidé de nous intéresser dans un premier temps à l'éclairement indirect. Bien entendu, du fait de l'indépendance des trois composantes de l'éclairement, nous appliquerons la même méthode aux deux autres composantes par la suite.

2.4 Algorithme de calcul d'une image

Lors du calcul d'une image en utilisant les *vecteurs d'éclairement*, l'algorithme utilisé est celui du lancer de rayons originel [27] pour les composantes de l'équation (2.4) qui ne donnent pas lieu au calcul de *vecteurs d'éclairement*.
Lorsque la composante directe, indirecte ou caustique doit être calculée en un point x, les *vecteurs d'éclairement* situés dans un voisinage du point x sont recherchés parmi ceux qui ont été stockés. Si leur nombre est suffisant et que certains critères sont remplis, une interpolation vectorielle est effectuée.
Lors de la phase d'initialisation de l'algorithme, il est nécessaire de calculer un germe de *vecteurs d'éclairement*. En effet, lors du calcul d'une image, les pixels sont calculés ligne par ligne, et les *vecteurs d'éclairement* utilisés pour l'interpolation ne seraient situés qu'au dessus de la ligne courante, ce qui introduirait un biais systématique dans le calcul. La création d'un germe initial aléatoirement réparti sur toute l'image permet d'éviter ce problème. L'algorithme de calcul d'une image en utilisant la notion de *vecteur d'éclairement* est présenté sur la Figure 3.

Création d'un germe
Pour *chaque pixel* **faire**
Générer un rayon primaire
Calculer l'intersection du rayon primaire avec la scène
Si *un objet est intersecté* **alors**
Déterminer le point d'intersection x le plus proche
{
Si *il n'y a pas de VE calculé en x* **alors**
Recherche des VE proches de x
Si *le nombre de VE est suffi sant*
Et Si *les critères d'interpolation sont remplis* **alors**
Interpoler
Fin Si
Sinon *Calculer le VE au point x*
Fin Si
Calculer la luminance en x
}
Fin Pour

Figure 3: Algorithme de calcul d'une image avec les VE

2.5 Principe de calcul des VEI

Si on discrétise l'espace des directions, la partie indirecte de l'équation(2.1) peut s'écrire :

$$L_{ind}(x,\overrightarrow{\omega_r}) \approx \frac{2\pi}{MN}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1}[\rho_d(x,\overrightarrow{\omega_{j,k}}\rightarrow\overrightarrow{\omega_r}) L_i(x,\theta_j,\phi_k)\cos\theta_j] \quad (2.5)$$

où M et N correspondent au nombre de découpages réguliers azimutaux et zénithaux de l'hémisphère centrée en x, où (θ_j,ϕ_k) sont les angles définissant le vecteur $\overrightarrow{\omega_{j,k}}$, et où ρ_d représente la partie diffuse de la $BRDF$.
Afin de récolter l'énergie incidente, on lance un rayon par cellule de l'hémisphère. Après avoir récupéré la luminance pour chaque cellule, on peut déduire la magnitude moyenne M_0 de la luminance reçue par le point x, ce qui permet d'obtenir les caractéristiques du VEI.

Chaque *vecteur d'éclairement* est associé à un point de l'espace et à une une direction de vue. Nous étudierons comment éliminer cette contrainte, en étendant la notion de *vecteur d'éclairement* au $5D$, dans la section 3.
Lors de l'évaluation de l'équation (2.5), on peut calculer un gradient d'éclairement.
Soit l'éclairement en un point x

$$E(x)=\int_{\Omega_i} L_i(x,\overrightarrow{\omega_i})\cos\theta_i d\omega_i$$

On discrétise cette équation et on la dérive au voisinage du point où un VEI a été calculé. En notant $d = x$, y, ou z, on obtient :

$$\begin{aligned}\frac{\partial E}{\partial d} &= \frac{2\pi}{MN}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1}\left[\frac{\partial(L_i(\theta_j,\phi_k)\cos\theta_j)}{\partial d}\right]\\ &= \frac{2\pi}{MN}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1}\left[\frac{\partial L_i(\theta_j,\phi_k)}{\partial d}\cos\theta_j + L_i(\theta_j,\phi_k)\frac{\partial\cos\theta_j}{\partial d}\right]\end{aligned}$$

Comme le terme $\frac{\partial L_i(\theta_j,\phi_k)}{\partial d}$ n'est pas calculable (à cause de la récursivité), on suppose que le déplacement au voisinage du point x est suffisamment faible pour pouvoir admettre que $\frac{\partial L_i(\theta_j,\phi_k)}{\partial d}=0$. Ainsi, on obtient :

$$\frac{\partial E}{\partial d}\approx\frac{2\pi}{MN}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1}-L_i(\theta_j,\phi_k)\sin\theta_j\frac{\partial\theta_j}{\partial d} \quad (2.6)$$

Ce gradient, même s'il n'est pas juste mathématiquement, permet de connaître le degré de perturbation de la zone considérée. Il sera utilisé lors du calcul des critères de validité des VEI, dans la section 4.2.

3 Extension multi-directionnelle

Dans cette section, nous allons expliquer comment nous allons étendre la notion de *vecteur d'éclairement* au cas où le point de vue bouge.

3.1 Motivations

Initialement, les *vecteurs d'éclairement* sont liés à une position de l'observateur. Dans le cas dynamique, où l'on peut modifier le point de vue, nous avons besoin d'étendre la notion de *vecteur d'éclairement*.
L'idée de base est d'utiliser un double cache : un pour les positions des points et l'autre pour les directions. Ainsi, le calcul d'un VEI sera effectué en réalisant une interpolation spatiale et une interpolation directionnelle.
Lors de la phase d'initialisation de l'algorithme de calcul d'une image suivant la méthode des *vecteurs d'éclairement*, il faudra calculer le germe pour plusieurs directions prédéfinies pour chaque point du germe de la méthode initiale.

3.2 Approche kd-arbre×icosaèdre

Le système de double cache permet de séparer la composante spatiale et la composante directionnelle afin d'optimiser le stockage et les recherches.
La composante spatiale, qui s'occupe des positions des points, est stockée dans un 3d-arbre [2].
La composante directionnelle est stockée dans un icosaèdre récursif. Ce polyèdre régulier comporte 20 faces triangulaires. Sur chaque face, on stocke les *vecteurs d'éclairement indirect* dont la direction appartient à l'angle solide délimité par la face. Lorsque le nombre de *vecteurs d'éclairement* stockés sur une face dépasse un seuil donné, on remplace la face par quatre nouvelles faces, en considérant le point de la sphère correspondant au milieu de chaque arête, comme indiqué sur la Figure 4. Cette façon de subdiviser permet un découpage adaptatif rapide de la sphère unitaire.

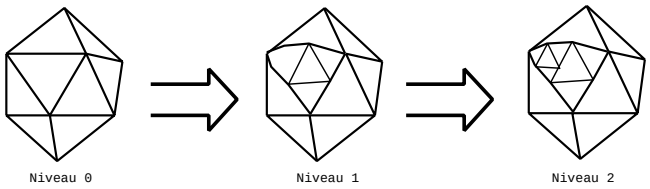

Figure 4: Subdivision d'une face d'icosaèdre

3.3 Résultats

Pour tester notre structure, une série d'expériences a été réalisée.
Des requêtes de recherche ont été lancées avec différents ensembles de points et de directions, tout d'abord en fixant le nombre de directions, puis celui des points, enfin en faisant varier les deux paramètres.
Le nombre de direction a été testé jusqu'à 1000 et le nombre de points jusqu'à 100 000.
Le temps d'exécution des requêtes s'est révélé relativement indépendant du nombre de points : il faut entre 10 et 15 millisecondes pour chaque requête. Cela peut s'expliquer car on descend peu dans la hiérarchie des structures.

4 Critères de validité

Dans les deux méthodes, le *render cache* et les *vecteurs d'éclairement*, un cache est utilisé pour éviter des calculs inutiles. En contre partie, il est nécessaire de connaître les conditions de validité pour pouvoir utiliser les caches sans commettre trop d'erreurs d'imprécision dans les calculs.
Dans cette section, nous allons d'abord rappeler les critères heuristiques suggérés pour le *render cache* et les *vecteurs d'éclairement*. Nous étudierons ensuite la façon dont nous les avons combinés dans notre nouvelle approche.

4.1 Critères pour le Render Cache

Dans l'article de Walter *et al.* [23], le critère principal pour déterminer la priorité est l'âge du pixel. En effet, plus un pixel est vieux, plus il a de chance d'être erroné. Ensuite un algorithme de diffusion d'erreur [7] est utilisé et permet d'obtenir une sorte de *lissage*.
La priorité d'un point absent du cache est calculée durant la phase d'interpolation. Elle dépend du nombre de points présents dans le cache au voisinage du point manquant : moins il y de voisins, plus la priorité sera élevée. Ceci permet de se focaliser sur des zones de taille réduite, afin d'éviter d'avoir des pixels erronés isolés, car l'oeil est sensible aux hautes fréquences.

4.2 Critères pour les VEI

Lorsque que l'on souhaite calculer la luminance en un point A et pour une direction de vue $\overrightarrow{\omega}$, on a besoin d'un VEI en ce point et pour cette direction. Si aucun vecteur n'a déjà été calculé et stocké en ce point et pour cette direction, on examine tous les VEI déjà calculés au voisinage du point A et de la direction $\overrightarrow{\omega}$, puis on vérifie si certains critères sont remplis selon la direction ou la position du point.

Critères spatiaux.
Pour être valide en un point A, un VEI calculé en un point B doit satisfaire les conditions suivantes :

1. $\|\overrightarrow{AB}\|$ est inférieure à la distance d_{moy_B} entre B et les autres objets de la scène, ce qui permet de limiter les fuites de lumière en adaptant la taille de la zone de validité.
2. La variation d'éclairement doit être inférieure à un seuil fixé E_B, ce qui permet un certain contrôle sur le lissage produit par l'interpolation.
3. La courbure de la surface doit être inférieure à un seuil fixé S_{courb}, car lors du calcul des VEI, seule la partie avant de la surface intervient et l'autre partie peut avoir une luminance très différente de celle de la partie avant.

Si un VEI est considéré comme valide, il est pondéré selon la formule suivante :

$$p_B = \left(1 - \frac{\|\overrightarrow{AB}\|}{d_{moy_B}}\right) \quad (4.1)$$

$$\cdot \frac{1}{1 + \frac{\|\overrightarrow{AB}\|.\Delta E}{E_B}} \quad (4.2)$$

$$\cdot \frac{\cos(\overrightarrow{N_A}, \overrightarrow{N_B}) - S_{courb}}{1 - S_{courb}} \quad (4.3)$$

Critères directionnels.
Lorsque l'on souhaite calculer la luminance en un point x pour une direction de vue $\overrightarrow{\omega}$, on cherche d'abord à quelle face de l'icosaèdre récursif cette direction appartient. Ensuite on détermine les directions valides parmi celles que l'on a trouvées.
Un VEI calculé pour une direction $\overrightarrow{\sigma}$ est valide pour une direction $\overrightarrow{\omega}$ si la différence entre $\overrightarrow{\sigma}$ et $\overrightarrow{\omega}$ est inférieure à un seuil arbitraire S_{dir}. Ce seuil permet de déterminer un poids directionnel $p_\sigma = \frac{\cos(\overrightarrow{\omega}, \overrightarrow{\sigma}) - S_{dir}}{1 - S_{dir}}$ pour la direction $\overrightarrow{\sigma}$.

Association des critères.
On fait le produit du poids directionnel et du poids spatial pour déterminer un poids global, noté p et associé au point B et à la direction $\overrightarrow{\sigma}$. Notons que $0 \leq p \leq 1$ et que l'interpolation est précise quand p est proche de 1.

Si le nombre de VEI valides est inférieur à un seuil donné, alors un nouveau VEI est calculé au point A et pour la direction $\overrightarrow{\omega}$ (et stocké dans la structure de données). Sinon, un VEI est calculé en faisant une interpolation pondérée des VEI valides trouvés.

4.3 Priorité des VEI 5D

La priorité d'un point du *render cache* donné est définie à partir des critères des VEI de la manière suivante :
durant la phase d'interpolation conduisant à un nouveau VEI, on associe au VEI un poids P, obtenu en calculant une moyenne normalisée des poids de chacun des VEI pris en compte lors de l'interpolation. La priorité associée au pixel donné est le produit de la priorité du point du *render cache*, liée à l'âge, par l'inverse du poids, $1/P$. Ainsi, une faible faible valeur de P augmente la priorité et le pixel aura plus de chance d'être recalculé par le lancer de rayons.

5 Résultats

Notre implémentation a été réalisée sur un ordinateur personnel doté d'un processeur Athlon à 1.2 GHz avec 512 Mo de mémoire, sans aucune accélération graphique matérielle. Nous avons généré des images de résolution 512x512 avec les scènes suivantes :

Scène A : Cornell Box 1. C'est une pièce contenant deux parallélépipèdes rectangles, composés de matériaux diffus et avec une source sphérique.

Scène B : Cornell Box 2. Scène semblable à la scène A mais avec un des parallélépipèdes rectangles composé d'un matériau brillant.

Scène C : Sphere Flake [8]. C'est une réunion récursive de sphères, composées de matériaux transparents, dans une pièce et avec une source surfacique.

Nous avons comparé les temps de rendu entre le lancer de rayons seul (LR), le lancer de rayons avec le *render cache* (LR + RC) et notre méthode (RC + VEI 5D). Actuellement, seul le point de vue change.

Comme nous pouvons le voir dans le tableau 1, nous obtenons avec notre méthode un gain de temps d'environ 66% pour la Scène A, d'environ 55% pour la scène B et d'environ 34% pour la scène C par rapport à l'utilisation du *render cache* seul. Le nombre de pixels à recalculer par le *render cache* est d'environ 21000 soit 8% de l'image.
Les artefacts liés au *render cache* restent présents (cf. les pixels noirs présents en haut des images, qui correspondent à des pixels non affichés dans l'image précédente).

	LR	LR + RC	5D VEI + RC
Cornell Box 1			
première image	46s	46s	47s
images suivantes	46s	13s	4.4s
Cornell Box 2			
première image	3mn 20s	3mn 20s	3min 21s
images suivantes	3min 20s	11s	4.96s
Sphere Flake			
première image	8mn 6s	8mn 6s	8mn 7s
images suivantes	8mn 6s	16.2s	10.7s

Table 1: Comparaison des méthodes

6 Conclusion

Cette méthode est basée sur la combinaison du *render cache* (dont l'objectif est l'interactivité) et des *vecteurs d'éclairement* (dont l'objectif est la réalisation d'images photo-réalistes). Les résultats présentés ici sont les premiers que nous ayons obtenus. Ils nous paraissent réellement prometteurs, et nous souhaitons continuer à travailler sur ce thème pour l'étendre et l'améliorer, l'objectif du photo-réalisme en temps interactif n'étant pas encore atteint.
Parmi les points qui pourraient être améliorés, nous pensons en particulier à :

- l'extension de la méthode aux vecteurs d'éclairement direct et caustiques ;
- l'amélioration du calcul de la priorité des pixels ;

- le remplacement du *render cache* par le *shading cache* [21] ;
- la prise en compte des objets mobiles.

References

[1] Kavita Bala and Julie Dorsey and Seth Teller. Interactive Ray Traced Scene Editing using Ray Segment Tree, *Rendering Techniques '99*, pp 31–44, 1999.

[2] J. L. Bentley K-d Trees for Semidynamic Point Sets, *Proceedings of the 6th Annual Symposium on Computational Geometry (SCG '90)*, pp 187–197, 1990.

[3] James F. Blinn and Martin E. Newell. Texture and Reflection in Computer Generated Images, *Communications of the ACM*, vol 19, pp 542–547, 1976.

[4] J. F. Blinn. Simulation of wrinkled surfaces, *Computer Graphics*, vol 12, pp 286–292, 1978.

[5] Cassidy J. Curtis and Sean E. Anderson and Joshua E.Seims and Kurt W. Fleischer and David H. Salesin. Computer-Generated Watercolor, *Computer Graphics*, vol 31, pp 407–414, 1997.

[6] Cyrille Damez and Kirill Dmitriev and Karol Myszkowski. State of the Art in Global Illumination for Interactive Applications and High-Quality Animations, *Computer Graphics Forum*, vol 22, pp 55–77, 2003.

[7] R. W. Floyd and L. Steinberg. An adaptive algorithm for spatial grey scale, *Proc. Soc. Inf. Display*, vol 17, pp 75–77, 1976.

[8] Eric Haines. Standard Procedural Databases, *http://info.acm.org/pubs/tog/resources/SPD/overview.html*.

[9] Henrik Wann Jensen. Global Illumination using Photon Maps, *Eurographics Workshop on Rendering*, pp 21–30, 1996.

[10] James T. Kajiya. The Rendering Equation, *Computer Graphics*, vol 20, pp 143–150 , 1986.

[11] Eric P. Lafortune and Yves D. Willems. Bidirectional Path Tracing, *Proceedings of Compugraphics '93*, pp 145–153, 1993.

[12] Gregory Ward Larson and Maryann Simmons. The Holodeck interactive ray cache, *Computer Graphics*, vol 33, pp 246–246, 1999.

[13] Erik Lindholm and Mark J. Kilgard and Henry Moreton. A User-Programmable Vertex Engine, *Computer Graphics*, vol 35, pp 149–158, 2001.

[14] Peter Litwinowicz. Processing Images and Video for an Impressionist Effect, *Computer Graphics*, vol 31, pp 07–414, 1997.

[15] Victor Ostromoukhov. Digital Facial Engraving, *Computer graphics*, vol 33, pp 417–424, 1999.

[16] K. Perlin. An Image Synthesizer, *Computer Graphics*, vol 19, pp 287–296, 1985.

[17] Mark Segal and Kurt Akeley. The OpenGL Graphics System: A Specification (revision 1.2.1), 1999.

[18] Francois Sillion and Claude Puech. Radiosity and Global Illumination, *Morgan Kaufmann*, ISBN 1-55860-277-1, 1994

[19] Sing Bing Kang. A Survey of Image-based Rendering Techniques, *Videometrics VI Proceedings*, pp 2–16, 1999.

[20] Osama Tolba and Julie Dorsey and Leonard McMillan. Sketching with Projective 2D Strokes, *Proceedings of the ACM Symposium on User Interface Software and Technology*, pp 149–157, 1999.

[21] Parag Tole and Fabio Pellacini and Bruce Walter and Donald P. Greenberg. Interactive Global Illumination in Dynamic Scenes, *ACM Transactions on Graphics*, vol 21, pp 537–546, 2002.

[22] Eric Veach and Leonidas Guibas. Bidirectional Estimators for Light Transport *Fifth Eurographics Workshop on Rendering*, pp 147–162, 1994.

[23] Bruce Walter and George Drettakis and Steven Parker. Interactive Rendering using Render Cache, *Rendering Techniques '99*, pp 19–30, 1999.

[24] Ingo Wald and Carsten Benthin and Philipp Slusallek and Thomas Kollig and Alexander Keller Interactive Global Illumination using Fast Ray Tracing, *Thirteenth Eurographics Workshop on Rendering*, pp 15–24, 2002.

[25] Bruce Walter and George Drettakis and Donald Greenberg. Enhancing and Optimizing the Render Cache, *Thirteenth Eurographics Workshop on Rendering*, pp 37–42, 2002.

[26] Gregory J. Ward and Paul Heckbert. Irradiance Gradients, *Third Eurographics Workshop on Rendering*, pp 85–98, 1992.

[27] Turner Whitted. An improved illumination model for shaded display, *CACM*, vol 23(6), pp 349-349, 1980.

[28] Jacques Zaninetti and Xavier Serpaggi and Bernard Péroche. A Vector Approach for Global Illumination in Ray Tracing, *Computer Graphics Forum*(Proc. Eurographics '99), vol 17, pp 149–158, 1999.

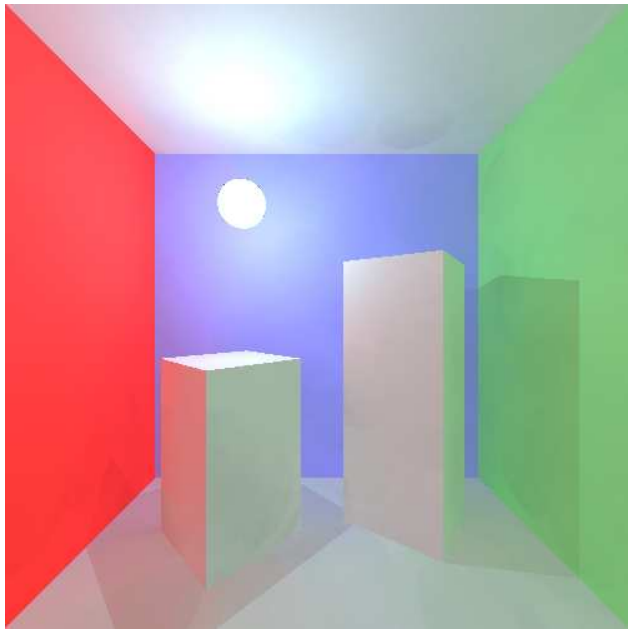

Figure 5: Cornell Box 1 : lancer de rayons (LR)

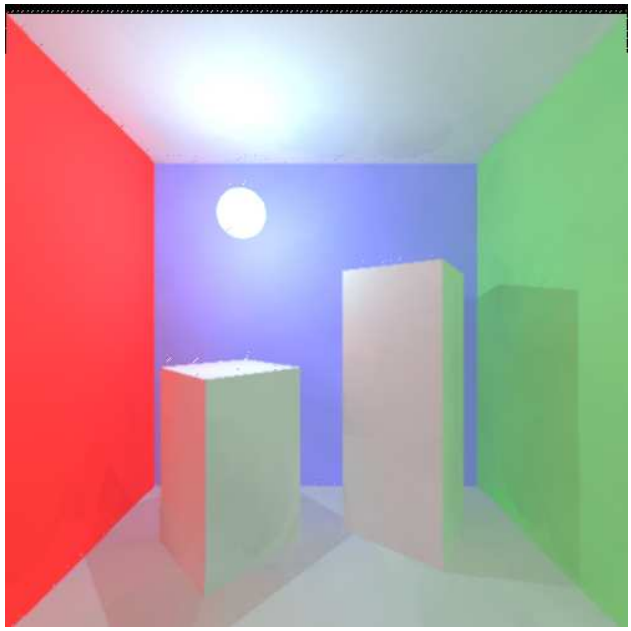

Figure 6: Cornell Box 1 : 5D VEI + RC

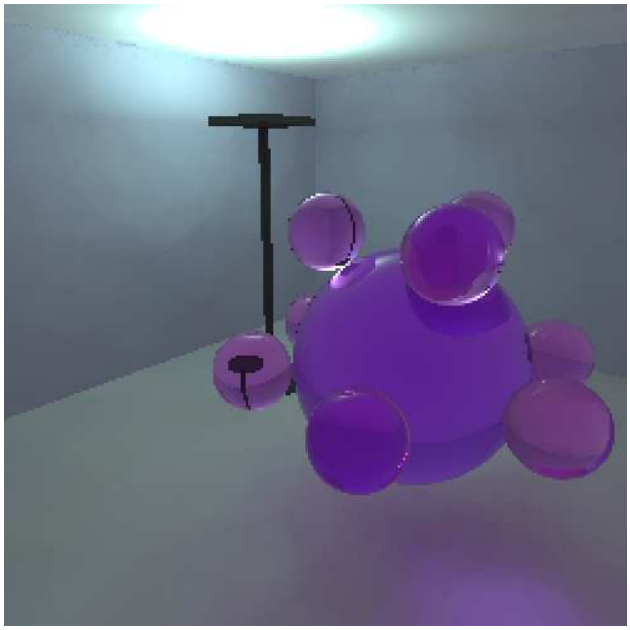

Figure 7: Sphere Flake : lancer de rayons (LR)

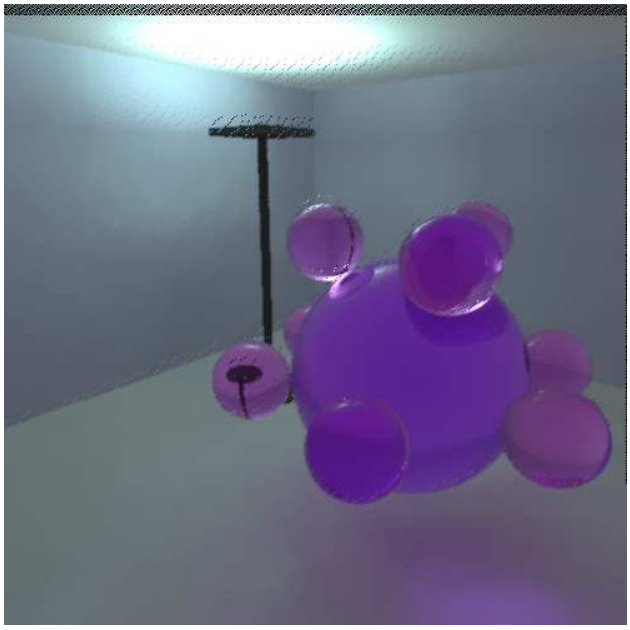

Figure 8: Sphere Flake : 5D VEI + RC