Université de Montréal

Échantillonnage de produits de fonctions

par

Fabrice Rousselle

Département d'informatique et de recherche opérationnelle

Faculté des arts et des sciences

Mémoire présenté à la faculté des études supérieures

en vue de l'obtention du grade de

Maître ès sciences (M.Sc.)

en informatique

Juillet 2007

Université de Montréal
Faculté des études supérieures

Ce mémoire de maîtrise intitulé

Échantillonnage de produits de fonctions

présenté par
Fabrice Rousselle

a été évalué par un jury composé des personnes suivantes :

Pierre Poulin
président-rapporteur

Victor Ostromoukhov
directeur de recherche

Neil F. Stewart
membre du jury

Mémoire accepté le 24 août 2007

Sommaire

Nous présentons une méthode générale et efficace d'échantillonnage suivant l'importance. Notre méthode peut être appliquée à des produits de fonctions multiples et utilise une représentation hiérarchique des maxima et moyennes locaux de chaque fonction considérée. Les échantillons sont générés durant la traversée de la hiérarchie, qui est faite de façon conservative en considérant le produit des maxima locaux des fonctions considérées. Les échantillons sont alors sélectionnés ou rejetés d'après une comparaison à des valeurs de seuils provenant d'une séquence à basse discrépance.

Nous appliquons notre méthode à l'évaluation de l'éclairage direct, dans un contexte de rendu photo-réaliste. Notre évaluation implique trois fonctions : la lumière distante encodée dans une carte d'environnement, la fonction de réflectance de la surface, et une carte de visibilité évaluée pour chaque pixel de l'image. Les échantillons sont générés d'après le produit des trois fonctions, ce qui permet une réduction considérable de variance par rapport aux méthodes de pointe tenant compte uniquement de l'éclairage et de la réflectance.

Mots clefs :
échantillonnage suivant l'importance, produits de fonctions, méthodes Monte Carlo, lancer de rayons, carte d'environnement, BRDF, visibilité

Abstract

We present a general method for efficient importance sampling of the product of multiple functions, using a hierarchical representation for the maximum and average values of each individual function. Samples are generated simultaneously by means of a conservative traversal of the hierarchies according to the product of their individual local maxima. The resulting samples are then thresholded against the product of the individual local averages.

We apply our method to the evaluation of direct illumination, which involves the product of distant lighting by an environment map, surface reflectance, and a visibility function estimated per pixel. Samples are generated according to the product of all three functions resulting in considerable noise reduction, compared to existing state-of-the-art methods sampling only the product of the lighting and the BRDF.

Keywords :

importance sampling, product of functions, Monte Carlo methods, ray tracing, environment map, BRDF, visibility

Table des matières

Table des figures

Liste des tableaux

Liste des algorithmes

Remerciements

Je remercie en premier lieu mon directeur de maîtrise, Victor Ostromoukhov, pour son encadrement, sa confiance et son financement sans lesquels ce projet n'aurait pu aboutir. Je tiens également à remercier les personnes ayant contribué à ce projet : Petrik Clarberg qui a participé aux discussions et fourni une aide précieuse pour l'aspect théorique, Luc Leblanc qui a largement contribué à l'implémentation du système, Nicolas Tittley qui a fourni le système de génération de sphères internes, Pierre Poulin qui a participé aux discussions, ainsi qu'à la rédaction de ce mémoire et de l'article y correspondant, Neil Stewart pour ses commentaires pertinents concernant la rédaction de ce mémoire, Henrik Wann Jensen qui a participé aux discussions. Je désire également souligner les contributions suivantes : Paul Debevec qui a gracieusement fourni les cartes d'environnement, Wojciech Matusik qui a mis à notre disposition ses BRDFs tabulées, et David Cline qui a fourni une implémentation de sa méthode. Je remercie l'ensemble des membres du laboratoire d'infographie de l'Université de Montréal pour ces deux dernières années en leur compagnie qui furent bien agréables. Enfin, je remercie ma famille pour son support continu et tout particulièrement ma conjointe, Cynthia, sans qui ce projet n'aurait pas démarré.

Chapitre 1

Introduction

Les méthodes d'échantillonnage procurent une solution simple et flexible lorsqu'il n'est pas possible d'évaluer une intégrale avec exactitude, que ce soit en raison de la complexité du problème, ou d'un temps d'évaluation prohibitif. Par exemple, les sondages d'opinions, très courants de nos jours, permettent d'obtenir rapidement une estimation de l'opinion d'une population potentiellement très nombreuse. Dans ce cas-ci, le problème est essentiellement d'estimer rapidement une valeur qui ne pourrait être évaluée exactement dans un temps, ou pour un coût, raisonnable. Dans d'autres cas, la complexité du problème dépasse nos outils analytiques.

Le premier élément remarquable des méthodes Monte Carlo est leur extrême simplicité. Il suffit d'être en mesure d'échantillonner et d'évaluer ponctuellement une fonction. Cela rend ces méthodes très flexibles et susceptibles d'être appliquées dans une très grande variété de cas. L'autre caractéristique intéressante des méthodes Monte Carlo est que leur taux de convergence est indépendant de la dimensionalité du domaine d'intégration. Si l'on prend le cas des règles de quadrature, une approche classique d'intégration numérique, leur taux de convergence est généralement de $O\left(N^{-r/s}\right)$, pour un entier $r \geq 1$ et s le nombre de dimensions. Une dimensionalité élevée dégrade donc la convergence de façon significative. Cet effet est souvent appelé la "malédiction de la dimensionalité" (*curse of dimensionality*) et est exposé, entre autre, dans la thèse de doctorat de Veach [Vea98]. Enfin, notons que l'application d'une méthode Monte Carlo n'implique pas une quelconque forme de continuité de la fonction intégrée, contrairement aux règles de quadrature qui supposent souvent que la fonction intégrée ait une ou plusieurs dérivées continues.

Les caractéristiques des méthodes Monte Carlo les ont rendues populaires dans le domaine de la finance (entre autres), où le nombre de variables (et donc de dimensions) peut être particulièrement élevé. Le principal inconvénient des méthodes Monte Carlo est que le taux de convergence de la méthode dite élémentaire, $O\left(\sqrt{N}\right)$, est relativement lent. On doit, pour diminuer de moitié la variance de l'estimation, quadrupler le nombre d'échantillons, ce qui conduit rapidement à des nombres d'échantillons prohibitifs. Plusieurs techniques ont été développées pour améliorer l'efficacité des méthodes Monte Carlo, parmi lesquelles on trouve l'échantillonnage suivant l'importance, base de notre approche.

Notre principale contribution est une méthode d'échantillonnage suivant l'importance basée sur le principe du *seuillage hiérarchique*. Le seuillage hiérarchique est une technique déterministe permettant de distribuer efficacement des échantillons suivant une certaine fonction d'importance, en parcourant une représentation hiérarchique de l'espace considéré. Nous étendons ce principe au produit de plusieurs fonctions, pouvant être définies dans différents référentiels, et nous montrons comment l'intégrer à une méthode Monte Carlo d'échantillonnage suivant l'importance. La distribution des points est effectuée de façon déterministe, et les différents aspects à considérer pour obtenir une distribution non-biaisée sont abordés.

Nous appliquons ensuite cette méthode d'échantillonnage au problème de l'évaluation de l'illumination directe en un point d'une scène virtuelle. Par lumière directe, nous entendons toute lumière provenant d'une source active (produisant de l'énergie), telle qu'une ampoule, le soleil, etc. Le but de l'évaluation est de déterminer la proportion de cette lumière directe qui est réfléchie vers l'observateur. Plusieurs travaux se sont attaqués à ce problème au cours des dernières années, mais ils ont été restreints au produit de l'éclairage et de la réflectance. Aussi, les approches récentes nécessitent toutes que les fonctions impliquées dans le produit soient exprimées dans le même référentiel. Nous proposons un système plus complet, où la visibilité locale est estimée dynamiquement et intégrée au processus d'échantillonnage. De plus, l'éclairage est conservé dans un référentiel local, alors que la fonction de réflectance est définie dans le référentiel local de la surface.

Ce document est divisé en chapitres, traitant des différents aspects de notre travail. Le chapitre 2 expose les concepts théoriques de base qui seront utilisés, notamment la

notion de nombres quasi-aléatoires et la méthode Monte-Carlo d'échantillonnage suivant l'importance. Dans le chapitre 3 seront présentés les principaux travaux antérieurs traitant de l'évaluation de l'illumination directe en rendu photo-réaliste. Notre approche, le *seuillage hiérarchique*, sera détaillée dans le chapitre 4. Nous discuterons particulièrement de l'extension au produit de fonctions, ainsi que des dispositions prises pour éliminer toute source de biais. Dans les chapitres 5 et 6 seront présentés, respectivement, les principaux éléments de notre implémentation et les résultats obtenus. Enfin, nous conclurons par une discussion de la méthode, de ses inconvénients et des principaux axes d'exploration à venir.

Chapitre 2

Méthodes Monte Carlo

Dans ce chapitre nous présentons les principaux concepts théoriques, relatifs aux méthodes Monte Carlo, qui nous seront utiles par la suite. En premier lieu, nous présentons les différents types de nombres aléatoires, et plus précisément les nombres pseudo-aléatoires, qui seront utilisés dans notre algorithme d'échantillonnage. En second lieu, nous présenterons les principes des méthodes Monte-Carlo, ainsi que les variantes permettant de diminuer la variance. Ici encore, nous présenterons plus précisément la variante choisie pour notre système, l'échantillonnage suivant l'importance.

2.1 Nombres aléatoires

À un certain moment, il est nécessaire, pour toute méthode Monte Carlo, de substituer à une variable aléatoire, des valeurs réelles. Ces valeurs sont dites *nombres aléatoires* car ils auraient pu être obtenus suivant un processus aléatoire. Par définition, il n'est pas possible de produire des nombres aléatoires, au sens strict du terme. Lorsque de tels nombres sont nécessaires, on a alors recours à des tables de nombres aléatoires. Ces tables sont obtenues par observations de phénomènes connus pour être aléatoires, comme la radio-activité.

En pratique, le caractère aléatoire n'est pas toujours désirable. En particulier, il ne permet pas de reproduire exactement une expérience, ce qui est souvent voulu (pour permettre la validation d'un résultat par une tierce partie par exemple). On a alors recours à des nombres produits à l'aide de règles de productions qui ne sont donc

plus aléatoires, mais *pseudo-aléatoires*. Il faut néanmoins que les nombres ainsi produits soient tels qu'aucun écart significatif, par rapport à une séquence aléatoire, ne soit détectable. En informatique, les fonctions permettant d'obtenir des nombres aléatoires (en C, Java, etc.), retournent en réalité des nombres pseudo-aléatoires.

Enfin, pour certaines applications, on peut préférer se dispenser totalement du caractère aléatoire, pour obtenir d'autres caractéristiques statistiques précises. On utilise alors des nombres dits *quasi-aléatoires*. De tels nombres sont généralement utilisés dans le but d'obtenir des distributions aussi uniformes que possible. Ces séquences de nombres sont dites à *basse discrépance*. La discrépance est une mesure de la distance d'une distribution donnée à une distribution parfaitement uniforme. Les séquences à basse discrépance sont particulièrement utiles pour distribuer des nombres sur une grille, sans pour autant que la résolution de la grille soit connue. Il est alors possible d'augmenter progressivement le pas de discrétisation de la grille, jusqu'à atteindre le seuil de convergence. Le terme quasi-Monte Carlo est utilisé pour distinguer les méthodes utilisant des séquences de nombres quasi-aléatoires.

Séquences van der Corput

Parmi les différentes séquences à basse discrépance, nous avons choisi d'utiliser les séquences van der Corput (VDC). Rappelons que l'objectif d'une séquence à basse discrépance est d'approximer le mieux possible une distribution parfaitement uniforme. En considérant des valeurs $0 \leq x \leq 1$, une telle distribution est triviale à obtenir pour n points : il suffit, en partant de $x = 0$, de choisir les nouveaux points par incrément de $1/n$. Cependant, si l'on veut ajouter un nouveau point $n + 1$ tout en conservant une distribution parfaitement uniforme, il faudrait redistribuer également les n premiers points à partir de 0 en utilisant un pas de $1/(n + 1)$. Les séquences VDC s'inspirent directement de ce processus, sans jamais invalider un point déjà défini, ce qui permet l'ajout incrémental.

Les séquences VDC existent pour toute base $b \geq 2$, et sont définies par l'inversion du radical. Pour le $n^{ème}$ nombre, on a :

$$v = \sum_{j=1}^{T} a_j b^{j-1} \tag{2.1}$$

où les a_j sont les T chiffres exprimant n dans la base b. On a $T = T(n) = \lceil log_b^n \rceil$. Afin

d'obtenir un nombre $\in [0, 1)$, il suffit de placer le radical après la virgule. Le tableau 2.1 donne les 16 premiers nombres de la séquence VDC en base 2.

n	Direct	VDC (radical inversé)
0 (0000_2)	0,0000 ($0,0000_2$)	0,0000 ($0,0000_2$)
1 (0001_2)	0,0625 ($0,0001_2$)	0,5000 ($0,1000_2$)
2 (0010_2)	0,1250 ($0,0010_2$)	0,2500 ($0,0100_2$)
3 (0011_2)	0,1875 ($0,0011_2$)	0,7500 ($0,1100_2$)
4 (0100_2)	0,2500 ($0,0100_2$)	0,1250 ($0,0010_2$)
5 (0101_2)	0,3125 ($0,0101_2$)	0,6250 ($0,1010_2$)
6 (0110_2)	0,3750 ($0,0110_2$)	0,3750 ($0,0110_2$)
7 (0111_2)	0,4375 ($0,0111_2$)	0,8750 ($0,1110_2$)
8 (1000_2)	0,5000 ($0,1000_2$)	0,0625 ($0,0001_2$)
9 (1001_2)	0,5625 ($0,1001_2$)	0,5625 ($0,1001_2$)
10 (1010_2)	0,6250 ($0,1010_2$)	0,3125 ($0,0101_2$)
11 (1011_2)	0,6875 ($0,1011_2$)	0,8125 ($0,1101_2$)
12 (1100_2)	0,7500 ($0,1100_2$)	0,1875 ($0,0011_2$)
13 (1101_2)	0,8125 ($0,1101_2$)	0,6875 ($0,1011_2$)
14 (1110_2)	0,8750 ($0,1110_2$)	0,4375 ($0,0111_2$)
15 (1111_2)	0,9375 ($0,1111_2$)	0,9375 ($0,1111_2$)

TAB. 2.1 – 16 premiers nombres de la séquence van der Corput en base 2.

Il est possible de construire une séquence VDC de façon récursive. La figure 2.1 illustre cette approche pour une séquence VDC en base 2. On part d'un segment initial, auquel on assigne un code vide ("*"). Le début de chaque segment est indiqué par une barre verticale. On procède alors à une subdivision récursive de chaque segment en deux enfants. L'enfant de gauche hérite du code du parent et du préfixe "0", alors que l'enfant de droite hérite du code du parent et du préfixe "1". Cette construction nous donne donc la position d'une valeur dans la séquence VDC. Cette construction illustre également le principe sous-jacent des séquences VDC : le domaine est discrétisé en utilisant un pas égal à la base, donnant une densité maximale. Les points subséquents sont placés sur une grille pouvant être raffinée à l'infini, permettant une densité toujours plus élevée. Remarquez que, si l'on place 2^n points exactement, ceux-ci sont parfaitement équidistants.

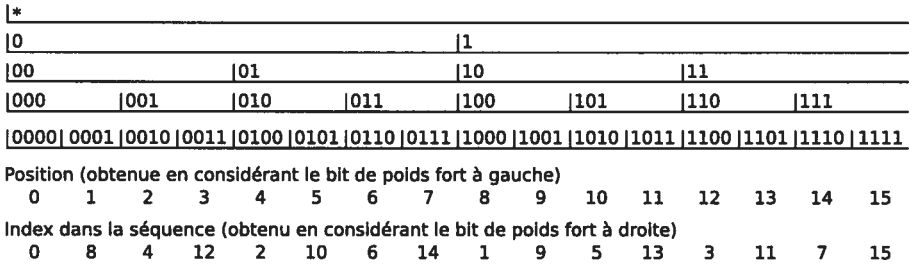


FIG. 2.1 – Construction récursive de la séquence van der Corput en base 2.

Discrépance

Comme nous l'avons mentionné, la discrépance est une mesure de l'uniformité d'une distribution. La définition formelle de la discrépance est :

$$D_N(\mathcal{B}; P) = \sup_{B \in \mathcal{B}} \left| \frac{A(B;P)}{N} - \lambda_s(B) \right| \tag{2.2}$$

Ici P est un ensemble de points $x_1, ..., x_N \in I^s$, et s le nombre de dimensions. $\mathcal{B}$ est la famille des sous-ensembles mesurables de I^s. $\lambda_s(B)$ est la mesure de Lebesgue du sous-ensemble B. La mesure de Lebesgue est une généralisation de la notion de longueur et d'aire aux sous-ensembles de $\Re^n$. $A(B;P)$ donne le compte de points tels que $1 \leq n \leq N$ et $x_n \in \mathcal{B}$. Intuitivement, ceci mesure la relation entre la proportion de points dans un sous-ensemble et le volume occupé par ce dernier. Plus la disparité entre les deux est grande, plus la discrépance est élevée.

À partir de l'équation 2.2, on distingue deux définitions de la discrépance. La discrépance *extrême* $D_N(P)$ d'un ensemble de points P est définie par $D_N(P) = D_N(\mathcal{J}; P)$, où $\mathcal{J}$ est la famille de tous les intervalles de I^s de la forme $\prod_{i=1}^{s} [u_i, v_i)$. Cette définition étant extrêmement coûteuse à évaluer, on peut lui préférer la discrépance *étoile*, définie par $D_N^*(P) = D_N^*(\mathcal{J}^*; P)$, où $\mathcal{J}^*$ est la famille de tous les intervalles de I^s de la forme $\prod_{i=1}^{s} [0, u_i)$. Pour une discussion plus approfondie de la notion de discrépance, voir le livre de Niederreiter [Nie92].

2.2 Méthodes Monte Carlo

Les méthodes Monte Carlo appartiennent à la branche expérimentale des mathématiques, par opposition à la branche théorique, et leur nom fait référence au casino de

Monte Carlo où sont pratiqués des jeux de hasard. Le principe des mathématiques expérimentales est de tirer des conclusions d'après des observations, plutôt que d'après des postulats. Certains problèmes, très complexes, peuvent être traités avec une relative facilité avec une approche expérimentaliste.

À titre d'exemple, en infographie, on peut prendre l'évaluation de l'éclairage global. Deux approches principales existent. La première, la "radiosité", est analytique et propose de modéliser les échanges d'énergie entre tous les éléments d'une scène. Elle permet de déduire la lumière qui sera reçue en un élément de surface. Le principe sous-jacent est en fait de réduire un problème complexe et insoluble en une série de problèmes simples dont la solution est connue. Ainsi, une scène est décomposée en éléments simples, pour lesquels les échanges d'énergie peuvent être modélisés. Bien que conceptuellement simple, cette approche présente un défi de taille : la décomposition automatique d'une scène complexe en éléments simples. La seconde approche, le "*photon mapping*", simule individuellement des millions de photons et observe leur chemin dans la scène. D'après la répartition des photons dans la scène, on obtient la lumière reçue en un point. Le lancer des photons est très simple et surtout très efficace. Bien que la compilation et l'interprétation de la répartition des photons soient relativement complexes, cette approche a l'avantage d'être indépendante de la complexité de la scène, contrairement à la méthode de radiosité.

Tout résultat quantitatif obtenu à l'aide d'une méthode Monte Carlo peut être vu comme l'estimation d'une intégrale multiple. Pour simplifier les explications, nous nous intéresserons ici uniquement à une intégrale à une dimension :

$$F = \int_0^1 f(x)dx. \tag{2.3}$$

Il existe plusieurs types de méthodes Monte Carlo, que nous présentons ici, en commençant par la méthode dite *élémentaire*.

2.2.1 Méthode élémentaire

On désire évaluer l'intégrale F. La méthode Monte Carlo élémentaire consiste à prendre la moyenne de N observations, qui est un estimateur sans biais de l'intégrale :

$$F_N = \frac{1}{N}\sum_{i=1}^{N} f(X_i). \tag{2.4}$$

Les valeurs X_i sont choisies aléatoirement dans le domaine d'intégration. L'approche élémentaire est extrêmement simple et sa convergence est garantie, ce qui en fait déjà un outil relativement intéressant. Elle présente néanmoins un inconvénient de taille : le taux de convergence est lent, de l'ordre de $O\left(\sqrt{N}\right)$, où N est le nombre d'échantillons. Cela veut dire qu'il est nécessaire de quadrupler le nombre d'échantillon afin de réduire de moitié la variance de l'estimation, ce qui devient rapidement prohibitif. Les techniques qui suivent ont toutes pour but de réduire la variance des méthodes Monte Carlo.

2.2.2 Échantillonnage stratifié

Une première approche, pour améliorer l'efficacité des méthodes Monte Carlo est l'échantillonnage stratifié. L'idée est de subdiviser le domaine d'intégration Ω en régions Ω_i disjointes, de telle sorte que

$$\bigcup_i \Omega_i = \Omega$$

Une première approche, naïve, du problème, est de subdiviser le domaine d'intégrations en sous-ensembles d'aire égale, suivant une grille régulière. En plaçant un échantillon aléatoirement dans chaque sous-ensemble, on obtient un effet similaire à la méthode élémentaire, mais en garantissant une certaine uniformité.

Il peut être montré que le gain lié à la stratification est fonction de la différence entre la somme des intégrales de chaque strate et l'intégrale sur tout le domaine [Vea98]. Autrement dit, afin de maximiser le gain lié à la stratification, il est préférable de segmenter le domaine en strates de valeurs homogènes. En effet, si la variance dans chaque strate est la même que sur la totalité du domaine d'intégration, il n'y aura pas de diminution de la variance. Il est préférable d'utiliser des strates compactes car celles-ci ont alors tendance à être plus homogènes. Il faut aussi, autant que possible, placer dans chaque strate un nombre d'échantillons proportionnel à la fraction volumique de cette strate par rapport au domaine.

2.2.3 Variables de contrôle

L'erreur d'échantillonnage dans la méthode Monte Carlo élémentaire est due aux variations de la fonction considérée. L'utilisation de variables de contrôle permet de

limiter ces variations et donc la variance de l'estimation. Le principe est de soustraire de la fonction considérée $f(x)$ une autre fonction $g(x)$ dont l'intégrale est connue. On a alors :

$$F = \int g(x)dx + \int [f(x) - g(x)]\,dx$$

et l'estimateur Monte Carlo devient

$$F = \int g(x)dx + \frac{1}{N}\sum_{i=1}^{N} f(X_i) - g(X_i).$$

L'objectif est d'avoir une fonction $g(x)$ qui soit aussi proche que possible de $f(x)$, tout en étant intégrable. Le cas idéal est celui où $f(x) - g(x) = c$, puisque toute variation est alors éliminée. Évidemment, ce cas idéal n'est pas normalement réalisable, mais il fixe l'objectif.

2.2.4 Échantillonnage suivant l'importance

L'échantillonnage suivant l'importance est une autre approche visant à compenser la variation de la fonction échantillonnée. L'idée est d'augmenter la densité d'échantillonnage dans les régions contribuant le plus à l'intégrale. Le principe sous-jacent est qu'une erreur d'évaluation dans une zone de faible contribution a moins d'impact. C'est sur ce principe que notre approche repose.

Plutôt que d'être tirés aléatoirement, les échantillons sont tirés suivant une Fonction de Densité de Probabilité (*Probability Density Function*, ou PDF). L'estimateur Monte Carlo est alors le suivant :

$$F_N = \frac{1}{N}\sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)}. \tag{2.5}$$

Rappelons qu'une PDF est un type particulier de fonction, telle que $p(x) \geq 0$ pour toute valeur de x et $\int p(x)dx = 1$. Plus la forme de la PDF utilisée est proche de celle de la fonction considérée, $f(x)$, plus le gain d'efficacité sera conséquent. En fait, si le rapport entre les deux est une constante, toute variance sera éliminée. En effet, prenons $p(x) = cf(x)$. On a alors :

$$\int f(x)dx = \frac{1}{c}\int p(x)dx = \frac{1}{c}.$$

et, par conséquent,

$$\frac{f(X_i)}{p(X_i)} = \frac{1}{c} = \int f(x)dx$$

Puisque c est une constante, la contribution de chaque échantillon est exactement la même et la variance est nulle. En pratique, il n'est pas possible d'avoir une fonction $p(x)$ exactement proportionnelle à $f(x)$, mais cela demeure l'objectif théorique. L'essentiel de la recherche en échantillonnage suivant l'importance traite de la définition de la fonction d'importance.

Il est possible de combiner l'échantillonnage suivant l'importance à l'utilisation de variables de contrôle. L'estimateur Monte Carlo devient alors :

$$F = \int g(x)dx + \frac{1}{N}\sum \frac{f(X_i) - g(X_i)}{p(X_i)}.$$

Chapitre 3

Autres travaux

Les méthodes Monte Carlo sont souvent utilisées dans le domaine du rendu photo-réaliste par lancer de rayon (*raytracing*). Les fonctions impliquées sont souvent discontinues (par exemple, la fonction de visibilité ne peut prendre que deux valeurs, 0 et 1) et certaines varient brusquement (comme les fonctions de réflectance très spéculaires). Aussi, ces fonctions sont souvent mesurées et par conséquent bruitées. Les méthodes Monte Carlo s'accommodent très bien de ces particularités, permettant de traiter une large gamme de cas, tout en demeurant très simple d'application. À la base du rendu photo-réaliste, on retrouve l'équation du rendu de Kajiya [Kaj86] :

$$L(\boldsymbol{x}, \vec{\omega}_o) = L_e(\boldsymbol{x}, \vec{\omega}_o) + \int_\Omega f_r(\boldsymbol{x}, \vec{\omega}_i, \vec{\omega}_o) L_i(\boldsymbol{x}, \vec{\omega}_i) cos\theta_i \, d\vec{\omega}_i, \tag{3.1}$$

qui donne la radiance en un point $\boldsymbol{x}$ de la scène dans la direction $\vec{\omega}_o$. Le premier terme, L_e, correspond à la lumière émise par ce point, alors que le second terme, l'intégrale sur l'hémisphère visible, correspond à la lumière réfléchie en ce point. f_r est la *Fonction de Distribution Bidirectionnelle de la Réflectance* (*Bidirectional Reflectance Distribution Function*, ou BRDF) de la surface (voir la section 5.2.2 pour une description). L_i est la lumière incidente. $cos\theta_i$ est un facteur d'atténuation de la lumière dû à l'angle incident. ω_i et ω_o indiquent respectivement le vecteur incident (depuis la lumière) à la surface et le vecteur sortant (vers l'oeil) depuis la surface.

Dans les applications de rendu, la variance des évaluations Monte Carlo se traduit en bruit dans l'image. Une approche très répandue pour réduire la variance est l'échantillonnage suivant l'importance. Il y a, par conséquent, eu beaucoup de travaux sur la définition de fonctions d'importance afin de résoudre l'équation du

rendu. Initialement, les travaux se concentraient sur des aspects spécifiques, comme l'échantillonnage de la lumière (généralement encodée dans des cartes d'environnement), ou encore de la BRDF. Certaines méthodes, plus générales, s'appliquaient à tout type de fonction. Enfin, puisque l'intégrale à évaluer est celle d'un produit, des méthodes ont été proposées, qui tenaient compte de tous les termes du produit. Dans ce chapitre, nous présenterons en premier lieu les méthodes d'échantillonnage de l'environnement, puis les méthodes spécifiques aux BRDFs, viendront ensuite les méthodes générales (s'appliquant aussi bien à l'environnement qu'aux BRDFs) et celles tenant compte du produit des deux.

3.1 Échantillonnage de l'environnement

Debevec [Deb98] a introduit la notion d'éclairage réaliste par carte d'environnement (*environment map*). La carte d'environment est une image omnidirectionnelle, encodant une large gamme de l'intensité lumineuse (*High Dynamic Range*, ou HDR) reçue en un point. La richesse de l'information contenue dans ces images permet d'obtenir des résultats de très bonne qualité (notamment pour l'intégration d'objets virtuels dans des scènes réelles), mais cela augmente considérablement le volume de données à traiter. L'échantillonnage est tout indiqué pour faciliter le traitement de ce type d'information.

Afin de traiter les cartes d'environnement, la première approche a été de les approximer par des constellations de sources lumineuses directionnelles. Chaque source lumineuse couvre une région spécifique de l'environnement, et son intensité est l'intégrale de la région couverte. Ces sources lumineuses sont alors utilisées directement. Il est important de noter que l'on ne parle pas ici de méthode Monte Carlo, mais simplement de discrétisation. Les images produites par ce type d'approche ne sont pas bruitées, mais sont également inexactes. La première méthode proposée était celle du module *LightGen* par Cohen et Debevec [CD01]. Il s'agit d'une application directe de l'algorithme de K-Moyennes. K points sont distribués aléatoirement dans l'image. Tous les pixels sont alors regroupés suivant leur proximité à l'un des points, créant K groupes. Chaque point est alors déplacé au centre de masse de son groupe. La masse d'un pixel correspond à son intensité lumineuse. Ce processus est itératif et s'arrête quand le déplacement des points est inférieur à une valeur seuil. Cette approche, très simple, a l'inconvénient de produire des distributions de points non-optimales, où les régions

sombres sont sur-échantillonnées.

Kollig et Keller [KK03] proposent une version améliorée de l'algorithme utilisé par *LightGen*. Le principe est fondamentalement le même, mais les points sont introduits un à la fois. Après chaque ajout de point, une passe complète de relaxation est effectuée. Le nouveau point est introduit à proximité du point d'intensité maximale. Le coût des relaxations additionnelles est compensé par une meilleure distribution de points, qui sont plus concentrés dans les zones de forte intensité lumineuse. Afin de contrer l'effet de palier dans les ombres, dû à la discrétisation des lumières, les auteurs proposent de choisir aléatoirement la position finale du point dans la région qu'il couvre, et ce pour chaque pixel considéré. Quelle que soit la position choisie, la valeur est toujours l'intégrale de la région couverte.

Agarwal et al. [ARBJ03] proposent une méthode conceptuellement similaire, bien que l'approche soit différente, appelée *Structured Importance Sampling* (SIS). Ils commencent par définir une valeur d'importance pour chaque pixel de l'environnement. Ils segmentent alors en $d = 6$ couches l'environnement et assignent à chaque couche un nombre de points proportionnel à son importance. Chaque couche est alors stratifiée en cellules de façon à minimiser l'écart maximal entre un point d'une cellule et son centre. Ces auteurs proposent également une pré-intégration de l'éclairage de chaque cellule et de positionner aléatoirement le point dans la cellule. Un exemple de stratification obtenue par application de cette méthode est donné dans la figure 3.1.

Ostromoukhov et al. [ODJ04] proposent un algorithme très efficace, basé sur une construction récursive du pavage de Penrose. L'idée est d'augmenter récursivement la densité locale de points, jusqu'à ce qu'elle soit proportionnelle à la valeur d'importance locale. Le procédé est déterministe et des vecteurs de correction précalculés peuvent être appliqués de façon à produire une distribution avec des caractéristiques de bruit bleu. Cette méthode est la base de celle que nous développons dans ce mémoire.

Wan et al. [WWL05] proposent d'utiliser la projection HEALPix (voir section 5.1.1) pour segmenter l'environnement. La projection pouvant être représentée hiérarchiquement, cela leur permet d'augmenter la densité locale de points, suivant la fonction d'importance. Le positionnement des points dans les feuilles de la hiérarchie HEALPix est déterministe.

Du fait que toutes ces méthodes sont basées strictement sur l'information lumineuse,

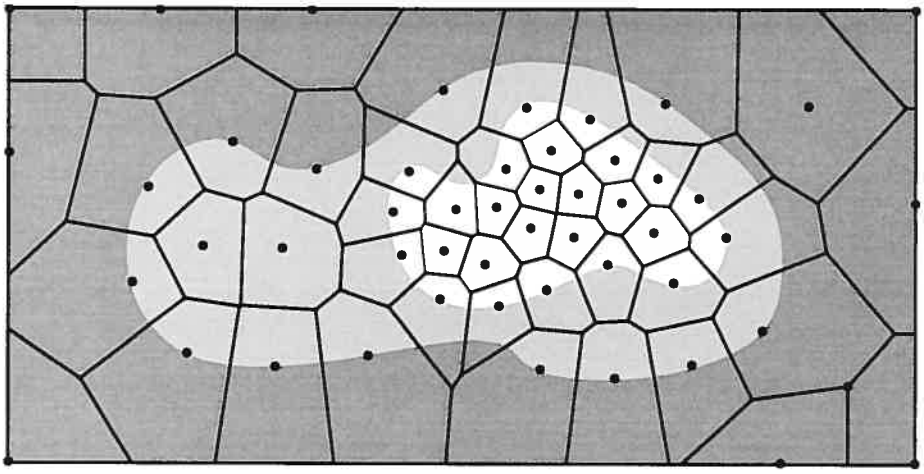


FIG. 3.1 – Stratification obtenue avec SIS pour $d = 3$ couches. Cette image est tirée de [ARBJ03].

elles sont réservées aux cas d'une BRDF diffuse ou ayant un lobe spéculaire plutôt évasé. De plus, l'orientation de la surface n'est pas prise en compte.

3.2 Échantillonnage de la BRDF

La BRDF définit comment la lumière provenant d'une direction donnée est réfléchie dans une autre direction. Si les vecteurs de direction sont définis avec des coordonnées sphériques, on a alors une fonction à 4 dimensions (deux pour le vecteur incident, et deux pour le vecteur sortant). On distingue deux types de BRDF : celles dites analytiques et celles dites mesurées. Une BRDF analytique est en fait un modèle mathématique définissant exactement la réflectance. Habituellement, les modèles mathématiques utilisés ont pour but d'offrir un maximum de flexibilité, pour un minimum de complexité. Certains types de réflections ne peuvent être modélisés facilement, on peut alors recourir à des mesures d'après de véritables matériaux, qui seront alors tabulées et interpolées.

Parmi les principaux modèles de BRDF analytiques, notons ceux de Phong [Pho75], Ward [War92] et Lafortune et al. [LFTG97]. Le modèle de Phong est à la fois l'un des plus anciens et des plus répandus. Sa grande simplicité en a fait un choix populaire, puisqu'il est à la fois intuitif et économique en terme de temps de calcul. Le modèle

de Lafortune peut être vu comme une généralisation de celui de Phong, mais avec la possibilité d'avoir plusieurs lobes spéculaires, et garantissant la conservation d'énergie (ce qui n'est pas le cas du modèle de Phong). L'objectif du modèle de Lafortune, tout comme celui de Ward, est d'offrir une structure offrant un maximum de flexibilité, dont les paramètres seront fixés de façon à reproduire des données mesurées. Lafortune a ainsi donné les paramètres à utiliser pour reproduire la fonction de réflectance de la peau. Ces BRDFs peuvent être échantillonnées directement, par inversion de la *Fonction de Densité Cumulative* (*Cumulative Density Function*, ou CDF). Le principe de cette technique est présenté par Pharr et Humphreys, de même que les principaux types de BRDFs analytiques dans [PH04]. L'échantillonnage par inversion de CDF est également présenté par Secord et al. [SHS02], où il est utilisé pour distribuer des primitives de dessin dans un contexte non photo-réaliste. Citons aussi les modèles de BRDF de Ashikhmin-Shirley [AS00], et Torrance-Sparrow [TS67], qui ne peuvent être échantillonnés directement par inversion de la CDF, et pour lesquels des approximations sont utilisées afin de guider l'échantillonnage.

Les modèles analytiques ont une portée limitée, dans la mesure où leur design est conçu pour répondre à certains besoins précis. Matusik et al. [MPBM03] proposent un système d'acquisition et une représentation de BRDFs isotropiques (3D plutôt que 4D) dont les effets ne peuvent pas toujours être reproduits avec les modèles analytiques existants (voir l'étude de Ngan et al. [NDM05]). Pour des modélisations où l'exactitude du rendu est essentielle, ce type de BRDF tabulée est souvent la meilleure solution. L'échantillonnage peut se faire à l'aide de modèles analytiques, comme celui de Lafortune, le modèle analytique est alors utilisé pour guider l'échantillonnage des données tabulées. Une autre approche, proposée par Lawrence et al. [LRR04], consiste à factoriser la BRDF en éléments de moindre dimensionnalité, pouvant facilement être échantillonnés.

Il est important de noter que les méthodes d'échantillonnage de la BRDF sont efficaces uniquement si la BRDF est très spéculaire où si l'éclairage est diffus.

3.3 Méthodes générales

Les méthodes présentées jusqu'à présent s'appliquaient exclusivement aux cartes d'environnement ou aux BRDFs. Certaines approches proposées sont plus générales

et peuvent s'appliquer indépendemment du type de fonction. Lawrence et al. [LRR05] proposent une méthode permettant de définir une CDF numérique de façon générique. Leur approche est spécialement conçue pour être appliquée à des données encodées dans des images, telles que les cartes d'environnement. L'idée est de réduire la complexité de la CDF en utilisant une approximation linéaire par morceau. L'approximation permet une compression allant de 1 à 3 ordres de grandeur et donc l'enregistrement des cartes d'environnement pour de multiples orientations de surface, évitant ainsi l'échantillonnage des parties cachées de l'environnement.

Les méthodes proposées jusqu'à présent traitent d'une fonction à la fois, l'éclairage ou la réflectance. Veach et Guibas [VG95] proposent une méthode permettant de combiner des échantillons tirés de deux fonctions distinctes, appelée *Multiple Importance Sampling* (MIS). Cette méthode permet ainsi d'utiliser au mieux les méthodes spécialisées qui ont été exposées jusqu'à présent.

3.4 Échantillonnage du produit

La méthode MIS permet de combiner différentes méthodes d'échantillonnage de façon à tirer le meilleur parti d'algorithmes spécialisés. Bien qu'offrant un compromis très intéressant, cette approche n'est pas aussi efficace dans le cas où aucune des fonctions n'est représentative du produit. En effet, l'équation du rendu est une intégration du produit de l'éclairage, la réflectance, le facteur dû à l'angle incident et la visibilité. Des travaux plus récents ont exploré l'échantillonnage direct du produit de l'environnement et de la BRDF.

Burke et al. [BGH05] proposent une approche suivant la même logique que celle de Veach et Guibas [VG95]. Ils proposent d'échantillonner l'environnement ou la BRDF, puis de retenir les échantillons pertinents d'après la valeur du produit des deux fonctions. Pour ce faire, ils suggèrent l'échantillonnage par rejet (voir la section 4.1 pour une définition) ou les méthodes de *Sampling Importance Resampling*. Cette méthode a l'inconvénient de rejeter une large proportion d'échantillons (les résultats présentés utilisent 15 échantillons sur 800 évalués).

Clarberg et al. [CJAMJ05] proposent une méthode basée sur les ondelettes de Haar, *Wavelet Importance Sampling* (WIS). L'idée est d'évaluer rapidement, en effectuant le produit des fonctions compressées, une fonction d'importance représentant le produit des

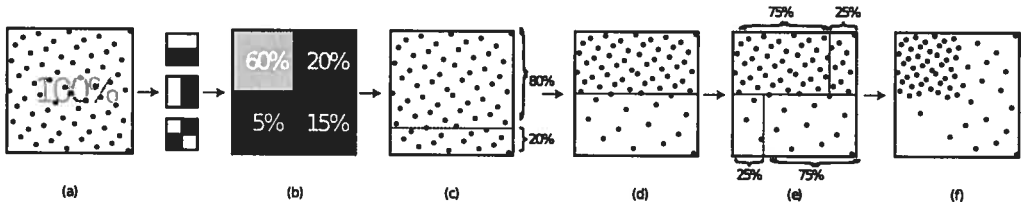


FIG. 3.2 – *Space Warping*. Adaptation d'une distribution uniforme initiale (a) d'après un niveau de coefficients d'ondelettes. Les pourcentages (b) sont tirés des coefficients d'ondelettes. La distribution initiale est partitionnée en deux rangées dont les hauteurs sont déterminées par leurs probabilités totales (c), qui sont ensuite remises à l'échelle (d). Finalement, l'opération est répétée pour chaque rangée (e), et résulte en une distribution respectant les probabilités spécifiées (f). Cette illustration est tirée de [CJAMJ05].

deux fonctions. La fonction d'importance résultante est une approximation constante par morceau du produit. Ces auteurs utilisent alors une technique appelée *Space Warping*, illustrée dans la figure 3.2, qui part d'une distribution uniforme de points (obtenue à l'aide d'une séquence Hammersley) et les redistribue dans les nœuds de la hiérarchie d'ondellette, de façon à obtenir une distribution suivant la fonction d'importance. Le principal inconvénient de cette méthode est qu'elle repose sur le précalcul et que les deux fonctions doivent être exprimées dans le même référentiel. Pour ce faire, la carte d'environnement est révaluée pour différentes orientations, à une moindre résolution (pour limiter la taille en mémoire et le temps de précalcul), ce qui réduit la qualité de l'estimation.

Cline et al. [CETC06] proposent une variante de la méthode WIS, appelée *Two-Stage Importance Sampling* (TSIS). Leur objectif est d'éliminer le recours au précalcul. Le principe de base reste le même : définir une fonction d'importance, puis appliquer l'algorithme de *Space Warping* pour obtenir une distribution appropriée de points. L'environnement est représenté à l'aide d'une *Summed Area Table*, et la BRDF est reconstituée par évaluations ponctuelles. Pour garantir la précision de la reconstruction de la BRDF, les positions des pics d'intensité sont définis et utilisés comme guides. Puisque la BRDF est évaluée dynamiquement, il n'est pas nécessaire de l'encoder pour toutes les orientations possibles, permettant une plus grande précision qu'avec WIS, ce qui augmente la qualité de l'évaluation. Un exemple illustrant le partitionnement généré

FIG. 3.3 – Partitionnement obtenu par application de TSIS sur le produit d'une carte d'environnement avec une BRDF très diffuse. Ces images sont tirées de [CETC06].

par cette méthode est donné dans la figure 3.3.

3.5 Résumé

L'évaluation de l'éclairage direct est un problème complexe, et la base de bien des recherches ces dernières années, particulièrement à l'aide de méthodes Monte Carlo. Rapidement, la nécessité de tenir compte, simultanément, des différentes fonctions s'est imposée. En ce sens, Veach et Guibas, avec leur méthode combinatoire, ont ouvert la voie vers les méthodes plus récentes, échantillonnant directement le produit des fonctions impliquées.

Les méthodes tenant compte du produit de l'éclairage incident et de la réflectance lors de l'échantillonnage donnent de très bons résultats. Si l'on se rapporte à l'équation 3.1, on constate que seule la visibilité n'est pas prise en compte. En pratique, pour les méthodes WIS et TSIS, la visibilité est la principale cause de la variance dans l'estimation et demeure d'une façon plus générale, au stade actuel de recherche, le principal problème à résoudre. Nous aborderons cette notion de visibilité et comment elle peut être intégrée au processus d'échantillonnage dans la section 5.4.

Chapitre 4

Seuillage hiérarchique

Notre méthode de *seuillage hiérarchique* combine deux approches. En premier lieu, nous obtenons, à partir d'une distribution uniforme, une distribution de points suivant approximativement une certaine fonction d'importance, à l'aide d'échantillonnage par rejet. Par la suite, les valeurs de ces échantillons sont utilisées pour alimenter un estimateur Monte Carlo de façon à estimer l'intégrale de la fonction. Une particularité de notre méthode est que la fonction d'importance est définie au fur et à mesure que les échantillons sont produits.

Dans ce chapitre, nous commençons par définir notre méthode d'échantillonnage. Par la suite, nous discutons des problèmes de biais et des solutions proposées, ainsi que des extensions à plusieurs dimensions et au produit de fonctions. Enfin, nous donnons un algorithme simple permettant de prédire le nombre d'échantillons obtenus, ainsi que l'estimateur Monte Carlo utilisé.

Notre méthode d'échantillonnage étant destinée à être utilisée dans des programmes informatiques, nous utilisons exclusivement des nombres pseudo-aléatoires. Cependant, afin de simplifier notre exposé, nous parlerons simplement de nombres aléatoires. De même, nous parlerons de distributions non biaisées, bien qu'elles soient en fait obtenues à l'aide de nombres pseudo-aléatoires.

4.1 Échantillonnage par rejet

Le seuillage hiérarchique est une variante d'échantillonnage par rejet, également appelée "tirage par blanc ou noir". Le principe du rejet est extrêmement simple et

permet d'échantillonner tous les types de fonctions. Prenons le cas d'une fonction $f(x)$ telle que $0 \leq f(x) \leq 1$, avec $0 \leq x \leq 1$. On trace $f(x)$ dans le carré unité $0 \leq x, y \leq 1$ et on tire aléatoirement des échantillons candidats (x, y), comme illustré dans la figure 4.1. Si un échantillon est situé sous la courbe, il est sélectionné, sinon, il est rejeté. En pratique, un échantillon est rejeté si $y \geq f(x)$. La simplicité de cette approche explique sa versatilité, puisqu'il suffit d'être en mesure d'évaluer la valeur d'une fonction en tout point pour l'échantillonner.

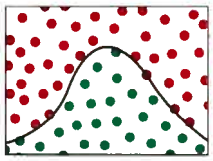


FIG. 4.1 – Échantillonnage par rejet.

Ce type d'échantillonnage s'applique directement à l'évaluation d'intégrales. En effet, l'intégrale d'une fonction étant égale à l'aire sous la courbe, il suffit d'évaluer la proportion d'échantillons sélectionnés pour obtenir l'intégrale de la fonction, avec une éventuelle mise à l'échelle si l'espace n'est pas unitaire. Une autre application classique, qui est celle qui nous intéresse, est l'obtention d'une distribution d'échantillons suivant une certaine densité de probabilité. L'obtention de cette distribution est implicite par le processus de rejet.

Bien que simple, l'échantillonnage par rejet n'est pas efficace. En effet, le nombre de tests à effectuer est inversement proportionnel à l'intégrale de la fonction considérée. Prenons une densité de probabilité $p(x)$, intégrant à 0,001, il faudra tester en moyenne 1000 candidats pour obtenir 1 échantillon suivant cette distribution. Afin de réduire le taux de rejet, on utilise alors une fonction enveloppe $q(x)$, que l'on sait échantillonner directement, approximant la forme de $p(x)$. On définit aussi une constante s, telle que $p(x) \leq sq(x)$ pour tout x du domaine considéré. Les échantillons candidats sont alors distribués suivant $q(x)$ et rejetés avec une probabilité $f(x)/sq(x)$.

Comme beaucoup d'autres approches visant à améliorer les méthodes Monte Carlo, le choix de $q(x)$ est délicat. Il faut que cette distribution soit facilement échantillonnable, et qu'elle approxime le mieux possible la fonction $f(x)$ considérée. Ces deux contraintes sont généralement incompatibles. Nous verrons dans la section suivante comment nous

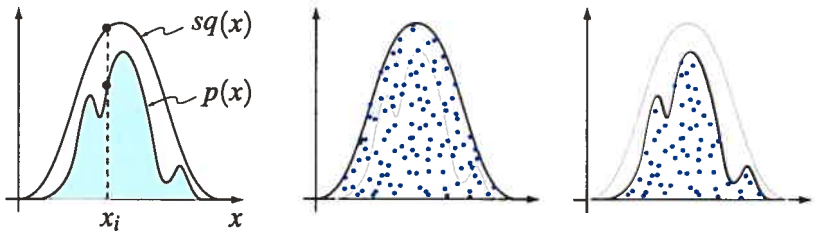


FIG. 4.2 – Échantillonnage par rejet avec une fonction enveloppe. Les échantillons candidats sont distribués suivant la distribution $q(x)$. Pour chaque candidat x, une valeur aléatoire $y \in [0, 1]$ est tirée et l'échantillon est conservé si $y \leq f(x)/sq(x)$.

construisons dynamiquement la fonction enveloppe de façon générique et récursive, permettant de traiter efficacement une très large gamme de fonctions.

4.2 Seuillage hiérarchique

Notre algorithme de seuillage hiérarchique a été conçu afin de répondre à deux critères de base : la simplicité, pour assurer l'efficacité du calcul, et l'extensibilité. Soit $f(x)$ la fonction à échantillonner.

Fondamentalement, le seuillage hiérarchique est simplement une approche récursive du principe d'échantillonnage par rejet. Par conséquent, notre objectif initial est le même : échantillonner uniformément l'espace de $f(x)$, puis effectuer le test de rejet. La nuance se situe au niveau de la génération des échantillons candidats, celle-ci se fait suivant un algorithme récursif, de façon à remplir graduellement "par le bas" l'espace, à la manière d'un verre que l'on remplit d'eau. Nous verrons, dans cette section, les trois éléments de notre algorithme : la génération des candidats, la définition de la fonction d'importance et la base du test de rejet.

Le premier élément de notre algorithme est la génération des candidats, qui a pour but d'échantillonner progressivement et uniformément l'espace. Nous utilisons, pour ce faire, des séquences à basse discrépance. Toute séquence peut être utilisée, pourvu qu'elle puisse être construite récursivement et qu'elle permette un remplissage progressif de l'espace. En pratique, nous avons choisi d'utiliser les séquences van der Corput (VDC) (définies dans la section 2.1), pour leur simplicité et leur applicabilité dans le contexte de l'évaluation de l'éclairage direct (voir chapitre 5). La figure 4.3

illustre l'approche, en utilisant une séquence VDC en base $b = 2$. On fixe tout d'abord le niveau maximal de subdivision, $L = 4$ dans ce cas, puis on génère récursivement les points de la séquence. On attribue alors à chaque point des coordonnées 2D : la position dans la séquence en abscisse, et l'indice dans la séquence en ordonnée. Il en résulte une distribution uniforme, mais qui est générée progressivement et "par le bas".

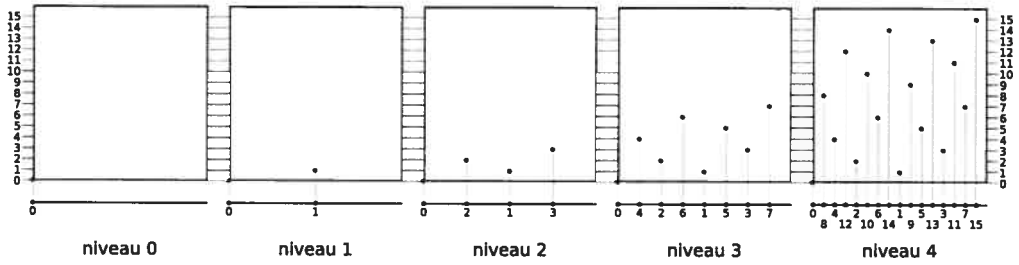


FIG. 4.3 – Génération récursive d'une distribution 2D uniforme d'échantillons. On utilise une séquence van der Corput en base $b = 2$, pour laquelle on a fixé un niveau maximal de subdivision $L = 4$. Les coordonnées de chaque point sont définies par sa position dans la séquence (en abscisse) et son indice dans la séquence (en ordonnée). Ce procédé introduit progressivement les points, remplissant graduellement l'espace.

Le deuxième élément de notre algorithme est la définition d'une fonction d'importance, approximant la forme de $f(x)$. Nos échantillons candidats ayant des valeurs entières bornées par $[0; b^L)$, alors que $f(x)$, est bornée par $[0; 1]$, il est nécessaire de mettre $f(x)$ à l'échelle en la multipliant par b^L. Cela fait, nous définissons la valeur d'importance de chaque feuille de la hiérarchie, comme étant la valeur moyenne locale de la fonction. Il en résulte une fonction d'importance constante par morceau, notée $h(x)$, approximant $f(x)$ par sa moyenne, à un facteur d'échelle près. La figure 4.4 illustre ce processus.

Le troisième élément de notre algorithme est le test de rejet proprement dit. Nous procédons comme à l'ordinaire, comparant chaque échantillon candidat à la fonction d'importance $h(x)$. La valeur d'un échantillon, qui est en fait l'indice dans la séquence VDC, est appelée *valeur de seuil*. À ce point-ci, nous avons un échantillon candidat ainsi qu'une valeur d'importance par segment de la séquence. Ces segments sont en fait les feuilles de la hiérarchie, obtenue par subdivision récursive du domaine initial.

L'algorithme complet de notre méthode est illustré par la figure 4.5 et le pseudo-code

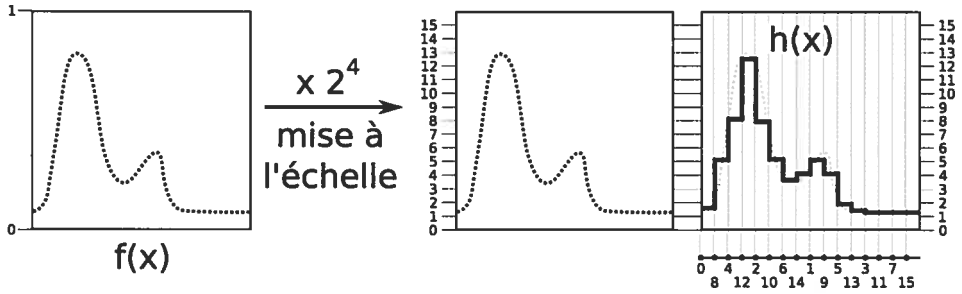


FIG. 4.4 – Définition de la fonction d'importance. La fonction $f(x)$ est mise à l'échelle en tenant compte de la base $b = 2$ de la séquence VDC et du niveau maximal de subdivision $L = 4$. À chacun des b^L éléments du domaine (tel que segmenté par la séquence), on assigne comme valeur d'importance la valeur moyenne de la fonction sur l'élément. La fonction d'importance résultante (illustrée par un trait noir continu), $h(x)$, est une approximation constante par morceau de $f(x)$ par sa moyenne, à un facteur d'échelle près.

de la fonction *Subdivise*() est également donné ci-après. À chaque niveau de subdivision, les segments sont illustrés par des boîtes bleues, dont la hauteur correspond à l'indice dans la séquence VDC, qui est également la valeur de l'échantillon. La fonction considérée (trait noir pointillé), $f(x)$, est visible à chaque niveau. Au niveau 4, la fonction d'importance (trait noir continu), est donnée. La dernière image sur la droite donne le résultat après rejet (rouge pour les candidats rejetés, vert pour les candidats

acceptés).

Fonction `Subdivise(`*nœud*`)`

Effectue une subdivision récursive d'un nœud de la hiérarchie. L est le niveau maximal de subdivision. MAX et AVG donnent le maximum local et la moyenne locale de la fonction. Le facteur d'échelle c permet de varier le nombre d'échantillons obtenus.

```
si nœud.niveau == L alors
    si nœud.seuil ≤ AVG(nœud) × c alors
        liste ← liste + PlaceEchantillon(nœud)
sinon
    si nœud.seuil ≤ MAX(nœud) × c alors
        pour chaque enfant faire
            enfant.niveau ← nœud.niveau + 1
            enfant.seuil ← CalculeSeuil(nœud, enfant)
            Subdivise(enfant)
```

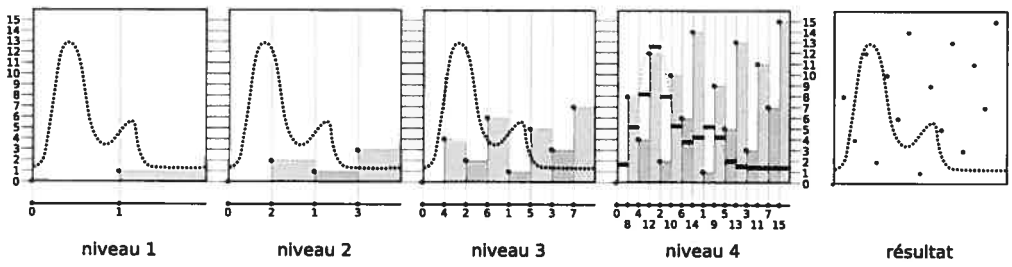


FIG. 4.5 – Seuillage hiérarchique. Les échantillons candidats (points bleus) sont générés par construction récursive de la séquence VDC et remplissent graduellement l'espace. La fonction $f(x)$ (en trait pointillé noir) est donnée pour chaque niveau de subdivision. Chaque segment de la séquence est illustré par une boîte bleue dont la hauteur correspond à l'indice dans la séquence. La fonction d'importance $h(x)$ (en trait plein noir) est donnée pour le niveau 4. La dernière image sur la droite présente, avec un code de couleur (rouge pour les candidats rejetés et vert pour ceux acceptés), le résultat de l'algorithme. Dans ce cas-ci le taux de rejet est de 11/16, soit 68,75%.

Nous avons exprimé, suivant une approche récursive, l'échantillonnage par rejet classique. Rappellons qu'il est nécessaire, comme pour toute méthode basée sur le rejet,

que $f(x)$ soit bornée. Nous ajoutons la contrainte que les bornes soient $[0;1]$, cependant toute fonction bornée peut être normalisée de façon à respecter cette contrainte additionnelle.

4.2.1 Identification des branches stériles

L'application directe de notre algorithme est foncièrement inefficace, puisqu'elle implique une évaluation de la fonction pour toutes les feuilles de la hiérarchie, ce qui revient à évaluer la totalité de la fonction. Bien que théoriquement valide, cette approche n'est donc pas applicable en pratique. Heureusement, comme avec l'échantillonnage par rejet classique, il est possible d'utiliser une fonction enveloppe, $q(x)$, de façon à limiter le nombre de candidats. Dans notre cas, ceci se fait en coupant des branches de la hiérarchie. Afin de garantir le bon fonctionnement de l'algorithme, il est nécessaire qu'aucune information ne soit perdue. Par conséquent, seules les branches stériles sont coupées. Une branche est considérée stérile si tous les candidats qu'elle produit sont rejetés.

La fonction enveloppe utilisée est simplement le maximum local, évalué individuellement pour chaque nœud de la hiérarchie. Au fil des subdivisions, cette valeur se raffine puisque le support des nœuds diminue. En couplant cette fonction enveloppe avec les valeurs de seuil des séquences VDC, il est facile d'identifier, sans ambiguïté possible, les branches stériles. Pour ce faire, nous exploitons une caractéristique implicite des séquences VDC : les valeurs de seuil croissent de façon monotone à chaque subdivision. La figure 4.6 illustre le résultat de l'application de ce principe. Prenons, par exemple, le nœud le plus à droite du niveau 2. On constate que la valeur de seuil de ce nœud (indiquée par la hauteur de la boîte) est supérieure au maximum local de $f(x)$ (indiqué par un trait plein bleu). À partir de ce point, au fil des subdivisions les valeurs de seuil ne feront que croître, alors que les valeurs locales de maximum vont décroître. Tous les nœuds auront donc un seuil supérieur au maximum local et, à plus forte raison, à la moyenne locale. Tous les candidats générés dans cette branche seront alors nécessairement rejetés, et elle peut être coupée.

Notez que, initialement, la fonction enveloppe et la fonction d'importance ne sont pas connues. Elles seront définies dynamiquement, durant la traversée de la hiérarchie, en même temps que les échantillons sont générés. En général, ces deux étapes (définition

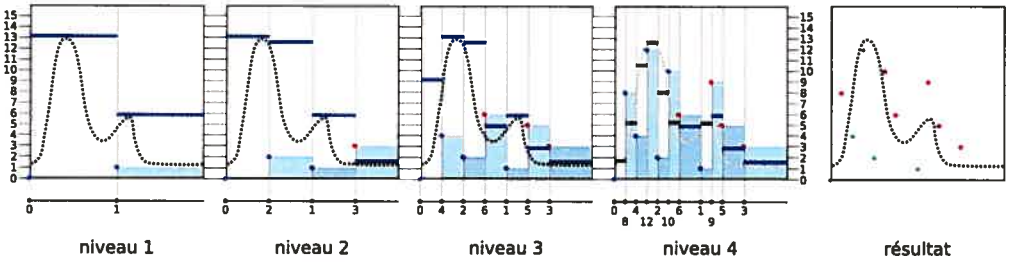


FIG. 4.6 – Seuillage hiérarchique efficace. À chaque niveau de la hiérarchie, on évalue pour chaque nœud le maximum local. Dès que la valeur de seuil d'un nœud est supérieure au maximum local (trait plein bleu), la branche est coupée (l'échantillon est immédiatement marqué en rouge, sans qu'il soit nécessaire d'évaluer la valeur d'importance). Au dernier niveau, pour chaque nœud dont la valeur de seuil est inférieure au maximum local, on évalue la valeur d'importance (trait plein noir). La dernière image sur la droite présente, avec un code de couleur (rouge pour les candidats rejetés et vert pour ceux acceptés), le résultat de l'algorithme. Dans ce cas-ci, le taux de rejet est de 6/11 (54,54%), au lieu de 11/16 (68,75%) pour la version non-hiérarchique.

des fonctions et échantillonnage) sont effectuées séquentiellement.

4.3 Extension à d dimensions

Notre algorithme de seuillage hiérarchique étant basé sur l'échantillonnage par rejet, il se généralise aisément à toutes les dimensions. Pour une fonction définie sur d dimensions, on échantillonne de façon uniforme l'espace de $d + 1$ dimensions en utilisant une séquence à basse discrépance appropriée. Chaque échantillon candidat est alors testé pour rejet en fonction de sa valeur dans la $(d+1)^{\text{ème}}$ dimension, c'est-à-dire la valeur de seuil obtenue avec la séquence.

Nous abordons ici le cas particulier de l'échantillonnage de fonctions définies sur deux dimensions. Ce cas est particulièrement répandu en infographie, où beaucoup de systèmes, y compris celui développé pour notre application de rendu photo-réaliste, utilisent de l'information encodée dans des images 2D. La distribution des échantillons candidats est obtenue en appliquant une séquence VDC en base 4 sur un schéma inspiré par celui de Bayer [Bay73], tel qu'illustré dans la figure 4.7. La représentation

hiérarchique est obtenue par subdivision récursive du domaine en quatre quadrilatères égaux. Il en résulte une structure pyramidale, déjà largement exploitée dans le domaine de l'infographie [Wil83]. Bien que la figure 4.7 utilise directement le schéma de Bayer pour indexer les enfants, en pratique, l'indexation est permutée aléatoirement de façon à éliminer le biais (voir section 4.4).

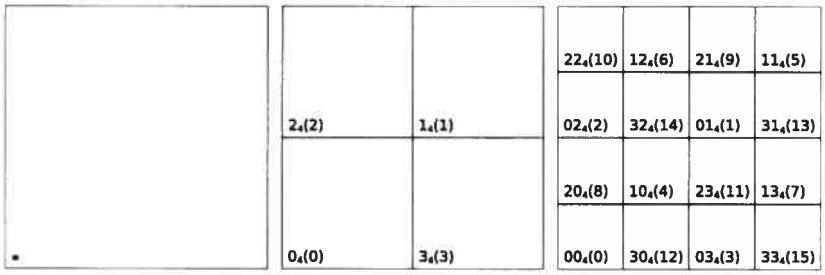


FIG. 4.7 – Échantillonnage uniforme d'un espace tri-dimensionnel, utilisant une séquence VDC en base 4 appliquée à un schéma de Bayer. Les valeurs des nœuds de l'arbre sont données en base 4, et en base 10 entre parenthèses. Les valeurs en base 10 sont obtenues en considérant le chiffre de poids fort à gauche.

4.4 Élimination du biais

Étant basées sur l'échantillonnage, les méthodes Monte Carlo ne donnent pas des résultats exacts. En infographie, la variance des résultats se traduit en bruit dans l'image. Cependant, si la méthode utilisée n'est pas biaisée, la couleur et l'intensité moyennes de l'image restent invariables et seule l'amplitude du bruit varie avec le nombre d'échantillons. Rappelons que notre approche appartient à la catégorie des méthodes quasi-Monte Carlo, puisqu'elle se base sur l'utilisation de nombres quasi-aléatoires, et qu'elle est par conséquent déterministe et nécessairement biaisée. La figure 4.8 montre des exemples d'images obtenues avec 16 et 64 échantillons par pixel, pour une distribution biaisée et l'autre non biaisée. On constate que le biais se traduit, visuellement, par un effet de palier dans les ombrages et par une couleur moyenne variant suivant le nombre d'échantillons. Ces deux artefacts visuels sont liés à la même cause : la distribution des échantillons varie peu d'un pixel à l'autre de l'image. Dans notre image, la normale au plan étant constante, seule la différence (minime) d'orien-

tation du vecteur vers la caméra affecte l'intégrale de la lumière réfléchie. Avec un processus déterministe, le positionnement des échantillons sera alors très similaire, comme illustré par la figure 4.9. Ce phénomène est communément appelé *effet rideau de douche*. C'est également la cause du changement de coloration que l'on observe entre les deux images. En pratique, ce changement de coloration est dû à l'apparition (ou à la disparition) soudaine d'un échantillon, dont l'intensité est suffisante pour que sa contribution soit remarquée.

Il est difficile de s'accommoder de biais en infographie. Si le résultat visuel peut être acceptable pour une image donnée, il l'est rarement pour une animation où les changements brusques dans l'ombrage et la couleur sont criants. Il est donc nécessaire d'éliminer, autant que possible, toute source de biais. Bien que, conceptuellement, il n'y ait qu'une seule source de biais dans notre algorithme (l'utilisation d'une séquence quasi-aléatoire), en pratique, on distingue trois éléments distincts qui contribuent au biais : l'attribution des codes aux différents nœuds de la hiérarchie, le positionnement des échantillons dans les nœuds, et la discrétisation des valeurs de seuil des nœuds. Chaque élément sera traité individuellement dans la suite de cette section.

4.4.1 Attribution des codes

Dans la section 2.1, nous avons donné l'expression mathématique, ainsi que la règle de production récursive, des séquences VDC. La règle de production est déterministe et donne une séquence fixe. C'est pour marquer l'aspect invariable des séquences à basse discrépance qu'on utilise le terme quasi-aléatoire, qui peut porter à confusion, puisque ces séquences n'ont en fait rien d'aléatoire. En pratique, le déterminisme des séquences VDC réside dans l'assignation des indices lors de la subdivision. Ainsi, pour les séquences VDC en base 2, l'enfant de gauche se voit toujours assigné l'indice 0, alors que l'enfant de droite se voit toujours assigné l'indice 1.

Il suffit, pour éliminer le déterminisme des séquences VDC, de permuter l'assignation des indices aux enfants, lors de la subdivision. Évidemment, la séquence obtenue n'est plus une séquence VDC à proprement parler, mais on conserve la propriété de basse discrépance et d'uniformité dans la distribution. Pour la séquence en base 2, il y a deux permutations possibles, pour le cas général de la base b, il y a $b!$ permutations possibles. La figure 4.10 illustre la distribution obtenue en plaçant les échantillons au

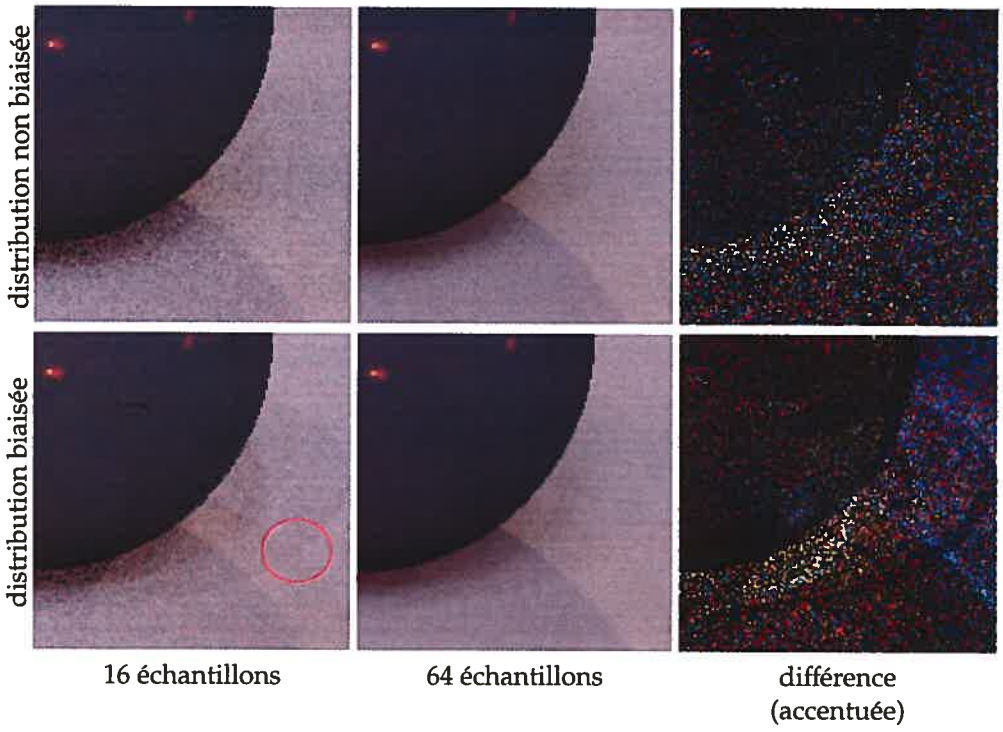


FIG. 4.8 – Effet du biais dans les distributions de points. Le niveau de bruit est moins élevé pour les images biaisées. Cependant, la couleur change suivant le nombre d'échantillons tel qu'on peut le voir dans l'image "différence", où les changements de couleurs, entre les images obtenues avec 16 et 64 échantillons, sont uniformes dans le cas biaisé et aléatoires dans le cas non biaisé. De plus, on constate des effets de paliers dans l'ombrage (le cercle rouge en indique un).

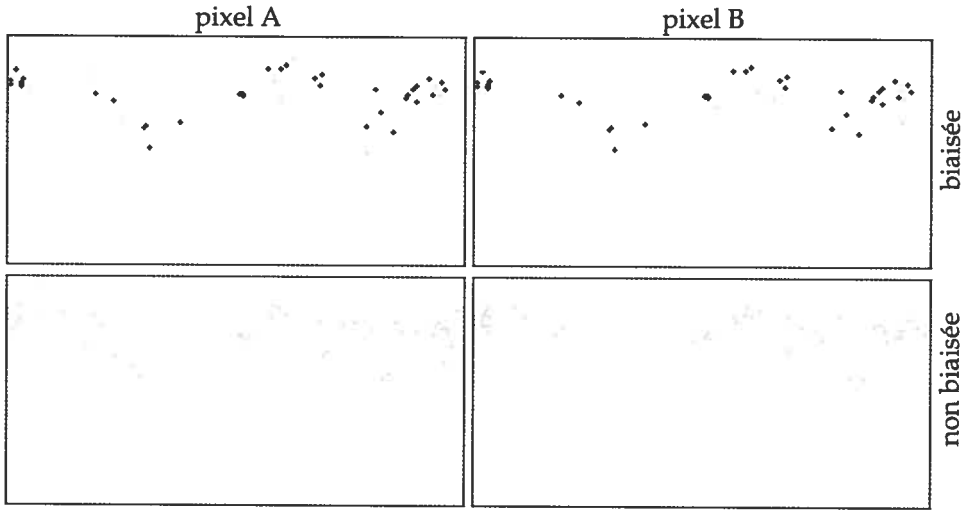
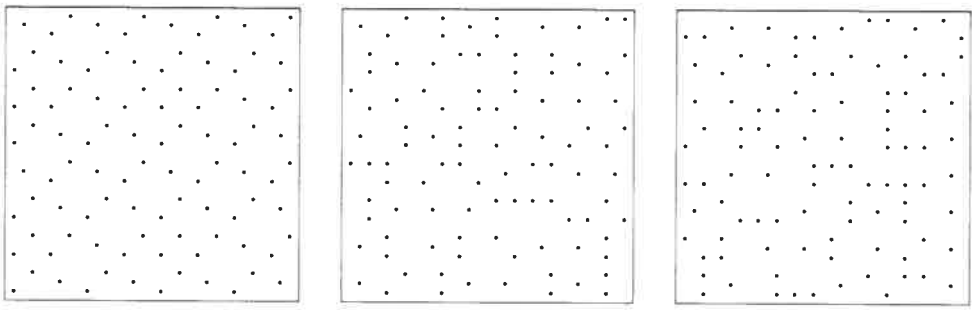


FIG. 4.9 – Échantillons obtenus pour deux pixels adjacents de la scène de la figure 4.9. Les deux pixels considérés sont sur le plan. La première ligne montre les échantillons obtenus par utilisation directe de la séquence VDC, alors que la seconde ligne montre les échantillons obtenus avec notre variante non biaisée. Dans le cas biaisé, les deux pixels partagent 80% de leurs échantillons (marqués en rouge). La qualité de la distribution non biaisée est, a priori, similaire à celle de la distribution biaisée.

centre des feuilles de la hiérarchie, en utilisant l'assignation suivant le schéma de Bayer et des permutations choisies aléatoirement.



Bayer	permutations aléatoires	permutations aléatoires
0,067225	0,06285	0,085975

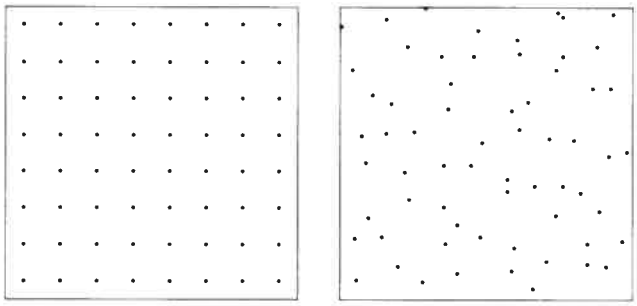
FIG. 4.10 – Effet des permutations sur la distribution de 100 points. L'image de gauche présente la distribution obtenue avec une séquence VDC en base 4 et une assignation des indices suivant le schéma de Bayer. Au milieu et à droite sont présentées des distributions obtenues en utilisant la même séquence, mais une assignation aléatoire des indices. Sous chaque image est également indiquée la discrépance étoile de la distribution (telle que définie dans la section 2.1).

4.4.2 Positionnement de l'échantillon

Dans la figure 4.10, les échantillons sont placés au centre des feuilles de la hiérarchie. Bien que les nœuds soient indexés aléatoirement, le pas de discrétisation n'en est pas moins visible. Plus les échantillons sont placés à un niveau profond de la hiérarchie et moins cet effet se fera sentir (et il disparaîtrait avec une profondeur infinie). La figure 4.11 illustre l'impact de la profondeur sur le positionnement des échantillons.

Il est nécessaire, pour que l'échantillonnage ne soit pas biaisé, que toute région du domaine puisse être échantillonnée. Si les échantillons sont positionnés de façon déterministe, seuls les sommets de la grille de discrétisation seront échantillonnés, ce qui implique qu'il faut une grille infiniment dense (ou une hiérarchie de profondeur infinie dans notre cas) pour que l'échantillonnage soit non biaisé.

En pratique, l'échantillon est simplement positionné aléatoirement dans le nœud dès que la densité est suffisante (à $L = 3$ dans l'exemple de la figure 4.11). Notez



$L = 3$ $L = 7$

FIG. 4.11 – Effet de la prodondeur de la hiérarchie sur la distribution de 64 points. Pour l'image de gauche, le niveau de subdivision maximal est fixé à $L = 3$, alors qu'il est fixé à $L = 7$ pour l'image de droite. Dans les deux cas, les 64 premiers échantillons (d'après les valeurs de seuil) sont tracés. Le pas de discrétisation est $2^4 = 16$ fois plus petit dans l'image de droite, et n'est déjà plus visible à l'œil nu.

que l'effet obtenu est exactement le même que si l'on avait placé les échantillons à une profondeur infinie (en assumant que l'indexation des enfants est faite de façon aléatoire).

4.4.3 Discrétisation des valeurs de seuil

La troisième et dernière source de biais provient de la discrétisation des valeurs de seuil. Comme mentionné, ces valeurs sont en fait des indices dans la séquence VDC, et donc des nombres entiers. Il est par conséquent impossible, avec un nombre fini d'échantillons, d'évaluer exactement une valeur donnée.

Pour illustrer ce problème, considérons l'intégrale d'une fonction. L'estimateur est trivial, on évalue simplement la proportion d'échantillons candidats sélectionnés, qui approxime l'aire sous la courbe et donc l'intégrale de la fonction. Soit une fonction constante $f(x) = 0,45$. On veut évaluer l'intégrale de cette fonction pour $0 \leq x \leq 1$ en utilisant une séquence VDC en base 2. Le résultat est donné pour différents niveaux de subdivision dans le tableau 4.1. Pour un niveau donné, le taux d'acceptation est constant, par contre il varie d'un niveau à l'autre. De plus, on observe un effet d'aliassage, puisque le taux d'acceptation varie par sauts et peut rester inchangé d'un niveau à l'autre.

Il suffit, pour éliminer l'effet de la discrétisation, d'ajouter à chaque valeur de seuil

Niveau de subdivision	Nombre de candidats	Valeur de $f(x)$ normalisée	Taux d'acceptation
3	8	3,6	0,5
4	16	7,2	0,5
5	32	14,4	0,46875
6	64	28,8	0,45313
7	128	57,6	0,45313
8	256	115,2	0,45313
9	512	230,4	0,45117

TAB. 4.1 – Évaluation de l'intégrale d'une fonction constante $f(x) = 0,45$ en utilisant une séquence VDC en base 2. Le taux d'acceptation varie suivant le niveau de subdivision utilisé, mais il est constant pour un niveau donné.

un nombre aléatoire $\in [0;1)$.

4.5 Extension au produit de plusieurs fonctions

Notre méthode peut s'appliquer à toute fonction $f(x)$. En particulier, il pourrait s'agir du produit de plusieurs fonctions $f_i(x)$:

$$f(x) = \prod_i f_i(x).$$

Une étape de précalcul est effectuée. Pour chaque fonction considérée, la moyenne et le maximum sont évalués pour chaque nœud de la hiérarchie. Ce précalcul est limité à une certaine profondeur de la hiérarchie, au-delà de laquelle la fonction est considérée constante.

Dans le cas où une seule fonction est considérée, les valeurs précalculées du maximum et de la moyenne sont directement utilisées dans l'algorithme de seuillage hiérarchique. Cependant, dans les cas où plusieurs fonctions sont impliquées, une estimation est calculée dynamiquement, pendant la traversée de l'arbre. Pour un nœud j, le maximum et la moyenne locaux du produit sont respectivement approximés par le produit des maxima et moyennes locaux de chaque fonction :

$$\begin{aligned} Max_j(f) &\approx \prod_i Max_j(f_i) \\ Avg_j(f) &\approx \prod_i Avg_j(f_i). \end{aligned}$$

Dans le cas du maximum, il s'agit d'une estimation conservatrice, ce qui est essentiel, puisque cette valeur sera utilisée pour couper des branches de la hiérarchie. Notez aussi que, plus on descend dans la hiérarchie, plus ces estimations sont précises. Puisque l'algorithme ne coupe pas les branches considérées importantes, la précision des estimations sera meilleure pour les régions contribuant le plus à l'intégrale du produit.

4.6 Nombre d'échantillons

Il est souhaitable d'être en mesure de spécifier le nombre d'échantillons utilisés pour une estimation Monte Carlo. Cela est cependant problématique dans le cas des méthodes par rejet. En effet, deux facteurs influent sur le nombre d'échantillons obtenus : le nombre de candidats, et l'intégrale de la fonction d'importance. Dans notre cas, le nombre de candidats est lié au facteur de subdivision b, ainsi qu'au niveau maximal de subdivision L. On note la fonction d'importance $h(x)$, et son intégrale H, et on obtient l'expression suivante pour l'espérance de N, le nombre d'échantillons obtenus :

$$E[N] = H \times b^L. \tag{4.1}$$

Puisque $h(x)$ est une fonction constante par morceau qui approxime $f(x)$, il est possible de modifier le nombre d'échantillons obtenus en multipliant $f(x)$ par une constante c. Soit $F = \int f(x)\,dx$, on a $H \approx cF$, qui peut être utilisé dans l'équation 4.1 pour obtenir c :

$$c \approx \frac{N}{F \times b^L}. \tag{4.2}$$

Il est important de rappeler que l'on assume que la fonction échantillonnée est bornée par $[0;1]$. Par conséquent, bien que l'on puisse avoir $c > 1$, il faut s'assurer que $0 \leq cf(x) \leq 1$, pour que le test de rejet soit valide.

La constante c est définie en fonction de F, or c'est justement cette intégrale que l'on cherche à calculer. Heureusement, une valeur approximative suffit pour nos besoins. Bien évidemment, la valeur de F étant approximative, N sera différent de la valeur attendue. Si cette différence est supérieure à un certain seuil, il est possible de raffiner c itérativement :

$$c_{t+1} = c_t \times N/M_t \tag{4.3}$$

avec N le nombre d'échantillons désirés et M_t le nombre d'échantillons obtenus à l'itération t.

Dans notre application, nous fixons $L = 15$, qui correspond au niveau maximal où les valeurs de seuils peuvent être encodées avec 32 bits. Nous utilisons la profondeur maximale, de façon à pouvoir traiter une grande variété de cas, y compris ceux où l'intégrale évaluée est extrêmement petite. F est estimé en subdivisant deux fois le nœud racine, puis en évaluant la valeur moyenne de $h(x)$ pour tous les nœuds de ce niveau. En utilisant une tolérance de 20%, le nombre moyen d'itération est d'environ 1,3 et dépasse très rarement 2. Bien sûr, si l'on échantillonne une seule fonction, et non pas un produit, l'intégrale de la fonction est connue (grâce au précalcul) et le nombre d'itération moyen est très proche de 1.

4.7 Estimateur Monte Carlo

Notre méthode utilise l'échantillonnage par rejet pour distribuer les échantillons suivant une certaine fonction d'importance. L'estimateur Monte Carlo pour l'échantillonnage suivant l'importance est :

$$F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)} \tag{4.4}$$

où $p(x)$ est la densité de probabilité (PDF) qui sert de fonction d'importance. Une PDF a deux caractéristiques : $p(x) \geq 0$ et $\int p(x)\, dx = 1$. Notre fonction d'importance, $h(x)$, satisfait à la première exigence, mais n'intègre pas à 1. Il est donc nécessaire de la normaliser par rapport à son intégrale H. Puisque $h(x)$ est définie comme étant l'union de la valeur moyenne de $f(x)$ pour chaque feuille de la hiérarchie (voir section 4.2), son intégrale est simplement la somme des contributions de chaque feuille :

$$H = \frac{1}{b^L} \sum_{i=1}^{m} h_i \times b^{L-l_i} = \frac{c}{b^L} \sum_{i=1}^{m} Avg_i(f) \times b^{L-l_i} \tag{4.5}$$

où h_i and $Avg_i(f)$ sont, respectivement, la valeur de $h(x)$ et la valeur moyenne de $f(x)$ pour le $i^{\text{ème}}$ nœud, l_i est le niveau du nœud dans la hiérarchie, m le nombre de feuilles de la hiérarchie, et c est le facteur d'échelle définit dans la section 4.6. L'aire relative d'un nœud par rapport au niveau maximal de subdivision est donnée par b^{L-l_i}, et est utilisée

pour pondérer la contribution de chaque nœud. L'estimateur Monte Carlo utilisé est :

$$F_N = \frac{H}{N} \sum_{i=1}^{N} \frac{f(x_i)}{h(x_i)}. \tag{4.6}$$

Chapitre 5

Application

Nous avons choisi d'appliquer notre méthode d'échantillonnage au problème d'évaluation de l'éclairage direct en un point. Ce sujet a été largement traité durant les dernières années, aussi bien dans le contexte des applications interactives, que dans celui du rendu photo-réaliste (voir chapitre 3).

Notre implémentation se base sur les travaux de Debevec [Deb98] sur l'éclairage à base d'images. Le principe de cette méthode est d'approximer les éléments éloignés de la scène par un modèle de lumière, et de modéliser uniquement la géométrie et les interactions des éléments locaux de la scène. Le modèle de lumière des éléments distants est obtenu en créant des cartes omnidirectionnelles encodant un large spectre dynamique (HDR). Ces cartes sont construites en photographiant sous différents points de vue, et à différents temps d'exposition, une sphère réflective placée au point d'intérêt. Pour plus d'information, reportez-vous aux travaux de Debevec et Malik [DM97], sur la création d'images à large HDR, et de Debevec [Deb98], sur la création de cartes omnidirectionnelles. À titre d'exemple, nous présentons, dans la figure 5.1, la carte d'environnement de la cathédrale de Grâce, qui est utilisée dans la plupart des résultats que nous présentons. Cette carte et d'autres peuvent être obtenues à ces adresses : http ://debevec.org/Probes/ et http ://gl.ict.usc.edu/Data/HighResProbes/.

Cette approche est généralement utilisée pour intégrer des éléments virtuels dans des scènes réelles. Plus récemment, Debevec et al. [DHT+00] ont développé une méthode de capture de la réflectance d'un visage permettant d'intégrer des personnages virtuels photo-réalistes dans des scènes filmées. Le volume d'information contenu dans ces cartes omnidirectionnelles est considérable et ne cesse d'augmenter (à titre d'exemple,

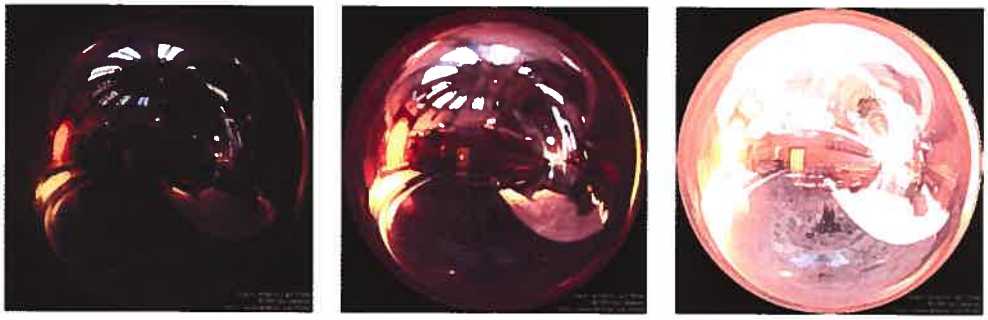


FIG. 5.1 – Carte d'environnement omnidirectionnelle de la cathédrale de Grâce. Le large spectre dynamique encodé permet de récupérer aussi bien l'information des zones claires (à gauche), que celle des zones sombres (à droite).

l'équipe de Paul Debevec a récemment rendu publique une carte de 18 méga pixels). Il n'est évidemment pas possible de tenir compte de la totalité de l'information contenue dans l'image et, si les premières approches utilisaient un sous-échantillonnage régulier pour limiter l'ampleur du calcul à effectuer, des méthodes d'échantillonnage suivant l'importance ont été développées depuis et donnent d'excellents résultats à moindre coût.

5.1 Projections

Une variété de projections ont été utilisées afin d'encoder de l'information omnidirectionnelle sur un plan. Nous présentons succintement les principales, ainsi que celle que nous avons choisi d'utiliser.

La projection panoramique est sans doute la plus courante en infographie, du moins dans le contexte du rendu photo-réaliste. Il s'agit d'une projection sur un domaine rectangulaire, où la longitude d'un point est reportée sur l'axe horizontal, alors que la latitude est reportée sur l'axe vertical. Elle a été souvent utilisée dans des travaux d'échantillonnage de l'éclairage direct, notamment dans [KK03, ODJ04, CJAMJ05, CETC06]. C'est une projection intuitive, qui a l'avantage d'être rectangulaire et donc de se prêter facilement au traitement numérique. Son principal inconvénient, qui est aussi celui de la plupart des autres projections, est la distorsion qu'elle cause, particulièrement dans les zones polaires. Aussi, les éléments discrets de cette projection n'ont pas une aire constante (ils ne couvrent pas le même angle solide).

La projection sur les faces d'un cube est principalement utilisée pour les applications interactives. Cette projection est d'ailleurs supportée directement sur les cartes graphiques actuelles. La distorsion est moindre que dans le cas du paramétrage panoramique, mais l'aire des éléments discrets n'est toujours pas constante. De plus, il existe des discontinuités aux bordures des faces du cube.

La projection sphérique correspond à la vue d'une scène se reflétant sur une sphère miroir. Elle est également supportée directement sur les cartes graphiques. Il est intéressant de noter que c'est la base même de la construction des cartes omnidirectionnelles produites par Debevec et al. [Deb98]. La projection angulaire est une variante très proche, où la distance radiale au centre est directement proportionnelle à l'angle couvert, offrant une résolution plus uniforme. La distorsion est, dans les deux cas, considérable à la périphérie de la projection. Les premières cartes d'environnement produites par Debevec étaient données suivant la projection angulaire, mais les dernières utilisent la projection panoramique.

Moins fréquente, la projection octaédrique offre une distorsion moindre, à la manière de la projection sur le cube. Son domaine de projection a l'avantage d'être carré (contrairement au cube, qui encode séparément les six faces), et elle a récemment été utilisée pour cette raison par Wang et al. [WZS+06], puisque cela permettait l'application directe de la compression en ondelettes de Haar.

Enfin, il existe la projection parabolique, qui a été introduite par Heidrich et Seidel [HS98]. Cette projection a été définie pour répondre au problème de sous-échantillonnage à la périphérie de la projection sphérique. Elle est également conçue de façon à permettre une projection rapide depuis l'espace 3D et d'être intégrable facilement dans les cartes graphiques.

5.1.1 HEALPix

Nous avons choisi, pour notre implémentation, d'utiliser la projection HEALPix (*Hierarchical Equal-Area isoLatitude Pixelisation*). Le nom même de cette projection souligne ses caractéristiques clés, qui sont la raison de notre choix. Il s'agit donc d'une projection pouvant être exprimée de façon hiérarchique, ce qui est évidemment essentiel au bon fonctionnement de notre algorithme. Elle permet une discrétisation en pixels d'aire égale. Par aire égale, nous entendons que l'angle solide couvert par chaque pixel de la

FIG. 5.2 – Projection HEALPix (*Hierarchical Equal-Area isoLatitude Pixelisation*) pour $N = 1, 2, 4$. Avec $N = 1$, la sphère est subdivisée en 12 quadrilatères curvilignes d'aire égale. Il est possible de passer d'un niveau à l'autre par subdivision récursive, créant la représentation hiérarchique, qui a donné son nom à la projection. On constate la faible distorsion des pixels de la projection, due à un traitement distinct des zones polaires (projection pseudo-cylindrique) et de la zone équatoriale (projection cylindrique).

projection est constant. Cette caractéristique simplifie le processus d'échantillonnage, puisqu'il n'est pas nécessaire de tenir compte d'un quelconque facteur de forme pour pondérer la contribution d'un pixel donné. Enfin, les centres des pixels sont répartis sur des anneaux parallèles de latitude constante. Cette caractéristique aura un impact sur le processus d'interpolation et l'encodage de données hémisphériques (comme les BRDFs). La projection HEALPix a déjà été utilisée dans le domaine de l'infographie par Wan et al. [WWL05].

La figure 5.2 illustre les trois premiers niveaux de subdivision de la projection HEALPix. Cette projection a été initialement développée dans le domaine de l'astrophysique par Górski et al. [GHB+05], pour le traitement de données omnidirectionnelles. Il s'agit en fait d'une famille de projections ayant toutes les mêmes caractéristiques. La variante que nous utilisons, et que nous appelons simplement projection HEALPix, est décrite en détail dans l'article de Górski et al. Une analyse mathématique plus poussée de la projection, ainsi qu'une présentation de certaines variantes, ont été faites par Calabretta et Roukema [CR04].

La projection HEALPix distribue, aussi uniformément que possible, $12N^2$ points sur la sphère unitaire. En fixant $N = 1, 2, 4$, on obtient les trois images de la figure 5.2. Pour $N = 1$, on a une subdivision de la sphère en 12 faces, et N correspond en fait au côté (en nombre de pixels) de chaque face de la projection. Dans la figure, les points bleus indiquent les centres des pixels, et sont alignés sur des anneaux parallèles de latitude

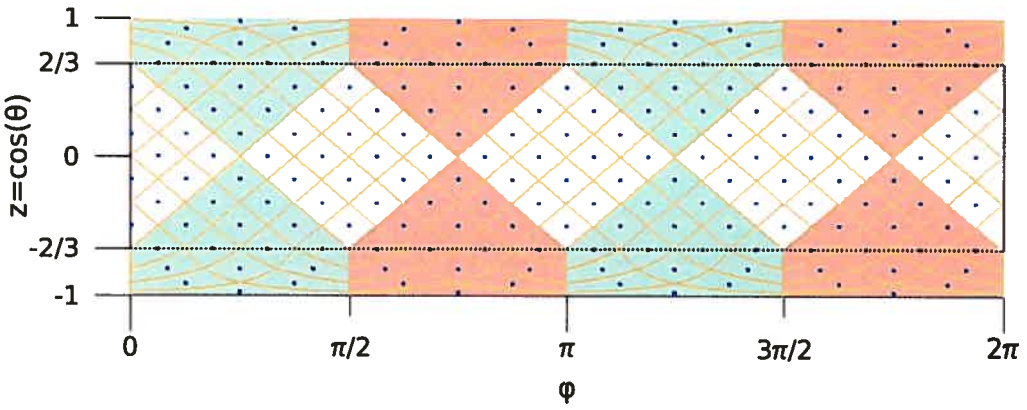


FIG. 5.3 – Projection HEALPix en coordonnées cylindriques pour $l = 2$. On constate la répartition équidistantes des centres des pixels sur des anneaux parallèles de latitude constante. La discontinuité dans la projection, entre les zones polaires et la zone équatoriale, à $|z| = 2/3$ est également visible.

fixe. Il est possible d'aller d'un niveau à l'autre par subdivision récursive des pixels. Ainsi, en notant l le niveau de subdivision, on a $N = 2^l$.

La latitude est mesurée par la coordonnée Z, ou encore l'angle θ, avec $z = \cos(\theta)$. La longitude est mesurée par l'angle ϕ à partir de l'axe X. On dénote trois sections dans la projection, les deux zones polaires et la zone équatoriale. Pour la zone équatoriale, la projection HEALPix est en fait une simple projection cylindrique avec préservation de l'aire Pour les zones polaires, il s'agit d'une projection pseudo-cylindrique minimisant la distorsion, assurant ainsi une meilleure uniformité des pixels. La figure 5.3 illustre la projection HEALPix dans un système de coordonnées cylindriques, pour $l = 2$. La discontinuité dans la projection, pour $|z| = 2/3$, apparaît clairement dans cette figure. La disposition des pixels, suivant des anneaux parallèles de latitude fixe, est également visible.

Les anneaux de la projection HEALPix ont quelques caractéristiques intéressantes. Il en existe $(4N-1)$, et les centres des pixels sur un anneau sont équidistants. L'indice d'un anneau, noté i, est calculé à partir du pôle nord, avec $i \geq 1$. En raison de la discontinuité dans la projection, les zones polaires et la zone équatoriale ont des caractéristiques différentes. Dans la zone équatoriale, il y a $4N$ pixels par anneau, alors qu'il y a $4i$ pixels par anneau dans la zone polaire nord. La zone polaire sud est caractérisée de façon

symétrique à la zone nord. La coordonnée z est définie par :

$$z = \begin{cases} \frac{4}{3} - \frac{2i}{3N} & \text{dans la zone équatoriale} \\ 1 - \frac{i^2}{3N^2} & \text{dans la zone polaire nord.} \end{cases}$$

Il existe deux systèmes de coordonnées permettant d'identifier un pixel de la projection HEALPix. Le premier système de coordonnées, que nous appellerons *par face*, identifie un pixel en spécifiant la face à laquelle il appartient, les coordonnées (u, v) dans la face, ainsi que le niveau de subdivision l. Les faces sont indexées, du nord au sud, suivant les anneaux, à partir de 0. Les coordonnées d'un pixel, dans le système par face, sont de la forme $(face, u, v, l)$. La figure 5.4 illustre cette notation. Il est facile de quantifier l'écart par rapport au centre d'une face à l'aide des coordonnées (u, v). Horizontalement, l'écart est de $(u - v)$, alors que verticalement, il est de $(u + v)$. Ceci permet de retrouver très rapidement la longitude ϕ d'un pixel, par rapport au centre de la face à laquelle il appartient. En effet, les pixels étant équidistants sur un anneau, il suffit d'évaluer l'écart horizontal $(u - v)$ et de le multiplier par le demi-pas entre deux pixels. De même, il est facile de retrouver l'indice de l'anneau sur lequel un pixel est situé, en additionnant l'écart vertical $(u + v)$ à l'indice du premier anneau d'une face. Ces deux mesures permettent de reprojeter efficacement un pixel HEALPix en 3D.

Le second système de coordonnées, que nous appellerons *par anneau*, identifie un pixel en donnant sa position j dans l'anneau i sur lequel il est situé, ainsi que le niveau de subdivision l. Les coordonnées d'un point, dans le système par anneau, sont de la forme (i, j, l). Comme nous le verrons plus loin, ce système est particulièrement approprié lorsque l'on veut interpoler les données, puisque l'interpolation se fait suivant les anneaux et non suivant les faces.

5.2 Représentation des différentes fonctions

Notre objectif est d'évaluer, par échantillonnage suivant l'importance, l'illumination directe en un point de la scène, définie par l'équation 3.1. Les différentes méthodes d'échantillonnage proposées jusqu'à présent (voir chapitre 3) tenaient compte de l'éclairage (encodé dans une carte d'environnement), de la réflectance (BRDF), ou du produit des deux. Dans notre implémentation, nous proposons une approche simple permettant de définir une carte de visibilité approximative, qui sera également prise en compte

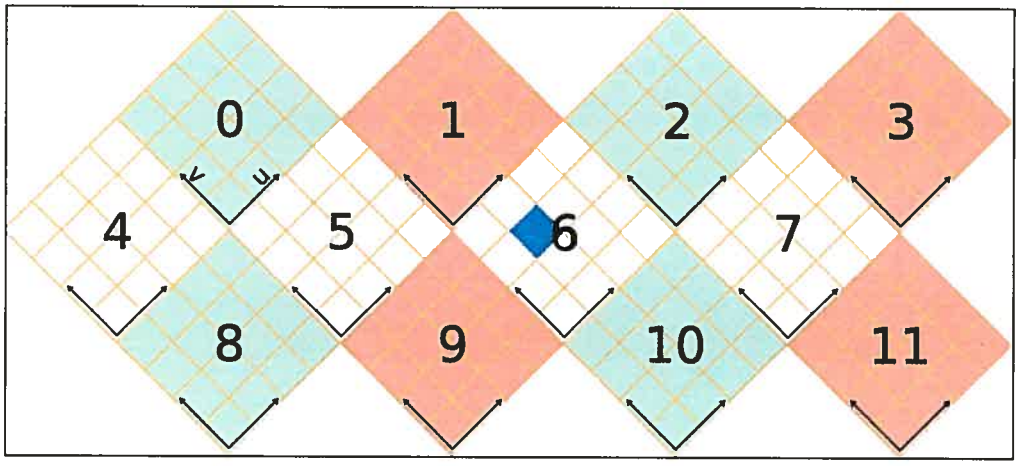


FIG. 5.4 – Aplat de la projection HEALPix pour le niveau de subdivision $l = 2$. Les faces sont indexées du nord au sud, suivant les anneaux. La base des coordonnées (u, v) est la même pour chaque face. Les coordonnées d'un pixel dans la hiérarchie sont de la forme $(face, u, v, l)$. Le pixel $(6, 1, 2, 2)$ est dessiné en bleu.

dans l'évaluation de la fonction d'importance (voir section 5.4).

Dans cette section, nous abordons la représentation de chaque fonction. Notre approche étant basée sur le précalcul, il est nécessaire d'enregistrer un volume conséquent de données, de façon aussi compacte que possible. Les données sont directement encodées suivant la projection HEALPix de façon à faciliter leur traitement. Pour chaque fonction considérée, le maximum local et la moyenne locale de la fonction sont évalués pour chaque pixel de la hiérarchie HEALPix, jusqu'à une certaine profondeur prédéfinie.

Dans notre implémentation, l'échantillonnage est effectué dans le référentiel global, et chaque face de la projection HEALPix est traitée séparément. En pratique, le seul point notable, par rapport à l'algorithme déjà présenté, est le fait que les données de la BRDF doivent être interpolées. Les détails seront donnés lorsque nous discuterons de la représentation de la BRDF.

5.2.1 Environnement

La carte d'environnement encode l'éclairage distant dans toutes les orientations. Puisqu'il s'agit d'un éclairage considéré à l'infini, cette information est constante

FIG. 5.5 – Projection HEALPix de la carte d'environnement de la cathédrale de Grâce. Le niveau de distorsion est appréciable, par rapport à celui observé dans la figure 5.1.

en tout point de la scène, et est donc exprimée dans un référentiel global. Puisque l'échantillonnage est effectué dans le référentiel global, les valeurs locales de moyenne et de maximum sont accessibles directement pour l'environnement. La figure 5.5 donne la projection de la carte d'environnement de la cathédrale de Grâce, dans le format HEALPix. En mémoire, la carte est enregistrée dans le système de coordonnées par face. Ce système est plus approprié puisqu'il offre une correspondance directe avec la hiérarchie utilisée pour l'échantillonnage.

5.2.2 BRDF

La BRDF définit la réflectance d'un matériau et est supposée uniforme sur toute la surface d'un objet. Nous ne traitons pas de fonctions variant spatialement ou temporellement parce que, notre méthode reposant sur le précalcul, le volume de données serait trop imposant et le temps de calcul serait prohibitif. Dans le chapitre 7, où nous abordons les travaux futurs, quelques pistes pour traiter ce type particulier de fonctions sont données.

Une BRDF encode généralement la réflectance d'un matériau dans une table 4D indexée par des paires de coordonnées sphériques (θ_i, ϕ_i) (vecteur incident, depuis la source de lumière), et (θ_o, ϕ_o) (vecteur sortant, vers la caméra). L'implémentation actuelle de notre système supporte uniquement les BRDFs isotropiques, qui peuvent

être réduites à des fonctions 3D dépendant de θ_i, θ_o, et $|\phi_i - \phi_o|$. Nous enregistrons des *tranches* de la BRDF en mémoire. Chaque tranche correspond à une carte 2D (θ_i, ϕ_i), pour une valeur de θ_o. La BRDF étant une caractéristique de la surface, elle est naturellement exprimée dans le référentiel de la surface (ou encore, le référentiel local). Chaque tranche est initialement définie dans le système de coordonnées par face, puis les valeurs sont transférées dans le système par anneau. L'orientation $\theta = 0$ correspond au pôle Nord dans la projection HEALPix, et seuls les anneaux de l'hémisphère nord sont conservés, puisque pour les autres la réflectance est nulle (l'hémisphère sud est sous la surface).

5.2.3 Visibilité

La carte de visibilité est évaluée durant l'exécution du programme, pour chaque pixel. Afin d'accélérer le processus d'échantillonnage, la visibilité est calculée dans le référentiel global, et est donc enregistrée, tout comme l'environnement, dans le système de coordonnées par face, de façon à pouvoir indexer directement un pixel dans la hiérarchie. Les détails de l'évaluation sont donnés dans la section 5.4.

5.3 Rotations et interpolation

L'échantillonnage se fait dans le référentiel global. L'environnement et la visibilité sont également définis dans le référentiel global. Pour ces deux fonctions, pour chaque nœud de la hiérarchie, il est possible d'obtenir directement les valeurs de la moyenne et du maximum. Par contre, ce n'est pas le cas de la BRDF, puisqu'elle est définie dans le référentiel local de la surface. Pour obtenir ces valeurs, il est nécessaire d'effectuer une rotation, pour se placer dans le référentiel local, puis d'interpoler la moyenne, ou d'évaluer le maximum d'après les pixels adjacents.

Pour chaque rayon primaire envoyé, il est nécessaire de définir la rotation permettant de passer du référentiel global à celui de la surface. Puisque nous traitons uniquement des BRDFs isotropiques, l'orientation du référentiel local n'a pas d'importance. Il est facile d'en définir un qui soit valide, à partir de la normale $\vec{n}$ à la surface. On commence par définir un vecteur arbitraire $\vec{v}(-n_y, n_z, n_x)$. On définit ensuite la bitangente $\vec{b} = \vec{n} \times \vec{v}$. Étant donnée la définition de $\vec{v}$, on est garanti que le $\vec{b}$ ne sera pas dégénéré. Enfin, on définit la tangente $\vec{t} = \vec{b} \times \vec{n}$. De toute évidence, pour une BRDF

anisotropique, cette approche n'est pas possible, et il serait alors nécessaire d'avoir une paramétrisation valide de la surface. Dans le référentiel local, la normale à la surface pointe vers le pôle Nord de la projection, et correspond donc à $\vec{z}$. On obtient directement la rotation permettant de passer dans le référentiel local :

$$\mathbf{R}_n = \begin{pmatrix} \vec{t} \\ \vec{b} \\ \vec{n} \end{pmatrix} = \begin{pmatrix} t_x & t_y & t_z \\ b_x & b_y & b_z \\ n_x & n_y & n_z \end{pmatrix}.$$

Les valeurs des tranches de la BRDF ont été calculées en assumant $\phi_o = 0$, ce qui n'est pas nécessairement le cas. Pour compenser, il suffit d'effectuer une rotation additionnelle, autour de $\vec{z}$, de $-\phi_o$. Avec $\vec{C} = \mathbf{R}_n \times \vec{c}$, la direction vers la caméra dans le référentiel local, on a $\phi_o = atan2(C_y, C_x)$. La rotation totale, $\mathbf{R}$, à effectuer est donc :

$$\mathbf{R} = \begin{pmatrix} \cos(-\phi_o) & -\sin(-\phi_o) & 0 \\ \sin(-\phi_o) & \cos(-\phi_o) & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} t_x & t_y & t_z \\ b_x & b_y & b_z \\ n_x & n_y & n_z \end{pmatrix}. \tag{5.1}$$

La rotation étant définie, il nous faut maintenant calculer la moyenne et le maximum de la BRDF pour un nœud de la hiérarchie. L'échantillonnage est effectué dans le référentiel global, suivant la hiérarchie de la projection HEALPix. Pour notre algorithme, nous avons besoin de la moyenne et du maximum de la BRDF, dans le référentiel global. Pour ce faire, il suffit de projeter un pixel dans le référentiel local, et d'évaluer la moyenne et le maximum. Puisqu'un pixel, après rotation, peut en couvrir plusieurs, les valeurs ne peuvent être obtenues directement. La moyenne est interpolée, d'après les pixels avoisinants, et le maximum est ajusté de manière à couvrir tout le voisinage et demeurer conservateur. La figure 5.6 illustre la situation.

5.3.1 Interpolation

Il peut sembler intuitif de considérer les différentes faces de la projection HEALPix comme une série d'images, et d'appliquer un schéma d'interpolation bilinéaire classique, mais ce ne serait pas approprié. Outre le fait que l'interpolation aux frontières des faces soit problématique, il faudrait aussi tenir compte des différentes discontinuités dans la projection (aux frontières des zones, à |z| = 2/3, et à la frontière des faces dans les zones polaires).

En pratique, l'interpolation se fait suivant la répartition des pixels sur les anneaux. La figure 5.8 illustre le procédé. Le point d'intérêt (ϕ_i, z_i) est indiqué en rouge. La

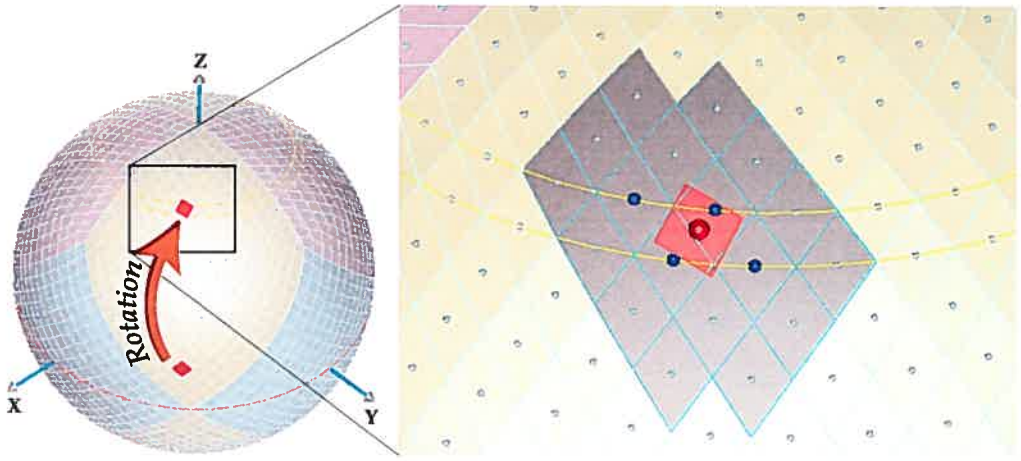


FIG. 5.6 – Rotation d'un pixel de la hiérarchie. Après rotation, un pixel peut en couvrir plusieurs. La valeur moyenne après rotation est obtenue en interpolant les valeurs des pixels avoisinants dans la projection HEALPix, alors que le maximum est élargi, de façon à rester conservateur.

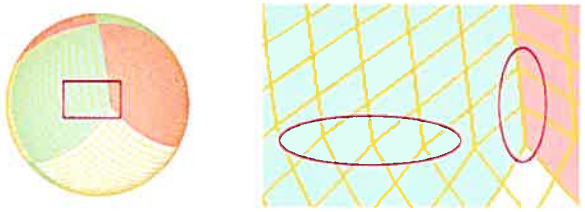


FIG. 5.7 – Interpolation dans la projection HEALPix. Des discontinuités (identifiées par des ellipses rouges) dans la projection empêchent l'utilisation d'un schéma d'interpolation bilinéaire classique.

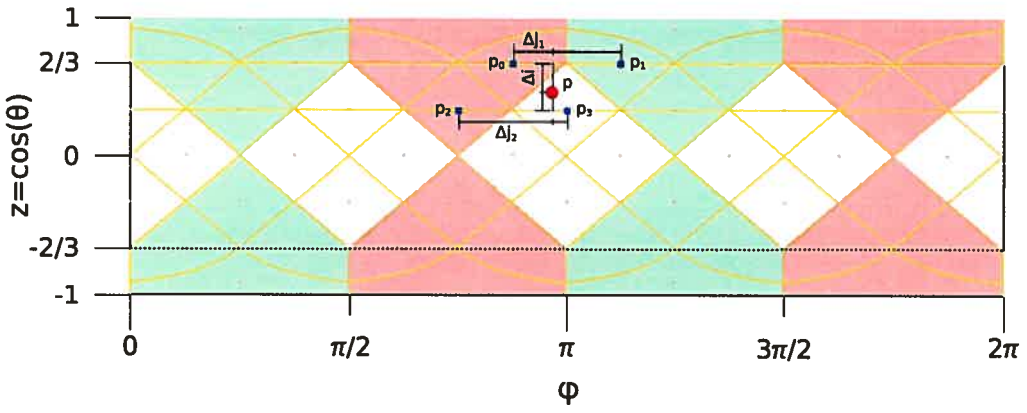


FIG. 5.8 – Interpolation dans la projection HEALPix. Afin d'obtenir la valeur au point p, on interpole les valeurs des points voisins sur les anneaux adjacents, p_0, p_1, p_2, et p_3. On effectue une interpolation bilinéaire, avec les pondérations Δj_1, Δj_2, et Δi.

coordonnée z_i est déjà connue, puisque la rotation est effectuée dans l'espace vectoriel ; par contre, la coordonnée $\phi_i = \arctan(y_i, x_i)$ doit être calculée. Étant donné le nombre de rotations effectuées, il est préférable d'utiliser une approximation pour le calcul de ϕ_i.

Il nous faut, tout d'abord, trouver les indices des deux anneaux adjacents. Étant donné un point de coordonnée z, au niveau de subdivision l, l'indice i de l'anneau au-dessus est :

$$i = \begin{cases} 2^l \times \sqrt{3 \times (1-z)} & \text{si } z > 2/3 \text{ (zone polaire nord)} \\ 2^l \times (2 - 1{,}5 \times z) & \text{si } |z| \leq 2/3 \text{ (zone équatoriale)} \\ 4 \times 2^l - \left(2^l \times \sqrt{3 \times (1+z)}\right) & \text{si } z < 2/3 \text{ (zone polaire sud).} \end{cases} \quad (5.2)$$

La valeur de i obtenue est en virgule flottante, la partie entière est l'indice de l'anneau, alors que la partie fractionnaire est utilisée comme facteur d'interpolation Δi. L'indice de l'anneau du dessous est la partie entière de $i + 1$.

Il nous faut maintenant trouver les pixels à interpoler. Puisque le nombre de pixel sur un anneau est connu et que ceux-ci sont équidistants, il est facile de trouver le pas, ϕ_Δ, entre deux pixels de l'anneau i. Rappelons qu'au niveau de subdivision l, nous

avons $1 \leq i < 4N$, avec $N = 2^l$.

$$\phi_\Delta = \begin{cases} 2\pi/4i & \text{si } z > 2/3 \text{ (zone polaire nord)} \\ 2\pi/4N & \text{si } |z| \leq 2/3 \text{ (zone équatoriale)} \\ 2\pi/(4 \times (4N - i)) & \text{si } z < 2/3 \text{ (zone polaire sud).} \end{cases}$$

En fonction de la zone considérée, la coordonnée ϕ_i du premier pixel d'un anneau varie. Pour les zones polaires, le premier pixel est à la position $\phi_i = \phi_\Delta/2$. Pour la zone équatoriale, la position du premier pixel alterne entre $\phi_i = 0$, lorsque $(i - N)$ est impair, et $\phi_i = \phi_\Delta/2$, lorsque $(i - N)$ est pair. L'indice j du pixel adjacent est :

$$j = 1 + (\phi - \phi_i)/\phi_\Delta. \tag{5.3}$$

Cette équation affecte l'indice $j = 1$ au premier pixel d'un anneau. j étant un nombre en virgule flottante, l'indice est en fait la partie entière. La partie fractionnaire, Δj, est la pondération utilisée dans l'interpolation.

La valeur moyenne d'un pixel, après rotation, est obtenue en interpolant les valeurs des quatre pixels voisins. En ce qui concerne la valeur maximum, nous utilisons le maximum des quatre pixels interpolés. Pour garantir que ce maximum est conservateur, nous modifions l'étape de précalcul de façon à inclure, pour chaque pixel, son voisinage immédiat (huit pixels voisins) lors du calcul du maximum local. La zone couverte par le maximum ainsi obtenu est indiquée en gris dans la figure 5.6.

5.4 Visibilité

Tout comme la carte d'environnement, la carte de visibilité est définie dans le référentiel global. Par conséquent, il n'y a pas de traitement particulier à effectuer, les valeurs pour chaque nœud de la hiérarchie pouvant être obtenues directement. Cette section détaille seulement le processus de création de la carte de visibilité. L'objectif est simplement de créer une carte valide, afin de l'intégrer dans notre système, et de vérifier le fonctionnement avec plus de deux fonctions dans le produit. L'approche suivie est naïve et a été choisie pour sa simplicité.

La carte de visibilité est une fonction binaire. Pour une direction donnée, l'importance est 0 si un objet bloque la source de lumière (c'est-à-dire, dans notre cas, une direction de la carte d'environnement), et 1 sinon. Ce type de fonction est délicat

pour les méthodes Monte Carlo suivant l'importance. En effet, la convergence de ces méthodes est garantie, quelque soit la fonction d'importance utilisée, en autant que la fonction soit non nulle en tout point. Si une direction a une contribution non nulle, mais une importance nulle, elle ne sera jamais échantillonnée, et le résultat de l'estimation sera biaisé. Par conséquent, il est nécessaire de bâtir une carte de visibilité conservatrice. Notre fonction étant discrétisée en pixels, il faut que la totalité de l'angle solide couvert par un pixel soit bloqué, pour lui affecter une valeur d'importance nulle.

L'évaluation de la carte de visibilité, puisqu'elle est effectuée pour chaque rayon primaire, se doit d'être très rapide. Afin de réduire le temps de calcul de cette carte, un compromis est fait au niveau de la précision. La visibilité est donc déterminée d'après une approximation, à l'aide de formes géométriques simples (appelées *primitives*), de la scène. En pratique, pour approximer les modèles polygonaux, nous utilisons une série de sphères internes au modèle. Cette approche est inspirée par les travaux de Wang et al. [WZS+06], sur le calcul interactif d'ombres douces. Une approche très similaire a été proposée par Hubbard [Hub96] pour accélérer la détection de collisions entre objets, dans le contexte de la réalité virtuelle.

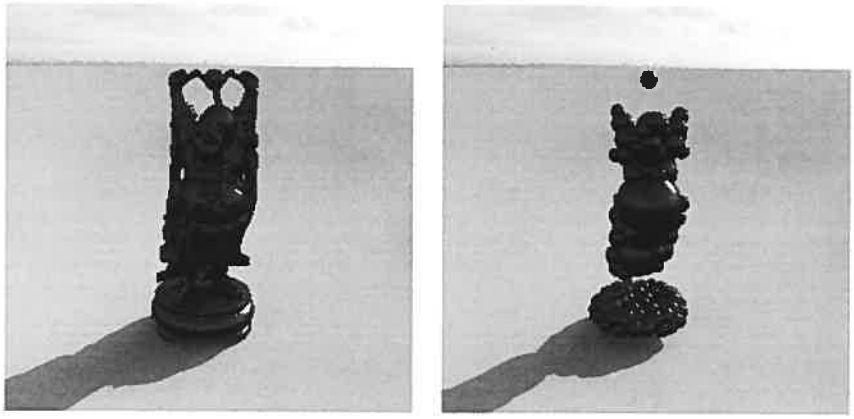


FIG. 5.9 – Approximation d'un modèle par un ensemble de sphères internes. L'ombre projetée par les sphères est un sous-ensemble de l'ombre réelle projetée par le modèle.

La figure 5.9 présente le modèle du bouddha, et son approximation par un ensemble de sphères. Les sphères utilisées sont *internes* à l'objet. Cela nous assure que l'ombre portée par l'approximation est un sous-ensemble de celle portée par le modèle original, et donc que notre évaluation est conservatrice. Étant donné le nombre de

sphères utilisées, il est nécessaire de les regrouper dans une hiérarchie, de façon à accélérer le traitement. La figure 5.10 donne la représentation hiérarchique des sphères approximant le modèle du bouddha. Cette hiérarchie est obtenue par regroupement récursif des sphères en deux groupes, créant un arbre binaire. Les nœuds de l'arbre sont des sphères englobantes. Le regroupement est fait de façon à favoriser des densités plus élevées. L'algorithme utilisé pour séparer les sphères en deux groupes est expliqué dans la figure 5.11. Les sphères englobantes sont générées avec le module *min_sphere_of_spheres* de la librairie CGAL [FGH+06].

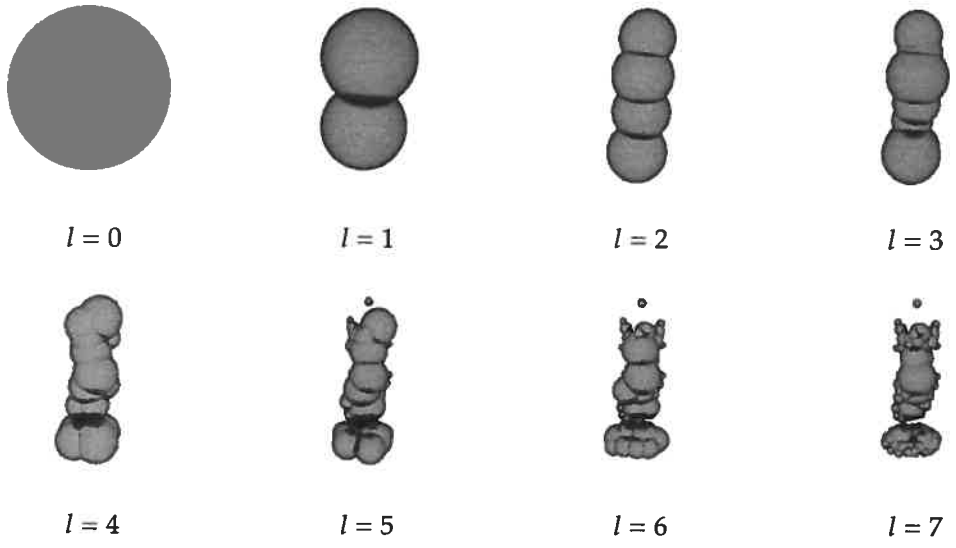


FIG. 5.10 – Hiérarchie construite à partir de 110 sphères internes. Aux niveaux $l = 7$, seules les sphères internes sont visibles. Certaines sphères internes sont situées à des niveaux moins profonds de la hiérarchie. Ceci est dû au fait que notre arbre n'est pas balancé (et les branches n'ont donc pas toutes la même profondeur), mais conçu de façon à avoir des nœuds aussi compacts que possible.

La carte de visibilité est générée en exploitant à la fois la hiérarchie de la projection HEALPix, et la hiérarchie de sphères. En partant du niveau $l = 0$, on teste, pour chacun des pixels, s'il y a occlusion. On définit un niveau maximal de subdivision L. Le test est

récursif et il est effectué en appliquant le pseudo-code suivant :

Fonction `TestObjets(`*nœud*`)`

Construit récursivement une carte de visibilité. Un test d'occlusion, pour tous les objets de la scène, est effectué pour le nœud passé en paramètre.

```
si SousEquateur(nœud) alors
    {max, avg} ← {0, 0}
sinon
    occlusion ← aucune
    pour chaque objet de la liste faire
        occlusion ← occlusion + Occlusion(objet,nœud)
        si occlusion.Type(totale) alors
            break
    si occlusion.Type(aucune) alors
        {max, avg} ← {1, 1}
    sinon si occlusion.Type(totale) alors
        {max, avg} ← {0, 0}
    sinon
        si nœud.l < L alors
            m1, m2, m3, m4, a1, a2, a3, a4
            {enfant1, enfant2, enfant3, enfant4} ← Enfants(nœud)
            {m1, a1} ← TestObjets(liste, enfant1)
            {m2, a2} ← TestObjets(liste, enfant2)
            {m3, a3} ← TestObjets(liste, enfant3)
            {m4, a4} ← TestObjets(liste, enfant4)
            {max, avg} ← {MAX(m1,m2,m3,m4), AVG(a1,a2,a3,a4)}
        sinon
            {max, avg} ← {1, 1}
carte[pixel] ← {max, avg}
return {max, avg}
```

Le test d'occlusion avec un objet est également récursif. Le pseudo-code est le

suivant :

Fonction `Occlusion(`*objet,nœud*`)`

Permet de tester l'occlusion d'un nœud de la hiérarchie par un objet.

```
occlusion ← aucune
liste ← objet.GetSphereEnglobante()
tant que liste.NonVide() faire
    sphere ← liste.Tete()
    si sphere.Type(englobante) alors
        si sphere.Occlusion(nœud) ≠ aucune alors
            liste ← liste + sphere.Enfants()
    sinon
        occlusion ← occlusion — sphere.Occlusion(nœud)
        si occlusion.Type(totale) alors
            retourner occlusion
retourner occlusion
```

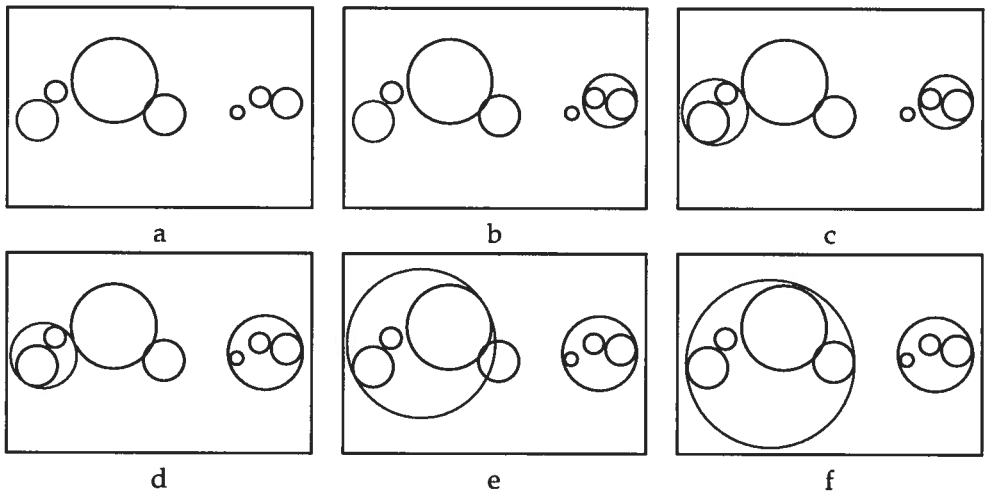


FIG. 5.11 – Regroupement des sphères. Les sphères sont triées suivant l'axe le plus long de la boîte englobante (l'axe horizontal dans ce cas-ci). Le premier groupement (vert) est initialisé avec la première sphère, et le deuxième groupement (bleu) est initialisé avec la dernière sphère. Pour chaque groupement, on évalue la distance, depuis son centre, au centre de la sphère la plus proche. La sphère dont la distance est la moindre, est associé au groupement correspondant. On répète l'opération jusqu'à ce que toutes les sphères soient associées à l'un des deux groupements.

Il nous reste à définir le test d'occlusion d'un pixel HEALPix par une sphère. Afin de simplifier le calcul, nous allons approximer chaque pixel par un cercle, tel qu'illustré dans la figure 5.12.

FIG. 5.12 – Approximation d'un pixel HEALPix par un cercle. Le cercle est centré sur le pixel. Le rayon est défini de façon à ce que le cercle passe par le sommet formant le plus grand angle avec le centre. De gauche à droite, on a les cercles utilisés pour approximer le pixel de coordonnées $(u, v) = (0, 0)$, de la face 0, aux niveaux $l = 0, 1, 2$.

Une fois le cercle défini, il ne reste plus qu'à tester l'occlusion. L'objectif est de déterminer si la sphère est positionnée de telle sorte qu'elle couvre partiellement, ou entièrement, l'angle solide couvert par le cercle approximant le pixel. Pour ce faire, nous définissons deux cônes à partir du point pour lequel on évalue l'éclairage, qui est simplement le centre de la sphère unitaire. Le premier cône passe par le cercle C approximant le pixel. Le second cône passe par le contour de la sphère S considérée. La figure 5.13 illustre la situation, dans le plan défini par le point de référence et les axes des deux cônes. Le test d'occlusion est très simple. Il y a occlusion totale si $\omega_s > (\widehat{srp} + \omega_p)$, et aucune occlusion si $\widehat{srp} > (\omega_s + \omega_p)$. Tout autre cas indique une occlusion partielle.

Le test d'occlusion peut être exprimé en fonction des valeurs de cosinus et sinus des différents angles, simplifiant le calcul. Ainsi, le test d'occlusion totale peut s'écrire

$$\cos(\omega_s) < \cos(\widehat{srp} + \omega_p) \tag{5.4}$$

et le test d'occlusion nulle peut s'écrire

$$\cos(\widehat{srp}) < \cos(\omega_s + \omega_p). \tag{5.5}$$

D'après l'équation

$$\cos(a + b) = \cos a \cos b - \sin a \sin b, \tag{5.6}$$

on peut exprimer le test d'occlusion en fonction de $\cos(\widehat{srp})$ et $\cos \omega_p$, avec $\cos(\widehat{srp}) = \vec{rs} \cdot \vec{rp}$. Les valeurs $\cos(\omega_p)$ et $\sin(\omega_p)$ sont fixes pour un pixel donné de la projection et

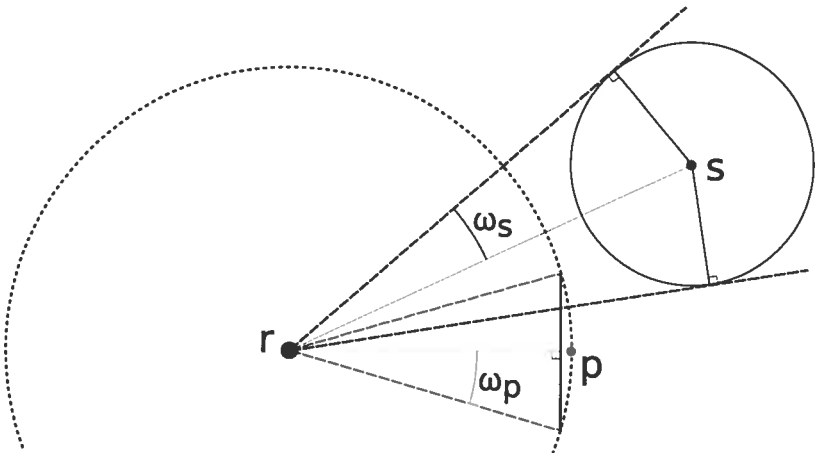


FIG. 5.13 – Évaluation de la visibilité. Coupe dans le plan défini par le point de référence r, et les axes des cônes $\vec{rp}$ et $\vec{rs}$. r est le point d'intersection entre le rayon primaire et la scène. p est le centre du cercle approximant le pixel HEALPix. s est le centre de la sphère créant l'occlusion.

peuvent donc être précalculées. Les valeurs $\cos(\omega_s)$ et $\sin(\omega_s)$ ne doivent être calculées qu'une fois par sphère, mais elles doivent être réévaluées pour chaque carte de visibilité, puisque ces valeurs varient relativement à la position du point r (le point d'intersection du rayon primaire avec la scène).

5.4.1 Plan infini

Notre implémentation intègre également l'occlusion par des plans infinis. Il suffit de déterminer la position relative d'un pixel (ou plutôt du cône approximant un pixel) par rapport au plan considéré. En prenant un point arbitraire du plan et sa normale, on définit un côté positif (celui indiqué par la normale) et un négatif. On a alors trois cas de figure : un cône peut être orienté du côté positif uniquement, du côté négatif uniquement, ou à cheval sur les deux (quand il est coupé par le plan). En reprenant la notation de la figure 5.13, on a ω_p l'angle d'ouverture du cône et $\vec{rp}$ l'axe du cône. On note $\vec{n}$ la normale au plan et α l'angle entre $\vec{n}$ et $\vec{rp}$. L'orientation d'un cône est

$$\begin{cases} \text{positive si } \alpha + \omega_p < \pi/2 \Leftrightarrow \cos(\alpha + \omega_p) > 0, \\ \text{négative si } \alpha - \omega_p > \pi/2 \Leftrightarrow \cos(\alpha - \omega_p) < 0, \\ \text{nulle sinon.} \end{cases}$$

En appliquant l'équation 5.6 et

$$\cos(a - b) = \cos a \cos b + \sin a \sin b, \tag{5.7}$$

on peut exprimer l'orientation d'un cône en fonction de $\cos\alpha$ et $\cos\omega_p$, avec $\cos\alpha = \vec{rp}\cdot\vec{n}$. On peut maintenant déterminer le type d'occlusion d'après l'orientation du cône. Si le point d'intersection r est du côté positif du plan ($\vec{rp} \cdot \vec{n} \geq 0$), une orientation positive correspond à une occlusion nulle et une orientation négative correspond à une occlusion totale. La correspondance est inversée si r est du côté négatif du plan. Dans les deux cas, une orientation nulle correspond à une occlusion partielle.

5.5 Optimisation

Il est possible d'améliorer significativement le temps de rendu, au détriment de la qualité. L'algorithme tel que décrit dans la section 4.2 produit les échantillons candidats uniquement au niveau maximum de subdivision, ce qui résulte en une discrétisation précise mais coûteuse du domaine. L'idée de notre optimisation est de couper non seulement les branches stériles (comme dans l'implémentation de base), mais aussi celle pouvant produire *au plus* un échantillon. Dès qu'une telle branche est détectée, la subdivision est arrêtée et l'échantillon est placé aléatoirement dans le nœud courant.

Il est facile d'identifier les branches pouvant produire au plus un échantillon. Nous savons que, lorsqu'un nœud parent est subdivisé, un de ses enfants hérite de sa valeur de seuil alors que les autres reçoivent des nouvelles valeurs plus élevées. Si tous les nouvelles valeurs sont supérieures au maximum local du parent, ces nouvelles branches seront nécessairement stériles et donc coupées. Seule la branche de l'enfant ayant hérité directement de la valeur de seuil du parent sera conservée (si la valeur de seuil est également inférieure au maximum local de l'enfant). En répétant ce raisonnement jusqu'au niveau maximal de subdivision, on voit que cette branche produira, au plus, un seul échantillon.

Un estimateur non biaisé utilise une évaluation exacte de la fonction à l'emplacement de l'échantillon. Avec notre nouvelle optimisation, certains échantillons seront placés aux niveaux intermédiaires de la hiérarchie, où les nœuds sont de plus grande taille. Plus la surface d'un nœud est grande, et plus la variance de la fonction est élevée, ce qui augmente la variance de notre estimation. Pour éliminer cet effet, il est cou-

original (non biaisé)
20,37 s

optimisé (non biaisé)
9,22 s

optimisé (biaisé)
8,53 s

FIG. 5.14 – Effet de l'optimisation du seuillage hiérarchique. En plaçant les échantillons dans les niveaux intermédiaires de la hiérarchie, il est possible de réduire considérablement le temps de calcul, au coût d'une augmentation significative de la variance, qui se traduit en bruit dans l'image. En utilisant les valeurs pré-intégrées, quand elles existent dans nos arbres précalculés, on élimine le bruit, en introduisant un léger biais. Notez que la version biaisée est légèrement plus rapide que l'alternative non biaisée, puisqu'on a moins d'évaluations à effectuer. On a utilisé 10 échantillons et la résolution des images est de 256×256 pixels.

rant d'utiliser des valeurs pré-intégrées, ce qui correspond, dans notre cas, à utiliser directement la valeur moyenne du nœud. En pratique, nous utilisons uniquement la valeur moyenne du nœud si elle existe dans notre hiérarchie précalculée. Par exemple, supposons que nos hiérarchies précalculées pour la BRDF et l'environnement aient des profondeurs de 5 et 8 respectivement. Si un échantillon est placé dans un nœud du niveau 7, la valeur de l'environnement est prise dans la table précalculée, par contre, celle de la BRDF est évaluée, puisque la résolution de la table est insuffisante. La figure 5.14 illustre le bruit causé par l'optimisation, et le résultat obtenu en utilisant les valeurs pré-intégrées. Le fait d'utiliser les valeurs pré-intégrées introduit un léger biais.

Chapitre 6

Résultats

Tous les résultats présentés dans ce chapitre ont été obtenus avec un Athlon64 3500+, avec 1 GB de RAM. Pour chaque fonction, on enregistre la hiérarchie HEALPix, jusqu'à une certaine profondeur l. Au niveau l, chaque face de la projection HEALPix contient $2^l \times 2^l$ pixels. La hiérarchie complète est constituée de $4/3 \times 2^l \times 2^l$ pixels par face.

Pour toutes les cartes d'environnement, la hiérarchie précalculée a une profondeur $l = 8$ et occupe 24 MB en mémoire. Pour une carte d'environnement, encodée dans la projection angulaire et de résolution 4096×4096 pixels, le temps de précalcul de la hiérarchie est de 2,12 s.

Pour les BRDFs, seuls les anneaux visibles sont conservés. Dans la grande majorité des BRDFs, une profondeur de $l = 5$ est utilisée. Pour les cas extrêmes (réflexion de type miroir), une profondeur de $l = 6$ est utilisée. À l'inverse, pour une BRDF très diffuse, une profondeur de $l = 4$ suffit. Le temps de précalcul varie en fonction de la BRDF considérée. Le tableau 6.1 donne les temps, avec des profondeurs de $l = 4, 5, 6$, pour une BRDF mesurée [MPBM03].

Profondeur l	Mémoire	Temps de précalcul
4	2.55 MB	0.5 s
5	9.78 MB	2.0 s
6	38.45 MB	8.0 s

TAB. 6.1 – Temps de précalcul et occupation en mémoire des tables de BRDF pour différentes profondeurs de hiérarchie. Le temps de précalcul croît linéairement par rapport à la taille en mémoire.

La profondeur de la hiérarchie des cartes de visibilité est toujours de $l = 4$ et occupe 8 KB en mémoire. La figure 6.1 illustre l'amélioration obtenue en approximant le modèle du bouddha par 110 sphères internes. Les temps de rendu de l'image, avec différents nombres d'échantillons, sont donnés dans le tableau 6.2. Notez que le temps de calcul de la carte de visibilité est indépendant du nombre d'échantillons, et que le coût additionnel de gérer une troisième fonction lors de la traversée de la hiérarchie est marginal. Comme on peut le constater dans la figure 6.1, quand l'occlusion est mal approximée par les sphères, l'amélioration du niveau de bruit est marginale (comme à la bordure de l'ombre du bouddha). Pour de telles régions, nous procédons à un échantillonnage adaptatif, où le nombre d'échantillons est augmenté pour compenser pour ceux bloqués, résultant en une image avec un niveau de bruit relativement homogène.

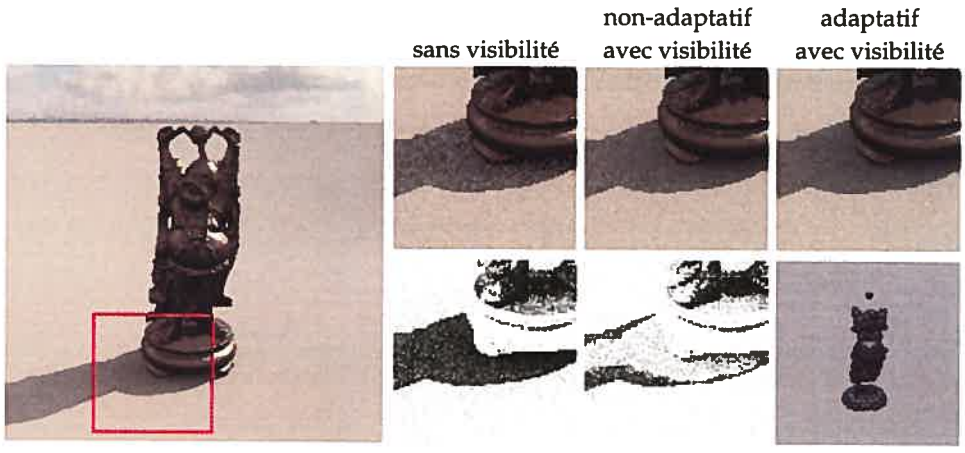


FIG. 6.1 – Images illustrant le gain obtenu avec la carte de visibilité et l'échantillonnage adaptatif. L'éclairage est évalué avec 16 échantillons, et la variante biaisée de l'algorithme a été utilisée. Les images en ton de gris du bas donnent le pourcentage de rayons de visibilité bloqués (noir pour 100% et blanc pour 0%). L'image en bas à droite donne les sphères utilisées pour approximer le bouddha. L'image de gauche est la référence.

La figure 6.2 illustre la flexibilité de notre approche en utilisant des fonctions de réflectance et d'éclairage variables. On affecte à la sphère une BRDF de Phong que l'on fait varier de parfaitement diffus à parfaitement spéculaire. De même, on applique un flou radial sur la carte d'environnement de plus en plus prononcé, résultant en un éclairage de plus en plus uniforme.

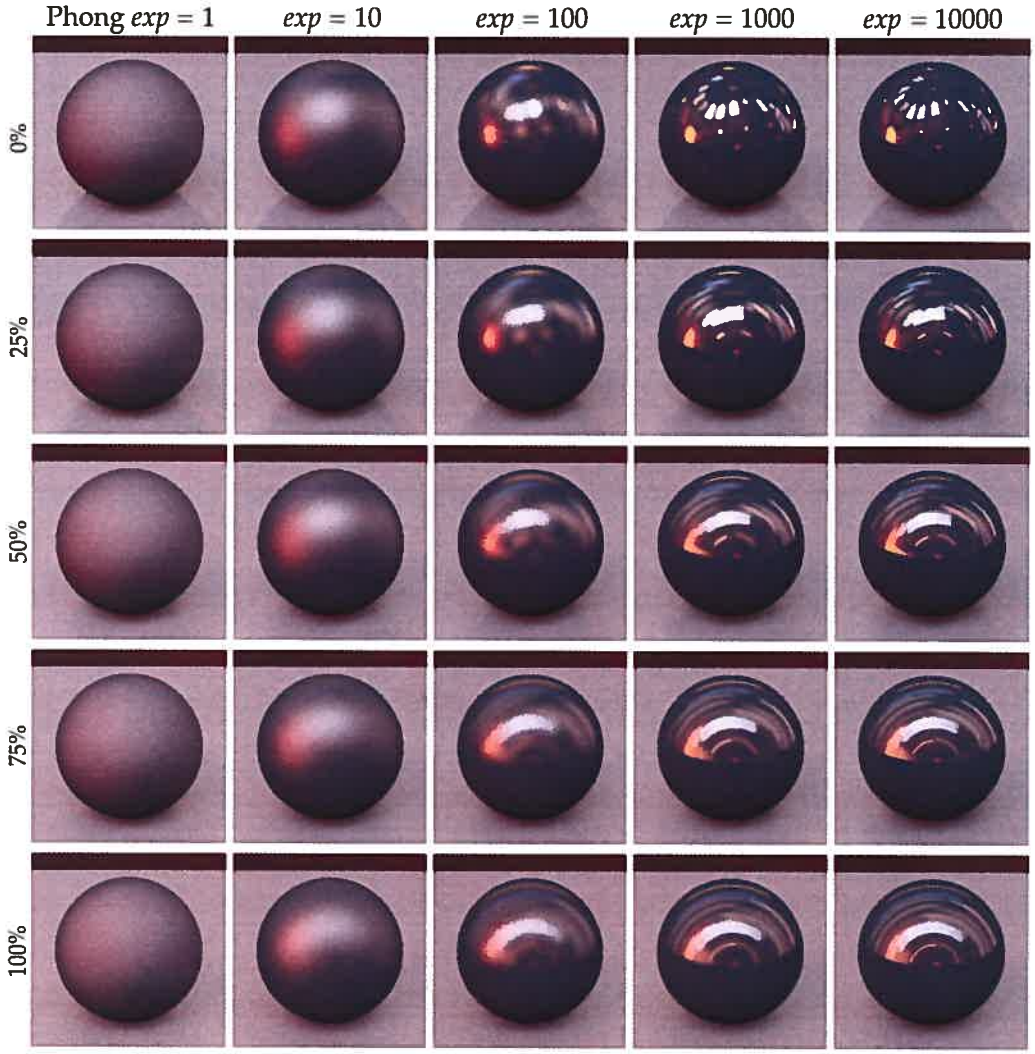


FIG. 6.2 – Variation de BRDF et d'éclairage. Horizontalement, on varie l'exposant de la BRDF de Phong utilisée pour la sphère. Verticalement, un flou radial de plus en plus prononcé est appliqué sur la carte d'environnement pour simuler un éclairage de plus en plus diffus (qui résulte en une ombre de plus en plus douce de la sphère sur le plan). À 100%, le flou radial étale un pixel sur un arc allant de $-\pi/4$ à $\pi/4$.

nombre d'échantillons	sans visibilité	non-adaptatif avec visibilité	adaptatif avec visibilité
16	4.29 s	13.57 s	14.77 s
128	19.70 s	30.04 s	35.87 s
512	78.86 s	92.40 s	118.93 s

TAB. 6.2 – Temps de rendu pour la scène de la figure 6.1 (gauche), à une résolution de 256×256 pixels.

La figure 6.3 présente les rendus d'une scène plus complexe, sous une variété d'éclairages. Une variété de types de BRDFs est présentée, allant du très diffus (le sol), au très spéculaire (la sphère en bas à gauche). L'environnement présente une gamme extrême d'intensités, entre les zones sombres et le soleil. L'image en haut à gauche, où le soleil est caché par les nuages, présente un cas avec un éclairage plus uniforme, avec des ombres douces portées sur le sol. Les quatre images ont été produites sans ajustement, seule la carte d'environnement est changée. On a utilisé 100 échantillons pour évaluer l'éclairage, 4 rayons primaires par pixel. On a utilisé l'évaluation de la visibilité, ainsi que l'échantillonnage adaptatif. La résolution des images est de 720×486 et le temps de rendu d'environ 12 mn.

La figure 6.4 donne l'évolution de la variance suivant le nombre d'échantillons de notre méthode, et la compare à d'autres techniques récentes. Pour comparaison, nous donnons aussi le résultat de l'échantillonnage exclusif de l'environnement ou de la BRDF. L'échantillonnage de la BRDF fonctionne relativement bien pour la sphère, qui est très spéculaire, mais mal pour le sol, qui est très diffus. À l'inverse, l'échantillonnage de l'environnement fonctionne bien pour le sol, mais mal pour la sphère. Les trois autres méthodes, tenant compte du produit des deux fonctions, sont efficaces dans tous les cas. Notez cependant que, avec moins de 64 échantillons, l'algorithme TSIS donne des résultats nettement moins bons. L'application de notre algorithme résulte en une plus faible variance. L'algorithme WIS donne, en théorie, une meilleure approximation, mais pas en pratique, puisque la résolution de la représentation en ondelettes est limitée à 128×128 pixels. Cette limite est liée au fait que l'environnement est représenté en 4D (plutôt que 2D dans notre cas, ou celui de l'algorithme TSIS), parce qu'il doit être défini dans le même référentiel que la BRDF. Les courbes de convergence des trois méthodes sont données dans la figure 6.6.

FIG. 6.3 – Scène où chaque objet a une BRDF différente, éclairée à plusieurs moments de la journée. Les BRDFs sont : velours (analytique) pour le lapin, bronze (mesuré) pour le bouddha, sucre (mesuré) pour les piliers, aluminium (mesuré), caoutchouc bleu (mesuré), or métallisé (mesuré), tungstène (mesuré) et phong (analytique) pour les sphères (de gauche à droite et de haut en bas). Les BRDFs mesurées ont été produites par Matusik et al. [MPBM03], et les cartes d'environnement par Stumpfel et al. [STJ+04].

Afin d'évaluer l'évolution de la variance de notre méthode, par rapport au nombre de fonctions considérées, nous avons procédé à un test empirique utilisant des fonctions de types variés. Pour chaque fonction considérée, le type et les attributs sont choisis aléatoirement. Les différents types utilisés sont donnés dans la figure 6.7. Chaque fonction est représentée par une image de 1024×1024 pixels. Nous avons alors échantillonné le produit, et calculé une référence par force brute, en évaluant le produit pour chaque pixel. La figure 6.8 donne la variance de l'estimation pour les deux variantes de notre algorithme (biaisé avec optimisation et non biaisé sans optimisation). La figure 6.9 présente un cas pratique avec 8 fonctions, et le résultat de l'échantillonnage. Remarquez que la variance est sensiblement la même pour les deux versions de l'algorithme. On remarque que l'augmentation de la variance est progressive, et qu'il n'y a pas de seuil critique, ni même de dégradation exponentielle des résultats.

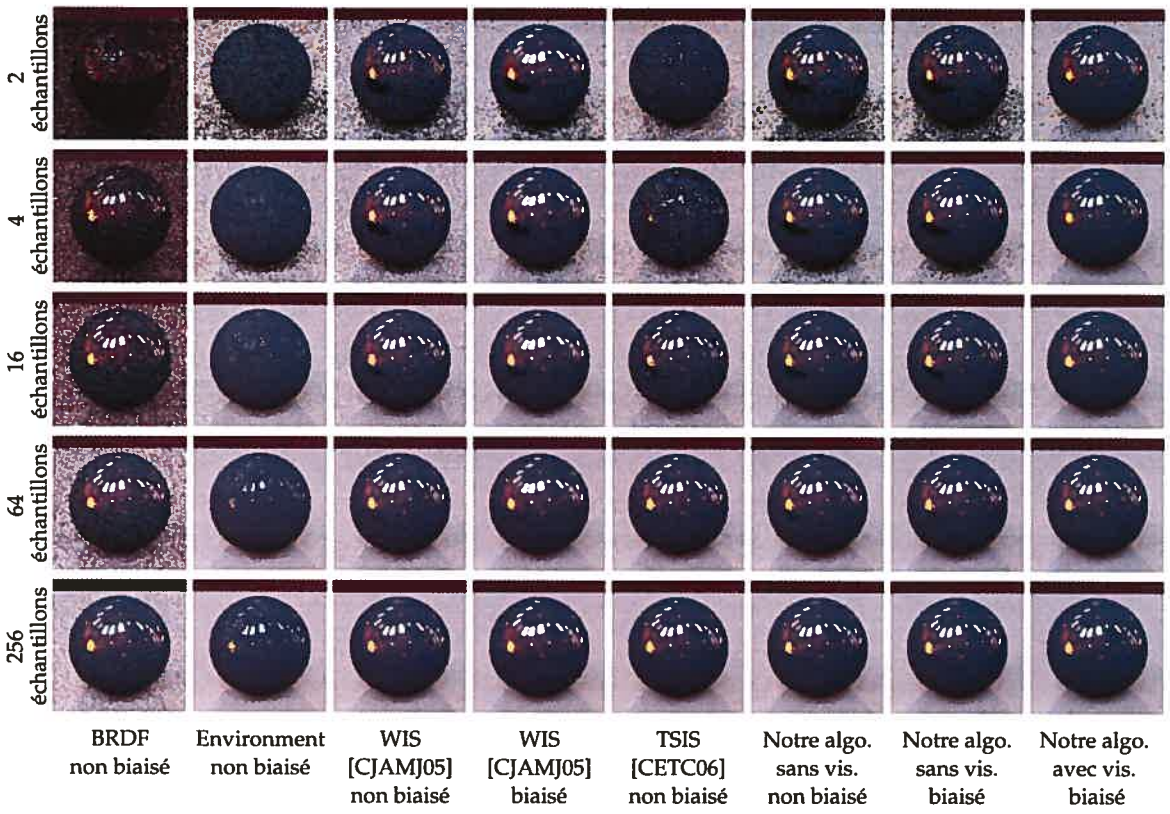


FIG. 6.4 – Tableau comparatif. Pour l'algorithme WIS, la résolution des ondelettes est de 128×128, avec 2–5% des coefficients conservés. Les extensions d'échantillonnage adaptatif de notre algorithme et de l'algorithme TSIS n'ont pas été utilisées. L'échantillonnage de la BRDF est fait par inversion de la fonction de distribution. L'échantillonnage de l'environnement est fait en appliquant notre algorithme sur cette seule fonction. Pour notre algorithme, les hiérarchies précalculées ont des profondeur de $l = 5$ pour la BRDF du sol, $l = 6$ pour la BRDF de la sphère, et $l = 8$ pour l'environnement. En tenant compte de la visibilité (dernière colonne à droite), on constate que le bruit dû à l'occlusion est significativement diminué.

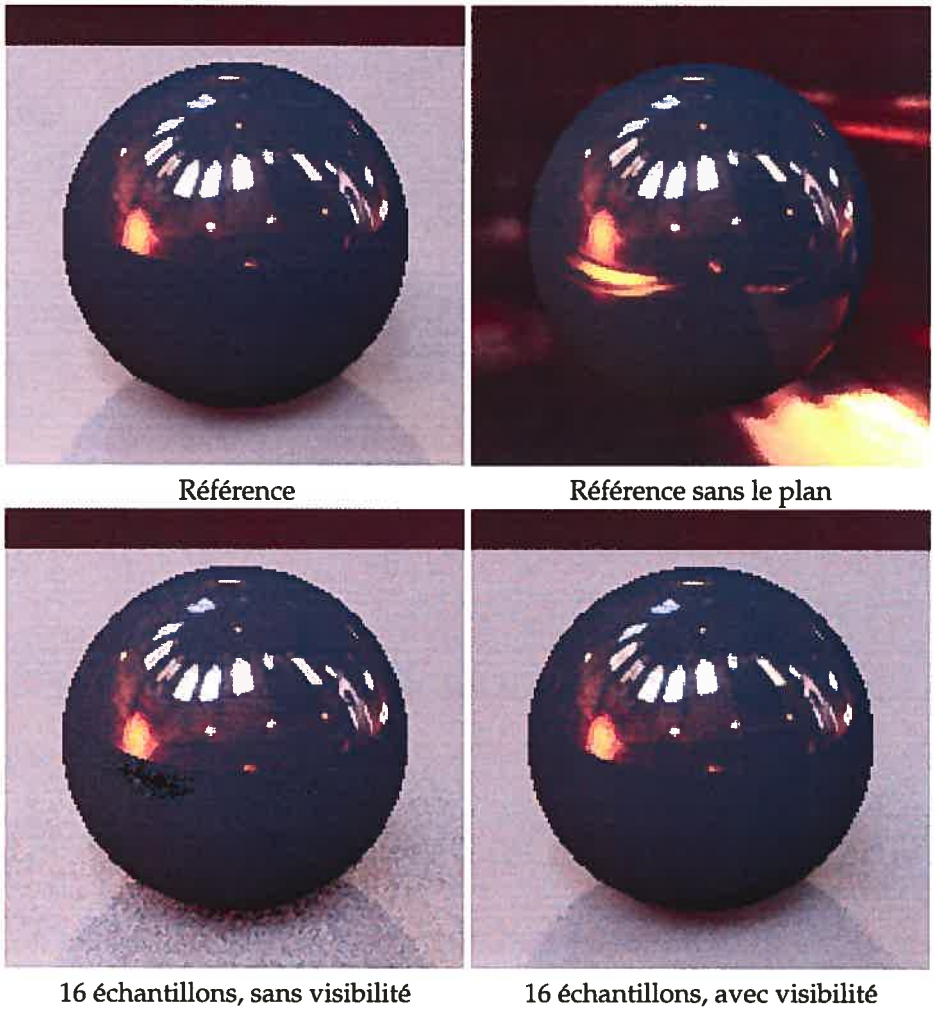


FIG. 6.5 – Référence de la scène de la sphère sur un plan utilisée dans la figure 6.4. La deuxième ligne reprend les images obtenues avec la version biaisée de notre algorithme. On constate que les zones les plus bruitées correspondent à des régions de forte intensité lumineuse cachées par les objets de la scène et qu'elles sont correctement traitées lorsque l'on inclue notre carte de visibilité.

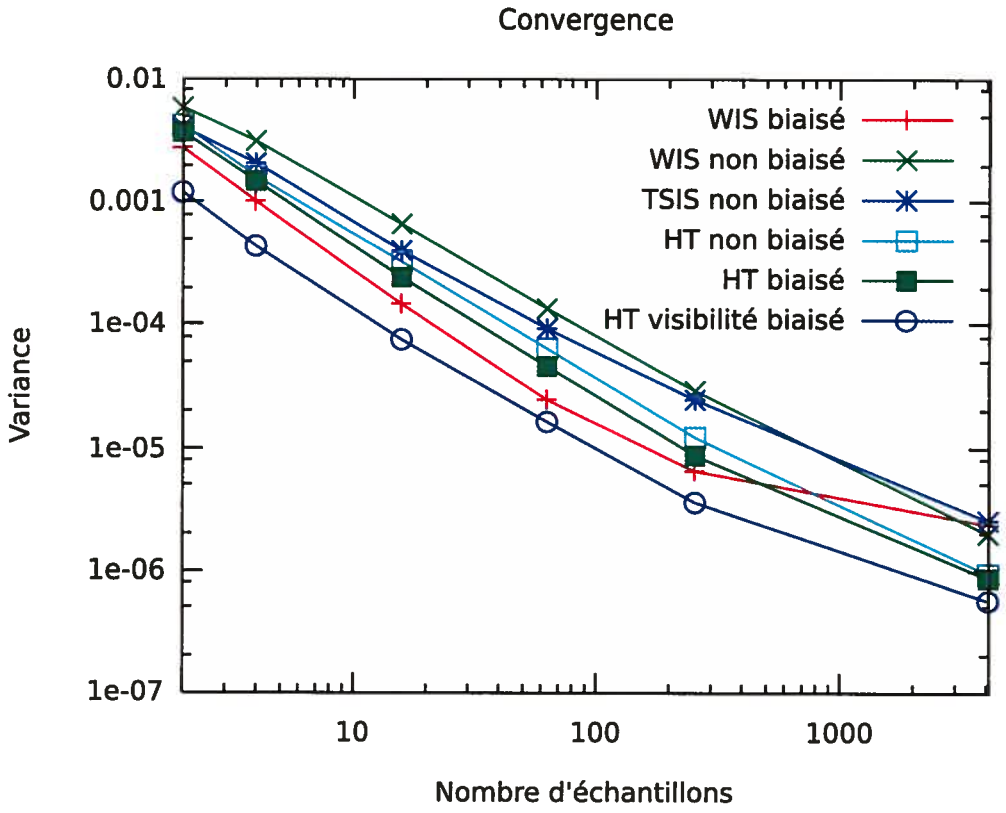


FIG. 6.6 – Courbes de convergence des principales méthodes d'échantillonnage. On constate que les résultats obtenus avec notre méthode ont une variance moindre que ceux obtenus avec la variante non biaisée de WIS et TSIS. La variante biaisée de WIS donne initialement de meilleurs résultats, mais ne converge pas vers une image exacte, contrairement à la variante biaisée de notre algorithme. Cela s'explique par le fait que, dans notre variante biaisée, nous utilisons des valeurs exactes lorsque la précision des tables est insuffisante. Enfin, on constate qu'il y a une diminution appréciable de la variance, lorsque l'on intègre la carte de visibilité.

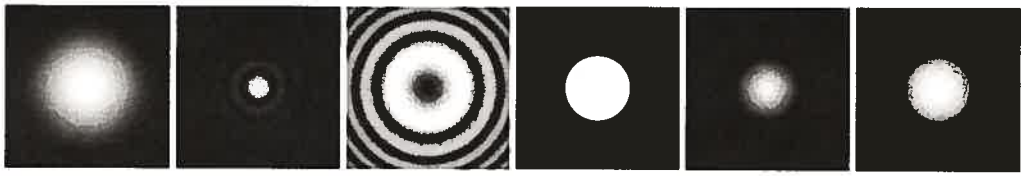


FIG. 6.7 – Les différents types de fonctions définies. Les attributs de chaque fonction (position du centre, fréquence, etc.) sont fixés aléatoirement.

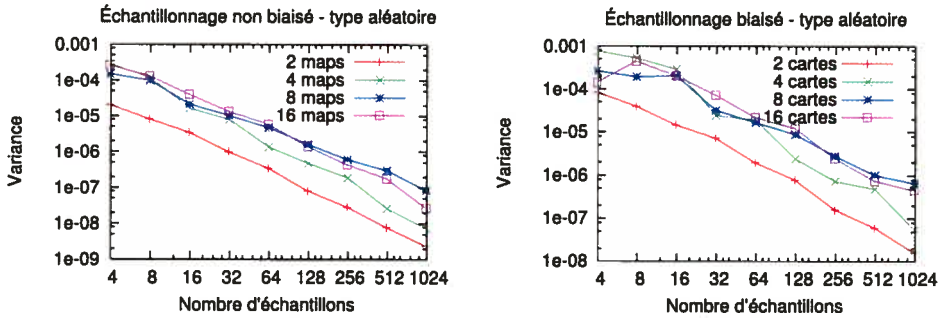


FIG. 6.8 – Comparaison de l'algorithme original et non biaisé, avec l'algorithme optimisé et biaisé. La variance est similaire avec les deux variantes, mais la convergence est plus régulière avec l'algorithme non biaisé.

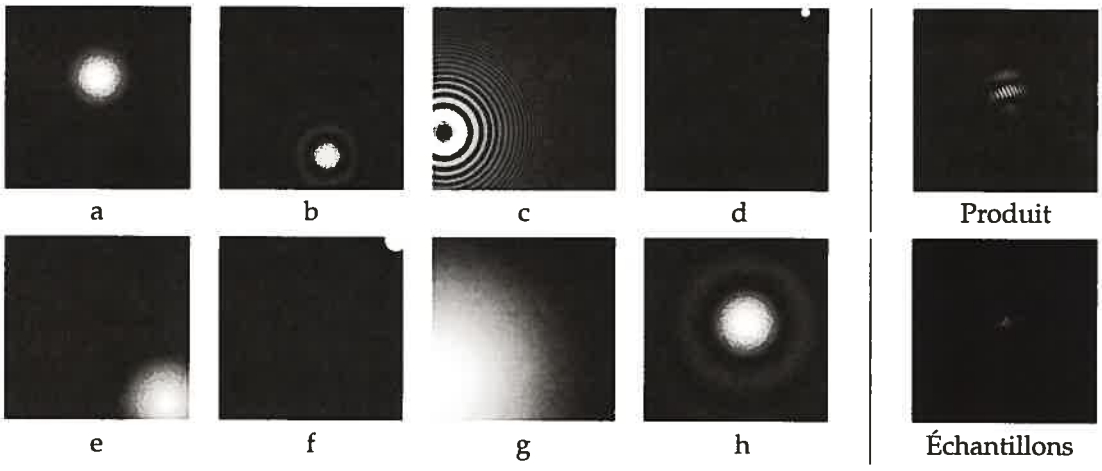


FIG. 6.9 – Échantillonnage du produit de huit fonctions, a–h. L'image en haut à droite a été obtenue en effectuant le produit, pixel par pixel, de toutes les fonctions. L'image en bas à droite donne les échantillons générés.

Chapitre 7

Conclusion

Nous avons présenté une méthode générale permettant d'échantillonner efficacement le produit de plusieurs fonctions. Les fonctions impliquées dans le produit peuvent être très variées. Notre application illustre les résultats obtenus avec un large spectre de caractéristiques, que ce soit des fonctions de réflectance allant du parfaitement diffus à la réflexion miroir, ou encore des fonctions d'éclairage présentant des gammes extrêmes de valeurs. L'utilisation de cartes de visibilité binaires illustre la robustesse de notre algorithme avec les fonctions présentant des discontinuités. Enfin, notre algorithme peut être appliqué pour évaluer le produit de fonctions exprimées dans des référentiels différents, tel que démontré par notre application.

Notre méthode présente deux inconvénients notables qui en limite l'applicabilité, tous deux liés au recours à une hiérarchie précalculée pour chaque fonction impliquée. Le temps de génération de ces hiérarchies est trop élevé dans certains cas. Par exemple, dans le contexte de notre application, il peut être utile de varier spatialement ou temporellement la fonction de réflectance d'un matériau. Afin de gérer de tels cas, il serait nécessaire de réévaluer, pour chaque pixel, la hiérarchie complète de la BRDF, ce qui serait prohibitif. L'autre inconvénient des hiérarchies précalculées est l'espace mémoire utilisé. Dans notre application, cela nous limite aux BRDFs isotropiques, puisque le volume d'information explose pour les BRDFs anisotropiques (approximativement 100 fois plus gros).

Notre méthode présente également certains avantages particulièrement intéressants. La qualité principale de notre approche est sa simplicité, qui permet d'incorporer, à faible coût, des fonctions additionnelles au produit. La deuxième qualité notable de

notre algorithme va de pair avec la simplicité, il s'agit de la robustesse. Puisque l'exploration de la hiérarchie est faite suivant une estimation conservatrice des maxima locaux, le nombre de fonctions impliquées peut augmenter sans jamais perdre aucun détail. Évidemment, nous assumons toujours que la discrétisation initiale de la hiérarchie précalculée de chaque fonction est suffisamment précise. Finalement, notre approche restreint le travail aux seules régions échantillonnées. Les techniques existantes d'échantillonnage suivant l'importance se préoccupent de produire une fonction d'importance en tout point, alors que notre approche concentre l'évaluation sur les régions contribuant à l'intégrale.

Nous sommes intéressés à explorer les solutions permettant de résoudre les deux inconvénients cités plus haut. Idéalement, nous voudrions évaluer dynamiquement l'information contenue dans les hiérarchies précalculées (valeurs locales de maximum et de moyenne). Cela aurait pour effet d'annuler les deux inconvénients cités plus haut, soit le temps de précalcul et l'occupation de mémoire. Ainsi, dans notre application, il suffirait de traiter dynamiquement la BRDF pour régler, à toute fin pratique, le problème d'utilisation de mémoire. Un traitement au cas par cas des différents modèles analytiques de BRDFs, pourrait mener à une estimation du maximum local. Pour la valeur moyenne, une approche similaire à celle de Cline et al. [CETC06], où la fonction est évaluée ponctuellement après avoir atteint un certain seuil de variance, pourrait être utilisée.

En ce qui concerne la visibilité, les résultats obtenus sont très encourageants et indiquent un potentiel considérable d'amélioration pour les méthodes d'échantillonnage suivant l'importance. La technique que nous avons développée, quoique fonctionnelle, n'est pas utilisable en pratique en raison de ses performances. Nous sommes particulièrement intéressés à explorer les possibilités de création efficace de cartes de visibilité. En particulier, l'évaluation adaptative nous semble une avenue particulièrement prometteuse, où le temps d'évaluation serait concentré sur les régions les plus susceptibles de poser problème. Nous envisageons aussi d'utiliser des types différents de formes géométriques pour approximer les différents modèles. Par exemple, un quadrilatère serait plus approprié pour approximer des objets minces, plats et allongés (tels que des tables, murs, etc.), qui se retrouvent fréquemment dans les scènes d'intérieur.

Bibliographie

[ARBJ03] Sameer Agarwal, Ravi Ramamoorthi, Serge Belongie et Henrik Wann Jensen. « Structured Importance Sampling of Environment Maps ». *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2003)*, volume 22, numéro 3, pages 605–612, 2003.

[AS00] Michael Ashikhmin et Peter Shirley. « An Anisotropic Phong BRDF Model ». *Journal of Graphics Tools*, volume 5, numéro 2, pages 25–32, 2000.

[Bay73] Bryce E. Bayer. « An optimum method for two-level rendition of continuous-tone pictures ». Dans *IEEE International Conference on Communications*, pages 11–15, 1973.

[BGH05] David Burke, Abhijeet Ghosh et Wolfgang Heidrich. « Bidirectional Importance Sampling for Direct Illumination ». Dans *Rendering Techniques 2005 (Proceedings of the Eurographics Symposium on Rendering)*, pages 147–156, 2005.

[CD01] Jonathan Cohen et Paul Debevec. « LightGen, HDRShop plugin ». http ://www.ict.usc.edu/~jcohen/lightgen/lightgen.html, 2001.

[CETC06] David Cline, Parris K. Egbert, Justin F. Talbot et David L. Cardon. « Two Stage Importance Sampling for Direct Lighting ». Dans *Rendering Techniques 2006 (Proceedings of the Eurographics Symposium on Rendering)*, pages 103–113, 2006.

[CJAMJ05] Petrik Clarberg, Wojciech Jarosz, Tomas Akenine-Möller et Henrik Wann Jensen. « Wavelet Importance Sampling : Efficiently Evaluating Products of Complex Functions ». *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005)*, volume 24, numéro 3, pages 1166–1175, 2005.

[CR04] M. R. Calabretta et B. F. Roukema. « Mapping on the HEALPix grid ». *ArXiv Astrophysics e-prints*, décembre 2004.

[Deb98] Paul Debevec. « Rendering synthetic objects into real scenes : bridging traditional and image-based graphics with global illumination and high dynamic range photography ». Dans *SIGGRAPH '98 : Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 189–198, New York, NY, USA, 1998. ACM Press.

[DHT+00] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin et Mark Sagar. « Acquiring the reflectance field of a human face ». Dans *SIGGRAPH '00 : Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 145–156, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[DM97] Paul E. Debevec et Jitendra Malik. « Recovering high dynamic range radiance maps from photographs ». Dans *SIGGRAPH '97 : Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 369–378, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[FGH+06] Kaspar Fischer, Bernd Gärtner, Thomas Herrmann, Michael Hoffmann, Eli Packer et Sven Schönherr. « Geometric Optimisation ». Dans CGAL Editorial Board, éditeur. *CGAL-3.2 User and Reference Manual*. 2006.

[GHB+05] K. M. Gorski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke et M. Bartelmann. « HEALPix – A Framework for High Resolution Discretization, and Fast Analysis of Data Distributed on the Sphere ». *The Astrophysical Journal*, volume 622, pages 759–771, 2005.

[HS98] Wolfgang Heidrich et Hans-Peter Seidel. « View-independent environment maps ». Dans *HWWS '98 : Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, pages 39–ff., New York, NY, USA, 1998. ACM Press.

[Hub96] Philip M. Hubbard. « Approximating polyhedra with spheres for time-critical collision detection ». *ACM Transaction on Graphics*, volume 15, numéro 3, pages 179–210, 1996.

[Kaj86] James T. Kajiya. « The rendering equation ». *Computer Graphics (Proceedings of SIGGRAPH '86)*, volume 20, numéro 4, pages 143–150, 1986.

[KK03] Thomas Kollig et Alexander Keller. « Efficient Illumination by High Dynamic Range Images ». Dans *Rendering Techniques 2003 (Proceedings of the Eurographics Symposium on Rendering)*, pages 45–50, 2003.

[LFTG97] Eric P. F. Lafortune, Sing-Choong Foo, Kenneth E. Torrance et Donald P. Greenberg. « Non-linear approximation of reflectance functions ». Dans *SIGGRAPH '97 : Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 117–126, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[LRR04] Jason Lawrence, Szymon Rusinkiewicz et Ravi Ramamoorthi. « Efficient BRDF Importance Sampling using a Factored Representation ». *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2004)*, volume 23, numéro 3, pages 496–505, 2004.

[LRR05] Jason Lawrence, Szymon Rusinkiewicz et Ravi Ramamoorthi. « Adaptive Numerical Cumulative Distribution Functions for Efficient Importance Sampling ». Dans *Rendering Techniques 2005 (Proceedings of the Eurographics Symposium on Rendering)*, juin 2005.

[MPBM03] Wojciech Matusik, Hanspeter Pfister, Matt Brand et Leonard McMillan. « A data-driven reflectance model ». *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2003)*, volume 22, numéro 3, pages 759–769, 2003.

[NDM05] Addy Ngan, Frédo Durand et Wojciech Matusik. « Experimental Analysis of BRDF Models ». Dans *Rendering Techniques 2005 (Proceedings of the Eurographics Symposium on Rendering)*, pages 117–226. Eurographics Association, 2005.

[Nie92] Harald Niederreiter. *Random number generation and quasi-Monte Carlo methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.

[ODJ04] Victor Ostromoukhov, Charles Donohue et Pierre-Marc Jodoin. « Fast hierarchical importance sampling with blue noise properties ». *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2004)*, volume 23, numéro 3, pages 488–495, 2004.

[PH04] Matt Pharr et Greg Humphreys. *Physically Based Rendering : From Theory to Implementation*. Morgan Kaufmann, 2004.

[Pho75] Bui Tuong Phong. « Illumination for computer generated pictures ». *Communications of the ACM*, volume 18, numéro 6, pages 311–317, 1975.

[SHS02] Adrian Secord, Wolfgang Heidrich et Lisa Streit. « Fast primitive distribution for illustration ». Dans Paul Debevec et Simon Gibson, éditeurs. *Rendering Techniques 2002 (Proceedings of the Eurographics Symposium on Rendering)*, Eurographics, pages 215–226. Springer-Verlag Wien New York, 2002. Proc. 13th Eurographics Rendering Workshop, Pisa, Italy, June 26–28, 2002.

[STJ+04] Jessi Stumpfel, Chris Tchou, Andrew Jones, Tim Hawkins, Andreas Wenger et Paul Debevec. « Direct HDR capture of the sun and sky ». Dans *AFRIGRAPH '04 : Proceedings of the 3rd international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, pages 145–149, New York, NY, USA, 2004. ACM Press.

[TS67] K. E. Torrance et E. M. Sparrow. « Theory for off-specular reflection from roughened surfaces ». *Journal of the Optical Society of America*, volume 57, numéro 9, pages 1105–14, 1967.

[Vea98] Eric Veach. *Robust monte carlo methods for light transport simulation*. Thèse de doctorat, Stanford University, 1998. Adviser-Leonidas J. Guibas.

[VG95] Eric Veach et Leonidas J. Guibas. « Optimally combining sampling techniques for Monte Carlo rendering ». Dans *SIGGRAPH '95 : Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 419–428, New York, NY, USA, 1995. ACM Press.

[War92] Gregory J. Ward. « Measuring and modeling anisotropic reflection ». *Computer Graphics (Proceedings of SIGGRAPH '92)*, volume 26, numéro 2, pages 265–272, 1992.

[Wil83] Lance Williams. « Pyramidal parametrics ». *Computer Graphics (Proceedings of SIGGRAPH '83)*, volume 17, numéro 3, pages 1–11, 1983.

[WWL05] Liang Wan, Tien-Tsin Wong et Chi-Sing Leung. « Spherical Q2-tree for Sampling Dynamic Environment Sequences ». Dans *Rendering Techniques 2005 (Proceedings of the Eurographics Symposium on Rendering)*, pages 21–30, 2005.

[WZS+06] Rui Wang, Kun Zhou, John Snyder, Xinguo Liu, Hujun Bao, Qunsheng Peng et Baining Guo. « Variational Sphere Set Approximation for Solid Objects ». *The Visual Computer*, volume 22, numéro 9, pages 612–621, 2006.