Université de Montréal

Étude des techniques d'estimation de densité et du tracé de chemins pour le rendu des milieux participatifs

par
Nicolas Vibert

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

décembre, 2016

RÉSUMÉ

L'utilisation d'images de synthèse photo réalistes est aujourd'hui devenue commune, que ce soit pour la réalisation de films, la création d'environnements virtuels à des fins vidéo ludiques comme le jeu, ou même pour la prévisualisation de projets architecturaux. La compréhension et les avancées technologiques de ces 20 dernières années permettent maintenant de créer des images très réalistes. Dans certains cas, il est même devenu difficile, voire même impossible, de faire la distinction entre une photo d'un objet réel et une image virtuelle.

Mais, pour atteindre un tel degré de réalisme, il faut créer des algorithmes complexes et coûteux, capables de simuler les lois physiques qui gouvernent les interactions de la lumière avec la matière. Les milieux participatifs en sont un parfait exemple. Ils sont incontournables et doivent être intégrés pendant les processus de création que ce soit pour simuler de la fumée, des nuages ou bien pour créer une atmosphère crédible.

Malgré les performances toujours plus grandes des ordinateurs modernes, il est encore actuellement impossible de reproduire fidèlement dans un temps raisonnable. Certains effets visuels comme les caustiques ou la diffusion multiple sont difficiles à reproduire pour certains types d'algorithmes, si bien que l'on se limite souvent à une simplification des interactions. C'est encore plus vrai pour les applications temps réels qui nécessitent souvent des rendus à 30 images par seconde.

La quantité et la qualité des images à calculer peuvent aussi être tellement importantes qu'il est nécessaire de mettre au point de nouveaux algorithmes. C'est le cas pour l'industrie cinématographique qui doit pouvoir anticiper et planifier ses rendus sur des fermes d'ordinateurs sur souvent plus d'une année et en même temps en contrôler rigoureusement l'aspect financier.

Nous allons présenter et analyser dans ce mémoire différentes techniques de simulation de milieux participatifs et proposer certaines directions pour de futurs travaux.

Mots clés: Milieux participatifs, diffusion, rendu, infographie.

ABSTRACT

Nowadays, the use of photo-realistic computer images is very common, whether for film making, creating virtual environments, video games, or even for pre-visualizing architectural projects. The understanding and technological advances of the last 20 years allow us to be able to create very realistic images. Indeed, it is sometimes difficult, if not impossible, to distinguish between a photo of a real object and a virtual image.

But to achieve such a high degree of realism, it is necessary to create complex and costly algorithms capable of simulating the physical laws governing interactions between light and matter. Participating media are a perfect example, as they are unavoidable and must be integrated during the creative processes, whether for simulating smoke or clouds, or for creating a credible atmosphere.

Despite the overpowering performance of modern computers, it is still currently impossible to reproduce participating media perfectly, and in a reasonable time. Some visual effects, such as caustics or multiple scattering, are difficult to reproduce for some kinds of algorithms, so one is often limited to a simplification of the interactions. This is even more true for real-time applications that often require rendering at 30 frames per second.

The quantity and the quality of the images to be calculated can also be so important that it is necessary to develop new algorithms. This is the case for the film industry, which must be able to anticipate and plan its renderings on computer farms, often over a year, and at the same time strictly control the financial aspect.

We will present in this thesis various techniques of simulation of participating media, analyse them, and propose certain directions for future work.

Keywords: Participating media, scattering, rendering, computer graphics.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES SIGLES

API	Interface applicative de programmation
BRE	Algorithme d'estimation de luminance par faisceaux
BVH	Hiérarchie de volumes englobants
CPU	Unité centrale de traitement
ETL	Équation du transport de la lumière
GPU	Unité de traitement graphique
PB	Algorithme de faisceau de photons (*photon beam*)
PDF	Densité de probabilité
PM	Algorithme du *photon mapping*
ETR	Équation du transfert radiatif
VPL	Lumière ponctuelle virtuelle
VRL	Algorithme des rayons de lumière virtuels
VSL	Lumière sphérique virtuelle

NOTATION

Symbole	Unité	Description
$\sigma_s(x)$	m^{-1}	Coefficient de diffusion au point x
$\sigma_a(x)$	m^{-1}	Coefficient d'absorption au point x
$\sigma_t(x)$	m^{-1}	Coefficient d'extinction au point x
α		Albedo
$f_r(x, \vec{\omega} \leftrightarrow \vec{\omega}')$	sr^{-1}	Fonction de réflectance bidirectionnelle (BRDF)
$p(x, \vec{\omega} \leftrightarrow \vec{\omega}')$	sr^{-1}	Fonction de phase normalisée
$\tau(x \leftrightarrow x')$	sans unité	Épaisseur optique : $\int_{x'}^{x} \sigma_t(x)dx$
$T_r(x \leftrightarrow x')$	sans unité	Transmittance de x à x' : $e^{-\tau(x \leftrightarrow x')}$
Ω	stéradian (sr)	Angle solide
$\Omega_{4\pi}$	stéradian (sr)	Angle solide sur la sphère
$\Omega_{2\pi}$	stéradian (sr)	Angle solide sur l'hémisphère
L	$\mathrm{W} \cdot \mathrm{m}^{-2} \cdot \mathrm{sr}^{-1}$	Luminance
$L(x \rightarrow \vec{\omega})$	$\mathrm{W} \cdot \mathrm{m}^{-2} \cdot \mathrm{sr}^{-1}$	Luminance sortante au point x
$L(x \leftarrow \vec{\omega})$	$\mathrm{W} \cdot \mathrm{m}^{-2} \cdot \mathrm{sr}^{-1}$	Luminance incidente au point x provenant de $\vec{\omega}$
L_e	$\mathrm{W} \cdot \mathrm{m}^{-2} \cdot \mathrm{sr}^{-1}$	Luminance émise
L_r	$\mathrm{W} \cdot \mathrm{m}^{-2} \cdot \mathrm{sr}^{-1}$	Luminance réfléchie
ξ		Nombre aléatoire uniforme : $\xi \in [0,1]$
E	$\mathrm{W} \cdot \mathrm{m}^{-2}$	Éclairement énergétique (*irradiance*)
A		Aire
$dA(x)$		Surface différentielle à x
$dA^{\perp}(x)$		Surface différentielle à x perpendiculaire à $\vec{\omega}$
V		Volume
x		Position
$\vec{n}$		Vecteur normal (normalisé) à la surface
$\vec{\omega}$		Direction normalisée qui pointe vers l'extérieur de la surface
$d\vec{\omega}$		Angle solide différentiel $= \sin\theta d\theta d\phi$
$d\vec{\omega}^{\perp}$		Angle solide projeté
$\Phi\omega^{\perp}$	lumen (lm)	Flux lumineux
$\int_{\mathcal{M}}$		Intégration sur la surface $\mathcal{M}$

À Mok-Mi et Émilie ainsi qu'à toute notre famille.

REMERCIEMENTS

Je remercie Derek Nowrouzezahrai et Pierre Poulin de m'avoir donné l'opportunité d'étudier dans le domaine qui me passionne et de m'avoir permis de faire en plus partie du laboratoire.

Je tiens particulièrement à remercier Derek de m'avoir fait faire un grand bond en avant dans mes connaissances en rendu et pour être surtout toujours disponible pour répondre à mes questions.

Je voudrais aussi dire merci à Mok-Mi de m'avoir maintes fois suggéré de retourner sur les bancs de l'école.

Et finalement, je remercie Émilie, qui malgré ses 7 mois a su m'apporter beaucoup de bonheur, sans pour autant m'en vouloir de parfois oublier l'heure du biberon.

CHAPITRE 1

INTRODUCTION

En regardant autour de nous, notre attention est souvent attirée par d'incroyables phénomènes naturels. Que ce soit des caustiques (voir la figure 1.1), le brouillard ou les nuages traversés par des rayons de lumière (voir les figures 1.2 et 1.3), tous ces effets sont produits par des interactions complexes entre d'infimes particules et la lumière.

Un des buts importants des images de synthèse est de créer des images qui donnent l'illusion de regarder la réalité. Pour arriver à les concevoir, il faut que la sensation de couleur d'un observateur qui regarde cette image soit la plus proche possible de la réalité.

Pour rendre crédible le résultat de ces simulations, il est nécessaire premièrement d'avoir une description aussi fidèle que possible du monde réel, comme la géométrie et les propriétés physiques des matériaux. Deuxièmement, il faut aussi avoir une modélisation physique correcte des émetteurs de lumière contenus dans le monde. Et finalement, il faut savoir intégrer les lois de la physique qui décrivent le fonctionnement de ces phénomènes optiques.

C'est en simulant ces lois par nos algorithmes que nous allons pouvoir connaître la distribution de la lumière dans notre environnement artificiel. Cette étape est généralement nommée l'étape globale, ou l'étape indépendante de la vue.

Étape globale : Cette étape détermine la quantité de lumière qui est réfléchie par une surface en un point. La lumière étant une onde électromagnétique, la distribution de la lumière en un point pour une direction donnée peut être représentée en fonction de la longueur d'onde. En général pour simplifier, on se limite à trois longueurs d'onde pour représenter les composantes rouge, vert et bleu, ce qui peut faire perdre un certain degré de réalisme suivant le type de simulation, mais se révèle souvent satisfaisant. Nous verrons que pour les scènes qui ne comportent pas de milieux participatifs, il suffit de calculer la luminance sur les points situés sur les surfaces. Dans le cas contraire, il faut calculer la luminance en un certain nombre de points de l'espace. Nous montrerons que la luminance sortante est affectée par la luminance incidente provenant des autres surfaces ainsi que par l'émission et les propriétés des matériaux, il s'agit ici de l'équation de rendu.

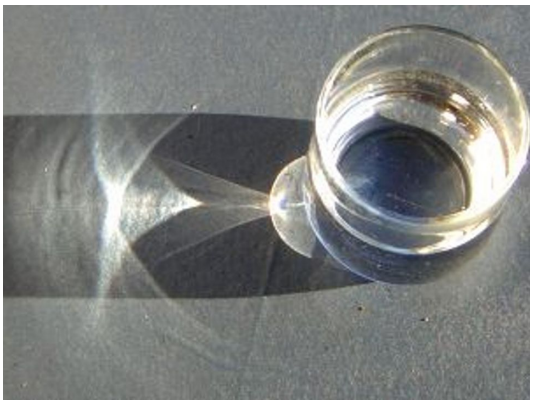
Ensuite, un outil de mesure qui simule l'œil ou la caméra est placé dans le monde virtuel. Cette fois, la distribution de la lumière va être mesurée suivant une position et une orientation. C'est l'étape locale ou l'étape dépendante de la vue.

Étape locale : L'étape locale consiste donc à mesurer la fonction de l'illumination globale par la caméra. Cette mesure est faite par chacun des capteurs de lumière qui correspond à un pixel dans l'image. Ces mesures sont linéaires par rapport à l'intensité reçue contrairement à notre œil et à nos écrans qui eux réagissent de manière non linéaire. Pour percevoir des images sur un écran de façon correcte, il faut donc compenser ces différences en utilisant une transformation non linéaire. C'est une transformation que l'on appelle « correction gamma » [37] que l'on doit appliquer aux images avant leur affichage. Il en existe différentes sortes suivant le type de matériel utilisé ainsi que l'utilisation finale des images. Toutes les images présentées dans ce document utilisent le standard sRGB [38].

Certains algorithmes ne font pas de différence entre ces deux étapes et calculent simultanément toutes les contributions qui affectent la caméra. C'est le cas du 1er algorithme que nous allons présenter, le *tracé de chemin* (voir la section 4.1), qui est simple et facile à mettre en place, et dont les résultats nous serviront de référence.

Par la suite, nous montrerons plusieurs autres techniques qui procèdent à une étape globale suivie d'une étape locale. Ces techniques peuvent être classées en deux groupes. Nous verrons premièrement le *tracé de photons* (voir la section 4.3), l'*estimation de luminance par faisceau* (voir la section 4.4) et les *faisceaux de photons* (voir la section 4.5) qui fonctionnent par estimation de densité de photons.
Puis les techniques de *lumière ponctuelle virtuelle* (voir la section 4.6) et les *rayons de lumière virtuels* (voir la section 4.7) qui fonctionnent par évaluation de l'illumination directe.

Mais avant d'entrer dans les détails du fonctionnement des algorithmes, nous allons commencer par expliquer les notions de base en radiométrie et en photométrie. Ces explications vont nous permettre de montrer comment résoudre mathématiquement l'équation de rendu pour les surfaces et les milieux participatifs. Et finalement, nous présenterons certaines techniques permettant d'améliorer nos estimations.



(a) Verre qui crée des caustiques sur le sol.
Heiner Otterstedt / Wikipedia sous licence Creative Commons.
`https://commons.wikimedia.org/wiki/File:Kaustik.jpg`

(b) Photo sous-marine avec des caustiques.
Mike Morris / Flickr sous licence Creative Commons.
`https://flic.kr/p/acbnqd`

Figure 1.1 : Exemples de caustiques réelles. La lumière suit un chemin spéculaire et termine sur une surface diffuse.

(a) Nuages avec diffusion de la lumière.
CrearSoft / Pixabay sous licence Creative Commons.
`https://pixabay.com/photo-1630574/`

(b) Nappe de brouillard.
Stevepb / Pixabay sous licence Creative Commons.
`https://pixabay.com/photo-404072/`

Figure 1.2 : Exemples de diffusion de la lumière dans un milieu participatif.

(a) Diffusion de la lumière.
MonikaP / Pixabay sous licence Creative Commons.
https://pixabay.com/photo-1503340/

(b) Diffusion de la lumière.
7854 / Pixabay sous licence Creative Commons.
https://pixabay.com/photo-54555/

Figure 1.3 : Exemples de diffusion de la lumière dans un milieu participatif avec des faisceaux très visibles.

CHAPITRE 2

RADIOMÉTRIE

Nous allons ici expliquer les notions de radiométrie et de photométrie nécessaires pour décrire le comportement de la lumière. À l'aide de ces outils, nous allons par la suite pouvoir dériver les équations de rendu.

2.1 Flux lumineux

En radiométrie, la quantité la plus petite est le photon. L'énergie e_λ d'un photon pour une longueur d'onde λ est :

$$e_\lambda = \frac{hc}{\lambda}, \tag{2.1}$$

avec la constante de Planck $\hbar \approx 6.62 \times 10^{-34}\,\mathrm{J}\cdot\mathrm{s}$ et c_0 la vitesse de la lumière (ou célérité) dans le vide $c_0 \approx 3 \times 10^8\,\mathrm{m}\cdot\mathrm{s}^{-1}$. Dans le domaine de la radiométrie, le flux lumineux Φ est la grandeur photométrique correspondante à la puissance rayonnée (ou flux énergétique) par une source dans un angle solide donné (voir l'équation 2.3). Le flux lumineux représente la fraction du flux énergétique correspondant à une lumière visible par l'homme, pondérée selon sa longueur d'onde. Un rayonnement électromagnétique produit une sensation visuelle d'intensité très variable et nulle en dehors du spectre visible. Son unité est le Watt $[\mathrm{W} = \mathrm{J}\cdot\mathrm{s}^{-1}]$ et il est formulé de la manière suivante :

$$\Phi = \frac{dQ}{dt}, \tag{2.2}$$

avec l'énergie rayonnante Q ayant pour unité le Joule [J]. Il s'agit de l'énergie d'une certaine quantité de photons intégrés sur toutes les longueurs d'onde :

$$Q(x) = \int_0^\infty Q_\lambda \, d\lambda. \tag{2.3}$$

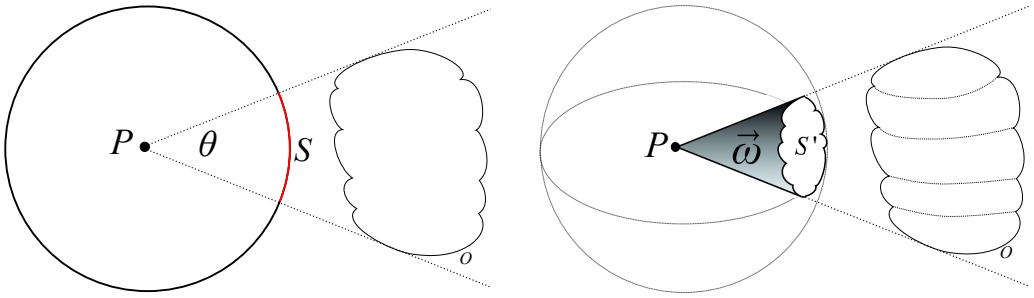
2.2 L'éclairement énergétique (ou irradiance)

L'irradiance E est la puissance d'un rayonnement électromagnétique frappant par unité d'aire surfacique. Son unité est le Watt par mètre carré $[\mathrm{W}\cdot\mathrm{m}^{-2}]$. On peut l'exprimer en termes de flux énergétique :

$$E(x) = \frac{d\Phi(x)}{dA(x)}. \tag{2.4}$$

L'exitance énergétique M est une grandeur qui désigne le flux émis par unité de surface d'une source étendue qui rayonne. Son unité est le Watt par mètre carré $[\mathrm{W} \cdot \mathrm{m}^{-2}]$.

2.3 Angle solide



(a) Angle planaire. (b) Angle solide.

Figure 2.1 : Illustration d'un angle planaire et solide.

L'*angle solide* est le pendant de l'angle en deux dimensions dans le plan à un angle sur une sphère. L'*angle planaire* est l'angle sous-tendant un objet par respect à un point P. Si l'on projette un objet O sur un cercle unitaire, une partie du cercle sera couverte par la projection. Cet arc S correspond à l'angle sous-tendant θ (voir la figure 2.1(a)) ; il est mesuré en *radian*. Le cercle unitaire entier correspond à un angle planaire de 2π. L'*angle solide* étend le cercle 2D unitaire à une sphère 3D unitaire. Si l'on projette un objet O sur une sphère unitaire, une partie S' de la surface de la sphère sera couverte par la projection. Cette surface correspond à l'angle solide sous-tendant (voir la figure 2.1(b)), il est mesuré en *stéradian* sr. La sphère unitaire entière a un *angle solide* de 4π, celui de l'hémisphère correspond à 2π. L'*angle solide* est noté par le symbole ω. C'est le rapport entre la surface S de la projection d'un objet sur une sphère et son rayon r au carré :

$$\vec{\omega} = \frac{S}{r^2}. \tag{2.5}$$

2.4 L'intensité lumineuse

L'intensité lumineuse I d'une source, dans une direction donnée, correspond au flux lumineux émis par unité d'angle solide. Son unité est le Watt par *stéradian* $[\mathrm{W} \cdot \mathrm{sr}^{-1}]$.

On peut formuler cette relation en termes de flux lumineux :

$$I(\vec{\omega}) = \frac{\mathrm{d}\Phi(\vec{\omega})}{d\vec{\omega}}. \tag{2.6}$$

2.5 Équation du transport de la lumière

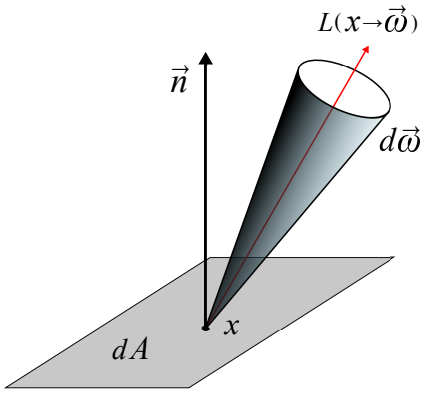


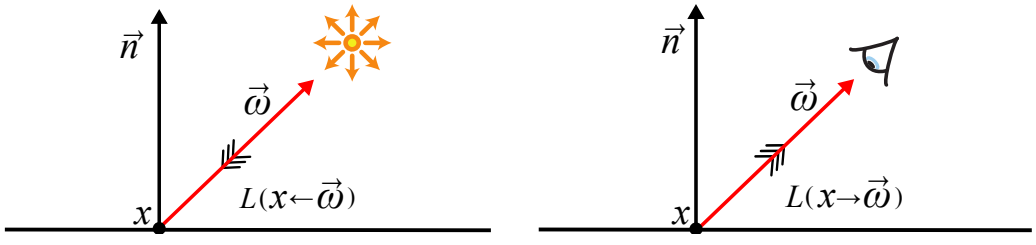
Figure 2.2 : Représentation de la luminance.

La luminance énergétique L, également nommée radiance, est la quantité de radiation émise (voir la figure 2.2) ou réfléchie par une surface dans un certain angle solide par unité d'aire surfacique. Son unité est le Watt par mètre carré par stéradian $[\mathrm{W} \cdot \mathrm{m}^{-2} \cdot \mathrm{sr}^{-1}]$. On l'exprime à partir de l'intensité lumineuse (voir l'équation 2.6) par :

$$L(x \rightarrow \vec{\omega}) = \frac{d^2\Phi(\vec{\omega})}{d\vec{\omega}\, dA^{\perp}(x)} = \frac{d^2\Phi(\vec{\omega})}{d\vec{\omega}\, dA(x)\, (\vec{\omega} \cdot \vec{n})}, \tag{2.7}$$

avec la luminance énergétique L, $dA^{\perp}$ l'aire projetée de dA sur une surface perpendiculaire à $\vec{\omega}$ et $d\vec{\omega}$ représente l'angle solide d'émission. En pratique on connaît la surface et donc la normale $\vec{n}$ au point x, ce qui permet de dire que : $dA^{\perp}(x) = dA(x)(\vec{\omega} \cdot \vec{n})$ avec $\vec{n}$ qui désigne la normale normalisée à la surface.

2.6 Luminance sortante et entrante



(a) Luminance incidente. (b) Luminance sortante.

Figure 2.3 : Illustration de la différence entre la luminance entrante et sortante.

La luminance entrante est notée par $L(x \leftarrow \vec{\omega})$ (voir la figure 2.3(a)) et la luminance sortante par $L(x \rightarrow \vec{\omega})$ (voir la figure 2.3(b)). Il est important de voir que le vecteur $\vec{\omega}$ est toujours orienté vers l'extérieur de la surface.

2.7 Relations radiométriques

Nous pouvons maintenant écrire l'éclairement énergétique (voir l'équation 2.4) en fonction de la luminance :

$$E(x) = \int_{\Omega_{2\pi}} L_i(x \leftarrow \vec{\omega})(\vec{\omega} \cdot \vec{n})\, d\vec{\omega}, \tag{2.8}$$

autrement dit, l'éclairement énergétique en un point x est la totalité de la luminance incidente L_i de tous les angles solides $\vec{\omega}$ composant l'hémisphère $\Omega_{2\pi}$. On peut aussi

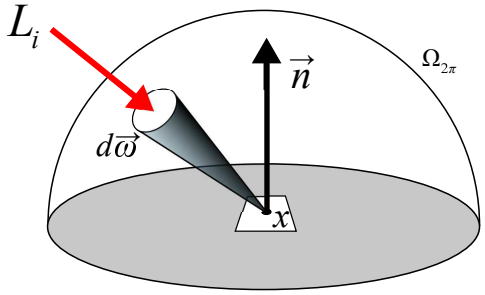


Figure 2.4 : Représentation de l'éclairement énergétique E en fonction de la luminance incidente L_i en un point x.

exprimer le flux Φ en fonction de l'aire A d'une surface et de l'angle solide $\vec{\omega}$ par :

$$\Phi = \int_A \int_{\Omega_{2\pi}} L(x, \vec{\omega})(\vec{\omega} \cdot \vec{n})\, d\vec{\omega} dx = \int_A E(x)\, dx. \tag{2.9}$$

2.8 Équation du rendu surfacique

L'équation du rendu telle que définie par Kajiya [18] décrit la luminance sortante comme la somme des luminances réfléchies et émises :

$$\underbrace{L_o(x \rightarrow \vec{\omega})}_{\text{sortante}} = \underbrace{L_e(x \rightarrow \vec{\omega})}_{\text{émise}} + \underbrace{L_r(x \rightarrow \vec{\omega})}_{\text{réfléchie}}. \tag{2.10}$$

Pour définir la luminance réfléchie, il est nécessaire d'introduire la fonction de réflectance bidirectionnelle *BRDF* (*bidirectional reflectance distribution function*). Cette fonction exprime la relation entre le flux énergétique et la luminance réfléchie au point x :

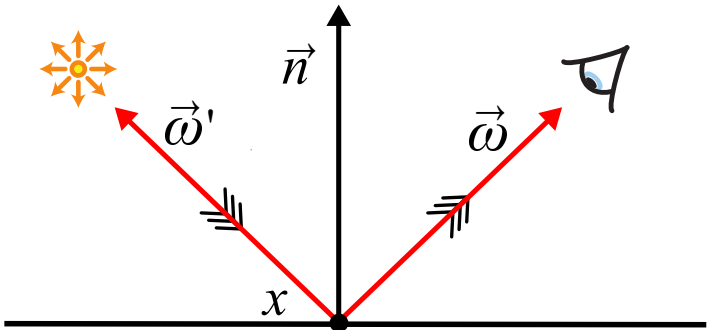


Figure 2.5 : Illustration des termes utilisés par la BRDF.

$$f_r(x, \vec{\omega}' \leftrightarrow \vec{\omega}) = \frac{dL(x \rightarrow \vec{\omega})}{dE(x \leftarrow \vec{\omega}')} = \frac{dL(x \rightarrow \vec{\omega})}{L(x \leftarrow \vec{\omega}')(\vec{\omega}' \cdot \vec{n})\, d\vec{\omega}'}. \tag{2.11}$$

On en déduit l'équation intégrale de Fredholm du second ordre suivante :

$$L_r(x \rightarrow \vec{\omega}) = \int_{\Omega_{2\pi}} f_r(x, \vec{\omega}' \leftrightarrow \vec{\omega}) L_i(x \leftarrow \vec{\omega}')(\vec{\omega}' \cdot \vec{n})\, d\vec{\omega}'. \tag{2.12}$$

Par substitutions de l'équation 2.12 dans l'équation 2.10 on trouve l'équation du rendu complète :

$$\underbrace{L_o(x \rightarrow \vec{\omega})}_{\text{sortante}} = \underbrace{L_e(x \rightarrow \vec{\omega})}_{\text{émise}} + \underbrace{\int_{\Omega_{2\pi}} f_r(x, \vec{\omega}' \leftrightarrow \vec{\omega}) L_i(x \leftarrow \vec{\omega}')(\vec{\omega}' \cdot \vec{n})\, d\vec{\omega}'}_{\text{réfléchie}}. \tag{2.13}$$

On notera que la luminance L apparaît des deux côtés de l'équation, ce qui signifie que pour calculer la luminance sortante $L_o(x \rightarrow \vec{\omega})$ en un point x dans la direction $\vec{\omega}$, nous devons aussi avoir la luminance entrante $L_i(x \leftarrow \vec{\omega}')$ toujours à la position x pour toutes les directions $\vec{\omega}'$.

On remarquera que si l'on considère que la lumière voyage dans le vide, nous avons :

$$L(x \rightarrow \vec{\omega}) = L(x \leftarrow -\vec{\omega}), \tag{2.14}$$

nous pouvons alors utiliser la luminance sortante à une position x_o comme luminance entrante à une position x_i (voir la figure 2.6) :

$$L(x_i \leftarrow \vec{\omega}) = L(x_o \rightarrow -\vec{\omega}). \tag{2.15}$$

Il est important de noter que la BRDF (voir l'équation 2.11) doit respecter deux propriétés de la réflectance. La première est la réciprocité d'*Helmholtz* (principe du retour inverse de la lumière) qui force la BRDF à être symétrique par rapport aux directions $\vec{\omega}$

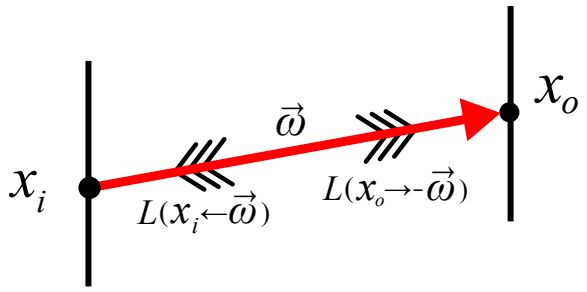


Figure 2.6 : Illustration de la réciprocité de la luminance entrante et sortante. La luminance entrante en x_i est égale à la luminance sortante en x_o.

et $\vec{\omega}'$: $f_r(x, \vec{\omega}' \leftrightarrow \vec{\omega}) = f_r(x, \vec{\omega} \leftrightarrow \vec{\omega}')$. La seconde nous dit que la BRDF doit suivre la loi de la conservation d'énergie, ce qui se traduit par :

$$\int_{\Omega_{2\pi}} f_r(x, \vec{\omega}' \leftrightarrow \vec{\omega})(\vec{\omega}' \cdot \vec{n})\, d\vec{\omega}' \leq 1,\ \forall \vec{\omega}. \tag{2.16}$$

Avant de décrire l'équation du rendu des milieux participatifs, il faut définir la fonction de phase qui est le pendant de la BRDF pour les volumes.

2.9 Fonction de phase

La fonction de phase (aussi appelée fonction de diffusion) va apparaître dans la définition de l'équation de transfert radiatif que nous verrons après (voir la section 2.10.1). Il est plus aisé de l'isoler et de la définir en premier.

La fonction de phase décrit la distribution de la diffusion angulaire de la lumière, c'est l'équivalent volumique de la BRDF (voir l'équation 2.11). Elle a deux propriétés importantes : elle doit premièrement être normalisée $\int_{\Omega_{4\pi}} \rho(\omega_i \rightarrow \omega_o)\, d\vec{\omega}_i = 1$, et comme pour la BRDF, elle doit respecter la réciprocité d'*Helmholtz* définie par $\rho(\omega_i \rightarrow \omega_o) = \rho(\omega_o \rightarrow \omega_i)$. Son unité est [$\text{sr}^{-1}$]. Il est important que cette fonction soit simple à calculer, paramétrable et si possible inversible afin de pouvoir réaliser un échantillonnage préférentiel (voir la section 3.3). La fonction ne dépend en général que de l'angle θ entre le rayon incident et le rayon sortant (diffusé). Elle est définie par une fonction $\rho(\theta)$ avec $\theta = 0$ correspondant à la direction vers l'avant et $\theta = \pi$ vers l'arrière. Elle est aussi dépendante des longueurs d'onde, mais par simplification on n'en tient généralement pas compte.

2.9.1 Diffusion isotrope

La diffusion isotrope est la fonction de phase la plus simple, car c'est une constante quel que soit l'angle incident

$$\rho(\theta) = \frac{1}{4\pi}. \tag{2.17}$$

2.9.2 Henyey-Greenstein

À l'origine, la fonction de phase d'Henyey-Greenstein a été créée pour expliquer la diffusion de la poussière inter-galactique [11].

$$\rho(\theta, g) = \frac{(1-g^2)}{4\pi(1+g^2-2\,g\cos\theta)^{1.5}}. \tag{2.18}$$

Elle prend un paramètre supplémentaire g, ce dernier est un facteur d'anisotropie qui permet de changer la forme de distribution angulaire :

$$g \in]-1,1[. \tag{2.19}$$

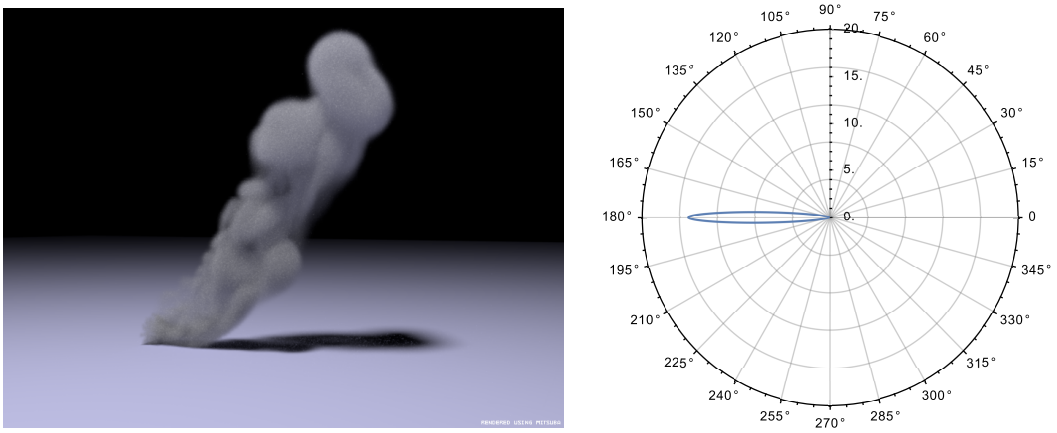
Elle est inversible, ce qui nous permet de l'évaluer avec un échantillonnage préférentiel (voir la section 3.3) afin de diminuer la variance dans le cas d'utilisation basée sur des techniques stochastiques telles que la méthode de Monte-Carlo. Pour trouver l'angle de la prochaine direction de diffusion, il suffit d'évaluer l'équation :

$$\cos\theta = \frac{1}{|2g|}\left(1+g^2-\left(\frac{1-g^2}{1-g+2\,g\,\xi}\right)^2\right), \tag{2.20}$$

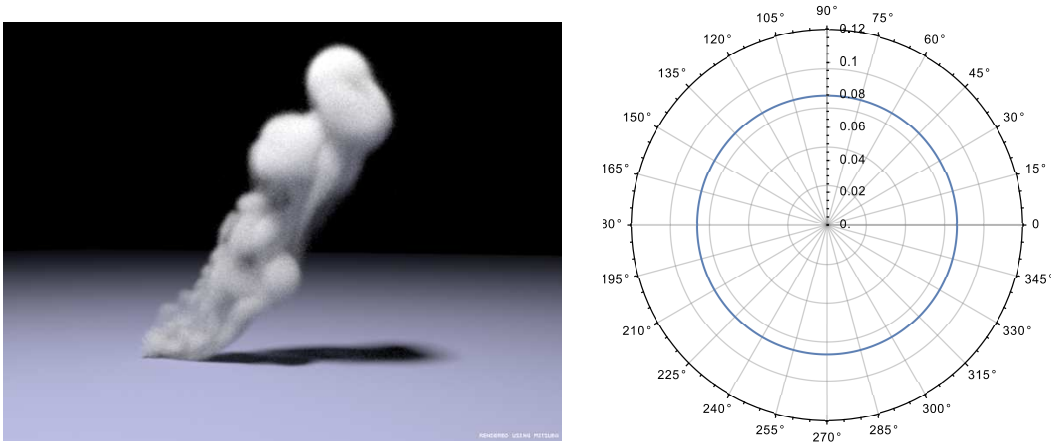
avec ξ un nombre aléatoire $\in [0,1]$. La figure 2.7 nous montre différents rendus utilisant la fonction de phase Henyey-Greenstein. On peut voir que le milieu participatif a une apparence qui diffère suivant la valeur du terme g.

2.9.3 Autres fonctions de phase

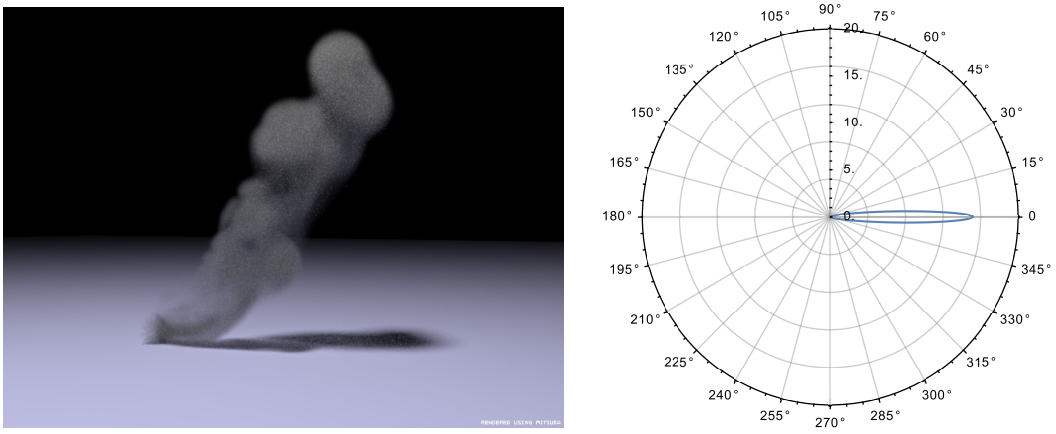
Il existe de nombreuses autres fonctions de phase, comme celles de Rayleigh, Mie [6, 12, 24, 30], Schlik [2]. Par simplification, nous utiliserons la fonction de phase isotrope pour nos résultats.



(a) $g = -0.9$, peu de lumière arrive à l'œil, elle repart principalement en arrière.



(b) $g = 0.0$, la diffusion est isotrope.



(c) $g = 0.9$, peu de lumière arrive à l'œil, elle traverse principalement en avant.

Figure 2.7 : Exemples de rendu volumique utilisant la fonction de phase Henyey-Greenstein et sa représentation en coordonnées polaires.

2.10 Équation du rendu des milieux participatifs

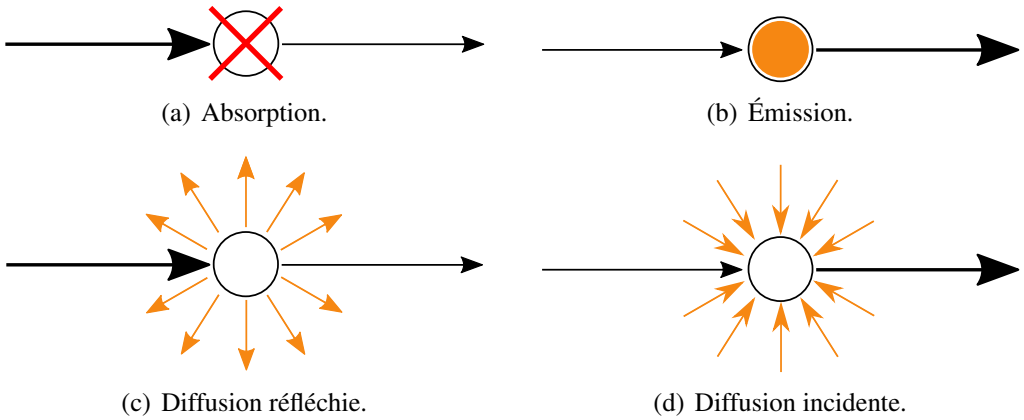
L'équation du rendu que nous avons vue précédemment, suppose que l'espace est fait de vide, ce qui n'est pas le cas dans la réalité. L'espace entre les surfaces est en général rempli de particules de différentes tailles, homogènes ou hétérogènes. La lumière qui passe à travers de tels milieux en est alors affectée. Comme il est impossible de calculer toutes les interactions entre chacune des particules prises individuellement et la lumière, nous utiliserons un modèle probabiliste pour approximer les comportements.

2.10.1 Équation du transfert radiatif

Le transport de la lumière dans les milieux participatifs est décrit par l'équation du transfert radiatif. Elle a été introduite en astrophysique pour décrire la propagation du rayonnement dans les milieux interstellaires [5], et en neutronique pour d'écrire la propagation des neutrons dans des réacteurs [4]. Pour simuler le comportement de la lumière dans un médium, on va décrire le changement de luminance le long d'un faisceau lumineux pour un pas infinitésimal. C'est-à-dire que l'on va regarder le changement différentiel de luminance au point x dans la direction $\vec{\omega}$:

$$(\vec{\omega} \cdot \nabla) L(x \rightarrow \vec{\omega}). \tag{2.21}$$

La luminance est affectée par quatre interactions (voir la figure 2.8) : l'absorption, l'émission, la diffusion réfléchie et la diffusion incidente. Nous allons détailler ces quatre termes puis écrire l'équation complète résultante.



(a) Absorption. (b) Émission.

(c) Diffusion réfléchie. (d) Diffusion incidente.

Figure 2.8 : Illustration des quatre interactions de l'équation.

Absorption : Cette interaction intervient quand la lumière est convertie dans une autre forme d'énergie (ex. la chaleur). La quantité de lumière absorbée est notée σ_a. C'est la

densité de probabilité que la lumière soit absorbée par unité de distance dans le milieu. Le changement de luminance L dans la direction $\vec{\omega}$ résultant de l'absorption σ_a est :

$$(\vec{\omega} \cdot \nabla) L(x \to \vec{\omega}) = -\sigma_a(x) L(x \to \vec{\omega}). \tag{2.22}$$

Cette fonction peut varier suivant la position x, la direction $\vec{\omega}$ ainsi qu'en fonction du spectre.

Auto émission : Un milieu peut aussi produire de la lumière, suite par exemple à une réaction chimique. La quantité de lumière émise est définie par la *fonction source* $Q(x \to \vec{\omega})$ [39] et peut varier suivant la position x et la direction $\vec{\omega}$:

$$(\vec{\omega} \cdot \nabla) L(x \to \vec{\omega}) = Q(x \to \vec{\omega}). \tag{2.23}$$

Dispersion réfléchie : Une partie de la lumière peut être diffusée dans des directions autres que $\vec{\omega}$. Cette quantité est notée σ_s. Le changement de luminance L résultant de la diffusion sortante σ_s est :

$$(\vec{\omega} \cdot \nabla) L(x \to \vec{\omega}) = -\sigma_s(x) L(x \to \vec{\omega}). \tag{2.24}$$

Dispersion incidente : Il faut aussi tenir compte de la lumière qui est diffusée depuis les autres directions $\vec{\omega}'$ vers la direction $\vec{\omega}$ et qui augmente la luminance tout le long de cette direction. Il faut calculer la luminance incidente L_i, qui sera intégrée sur la sphère et multipliée par une *fonction de phase* $\rho(x, \vec{\omega}' \leftrightarrow \vec{\omega})$. La fonction de phase décrit la distribution de la diffusion angulaire de la lumière, c'est l'équivalent volumique de la BRDF (voir l'équation 2.11).

$$\begin{aligned} (\vec{\omega} \cdot \nabla) L(x \to \vec{\omega}) &= \sigma_s(x) L_i(x \to \vec{\omega}) \\ &= \sigma_s(x) \int_{\Omega_{4\pi}} \rho(x, \vec{\omega} \leftrightarrow \vec{\omega}') L(x \leftarrow \vec{\omega}') \, d\vec{\omega}'. \end{aligned} \tag{2.25}$$

2.10.2 Coefficient d'extinction

On peut remarquer que la luminance est réduite par l'absorption (voir l'équation 2.24) et par la diffusion réfléchie (voir l'équation 2.22). Pour simplifier, on les combine en un seul terme, le *coefficient d'extinction* noté σ_t et qui est défini par :

$$\sigma_t = \sigma_a + \sigma_s, \tag{2.26}$$

ce qui donne :

$$(\vec{\omega} \cdot \nabla) L(x \to \vec{\omega}) = -\sigma_t(x) L(x \to \vec{\omega}). \tag{2.27}$$

2.10.3 Équation complète du transfert radiatif

En ajoutant les trois termes, extinction (voir l'équation 2.27), émission (voir l'équation 2.23) et diffusion (voir l'équation 2.24), nous obtenons l'équation complète du transfert radiatif :

$$(\vec{\omega}\cdot\nabla)L(x\rightarrow\vec{\omega}) = -\sigma_t(x)L(x\rightarrow\vec{\omega}) + Q(x\rightarrow\vec{\omega}) + \sigma_s(x)\int_{\Omega_{4\pi}}\rho(x,\vec{\omega}\leftrightarrow\vec{\omega}')L(x\leftarrow\vec{\omega}')\,d\vec{\omega}'. \tag{2.28}$$

Nous avons donc maintenant une équation qui nous donne la luminance sortante pour un point de l'espace et une direction donnée en tenant compte des différentes interactions avec le médium.

2.10.4 Luminance énergétique volumique

Pour un volume, nous avons quelque chose de très semblable à ce que nous avons vu pour les surfaces (voir l'équation 2.7) :

$$L(x,\vec{\omega}) = \frac{d^2\Phi(x,\vec{\omega})}{\sigma_s(x)\,d\vec{\omega}\,dV}. \tag{2.29}$$

2.10.5 Équation du rendu volumique

À partir de l'équation de transfert radiatif vue précédemment, on va pouvoir décrire comment la lumière est transportée d'une surface à l'autre. Pour des raisons de simplification, nous ne tiendrons compte uniquement de l'extinction (absorption et diffusion réfléchie) et de la diffusion incidente, l'émission sera donc omise, il est très facile de l'ajouter si besoin. En intégrant l'extinction (voir l'équation 2.27) sur $x \leftrightarrow x_s$ nous obtenons la transmittance T_r :

$$T_r(x\leftrightarrow x_s) = e^{-\tau(x\leftrightarrow x_s)}, \tag{2.30}$$

le terme τ est la profondeur optique, il est défini par cette équation :

$$\tau(x\leftrightarrow x_s) = \int_x^{x_s}\sigma_t(x)\,dx. \tag{2.31}$$

Il est important de noter que σ_t est constant dans les milieux homogènes, ce qui permet la simplification suivante pour ce cas :

$$\tau(x\leftrightarrow x_s) = \sigma_t \parallel x_s - x \parallel . \tag{2.32}$$

À partir de l'équation (voir l'équation 2.30) on peut définir la *luminance surfacique réduite*, qui est la luminance au point x' résultant de la luminance sortant au point x_s (voir la fi-

gure 2.10) :

$$L(x \leftarrow \vec{\omega}) = T_r(x \leftrightarrow x_s) L(x_s \rightarrow -\vec{\omega}). \tag{2.33}$$

On ajoute maintenant la diffusion incidente (voir l'équation 2.25) accumulée sur $x \leftrightarrow x_s$ (voir la figure 2.9) pour avoir l'équation du rendu volumique. L'émittance accumulée (voir l'équation 2.23) a été omise pour simplification :

$$\begin{aligned} L(x \leftarrow \vec{\omega}) = & \underbrace{T_r(x \leftrightarrow x_s) L(x_s \rightarrow -\vec{\omega})}_{\text{luminance surfacique réduite}} + \\ & \underbrace{\int_x^{x_s} T_r(x \leftrightarrow x_t) \sigma_s(x_t) L_i(x_t \rightarrow -\vec{\omega}) \, dx_t}_{\text{dispersion incidente accumulée}}, \end{aligned} \tag{2.34}$$

avec la luminance incidente L_i définie par l'équation intégrale de Fredholm du second ordre suivante :

$$L_i(x_t \rightarrow \vec{\omega}) = \int_{\Omega_{4\pi}} \rho(x, \vec{\omega} \leftrightarrow \vec{\omega}') L(x \leftarrow \vec{\omega}') \, d\vec{\omega}'. \tag{2.35}$$

Le premier terme est la luminance réduite surfacique (voir la figure 2.10) qui est responsable de la réduction de la luminance à partir de la surface x_s. Le deuxième terme est la luminance incidente accumulée (voir la figure 2.9) qui représente l'accumulation de la luminance diffusée dans la direction $-\vec{\omega}$ depuis l'intérieur du milieu participatif sur la distance $x \leftrightarrow x_s$. On remarquera que la luminance incidente accumulée est le terme le plus coûteux, il faut l'intégrer sur la distance $x \leftrightarrow x_s$ et **elle est récursive**.

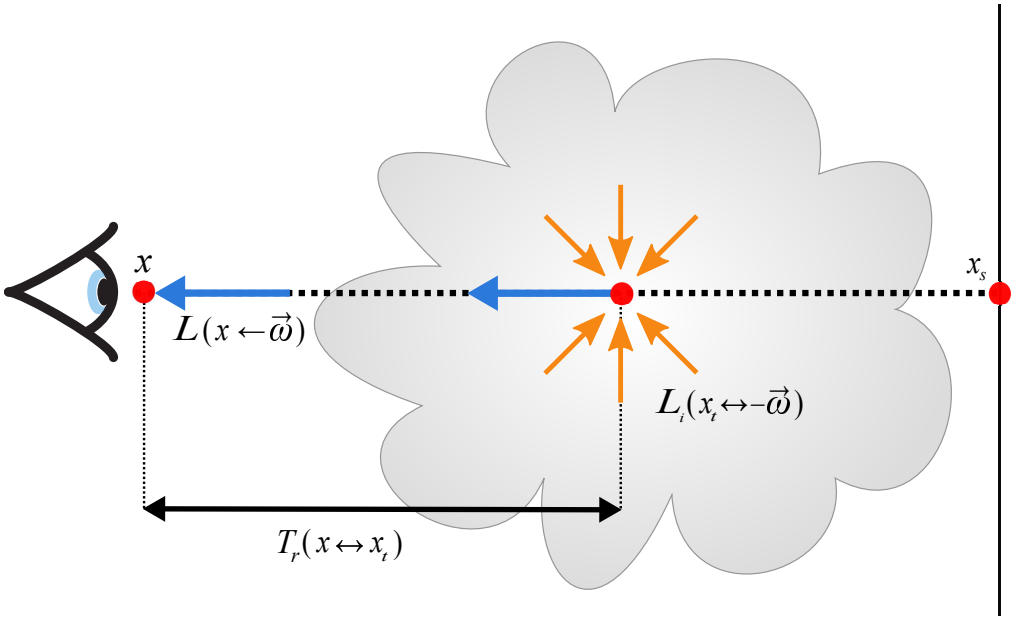


Figure 2.9 : Illustration de la diffusion incidente accumulée. On accumule L_i au point x_t le long du segment $[x, x_s]$ en modulant par la transmittance T_r de x_t au point x.

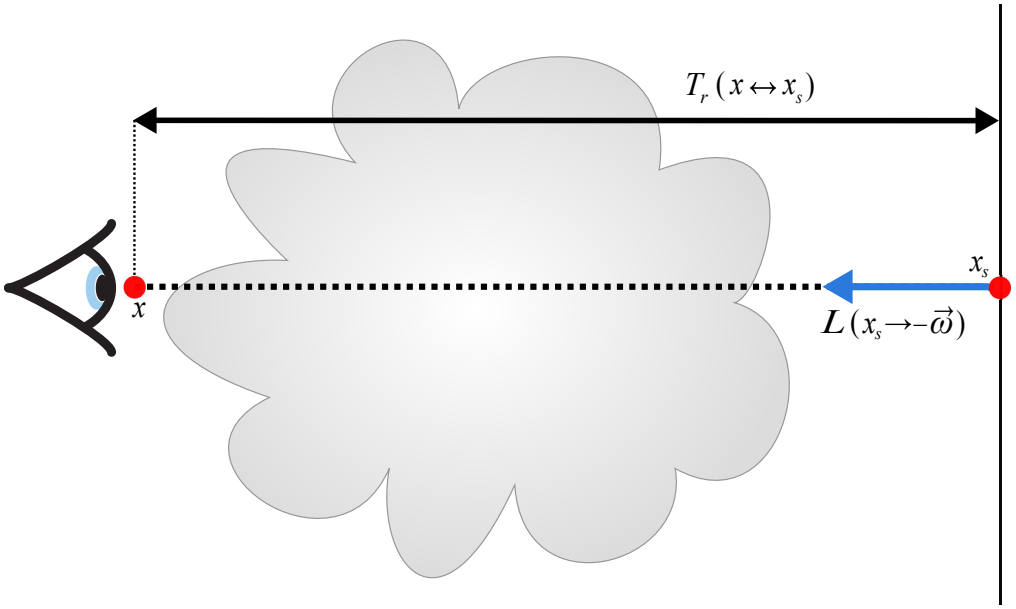


Figure 2.10 : Illustration des termes de la luminance surfacique réduite. L est calculée au point x_s et est modulée par la transmittance T_r de x_s au point x.

CHAPITRE 3

INTÉGRALE DE CHEMIN ET STRATÉGIE D'ÉCHANTILLONNAGE

Dans cette section, nous allons montrer comment transformer l'équation du rendu sous une forme exploitable par nos algorithmes. Nous exposerons une brève introduction au calcul intégral avec la méthode de Monte Carlo ainsi que des techniques pour améliorer ses performances.

3.1 Intégrale de chemin

La formulation d'intégrale de chemin (*path integral form*) [31, p.30 ; 16, p.30] est une autre façon d'exprimer l'équation du rendu pour la rendre utilisable par nos algorithmes. Pour la développer, il faut dans un premier temps changer la paramétrisation de l'intégration (voir l'équation 3.3), où l'on passe d'une intégration par angle solide à une intégration par surface. Pour rappel, la différentielle de l'aire est reliée à l'angle solide différentiel (voir la figure 3.1) par :

$$d\vec{\omega}(x) = \frac{dA'(x')\,\cos\theta'}{r^2}, \tag{3.1}$$

avec x un point, dA' une surface, x' un point sur cette surface, $\vec{n}'$ une normale à la surface,

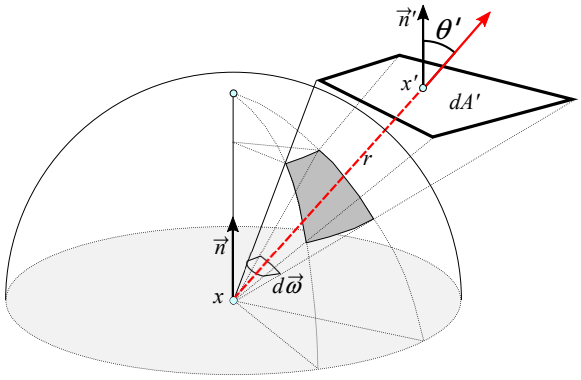


Figure 3.1 : L'angle solide différentiel $d\vec{\omega}$ s'exprime en fonction de la distance r et de l'angle θ' à la surface dA'. Si $r = 1$ et $\cos\theta' = 1$ alors $d\vec{\omega} = dA'$.

$r = \|x' - x\|^2$ et θ' l'angle entre n' et le vecteur $\overrightarrow{xx'}$. Intuitivement r^2 compense pour la distance et θ' pour l'alignement. On va aussi créer un nouveau terme G qui reprend $\cos\theta$ de l'équation du transport de la lumière (voir l'équation 2.7) avec le Jacobien vu

précédemment tel que :

$$G(x,x') = \frac{(\vec{\omega}\cdot\vec{n}')(\vec{\omega}'\cdot\vec{n})}{\|x'-x\|^2}, \tag{3.2}$$

avec les vecteur $\vec{\omega}, \vec{\omega}', \vec{n}$ et $\vec{n}'$ orientés vers le dessus de la surface (*front facing*).

On obtient la version surfacique de l'équation du transfert de lumière :

$$\begin{aligned} L_o(x \to \vec{\omega}) = L_e(x \to \vec{\omega}) + \\ \int_S f_r(x, x' \to x, \vec{\omega}) L_i(x' \to x) V(x,x') G(x,x')\, dA', \end{aligned} \tag{3.3}$$

$$V(x,x') = \begin{cases} 1 \text{ si } x \text{ et } x' \text{ sont mutuellement visibles,} \\ 0 \text{ sinon.} \end{cases} \tag{3.4}$$

Il est maintenant possible d'exprimer l'équation du rendu à partir de trois points (voir la figure 3.2), x, x' et x'' :

$$\begin{aligned} L_o(x' \to x) = L_e(x' \to x) + \\ \int_S f_r(x'' \to x' \to x) L_i(x'' \to x') V(x',x'') G(x',x'')\, dA''. \end{aligned} \tag{3.5}$$

On voit que l'on ne peut pas directement utiliser l'équation du rendu (voir l'équation 3.5)

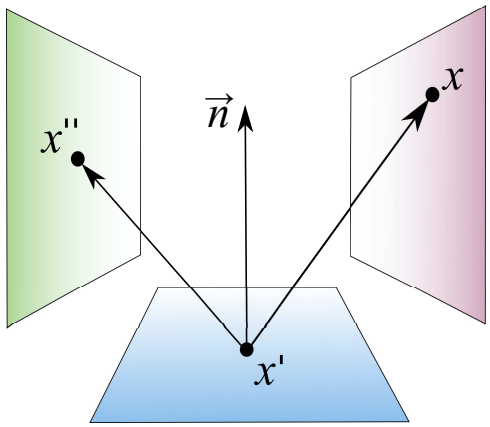


Figure 3.2 : Interprétation géométrique des points x, x' et x'' dans l'équation 3.5. La forme en trois points de l'équation du transport de la lumière convertit l'intégrande sur les angles solides en une intégrande avec un domaine basé sur des points sur les surfaces.

car la luminance apparaît des deux côtés. Il y a trois catégories de solutions à ce type de problème : l'inversion, l'expansion et l'itération. C'est l'expansion que nous allons utiliser pour exprimer le problème sous une forme d'une multitude de chemins. Il suffit de remplacer la luminance L de façon récursive dans la partie de droite par son expression.

Ce qui donne une équation d'une taille infinie et difficilement manipulable. L'approche la plus pratique est l'utilisation des séries de Neumann, ce qui permet d'exprimer l'équation du rendu de façon toujours récursive, mais compacte :

$$L = L_e + \mathcal{T}L, \tag{3.6}$$

avec un opérateur $\mathcal{T}$:

$$\mathcal{T}\{F\} = \int_{\Omega_{2\pi}} f_r(x, \vec{\omega}' \leftrightarrow \vec{\omega}) F(x \leftarrow \vec{\omega}')(\vec{\omega}' \cdot \vec{n})\, d\vec{\omega}'. \tag{3.7}$$

Ce qui de façon récursive donne :

$$L = L_e + \mathcal{T}L_e + \mathcal{T}^2 L_e + \mathcal{T}^3 L_e + \mathcal{T}^4 L_e + ... = \sum_{m=0}^{\infty} \mathcal{T}^m L_e. \tag{3.8}$$

On peut interpréter cette série de termes comme la somme des chemins organisés par le nombre de rebonds (voir la figure 3.4). On voit dans la colonne de droite que le rendu est de plus en plus sombre car l'opérateur $\mathcal{T}$ est une contraction, c.-à-d. $\lim_{n\to\infty} \mathcal{T}^{n+1} L = 0$. Le 1ier terme L_e est l'émission, il n'y a donc aucun rebond. Le 2ème terme $\mathcal{T}L_e$ est l'émission suivie d'un rebond, etc. On peut maintenant réécrire l'équation du rendu comme une somme de chemins de longueurs k :

$$\begin{aligned} L_o(x_o, \vec{\omega}') = \sum_{k=0}^{\infty} \int_{\mathcal{M}} \int_{\mathcal{M}} ... \int_{\mathcal{M}} K(x_k, x_{k-1}, x_{k-2}) ... K(x_1, x_0, x_{-1}) \times \\ L_e(x_k \to x_{k-1})\, dA_k\, dA_{k-1} ...\, dA_0, \end{aligned} \tag{3.9}$$

avec $\vec{\omega} = x_{-1} - x_0$, $x-1$ comme étant la position de l'observateur et avec :

$$K(x'', x', x) = f_r(x'' \to x' \to x) V(x'', x') G(x'', x'). \tag{3.10}$$

Pour illustrer ce résultat, nous allons le développer pour quatre points (voir l'équation 3.8) :

$$\begin{aligned} L(x_1 \to x_0) = L_e(x_1 \to x_0) + \\ \int_{\mathcal{M}} L_e(x_2 \to x_1) f_r(x_2 \to x_1 \to x_0) V(x_2, x_1) G(x_2, x_1)\, dA_{x_2} + \\ \int_{\mathcal{M}} \int_{\mathcal{M}} L_e(x_3 \to x_2) f_r(x_3 \to x_2 \to x_1) V(x_3, x_2) G(x_3, x_2)\, dA_{x_0} \times \\ \int_{\mathcal{M}} L_e(x_2 \to x_1) f_r(x_2 \to x_1 \to x_0) V(x_2, x_1) G(x_2, x_1)\, dA_{x_2} + ... \end{aligned} \tag{3.11}$$

Nous obtenons une somme d'intégration infinie que l'on peut représenter géométriquement par la figure 3.3 et un exemple visuel dans la figure 3.4.

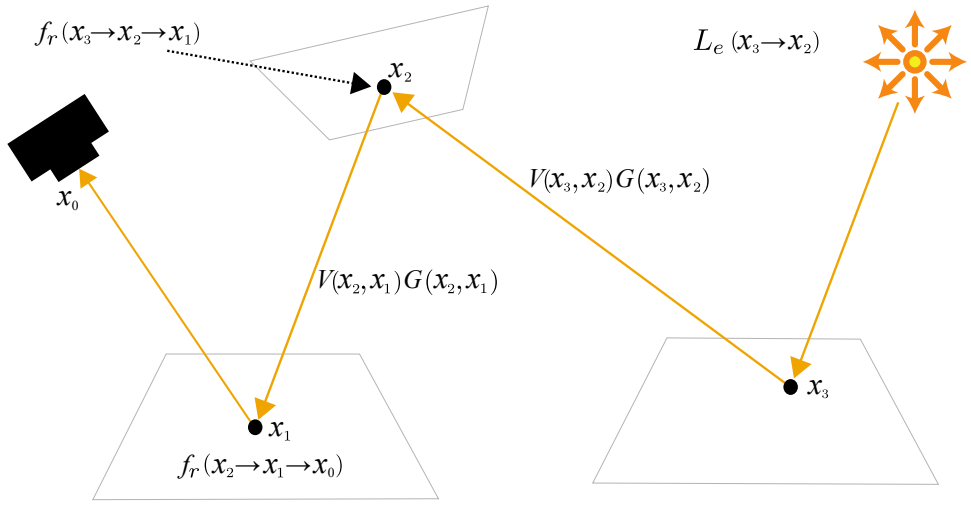


Figure 3.3 : Exemple d'un chemin de l'intégrale pour quatre points.

3.2 Intégration par la méthode de Monte Carlo

Prenons une fonction $f(x)$ que nous voulons intégrer sur $[a,b]$:

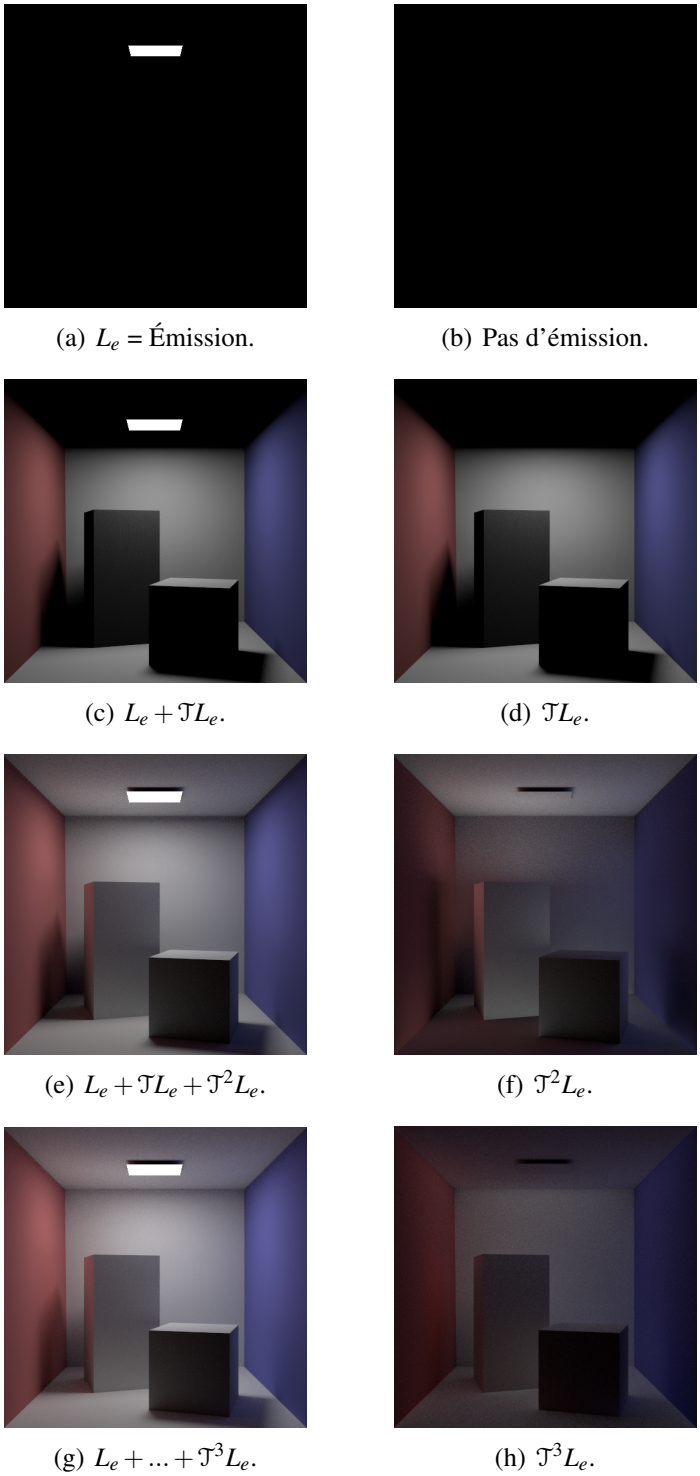
$$F = \int_a^b f(x)\,dx. \tag{3.12}$$

On sait que l'on peut approximer F en moyennant un certain nombre d'évaluations de $f(x)$ que l'on multiplie ensuite par $(b-a)$. Avec $X_i \in [a,b]$ une variable aléatoire uniformément distribuée entre a et b, on a :

$$F_N = (b-a)\frac{1}{N}\sum_{i=1}^{N} f(X_i). \tag{3.13}$$

F_N est l'estimateur de Monte Carlo de l'intégrale F. Plus le nombre d'échantillons N est grand, plus l'estimation est précise :

$$\lim_{N\to\infty} F_N = F. \tag{3.14}$$



(a) L_e = Émission. (b) Pas d'émission.

(c) $L_e + \mathcal{T}L_e$. (d) $\mathcal{T}L_e$.

(e) $L_e + \mathcal{T}L_e + \mathcal{T}^2L_e$. (f) $\mathcal{T}^2L_e$.

(g) $L_e + ... + \mathcal{T}^3L_e$. (h) $\mathcal{T}^3L_e$.

Figure 3.4 : Les quatre premiers termes de l'équation du transport de la lumière décomposée en série de Neumann, avec l'accumulation dans la colonne de gauche des différents termes de la colonne de droite.

Montrons que l'espérance de l'estimateur F_n est égale à l'intégrale de f :

$$\begin{aligned} E[F_N] &= E\left[\frac{(b-a)}{N}\sum_{i=1}^{N} f(X_i)\right] \\ &= \frac{(b-a)}{N}\sum_{i=1}^{N} E\left[f(X_i)\right] \\ &= \frac{(b-a)}{N}\sum_{i=1}^{N}\int_a^b f(x)p(x)\,dx \\ &= \frac{1}{N}\sum_{i=1}^{N}\int_a^b f(x)\,dx \\ &= \int_a^b f(x)\,dx. \end{aligned} \tag{3.15}$$

On peut voir sur la figure 3.5 l'exemple de la fonction $g(x) = \sin(x)$ sur $[0..\pi]$. On cherche ici à trouver F_g avec l'équation 3.13 tel que (on sait que la solution est égale à 2) :

$$F_g = \int_0^{\pi} g(x)\,dx. \tag{3.16}$$

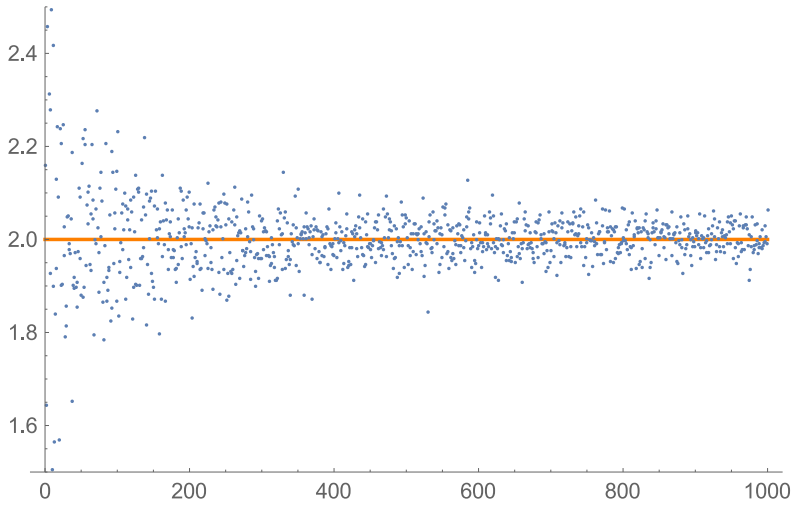


Figure 3.5 : Convergence de l'estimateur de Monte Carlo pour la fonction $\sin(x)$ avec un échantillonnage uniforme. L'axe vertical représente la valeur de l'estimateur et l'axe horizontal, le nombre d'échantillons N. On voit que plus N augmente, plus F_N converge vers la solution connue, ici la valeur 2.0.

En calculant l'écart-type σ de l'estimateur F_N on peut connaître la vitesse de convergence de l'estimateur F_N. Commençons par calculer la variance σ^2 :

$$\sigma^2 = \frac{1}{N}\left(\int_0^\pi g(x)^2 dx - I^2\right). \tag{3.17}$$

L'écart-type σ étant la racine carrée de la variance σ^2 :

$$\sigma = \frac{1}{\sqrt{N}}[Y] \quad \text{avec} \quad [Y] = \left[\sqrt{\int_0^\pi g(x)^2 dx - I^2}\right]. \tag{3.18}$$

Ce qui nous dit par exemple que pour diviser l'erreur par deux, il faut un N quatre fois plus grand $\frac{1}{\sqrt{N_2}} = \frac{1}{2}\frac{1}{\sqrt{N_1}} \Rightarrow 4N_1 = N_2$.

On a vu que la convergence est très lente, mais nous allons voir qu'il existe des techniques d'échantillonnage préférentiel qui permettent de réduire cette variance. La méthode de Monte Carlo est très simple à mettre en place et fonctionne à des dimensions arbitraires ; c'est l'approche que nous utiliserons (ou plutôt sa variante déterministe, le quasi-Monte-Carlo) dans nos algorithmes de rendu.

3.3 Échantillonnage préférentiel

Pour améliorer notre estimateur, nous pouvons utiliser ce que l'on connaît de la fonction que l'on intègre pour diminuer la variance. Le principe de l'échantillonnage préférentiel est d'échantillonner dans les zones les plus importantes. Ce sont évidemment ces valeurs qui vont avoir le plus de poids (PDF) dans l'intégrale. Pour trouver ces zones, on crée une fonction de densité de probabilité p, qui doit ressembler le plus possible à la forme générale de l'intégrale. Le passage d'une distribution uniforme à une fonction $p(x)$ donne :

$$F_N = \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{p(X_i)}, \tag{3.19}$$

avec les deux propriétés importantes des PDFs :

$$p(x) \geq 0 \quad \text{et} \quad \int_{-\infty}^{+\infty} p(x)\,dx = 1. \tag{3.20}$$

Montrons que l'espérance de l'estimateur est égale à l'intégrale de f :

$$\begin{aligned} E[F_N] &= E\left[\frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{p(X_i)}\right] \\ &= \frac{1}{N}\sum_{i=1}^{N}\int_a^b \frac{f(x)}{p(x)}p(x)\,dx \\ &= \frac{1}{N}\sum_{i=1}^{N}\int_a^b f(x)\,dx \\ &= \int_a^b f(x)\,dx. \end{aligned} \tag{3.21}$$

On peut se demander pourquoi ce résultat est utile sachant qu'ici la variance est aussi proportionnelle à $1/N$. En fait, en choisissant une PDF $p(x)$ adéquate, on va pouvoir diminuer la variance et donc N, autrement dit notre estimateur va converger plus rapidement.

Si l'on reprend l'exemple de la fonction $g(x)$, une bonne fonction PDF est $p(x) = \sin(x)/2$, on divise par deux pour normaliser. On a donc :

$$F_g = \frac{1}{N}\sum_{i=1}^{N}\int_a^b \frac{g(x)}{\sin(x)/2}p(x)\,dx = 2. \tag{3.22}$$

On est ici dans le cas spécial où la PDF correspond exactement à f ; on a alors une variance nulle. Généralement on ne connaît pas f, alors on utilise la méthode d'inversion avec la fonction de distribution cumulative de $p(x)$ pour sélectionner les X_i.

Cette technique étant primordiale, nous y ferons souvent référence tout au long de ce mémoire. Nous l'utilisons dès que nous avons une bonne idée des composantes importantes d'une intégrale. Il existe beaucoup d'autres techniques pour réduire la variance comme l'échantillonnage stratifié, le quasi-Monte-Carlo, mais aussi l'échantillonnage préférentiel multiple [33].

3.4 Roulette russe

La *roulette russe* est une technique probabiliste utilisée pour accélérer les calculs en physique des particules [29]. Pour ce qui nous concerne, elle va permettre d'éliminer les parties non importantes de notre domaine d'intégration utilisées par la méthode de Monte Carlo. Regardons l'exemple suivant [16, p.62] avec p, la probabilité que l'on

estime ou pas la prochaine luminance L_n :

$$L_n = \begin{cases} \frac{L}{p} & \text{si } \varepsilon < p, \\ 0 & \text{sinon.} \end{cases} \tag{3.23}$$

On montre que cela fonctionne en calculant l'espérance (L est évaluée par un tracé de rayons) :

$$E\{L\} = (1-p) \times 0 + p \times \frac{E\{L\}}{p} = E\{L\}. \tag{3.24}$$

Ce résultat nous dit que l'espérance en suivant un nombre fini de rebonds est la même qu'avec une infinité de rebonds. Ce résultat est très important, car il va permettre par exemple de décider s'il faut continuer ou arrêter l'évaluation d'un chemin (voir l'équation 3.9) sans pour autant biaiser l'évaluation. On l'utilise aussi pour décider du chemin à suivre dans le cas d'une BRDF ayant une composante diffuse et spéculaire.

CHAPITRE 4

MÉTHODES DE RENDU

Il existe de nombreux algorithmes pour calculer des images de synthèse réalistes. Ces algorithmes ne sont pas tous égaux en termes de temps de calcul et de qualité de résultats pour simuler les différentes interactions de la lumière avec l'environnement. Certains ne peuvent simuler qu'une partie de ces interactions, et d'autres en seront incapables, ou alors de façon inefficace.
Dans ce chapitre, nous allons nous intéresser aux milieux participatifs en regardant certains algorithmes capables de les simuler.

Pour notre étude, ce sont les interactions avec les milieux participatifs qui nous intéressent. Il y a trois comportements possibles : milieu participatif/surface, surface/milieu participatif et milieu participatif/milieu participatif. Nous allons nous limiter pour des raisons de simplicité à la dernière d'entre elles.

Pour illustrer le fonctionnement des algorithmes avec les milieux participatifs, nous utiliserons une scène de test. Cette scène a été choisie pour être géométriquement très simple afin de diminuer les temps de rendu. Elle est composée d'un cube contenant un milieu participatif, d'une lumière ponctuelle rouge en arrière du cube et d'une lumière ponctuelle blanche sur le dessus. Le milieu participatif est homogène avec $\sigma_s = 4$ et $\sigma_a = 1$.

Nous commencerons par présenter un algorithme stochastique non biaisé en une passe, le tracé de chemins (*path tracing*), qui nous servira quand cela est possible de référence (*ground truth*) pour nos résultats. Nous continuerons ensuite par une technique bi-directionnelle en deux passes, le *photon mapping*, qui nous servira de fondation pour la plupart des algorithmes suivants.

4.1 Algorithme du tracé de chemins (*path tracing*)

L'algorithme du tracé de chemins [18] repose sur une approche purement stochastique. Il fonctionne en une passe et calcule une solution numérique en utilisant la méthode d'intégration de *Monte Carlo* (voir la section 3.2). Il assure un résultat non biaisé par nature, et c'est pour cela qu'on l'utilise comme référence (*ground truth*). C'est aussi la plus simple des solutions à implémenter, mais elle souffre d'une importante variance, ce qui a pour résultat une convergence très lente. Il est toutefois possible de réduire cette variance avec des techniques d'échantillonnage préférentiel (voir la section 3.3).

Dérivation : Cet algorithme n'utilise pas l'équation du rendu (voir l'équation 2.13) telle que nous l'avons vue, mais son expression en intégrale de chemin (voir la section 3.1). Pour la résoudre, nous utiliserons la méthode de Monte Carlo (voir la section 3.2).

Algorithme : Son fonctionnement est relativement simple : on va évaluer l'équation du rendu à l'aide d'un échantillonnage en marche aléatoire (*random walk sampling*) en commençant depuis l'oeil.

Algorithme 1 Calcul de la luminance d'un chemin en marche aléatoire pour les surfaces et milieux participatifs.

procedure LUMINANCE(x, $\vec{\omega}$)
  $t \leftarrow$ TRACE$(x, \vec{\omega})$ ▷ Distance à la surface la plus proche.
  $(d, pdf_m, pdf_s) \leftarrow$ ÉCHANTILLONNEDISTANCE
  si $d < t$ **alors** ▷ Diffusion dans un milieu participatif.
    $x \leftarrow x + d\vec{\omega}$ ▷ On avance au nouveau point.
    $(\vec{\omega}_i, pdf_i) \leftarrow$ ÉCHANTILLONNEPHASE$(x, \vec{\omega})$
    retour $\frac{\sigma_s T_r(d) \rho(\vec{\omega}' \to \vec{\omega})}{pdf_m \, pdf_i}$LUMINANCE$(x, \vec{\omega}_i)$ ▷ Récursion.
  sinon ▷ Diffusion sur une surface.
    $x \leftarrow x + t\vec{\omega}$ ▷ On se place sur la surface.
    $(\vec{\omega}_i, pdf_i) \leftarrow$ ÉCHANTILLONNEBRDF$(x, \vec{\omega})$
    retour $L_e + \frac{T_r(t) f_r(x, \vec{\omega}' \leftrightarrow \vec{\omega})}{pdf_s \, pdf_i}$LUMINANCE$(x, \vec{\omega}_i)$ ▷ Récursion.
  fin si
fin procedure

Pour chacun des pixels de notre image, l'algorithme va suivre un chemin aléatoire (algorithme 1). Le chemin commence à la position x de l'œil avec une direction aléatoire $\vec{\omega}$. Ensuite, on échantillonne aléatoirement une distance d pour la prochaine interaction ainsi que sa probabilité.
Il y a maintenant deux cas de figure, soit le nouveau point x est dans un milieu participatif, soit il est sur une surface.

Diffusion volumique : $d < t$
Le point x est dans un milieu participatif car la distance d est inférieure à la distance t de la surface la plus proche.
On commence par déplacer le point x à la nouvelle position $x + d\vec{\omega}$. Ensuite, la fonction **ÉchantillonnePhase** calcule une nouvelle direction incidente $\vec{\omega}_i$. Finalement, on retourne la luminance incidente à la position et direction, x, $\vec{\omega}_i$. On doit aussi multiplier la luminance par la fonction de phase ρ, la transmittance $T_r(d)$ et le coefficient de dispersion réfléchie σ_s au point x.
On peut réduire la variance avec un échantillonnage préférentiel de la distance et de la

fonction de phase. Si le milieu participatif est homogène, alors la probabilité que la lumière avance d'une distance d est égale à la transmittance $T_r(d)$ (voir l'équation 2.30). On utilise la méthode d'inversion pour trouver d, et pdf_m comme densité de probabilité :

$$d = -\frac{\log(1-\xi)}{\sigma_t} \quad \text{avec} \quad pdf_m = \sigma_t \exp^{-\sigma_t d}, \tag{4.1}$$

et $\xi \in [0,1)$ un nombre aléatoire, $\sigma_t = \sigma_s + \sigma_a$ le coefficient d'extinction.
Si l'échantillonnage préférentiel est parfait alors on peut simplifier l'équation :

$$\frac{\sigma_s Tr(d)\rho(\vec{\omega}' \to \vec{\omega})}{pdf_m \; pdf_i} = \frac{\sigma_s}{\sigma_t} = \alpha. \tag{4.2}$$

Dans le cas d'un milieu participatif hétérogène, il faut faire un échantillonnage de la distance en utilisant les techniques du *delta-tracking* ou du *ratio-tracking* [27].

Diffusion surfacique : $d \geq t$
Le point x est sur une surface car la distance d est supérieure ou égale à la distance t de la surface la plus proche.
On commence par déplacer le point x à la nouvelle position $x + d\vec{\omega}$. Ensuite, la fonction **ÉchantillonneBRDF** calcule une nouvelle direction incidente $\vec{\omega}_i$. Finalement, on retourne la luminance émise à laquelle on ajoute la luminance incidente à la position et direction, x, $\vec{\omega}_i$. On doit aussi multiplier la luminance par la fonction de réflectance bidirectionnelle f_r et la transmittance $T_r(d)$.
On peut réduire la variance avec un échantillonnage préférentiel de la distance et de la fonction de réflectance. La densité de probabilité pdf_s de l'échantillonnage de la distance d est calculée par :

$$pdf_s = \int_{s=t}^{\infty} \sigma_t \, e^{-\sigma_t s} ds = e^{-\sigma_t t}. \tag{4.3}$$

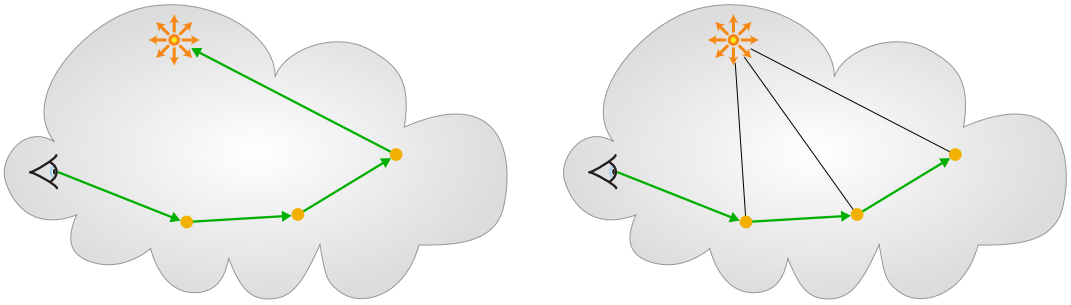
Si l'échantillonnage préférentiel est parfait alors on obtient la même simplification que pour le cas volumique.

L'algorithme a été simplifié, nous ne tenons pas compte de l'équation du rendu volumique complète mais uniquement de la luminance surfacique réduite (voir la figure 2.10) et de la diffusion incidente accumulée (voir la figure 2.9). Et, il s'arrête quand le chemin ne rencontre rien, un émetteur, quand la récursion arrive à une certaine profondeur fixée à l'avance ou par une roulette russe (voir la section 3.4).

L'algorithme est exécuté plusieurs fois pour un même pixel afin de converger vers la solution.

La version de l'algorithme que nous venons de voir est dite "naïve" (voir la figure 4.1(a)).

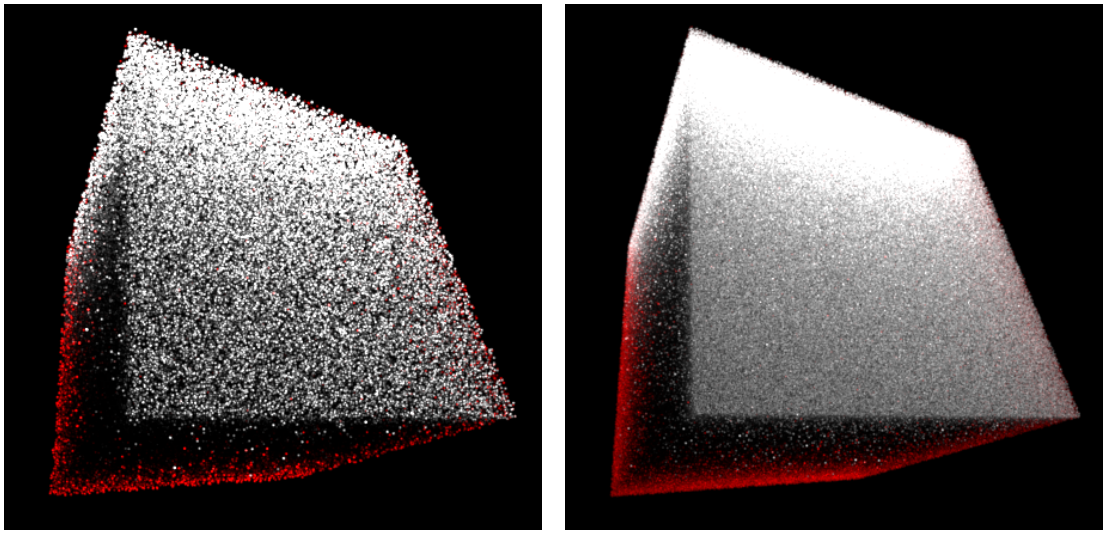
Elle est particulièrement inefficace étant donné que la probabilité qu'un chemin aléatoire rencontre un émetteur est généralement très faible, elle est même nulle dans le cas de lumière ponctuelle. Une très grande majorité de chemins se termineront sans apporter de contribution, la convergence sera très longue et l'image finale presque noire.
Une variante de l'algorithme du tracé de chemins dite "explicite" (voir la figure 4.1(b)), va évaluer l'illumination directe pour chacun des points du chemin. Tous les chemins vont apporter une information, la convergence sera donc plus rapide.



(a) Tracé de chemins naïf. (b) Tracé de chemins explicite.

Figure 4.1 : Chemin oeil-lumière utilisé dans l'algorithme du tracé de chemins naïf/explicite.

Résultat : Les résultats (voir les figures 4.3 et 4.2) montrent que plus nous calculons de chemins par pixel, plus la variance diminue. Il suffit donc de calculer un grand nombre de chemins pour que notre image converge vers la solution exacte. Cet algorithme converge vers la solution exacte, mais le temps de calcul est généralement très long. Comme nous avons pu le voir, de nombreuses techniques comme *l'échantillonnage préférentiel* (voir la section 3.3) permettent d'accélérer la convergence. Il est aussi parfaitement parallélisable et facile à adapter sur GPU. De par sa simplicité d'implémentation, c'est aujourd'hui l'algorithme le plus utilisé dans les applications de rendu professionnelles. Nous l'utiliserons aussi pour valider certains de nos résultats.



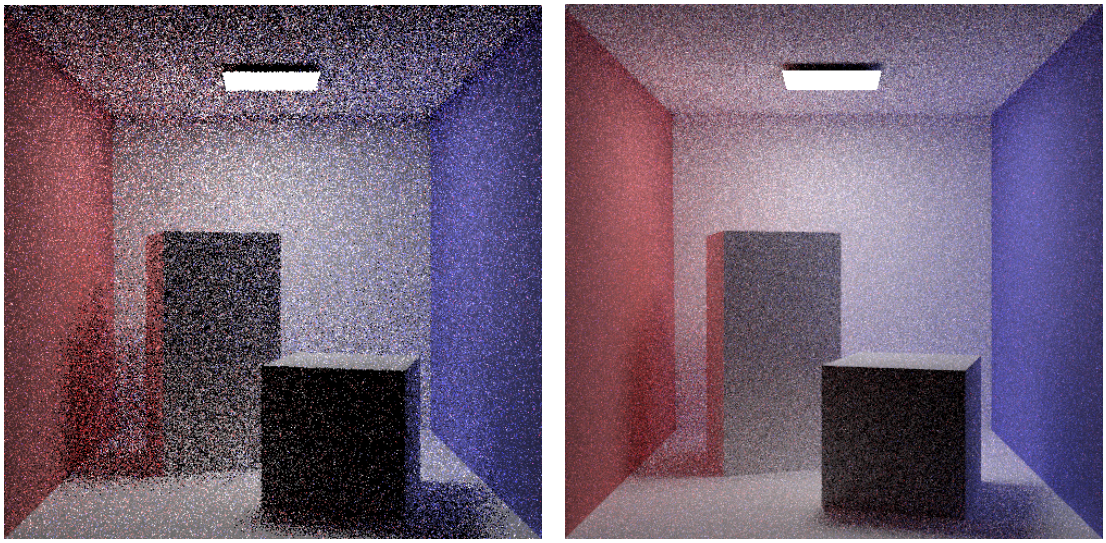
(a) 1 chemin par pixel, 0min1s.

(b) 10 chemins par pixel, 0min1s.

(c) 100 chemins par pixel, 1min7s.

(d) 1000 chemins par pixel, 1min0s.

Figure 4.2 : Rendu avec l'algorithme du tracé de chemins (Mitsuba). La scène est composée d'un cube contenant un milieu participatif homogène, d'une lumière ponctuelle rouge en arrière du cube et d'une lumière ponctuelle blanche au-dessus du cube.



(a) 1 chemin par pixel, 0min1s.

(b) 10 chemins par pixel, 0min2s.

(c) 100 chemins par pixel, 0min11s.

(d) 1000 chemins par pixel, 1min35s.

Figure 4.3 : Rendu surfacique avec l'algorithme du tracé de chemins explicite.

4.2 Intégration par marche (*ray marching*)

L'équation du transfert radiatif (voir l'équation 2.28) ne peut être résolue en général de façon numérique. Prenons l'exemple de l'intervalle fermé $[a,b]$, partitionné par les points $a < x_1 < x_2 < ... < x_{n-1} < b$, avec $\Delta x_1, \Delta x_2, ..., \Delta x_n$ l'intervalle entre les points et x'_k un point arbitraire dans le $k^{\text{ème}}$ intervalle, alors :

$$\sum_{k=1}^{n} f(x'_k)\Delta x_k, \tag{4.4}$$

est la somme de Riemann pour la fonction $f(x)$. Si $\Delta_x \to 0$ alors l'équation 4.4 est l'intégrale de Riemann de $f(x)$. Prenons l'exemple de $f(x) = \log(x+1)$ sur l'intervalle $[0,5]$ avec un pas $\Delta = 1$. L'intégrale de Riemann sur cet exemple donne 5,7833. La solution exacte peut être calculée analytiquement ; elle est égale à 5,7506. On comprend que cette méthode est une approximation dont la précision du résultat dépend de Δ.

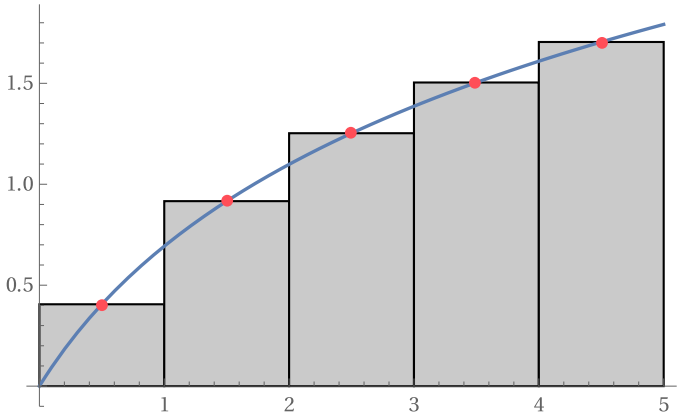


Figure 4.4 : Approximation de l'intégrale de la fonction $f(x) = \log(x+1)$ par la méthode d'intégration de Riemann avec les valeurs (en rouge) prises au milieu de l'intervalle Δ.

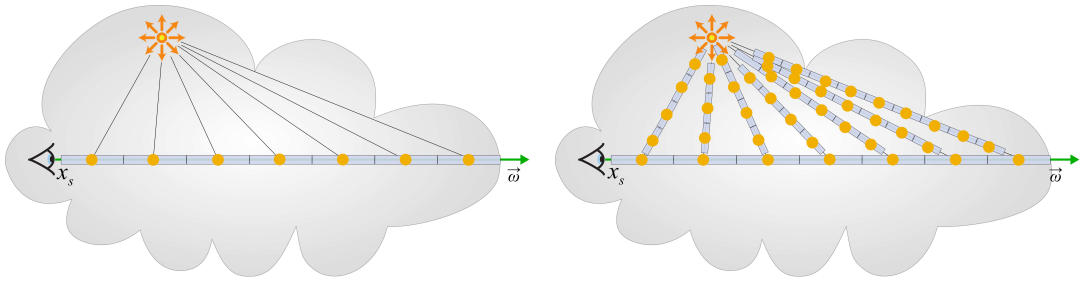
Appliquons maintenant cette méthode à l'équation du rendu volumique (voir l'équation 2.28). Le principe est de diviser le domaine de l'intégrale. C'est le rayon $\vec{\omega}$ que nous allons diviser en sous segments de longueur Δ_x pour lesquels l'éclairage incident, la fonction de phase et la transmittance sont considérés comme constants. Avec un émetteur et un milieu participatif homogène, on obtient l'équation suivante (voir la figure 4.5(a)) :

$$L(x_s, -\vec{\omega}) = \sum_{t=0}^{N} T_r(x_s \leftrightarrow x_t)\sigma_s(x_t)L_i(x_t, \vec{\omega}_{lt})\rho(x_t, \vec{\omega}_{lt}, -\vec{\omega})\Delta_x, \tag{4.5}$$

avec $\vec{\omega}_{lt}$ la direction vers l'émetteur au segment t et x_t un point sur le segment Δ_t. Pour un milieu participatif hétérogène, la luminance incidente L_i ne pourra pas être calculée directement car la transmittance le long du rayon vers l'émetteur n'est pas connue. Il

faut alors procéder à une intégration par marche (voir la figure 4.5(a)).

Pour simuler la diffusion multiple, il faut relancer récursivement une intégration par marche pour chaque point x_k, ce qui fait que cet algorithme a un coût exponentiel et est dans la pratique inutilisable pour simuler la diffusion multiple.



(a) Intégration par marche dans un volume homogène

(b) Intégration par marche dans un volume hétérogène

Figure 4.5 : Évaluation de la luminance $-\vec{\omega}$ dans un milieu participatif par intégration par marche dans le cas de la diffusion simple (*single scattering*).

Choisir une "bonne" valeur pour le pas Δ_t d'intégration n'est pas facile. Une valeur trop petite donnera un temps de calcul très long, mais visuellement un bon résultat, un pas trop grand donnera un effet de seuil (voir la figure 4.6). On peut diminuer cet effet en ajoutant un $\xi \in [0,\Delta]$ au point de départ x_s, ce qui ajoute un léger bruit. Il est possible aussi de choisir un pas adaptatif.

Dans le cas d'un milieu participatif infini ou procédural pour lequel il est possible de définir une boîte englobante, la méthode précédente fonctionne. Si ce n'est pas possible, il faut alors utiliser une approche locale comme le *distance sampling* (voir l'équation 4.3), le *delta-tracking* ou le *ratio-tracking* [27].

Résultats : Nous avons calculé deux images avec notre implémentation de l'intégration par marche. Un rendu d'un milieu homogène et un rendu d'un milieu hétérogène sous forme de grille de densité (voir la figure 4.7).

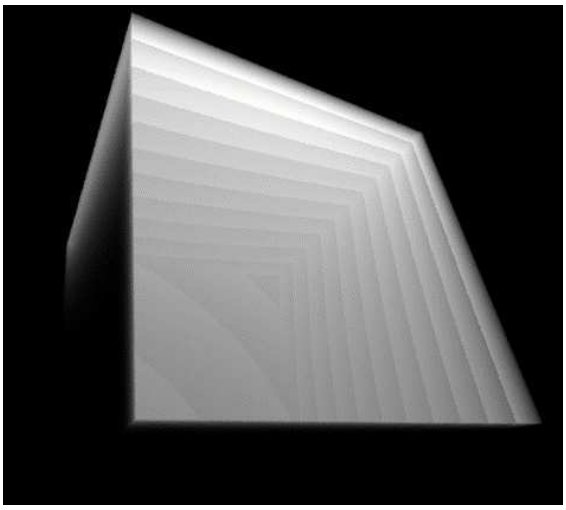
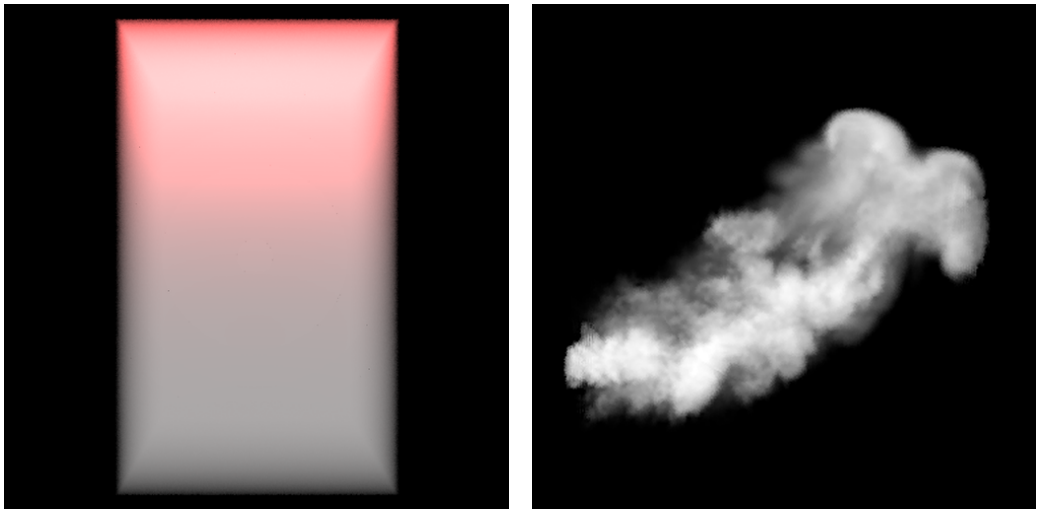


Figure 4.6 : Exemple de rendu avec une intégration par marche. Deux types d'effets de seuil causés par un choix de Δ_t trop grand sont particulièrement visibles.



(a) *Ray marching* avec un volume homogène et une diffusion simple.

(b) *Ray marching* pour un volume hétérogène encodé dans une grille de densités et une diffusion simple.

Figure 4.7 : Exemple de rendu de l'algorithme du *ray marching* avec une diffusion simple. La figure 4.7(a) représente un cube avec un milieu homogène et une lumière rouge ponctuelle située au-dessus. La figure 4.7(b) représente un volume hétérogène émissif.

4.3 Algorithme du placage de photons (*photon mapping*)

L'algorithme du *photon mapping* a été développé par Jensen [16, 17]. C'est une technique biaisée, mais qui converge dans sa variante progressive [8].

Dérivation : Dans le cas surfacique, considérons que nous connaissons l'éclairement énergétique (flux) incident en de nombreux points sur les surfaces. Alors, en utilisant la relation de l'équation 2.11 entre le flux et la luminance ainsi que l'équation 2.7, nous avons :

$$L(x \to \vec{\omega}) = \int_S \rho(x, \vec{\omega}' \leftrightarrow \vec{\omega}) \frac{\mathrm{d}^2\Phi(x, \vec{\omega}')}{dA(x)}\, dA(x). \tag{4.6}$$

En utilisant le flux des plus proches voisins, on obtient (voir la figure 4.8) :

$$L(x \to \vec{\omega}) \approx \sum_{p=1}^{n} \rho(x, \vec{\omega}_p \leftrightarrow \vec{\omega}) \frac{\Delta\Phi_p(x, \vec{\omega}_p)}{\Delta A}. \tag{4.7}$$

On obtient la luminance sortante avec le flux Φ approximé par la densité des n photons aux alentours de x sur une surface que l'on assume plane et d'aire $\Delta A = \pi r^2$:

$$L(x \to \vec{\omega}) \approx \sum_{p=1}^{n} \rho(x, \vec{\omega}_p \leftrightarrow \vec{\omega}) \frac{\Delta\Phi_p(x, \vec{\omega}_p)}{\pi r^2}. \tag{4.8}$$

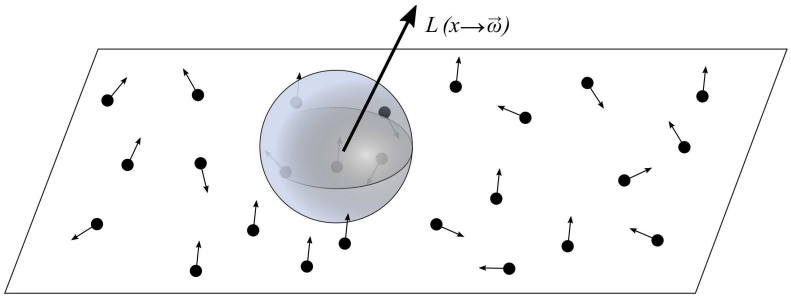


Figure 4.8 : Illustration de la luminance sortante $L(x \to \vec{\omega})$ calculée par l'estimation de densité du flux pour une surface πr^2 représentée par l'intersection du plan et de la sphère. Les points noirs représentent le flux avec la direction incidente.

Dans le cas volumique (voir la figure 2.8), considérons que nous connaissons l'éclairement énergétique (flux) incident en de nombreux points dans le volume. Alors, en utilisant la relation de l'équation 2.29 entre le flux et la luminance ainsi que l'équation 2.25, nous avons :

$$L(x \to \vec{\omega}) = \int_{S^2} \rho(x, \vec{\omega}' \leftrightarrow \vec{\omega}) \frac{\mathrm{d}^2\Phi(x, \vec{\omega}')}{\sigma_s(x)\, dV(x)}\, dV(x). \tag{4.9}$$

En utilisant le flux des plus proches voisins, on obtient (voir la figure 4.9) :

$$L(x \to \vec{\omega}) \approx \sum_{p=1}^{n} \rho(x, \vec{\omega}_p \leftrightarrow \vec{\omega}) \frac{\Delta\Phi_p(x, \vec{\omega}_p)}{\Delta V}. \tag{4.10}$$

On obtient la luminance sortante avec le flux Φ approximé par la densité des n photons aux alentours de x dans un volume sphérique de rayon r :

$$L(x \to \vec{\omega}) \approx \sum_{p=1}^{n} \rho(x, \vec{\omega}_p \leftrightarrow \vec{\omega}) \frac{\Delta\Phi_p(x, \vec{\omega}_p)}{\frac{4}{3}\pi r^3}. \tag{4.11}$$

La densité donnera un résultat flouté si la quantité de photons est faible dans la région, ce qui est acceptable pour l'illumination indirecte. Pour une simulation de caustiques, on préfère avoir des bords beaucoup plus saillants. Dans ce cas il est nécessaire de filtrer les photons en donnant un poids plus important à ceux qui sont les plus proches du point d'évaluation. Il existe différents types de filtres [16 ; 17, p.82] comme le linéaire, le gaussien ou le cône.

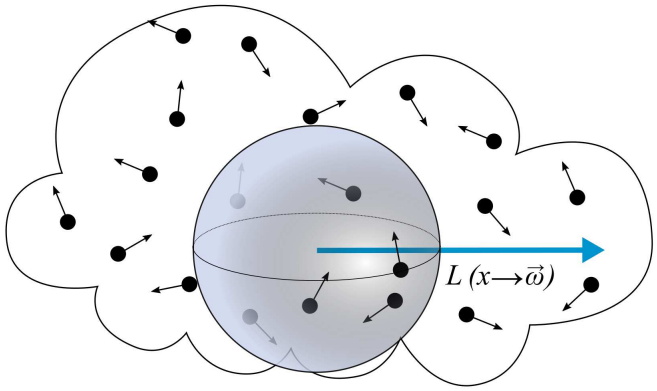


Figure 4.9 : Illustration de la luminance sortante $L(x \to \vec{\omega})$ calculée par estimation de densité volumique du flux pour un volume sphérique. Les points noirs représentent le flux avec la direction incidente.

Algorithme : L'algorithme fonctionne en deux étapes. La 1ère est la création de la *photon map*. C'est une passe de précalcul qui va évaluer et enregistrer dans une structure de données l'éclairement énergétique en de nombreux points. La 2ème étape est la passe de rendu pour laquelle on utilisera le résultat de la 1ère étape pour l'évaluation de la luminance.

1ère étape : Des photons sont émis dans des directions aléatoires depuis les sources lumineuses (voir la figure 4.10). On sélectionne un émetteur ainsi que la probabilité de

ce choix, puis on émet une particule depuis cet émetteur. Cette particule d'énergie Φ qui est une fraction de l'énergie totale émise, va avancer le long d'un chemin aléatoire (voir la figure 4.10). L'algorithme 2 détaille cette procédure. Comme pour l'algorithme 1 du

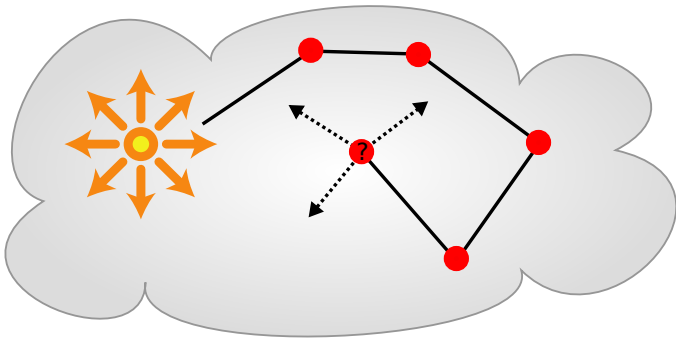


Figure 4.10 : Parcours d'un photon dans un milieu participatif avec diffusion multiple.

Algorithme 2 Émission de particules.

1: **procedure** CRÉATIONPHOTONMAP
2: $n_{photon} \leftarrow 0$
3: **répéter**
4: $(l, pdf_l) \leftarrow EmetteurSelection()$
5: $(X_p, \vec{\omega}, \Phi_p) \leftarrow EmitPhoton(l)$
6: TRACEPHOTON$(x_p, \vec{\omega}_p, \Phi_p / pdf_l)$
7: $n_{photon} \leftarrow n_{photon} + 1$
8: **jusqu'à** PhotonMap est pleine
9: **fin procedure**

tracé de chemins, les particules interagissent avec les surfaces et les milieux participatifs en modifiant l'énergie transportée. À chacune des interactions (points verts, figure 4.10) on enregistre la position, la direction incidente ainsi que le flux dans une structure de données appelée "photon map" (voir la figure 4.11).

```
struct sPhoton {
        vec3 position;
        vec3 directionIncidente;
        vec3 flux; //rgb
};
```

Listing 4.1 : Structure d'un photon.

Cette structure 3D est un arbre-kd balancé [1]. C'est une structure de données efficace pour la recherche du plus proche voisin ($O(\log N)$). Nous aurons besoin de réaliser cette étape un grand nombre fois pour estimer la densité locale. On arrête le parcours

Algorithme 3 Tracé de photons sur des surfaces et dans des milieux participatifs.

procedure TRACEPHOTON($X_p, \vec{\omega}_p, \Phi_p$)
  $profondeur \leftarrow 0$
  tantque $profondeur < max_{profondeur}$ **faire**
    $t \leftarrow$ TRACE$(x, \vec{\omega}_p)$ ▷ Distance à la surface la plus proche.
    $(d, pdf_m, pdf_s) \leftarrow$ ÉCHANTILLONNEPROPAGATION ▷ Échantillonne une distance.
    si $d < t$ **alors** ▷ Diffusion dans un milieu.
      si non DIFFUSEMEDIUM($X_p, \vec{\omega}_p, \Phi_p, d, pdf_m$) **alors retour**
      fin si
      ENREGISTREPHOTON($X_p, \vec{\omega}_p, \Phi_p, profondeur$)
    sinon ▷ Diffusion sur une surface.
      si non DIFFUSESURFACE($X_p, \vec{\omega}_p, \Phi_p, t, pdf_s$) **alors retour**
      fin si
      ENREGISTREPHOTON($X_p, \vec{\omega}_p, \Phi_p, profondeur$)
    fin si
    $profondeur \leftarrow profondeur + 1$
  fin tantque
fin procedure

Algorithme 4 Diffusion sur une surface.

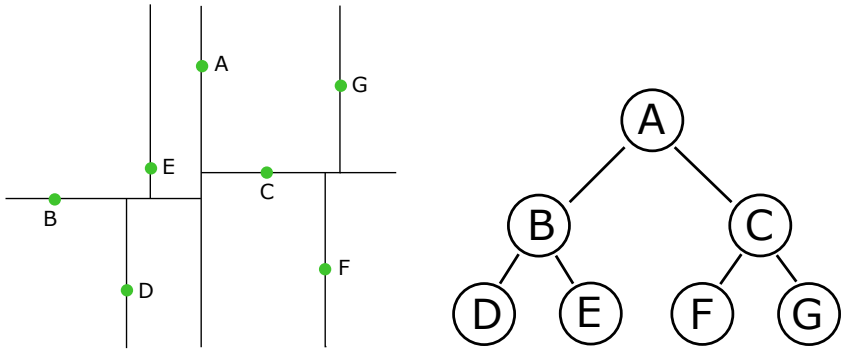
procedure DIFFUSESURFACE($X_p, \vec{\omega}_p, \Phi_p, t, pdf_s$)
  si $\xi < \alpha$ **alors** ▷ Roulette russe pour arrêter le chemin.
    $X_p \leftarrow X_p + t\vec{\omega}_p$
    $(\vec{\omega}_i, pdf_i) \leftarrow$ ÉCHANTILLONNEBRDF$(X_p, \vec{\omega}_p)$)
    $\Phi_p \leftarrow \Phi_p \frac{T_r(t) f_r(x_p, \vec{\omega}_i \leftrightarrow \vec{\omega}_p)}{pdf_s \, pdf_i \, \alpha}$
    $\vec{\omega}_p \leftarrow \vec{\omega}_i$
    retour true ▷ On continue le chemin.
  sinon
    retour false ▷ On arrête le chemin.
  fin si
fin procedure

Algorithme 5 Diffusion dans un milieu participatif.

procedure DIFFUSEMEDIUM($X_p, \vec{\omega}_p, \Phi_p, d, pdf_m$)
  si $\xi < \alpha$ **alors** ▷ Roulette russe pour arrêter le chemin.
    $X_p \leftarrow X_p + d\vec{\omega}_p$
    $(\vec{\omega}_i, pdf_i) \leftarrow$ ÉCHANTILLONNEPHASE$(X_p, \vec{\omega}_p)$
    $\Phi_p \leftarrow \Phi_p \frac{\sigma_s T_r(d) \rho(x_p, \vec{\omega}_i \rightarrow \vec{\omega}_p)}{pdf_m \, pdf_i \, \alpha}$
    $\vec{\omega}_p \leftarrow \vec{\omega}_i$
    retour true ▷ On continue le chemin.
  sinon
    retour false ▷ On arrête le chemin.
  fin si
fin procedure

aléatoire de la particule avec une roulette russe (voir la section 3.4) ou si la profondeur de récursion maximum est atteinte. Cette étape est illustrée par l'algorithme 3.

Comme pour le tracé de chemins, les particules ont deux interactions possibles, soit avec une surface (algorithme 4) ou bien avec un milieu participatif (algorithme 5). Si on utilise l'échantillonnage avec importance, on peut simplifier comme on a pu le voir précédemment (voir la section 4.2).



(a) Liste de points et axes de division de l'espace planaire.

(b) Arbre binaire associé représentant l'arbre-kd.

Figure 4.11 : Subdivision hiérarchique de l'espace 2D d'une liste de points et l'arbre-kd balancé associé.

2ème étape : Pour chaque pixel de l'écran, on envoie un rayon qui part de l'œil et qui se dirige dans la scène. La luminance arrivant vers l'œil à partir d'une surface est calculée avec l'équation (voir l'équation 4.8). Pour un volume, on utilise la méthode d'intégration avec marche (voir la section 4.2) que nous avons vue précédemment avec la luminance incidente L_i évaluée par estimation de densité par l'équation 4.11 à l'aide de la *photon*

map. On obtient finalement la luminance totale arrivant vers l'œil :

$$L(x,-\vec{\omega}) \approx \sum_{t=0}^{s} T_r(x \leftrightarrow x_t)\sigma_s(x_t)L_i(x_t,\vec{\omega})\Delta_t, \tag{4.12}$$

avec L_i calculée avec l'équation d'estimation de densité (voir l'équation 4.11).

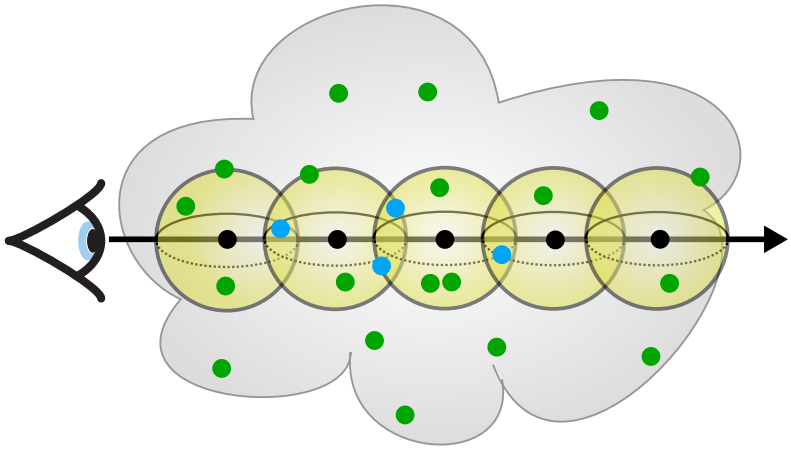


Figure 4.12 : Illustration du *ray marching* avec un pas Δt constant et une estimation de densité avec un noyau sphérique.

Il est assez fréquent d'avoir une mauvaise distribution des photons, la densité locale n'étant alors qu'une mauvaise estimation qui fait apparaître des artefacts (voir la figure 4.13) sur les surfaces. On peut remarquer que parfois on utilise les mêmes particules (points bleus sur la figure 4.12) dans des estimations de densités différentes. Ceci rend notre estimation sous optimale.

Pour adoucir les surfaces diffuses, on peut procéder à une étape de *final gather* (voir la figure 4.16(b)) qui consiste à calculer un niveau de diffusion supplémentaire. Dans la pratique, on procède comme pour le tracé de chemins en lançant une petite quantité de rayons avec une profondeur généralement égale à un ou deux rebonds. Mais cette étape augmente le temps de calcul de façon importante, mais on peut alors utiliser un système de caches comme l'*irradiance caching* introduite par Ward [36]. Cette technique exploite le fait que l'éclairement énergétique le long d'une surface diffuse varie peu. Il suffit alors de déterminer des points d'intérêts, et d'interpoler pour approximer l'éclairement énergétique sur la surface. Dans la pratique, l'utilisation d'un cache n'est pas simple et de nouvelles questions se posent sur comment interpoler les valeurs ou sur comment trouver les points d'intérêts.

Pour améliorer les résultats, on utilise en général l'algorithme du *photon mapping* uniquement pour l'illumination indirecte. Dans ce cas, on n'enregistre pas la 1ère interaction

dans la *photon map*, et l'illumination directe est explicitement évaluée pour un medium homogène (voir la figure 4.5(a)) ou par une intégration par marche (voir la section 4.2) dans le cas hétérogène (voir la figure 4.5(b)).

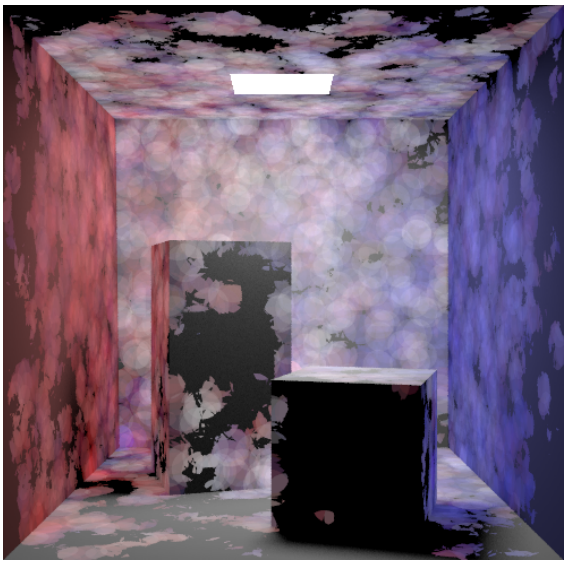
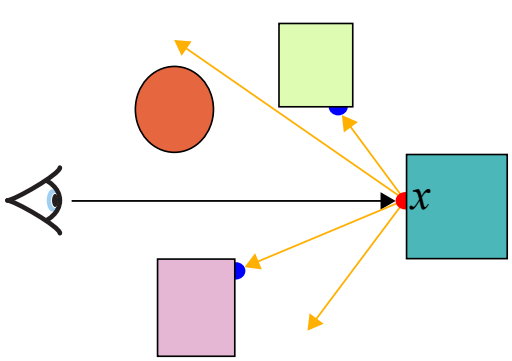
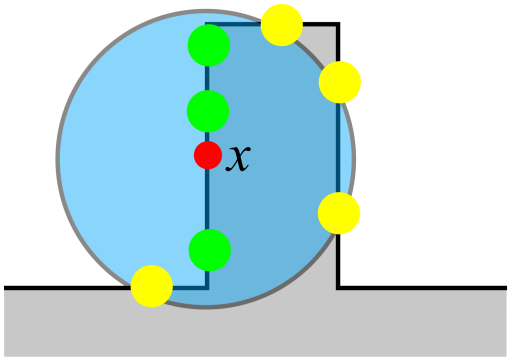


Figure 4.13 : Exemple de rendu avec l'algorithme du *photon mapping*. L'illumination directe est calculée explicitement. L'illumination indirecte est calculée avec une *photon map* contenant 12500 photons. On voit que la quantité de photons est insuffisante pour obtenir une estimation de densité visuellement correcte.

Conclusion : Comme on a pu le voir, cet algorithme utilise une quantité réduite de paramètres qui doivent être choisis par l'utilisateur comme la quantité de photons de la *photon map* ou pour l'estimation de densité, et le rayon des photons. Ces paramètres ont une très grande influence sur le temps de calcul ainsi que sur la qualité des résultats. Mais, il n'est pas toujours facile de les choisir adéquatement ce qui peut rendre cet algorithme plus difficile à utiliser dans la pratique que le tracé de chemins (voir la section 4.1). On est aussi souvent limité par la mémoire disponible pour enregistrer la *photon map*.
Cet algorithme est biaisé, car le noyau utilisé pour l'estimation de la densité fait une interpolation locale qui floute le résultat. De même, si l'on procède à la moyenne de plusieurs images, nous n'aurons pas plus de détails et c'est encore plus vrai quand le nombre de photons est insuffisant dans certaines régions. En théorie, il est cohérent à la limite, c'est-à-dire qu'avec une quantité infinie de photons on devrait arriver au bon résultat. Mais ceci suppose d'avoir une très bonne répartition des photons et beaucoup de mémoire disponible. Dans certains cas, la géométrie cause des erreurs, c'est assez visible dans les coins de murs (voir la figure 4.16(b)). Il faut aussi tenir compte des problèmes d'occultation, ce qui oblige à filtrer les photons que l'on trouve aux alentours de



(a) Évaluation de l'illumination incidente au point x pour le *final gather*. On relance plusieurs rayons pour ajouter un niveau de diffusion. Une estimation de densité est réalisée aux points bleus.

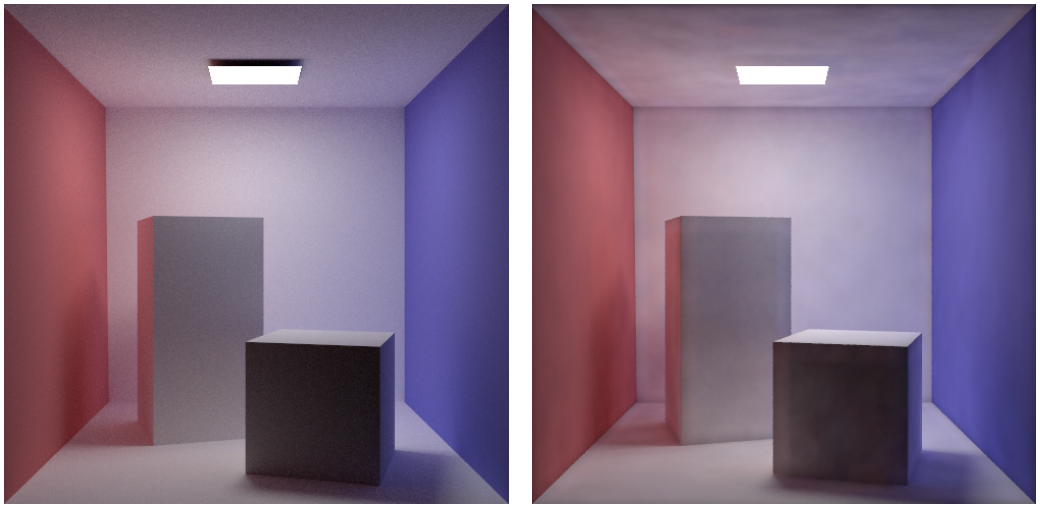


(b) Estimation de densité au point x. Par leurs orientations ou leurs positions, certains photons peuvent être problématiques (photons jaunes) pour le calcul de l'estimation de densité.

notre point (voir la figure 4.14(b)). Il est possible de corriger jusqu'à un certain point ces cas pathologiques, mais cela a aussi un coût. Une passe supplémentairement de lancé de rayons est nécessaire si l'on souhaite ajouter les effets spéculaires qui ne peuvent pas être simulés avec cette technique.

Parmi les avantages, on notera que le *photon mapping* peut calculer l'illumination globale avec des surfaces et des milieux participatifs, les caustiques sont aussi très faciles à reproduire contrairement aux algorithmes basés sur Monte Carlo. Il est aussi relativement simple de reproduire des effets de transluminescence (*subsurface scattering*) et de caustiques. La *photon map* ne change pas si la géométrie et les émetteurs sont statiques, ce qui permet de recalculer une image avec différents points de vue sans avoir à refaire la 1ère étape.

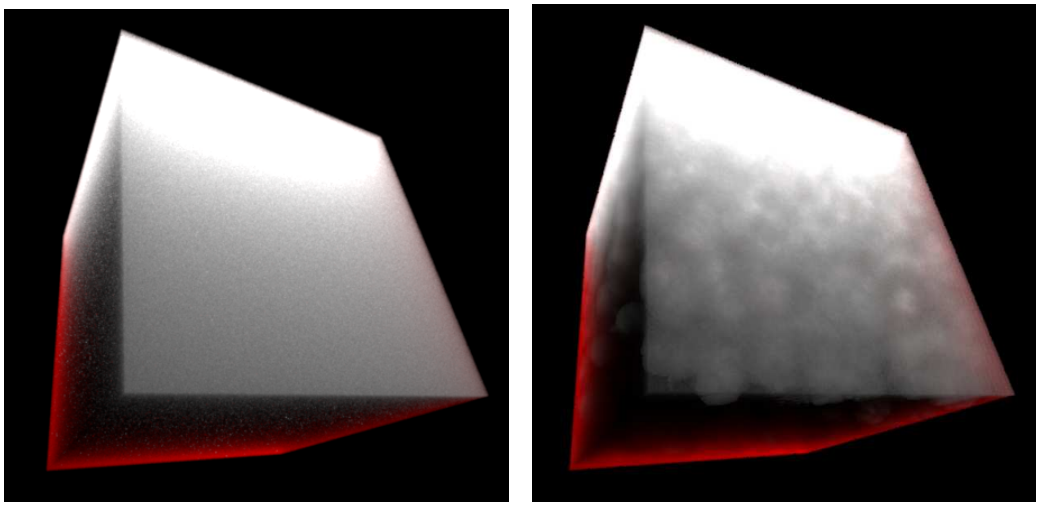
Résultats : Nous avons calculé plusieurs images avec nos implémentations et avec Mistuba [13] pour comparaison. Nous avons comparé le rendu surfacique du *path tracing* avec le *photon mapping* sans *final gather* (voir la figure 4.14). Nous avons aussi calculé un rendu surfacique avec le *photon mapping* en utilisant la technique du *final gather* (voir la figure 4.16), ainsi qu'un rendu volumique homogène avec diffusion multiple avec le *path tracing* et le *photon mapping* (voir la figure 4.15).



(c) Rendu avec tracé de chemins, 0min49s.

(d) Rendu avec *photon mapping* sans *final gather*, 10^5 photons, 0min28s.

Figure 4.14 : Comparaison du rendu surfacique avec *path tracing* et *photon mapping* avec illumination directe et indirecte. L'image de droite est plus rapide à calculer, mais il y a de nombreuses erreurs dues à l'estimation de densité. L'erreur est visible sous forme de flou 2D et est due au fait que l'on utilise les photons présents dans une sphère de proximité.

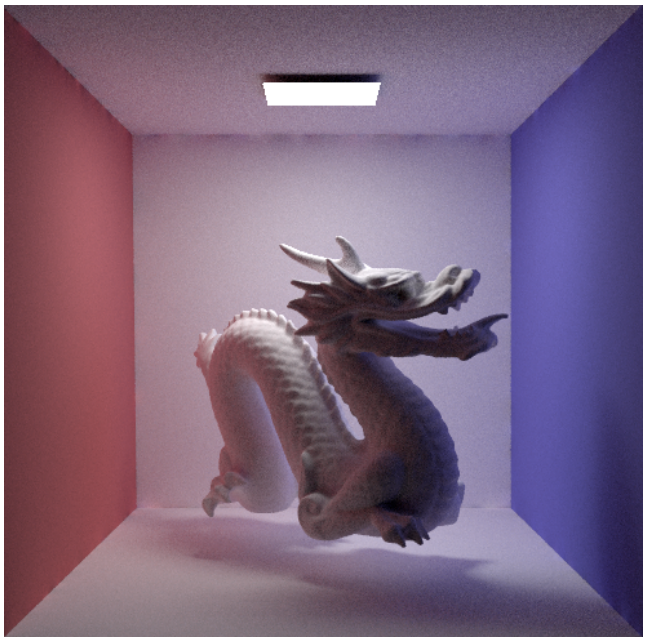


(a) Rendu avec *path tracing*, 0min31s.

(b) Rendu avec *photon mapping*, 10^5 photons, 0min32s.

Figure 4.15 : Rendu d'un volume homogène avec *path tracing* explicite et *photon mapping*. Le bruit visible dans l'image de gauche a été remplacé par du flou dans l'image de droite.

(a) Rendu avec *photon mapping* sans *final gather*, 10^5 photons, 0min41s.



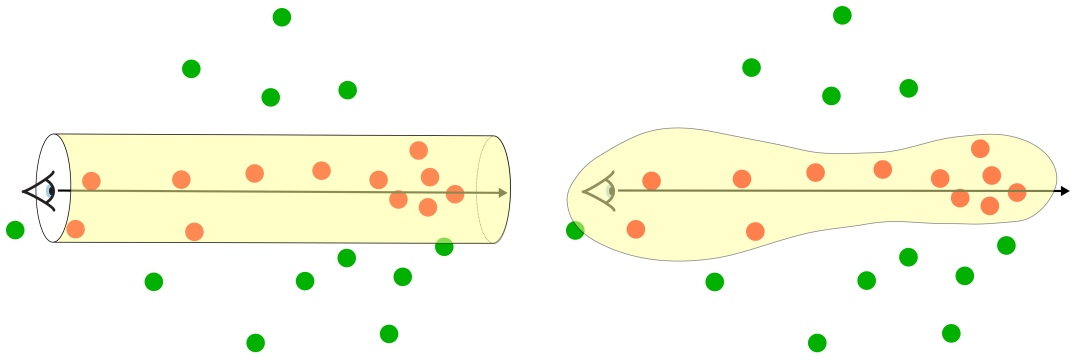
(b) Photon mapping avec 10^5 photons et final gather avec 1 niveau de diffusion et 8 rayons, 2min6s.

Figure 4.16 : Rendu surfacique avec *photon mapping*, *final gather*, illumination directe et indirecte. L'utilisation du *final gather* donne un résultat beaucoup plus diffus et élimine pratiquement tous les artefacts sur les surfaces, mais le temps de calcul est beaucoup plus grand.

4.4 Estimation de luminance par faisceaux (*beam radiance estimate*)

L'estimation de la luminance par faisceaux (*beam radiance estimate, BRE*) [15] est une technique récente pour le rendu de milieu participatif. Elle améliore les résultats et les performances de l'algorithme du *photon mapping* vu précédemment (voir la section 4.3).

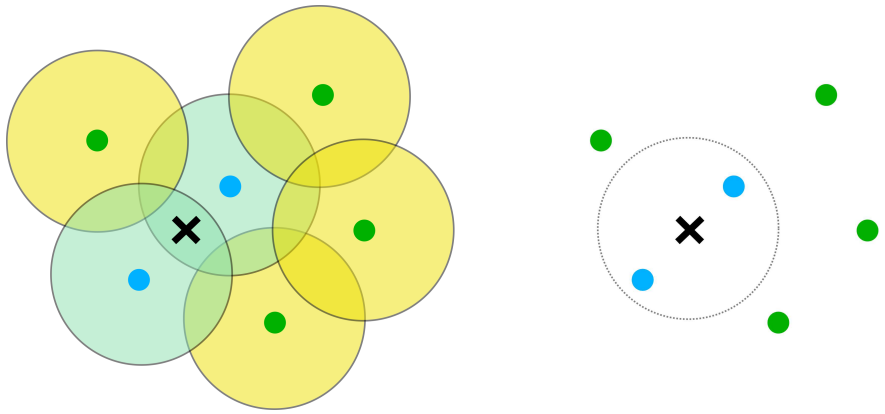
Dérivation : L'algorithme du *photon mapping* utilise une intégration avec marche et comme nous l'avons vu, cela pose plusieurs problèmes. Le BRE va remplacer cette intégration itérative par une seule évaluation. Comme on peut le voir (voir la figure 4.17), le principe est d'utiliser un faisceau, ce qui permet de ne plus avoir d'intégration par marche. Le rayon le long du faisceau peut être fixe ou variable, les meilleurs résultats étant obtenus par rayon variable.



(a) Faisceau avec rayon fixe. (b) Faisceau avec rayon variable.

Figure 4.17 : Illustration de l'estimation par faisceau avec rayon fixe et variable. On voit que pour la même quantité de photons, la densité n'est pas la même car le volume du faisceau n'est pas le même dans les deux cas.

Géométriquement, l'utilisation de faisceaux de taille fixe est très simple ; il suffit de trouver les photons qui sont dans le cylindre. Dans le cas d'un rayon variable, c'est différent, car il n'est pas facile de trouver les particules qui sont à l'intérieur, et ce de façon efficace. L'idée est d'utiliser l'interprétation primale vs duale de l'estimation de densité, qui permet de changer la façon dont on pose le problème. Cette réciprocité est très facile à comprendre géométriquement (voir la figure 4.18). L'algorithme considère les photons non pas comme des points, mais comme des disques. Dans le cas d'un faisceau de taille fixe, on choisit un seul rayon arbitraire pour tous les photons. Pour la version adaptative, on trouve le rayon du faisceau par la méthode du noyau variable (*variable kernel method*) [3], on calcule l'estimation de densité locale puis on utilise la distance du $n^{ème}$ photon comme rayon.



(a) Le faisceau sous forme de rayon intersect les deux disques verts au point x.

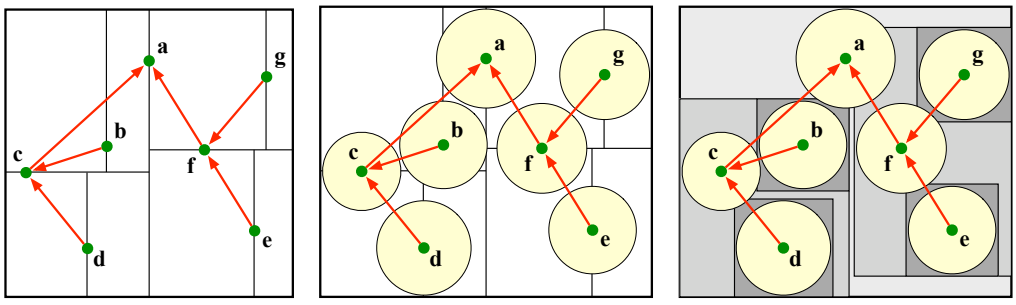
(b) Le faisceau sous forme de cylindre intercepte les deux particules bleues.

Figure 4.18 : Illustration primale vs duale de l'estimation de densité. On voit que l'intersection d'un cylindre avec des points (à droite) peut être vue comme l'intersection d'un rayon et de disques (à gauche). Si les disques sont de taille fixe alors on a l'équivalent d'un faisceau de rayon fixe, sinon de rayon variable.

La méthode propose de calculer le rayon du photon i de la manière suivante :

$$r_i = d_{i,m} \sqrt[3]{\frac{n}{m}}, \tag{4.13}$$

avec $m = \sqrt{n}$, et $d_{i,m}$ égale à la distance aux m n$^{\text{ème}}$ photons. Pour trouver les photons contenus dans le faisceau, il faut trouver les sphères qui intersectent le rayon $\vec{\omega}$. Il est proposé d'utiliser un volume englobant hiérarchique (BVH) qui est calculé à partir de l'arbre-kd (voir la figure 4.19). Pour l'étape de rendu, on va estimer la luminance inci-



(a) Arbre-kd. (b) Calcul des rayon par photon. (c) Construction de la BVH.

Figure 4.19 : Les trois étapes de la construction de la BVH des photons pour le *beam radiance estimate* (Jaroz et al. [15, p.4]).

dente accumulée le long du rayon de la caméra en collectant tous les photons dont la sphère englobante intersecte le rayon. La contribution de chaque photon est accumulée avec l'équation 4.14 :

$$L(x,-\vec{\omega}) = \frac{1}{N}\sum_{i=1}^{N} K_i(x,\vec{\omega},x_i,r_i)T_r(x \leftrightarrow x'_i)\sigma_s(x')\rho(x_i,\vec{\omega},\vec{\omega}_i)\alpha_i, \tag{4.14}$$

avec $x' = x + t_i \times \vec{\omega}$ la projection de la position du photon i sur le rayon de direction $\vec{\omega}$ et $t_i = (x_i - x) \times \vec{\omega}$ (voir la figure 4.20). Le noyau K_i est différent par photon ; on utilise celui à double poids en deux dimensions de Silverman [28] :

$$K_i(x,\vec{\omega},x_i,r_i) = \begin{cases} {r_i}^{-2}K_2(\frac{d_i}{r_i}) & \text{si } d_i \in [0,r_i], \\ 0 & \text{sinon} \end{cases} \tag{4.15}$$

avec $K_2(x) = 3\pi^{-1}(1-x^2)^2$.

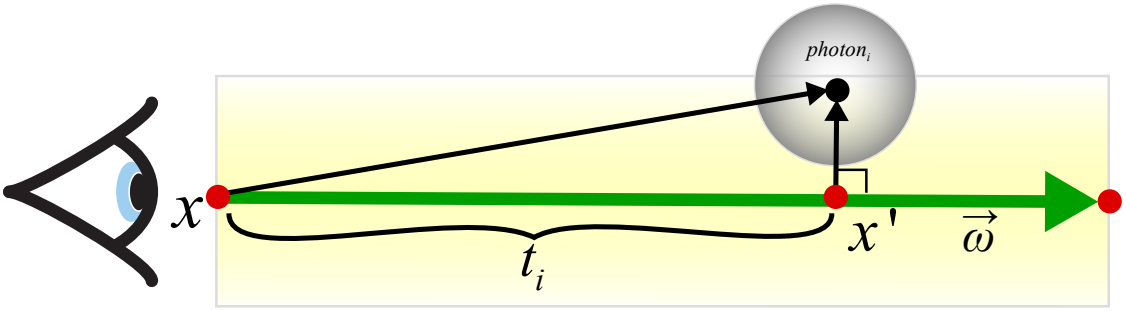


Figure 4.20 : Projection du photon i sur le rayon $\vec{\omega}$ en x'.

Algorithme : Pour implémenter l'algorithme il faut suivre les étapes suivantes :

1ère étape :
Cette étape est identique à la 1ère étape utilisée par le *photon mapping* (voir la section 4.3), (voir la figure 4.19(a)).

2ème étape :
Dans cette étape, on calcule un rayon pour chacun des photons enregistrés dans la *photon map* à l'aide de l'arbre-kd. On utilise simplement la distance au photon le plus proche comme rayon, ce qui est relativement rapide à trouver en utilisant la *photon map*. Les particules seront maintenant vues comme des sphères (voir la figure 4.19(b)).

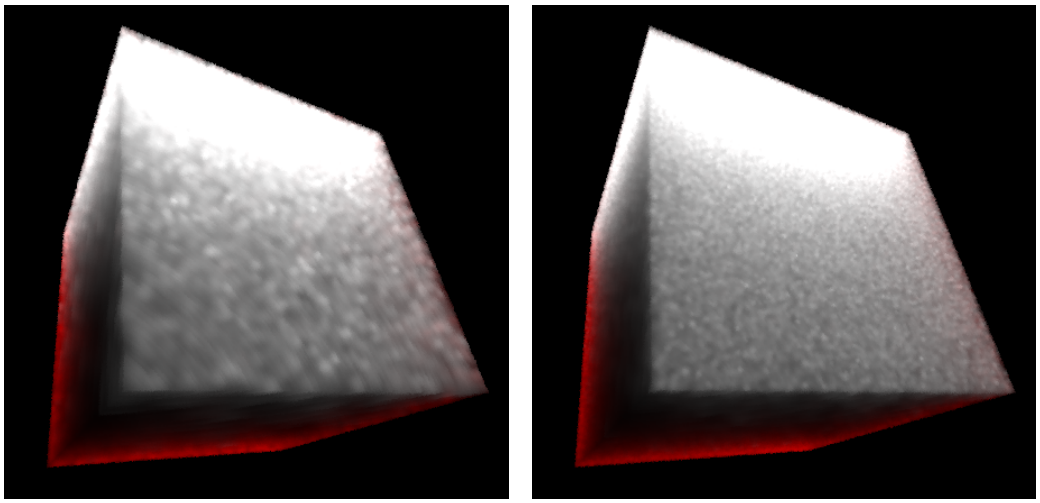
3ème étape :
On construit une hiérarchie de volumes englobants qui va contenir tous les photons.

Algorithmiquement parlant, on utilise l'arbre-kd de la *photon map* pour construire récursivement la BVH. La construction de la BVH bénéficie de la représentation spatiale hiérarchique déjà calculée par l'arbre-kd (voir la figure 4.19(c)).

4ème étape :
On procède au rendu comme pour l'algorithme du *photon mapping* mais cette fois-ci on utilise l'équation d'estimation de luminance (voir l'équation 4.14) une seule fois. Pour les surfaces il faut utiliser une autre technique, par exemple le *photon mapping* ou des *virtual point lights*.

Contrairement au *photon mapping*, nous n'avons plus de paramètre de pas d'intégration et de rayon pour les particules, ce qui rend cette technique très simple d'utilisation. Elle est aussi beaucoup plus rapide, ce qui permet d'augmenter la quantité de photons et d'avoir une meilleure convergence. Par contre l'algorithme utilise un peu plus de mémoire pour la création d'une structure hiérarchique de volume englobant. Pour les surfaces, il faut utiliser en complément une autre technique comme le *photon mapping* ou les *virtual point lights*.

Résultats : La figure 4.21 compare deux résultats avec une quantité de photons différente. Comme pour le *photon mapping*, la quantité de photons influe sur le bruit.

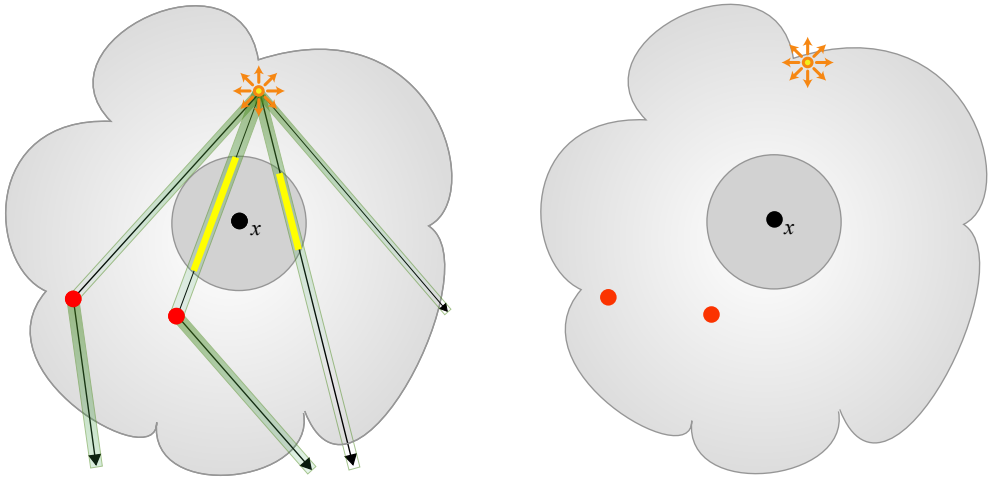


(a) 10^5 photons. (b) 10^7 photons.

Figure 4.21 : Rendu d'un milieu participatif homogène avec l'algorithme du *beam radiance estimate* avec illumination directe et indirecte.

4.5 Faisceaux de photons (*photon beams*)

Nous avons vu que l'algorithme du *photon mapping* n'utilise que les points de diffusion de la marche aléatoire des photons pour l'estimation de la densité. L'algorithme des faisceaux de photons va tenir compte de tous les segments du chemin (voir la figure 4.22). Ceci permet d'avoir une plus grande quantité d'information pour l'estimation de la densité locale. Cette technique est utile pour le rendu des milieux participatifs, et pour les surfaces on peut par exemple utiliser le *photon mapping* (voir la section 4.3) ou des *virtual point lights* (voir la section 4.6).

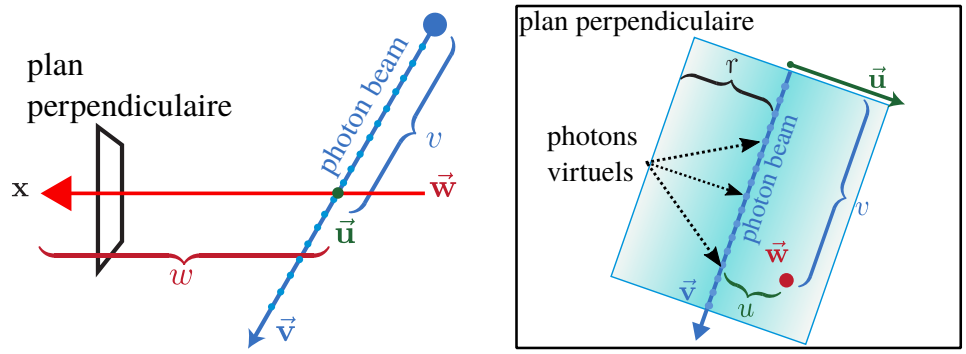


(a) Représentation continue de photons par faisceaux. La partie jaune donne de l'information pour le calcul de densité aux alentours du point x.

(b) Représentation ponctuelle de photons (points rouges). Il n'y a pas d'information pour le calcul de densité aux alentours du point x.

Figure 4.22 : Estimation de la densité au point x avec une représentation ponctuelle et une par des faisceaux des photons dans un milieu participatif. Dans l'image de droite, il n'y a pas d'information disponible pour l'évaluation de la densité aux alentours du point x. A contrario dans l'image de gauche, on a les faisceaux jaunes.

Dérivation : La dérivation est basée sur le principe du *photon marching* c'est une suite de photons à intervalles réguliers avec un flux réduit par la transmittance le long du segment. A la limite, avec des photons espacés d'une distance infinitésimale, on obtient un faisceau continu. On peut représenter un faisceau en utilisant la structure des photons (voir la structure 4.1) en lui ajoutant une information de longueur d. Avec un milieu participatif homogène nous savons calculer la transmittance de façon analytique (voir l'équation 2.30) en tout point. Pour un milieu participatif hétérogène, il faut créer une fonction constante par morceaux (*piecewise function*) pour enregistrer l'information.



(a) Vue de côté de la direction de l'estimation $\vec{\omega}$.

(b) Vue dans le plan perpendiculaire à la direction de l'estimation -$\vec{\omega}$.

Figure 4.23 : On peut voir ici la géométrie des termes utilisés dans l'estimation faisceau/faisceau-1D (Jaroz et al. [15, p.3]).

La dérivation proposée par Jaroz et al. [14] est la suivante. On se place dans le système de coordonnées $(\vec{\mathbf{u}}, \vec{\mathbf{v}}, \vec{\mathbf{w}})$ avec $\vec{\mathbf{w}}$ comme direction d'évaluation et $\vec{\mathbf{v}}$ comme direction du faisceau. $\vec{\mathbf{u}}$ est le vecteur perpendiculaire au plan $\vec{\mathbf{v}} \times \vec{\mathbf{w}}$. Le faisceau est vu comme une suite infinie continue de photons sur sa longueur avec une diffusion 1D le long de l'axe $\vec{\mathbf{u}}$. L'estimation de l'illumination incidente d'un faisceau le long d'une direction $\vec{\mathbf{w}}$ est :

$$L_m(x, \vec{w}, r) \approx k_r(u)\sigma_s(x_w)T_r(w)T_r(v)\frac{f(\vec{\mathbf{w}}, \vec{\mathbf{v}})}{\sin(\vec{\mathbf{w}}, \vec{\mathbf{v}})}, \tag{4.16}$$

avec un noyau de diffusion 1D du photon le long de $\vec{\mathbf{u}}$: $k_r = (2r)^{-1}$. (u, v, w) sont ici les distances signées le long des trois axes jusqu'au photon virtuel (voir la figure 4.23(b), les points bleus) le plus près de $\vec{\mathbf{w}}$.

Algorithme : Pour implémenter l'algorithme il faut suivre les étapes suivantes :

1ère étape :
Cette étape ressemble à la 1ère étape du *photon mapping* (voir la section 4.3). Des photons sont émis depuis les émetteurs puis on enregistre les interactions dans une structure de données de type arbre-kd/BVH. Mais cette fois comme nous l'avons vu, la structure contient les informations des faisceaux.

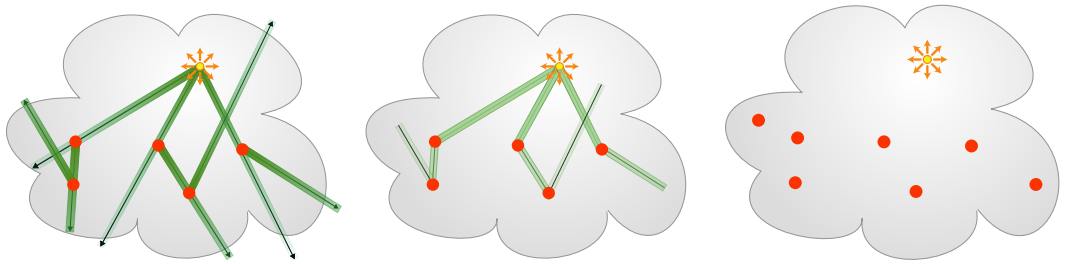
2ème étape :
Cette étape est le rendu. On cherche les faisceaux qui intersectent le rayon de la caméra $\vec{w}$ à l'aide de la structure de données. Puis pour chaque faisceau on évalue l'équation 4.16. Cette technique est cohérente comme le *photon mapping*, et elle est non biaisée si l'on utilise une infinité de faisceaux de rayons infiniment petits. Comme nous sommes limi-

tés par la mémoire disponible sur les machines, il est préférable d'utiliser une version progressive de cet algorithme qui fonctionne par passes successives moyennées et par réduction du rayon des faisceaux.

Faisceaux longs et courts : Krivanek et al. [23] ont proposé deux types de faisceaux : des courts et des longs. Un faisceau long (voir la figure 4.24(b)) se prolonge jusqu'à la prochaine intersection. Si le milieu participatif n'a pas de limite alors la distance est infinie ; dans ce cas on peut utiliser un volume englobant le milieu participatif. Pour les faisceaux courts (voir la figure 4.24(b)), c'est le prochain point de diffusion qui sera utilisé comme point de fin du segment. Leur transmittance est constante, et égale à celle du faisceau parent au point de diffusion. Il n'est donc plus nécessaire pendant le rendu de tenir compte de la transmittance $T_r(v)$ le long du faisceau.

Pendant l'étape de rendu il faut chercher les faisceaux qui intersectent le rayon de la caméra, et pour avoir de bonnes performances on utilise un arbre-kd/BVH pour les stocker. Pour les faisceaux longs, l'arbre-kd/BVH est inefficace car ils sont tous très longs, par contre on peut procéder à une subdivision avec une grille (*octree*) qui englobe l'objet.

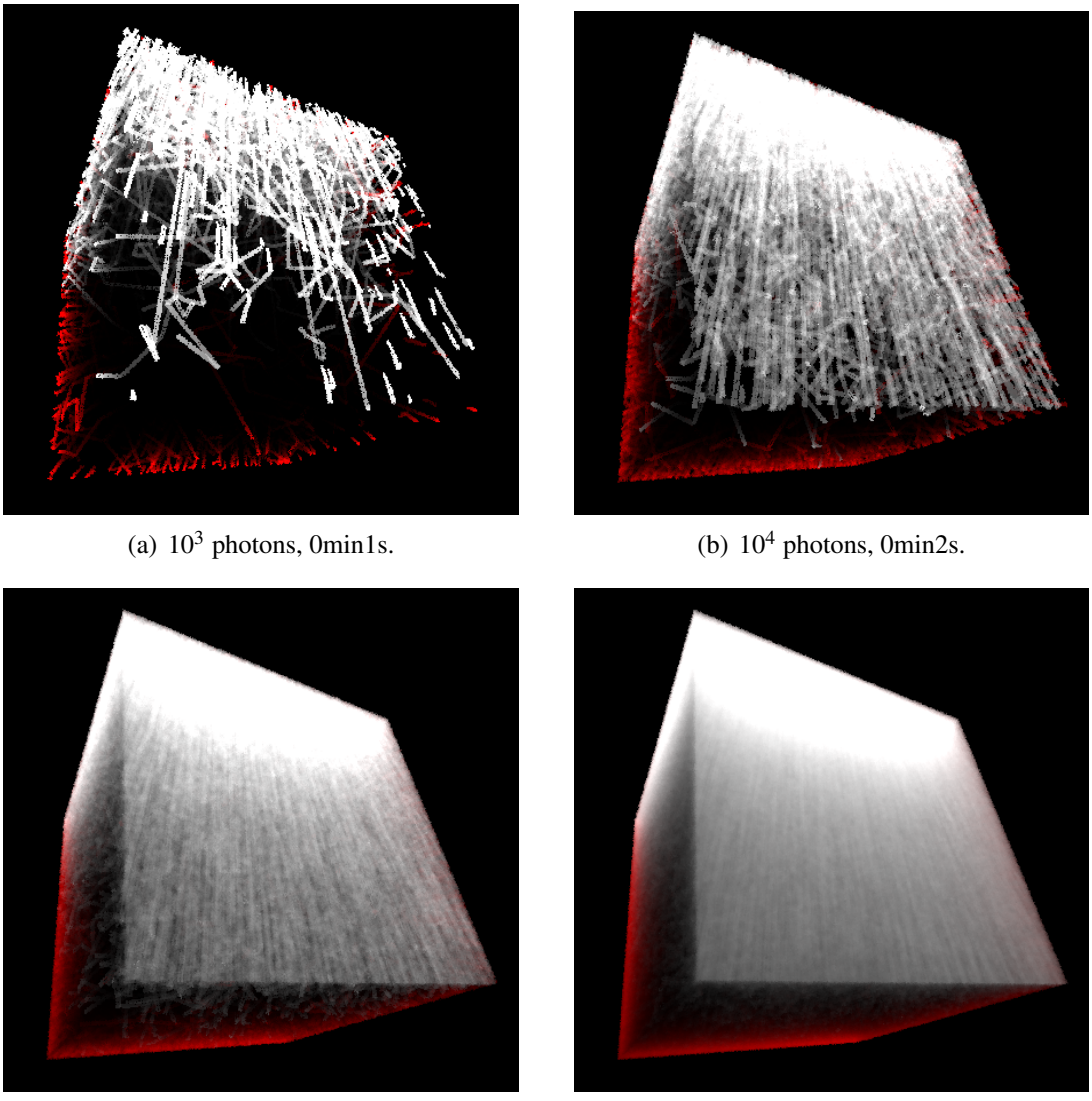
Dans la pratique, il est plus simple d'utiliser des faisceaux courts. Il n'est pas nécessaire de les subdiviser et l'évaluation lors du rendu est simplifiée.



(a) Faisceaux longs. (b) Faisceaux courts. (c) Photons ponctuels.

Figure 4.24 : Faisceaux longs, courts et photons ponctuels.

Résultats : Nous avons implémenté une version progressive et non progressive avec des faisceaux courts et longs, ainsi que la possibilité de subdiviser les faisceaux. La version la plus rapide à calculer est celle avec les faisceaux courts (voir les figures 4.25, 4.26).



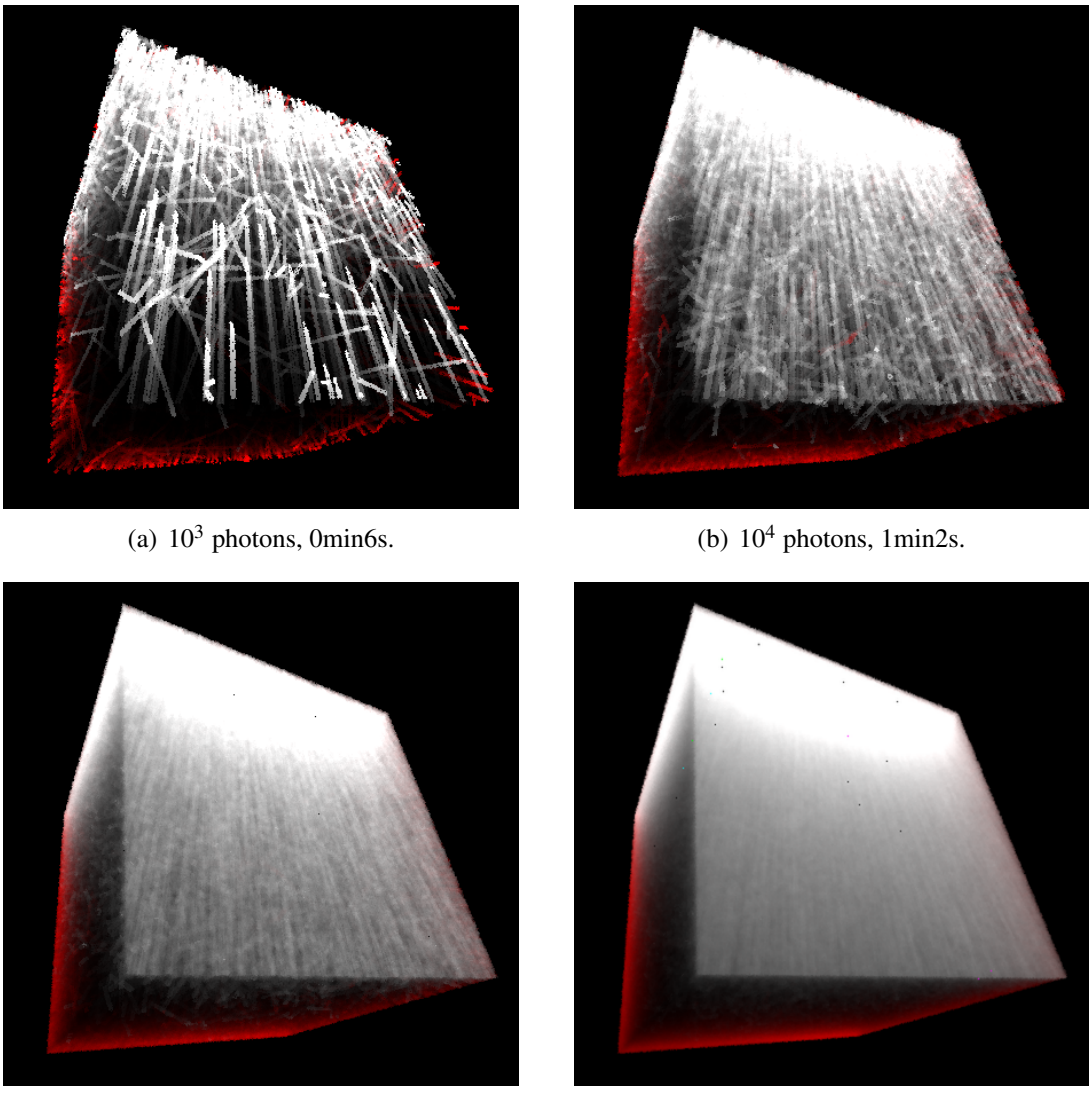
(a) 10^3 photons, 0min1s.

(b) 10^4 photons, 0min2s.

(c) 10^5 photons, 0min30s.

(d) 10^6 photons, 7min11s.

Figure 4.25 : Rendu non progressif d'un milieu homogène par faisceaux courts.



(a) 10^3 photons, 0min6s.

(b) 10^4 photons, 1min2s.

(c) 10^5 photons, 13min39s.

(d) 10^6 photons, 3h20min2s.

Figure 4.26 : Rendu non progressif d'un milieu homogène par faisceaux longs.

4.6 Lumière ponctuelle virtuelle (*virtual point light*)

L'algorithme des lumières ponctuelles virtuelles ou *virtual point light (VPL)*, connu aussi sous le nom de radiosité instantanée (*instant radiosity*) [19] permet d'approximer facilement l'illumination indirecte sans bruit, contrairement au *path tracing*. Le principe est de transformer l'évaluation de l'illumination indirecte en une illumination directe en évaluant de nombreux émetteurs ponctuels.

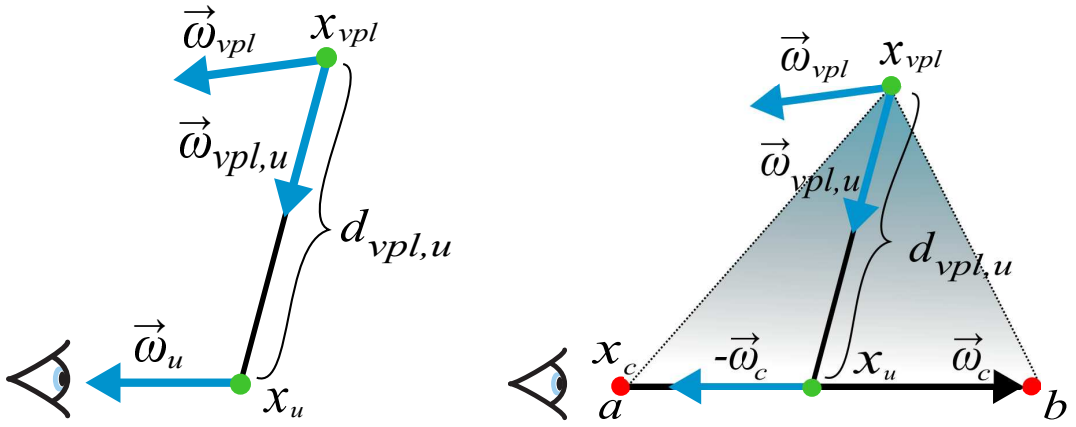
Dérivation : La contribution d'une VPL x_{vpl} en un point x_u dans un milieu participatif est donnée par l'équation 4.17 (voir la figure 4.27(a)) :

$$L_i(x_u \rightarrow \vec{\omega}_u) \approx \Phi \frac{\rho(x_{vpl}, \vec{\omega}_{vpl} \rightarrow \vec{\omega}_{vpl,u}) \rho(x_u, \vec{\omega}_{vpl,u} \rightarrow \vec{\omega}_u) T_r(d_{vpl,u}) V(x_{vpl}, x_u)}{{d^2}_{vpl,u}}, \quad (4.17)$$

avec Φ la puissance du photon, d la distance, V la visibilité, $\rho(x_{vpl}, \vec{\omega}_{vpl} \rightarrow \vec{\omega}_{vpl,u})$ la fonction de phase à la position x_{vpl} de la VPL, et $\rho(x_u, \vec{\omega}_{vpl,u} \rightarrow \vec{\omega}_u)$ la fonction de phase au point d'évaluation x_u. La contribution d'une VPL sur un rayon est égale à l'intégration de l'équation 4.17 sur la longueur du segment :

$$L_m(x_u \leftarrow \vec{\omega}_c) \approx \Phi \int_0^t \frac{\sigma_s(x_{vpl}) \rho(x_{vpl}, \vec{\omega}_{vpl} \rightarrow \vec{\omega}_{vpl,u}) \rho(x_u, \vec{\omega}_{vpl,u} \rightarrow \vec{\omega}_u) T_r(d_{vpl,u}) V(x_{vpl}, x_u)}{{d^2}_{vpl,u}} du, \quad (4.18)$$

avec $t \in [a,b]$, a et b étant les deux points du segment (voir la figure 4.27(b)).



(a) Contribution d'une VPL x_{vpl} en un point x_u dans la direction $\vec{\omega}_u$.

(b) Contribution d'une VPL x_{vpl} pour un segment (a,b) dans la direction $-\vec{\omega}_c$.

Figure 4.27 : Géométrie de la contribution incidente directe totale d'une VPL x_{vpl} pour un point et un segment.

Échantillonnage préférentiel : Kulla et al. [22] ont proposé un échantillonnage préférentiel très simple à mettre en place dans le cas de diffusion simple pour des milieux participatifs homogènes. Avec une lumière ponctuelle, il est assez évident que la luminance incidente directe sur le rayon de la caméra est plus importante pour les points les plus proches de la lumière. Nous aurons clairement une convergence plus rapide si l'on procède à un échantillonnage des points avec une importance inverse au carré de la distance plutôt qu'uniforme.

Kulla et al. [22] ont premièrement procédé à un changement de paramétrisation de t pour que l'origine du rayon de la caméra soit maintenant à la position de l'émetteur projetée perpendiculairement sur le rayon, ce qui donne (les fonctions de phase ont été omises) :

$$L(x,\vec{\omega}) = \Phi\sigma_s \int_a^b \frac{e^{-\sigma_t(t+\Delta+\sqrt{D^2+t^2}}}{D^2+t^2} dt, \tag{4.19}$$

avec Δ la distance entre la vraie origine et la nouvelle (voir la figure 4.28). La PDF

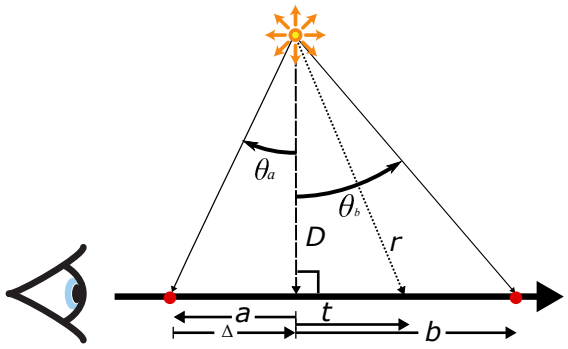


Figure 4.28 : Interprétation géométrique de la luminance incidente totale d'une VPL pour un rayon avec le changement paramétrique de t proposée par Kulla et al. [22].

proportionnelle à $1/r^2$ proposée est :

$$\begin{aligned} pdf(t) &= \frac{D}{(\theta_b-\theta_a)(D^2+t^2)} \\ t(\xi) &= D\tan((1-\xi)\theta_a + \xi\theta_b), \end{aligned} \tag{4.20}$$

avec $\xi \in [0,1)$.

Algorithme : Pour calculer une image avec les VPLs, il suffit de faire la 1ère étape de l'algorithme du *photon mapping* (voir la section 4.3). Puis pour la 2ème étape, de traiter chaque photon comme une VPL et d'accumuler l'évaluation de l'équation 4.18.

Conclusion : Comme pour tous les algorithmes du type *many lights*, il faut utiliser une autre technique pour calculer la contribution de l'illumination directe. Il est facile de faire un rendu progressif, car il suffit de moyenner les passes de rendu. Cette technique n'utilise que très peu de mémoire, mais elle est assez lente. Il faut en fait un grand nombre de VPLs pour arriver à un résultat satisfaisant. Il est très facile de porter cette technique sur GPU, ce qui permet d'améliorer les performances.

Le problème le plus important que l'on rencontre avec les VPLs est la présence de singularités aux abords des émetteurs. Elles sont dues au dénominateur qui divise par la distance, menant alors à des valeurs infinies. Il est possible de limiter l'énergie à un maximum, mais cela crée une perte. Il existe d'autres approches pour corriger ce problème comme la compensation du biais [20] ou par l'utilisation de lumières sphériques [9]. Du fait de sa lenteur et du grand nombre de VPLs à générer et évaluer, de nombreuses techniques de regroupement (*cluster*) ont été développées ; on pense ici principalement aux *lightcuts* [34, 35].

Résultats : Pour montrer le fonctionnement de l'algorithme, nous avons calculé un rendu volumique (voir la figure 4.29) ainsi qu'une série de rendus surfaciques (voir les figures 4.30 et 4.31) comparatifs pour illustrer l'effet de la limitation de l'énergie.

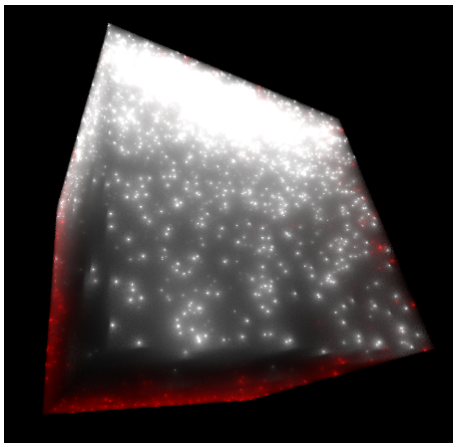
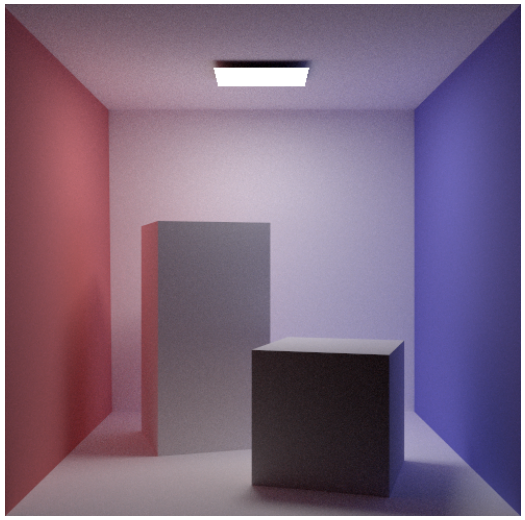


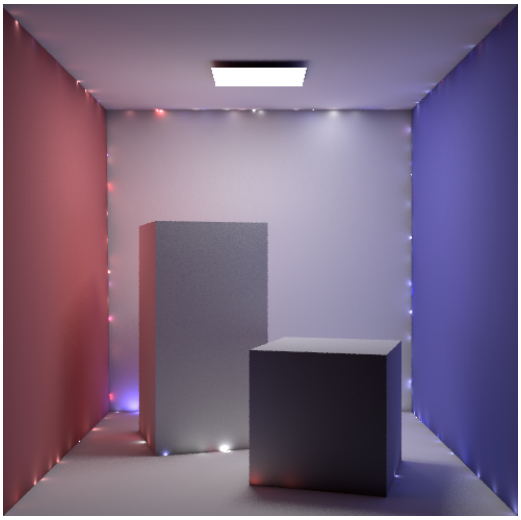
Figure 4.29 : Rendu d'un milieu participatif homogène avec VPLs et illumination directe. On distingue très bien les singularités sous forme de points très lumineux (rendu en 9min52s).

4.7 Rayons de lumière virtuels (*virtual ray light*)

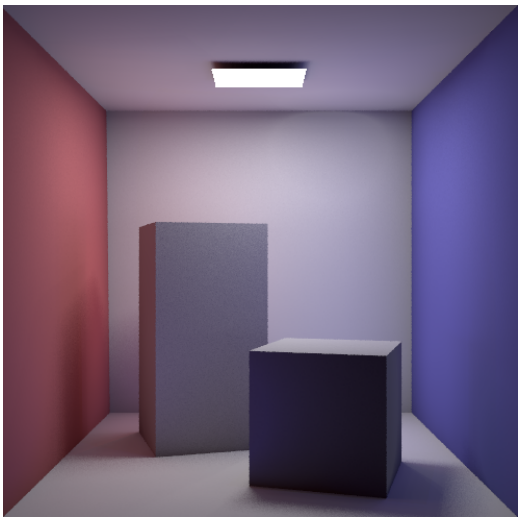
L'algorithme des rayons de lumière virtuels ou *virtual ray lights (VRL)* [26] fait partie des techniques de lumières multiples (*many lights*) comme les VPLs (voir la section 4.6).



(a) Résultat de référence avec rendu *path tracing*, 0min49s.

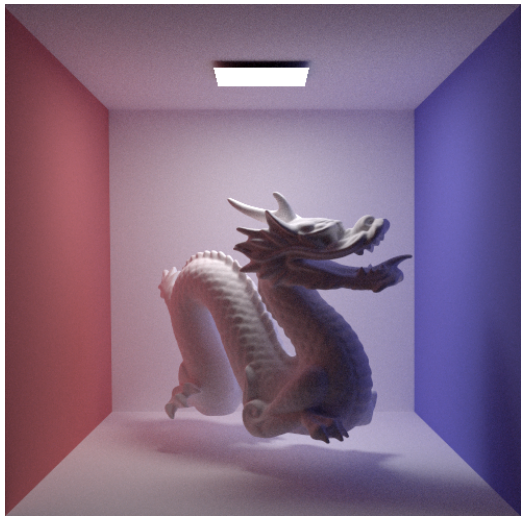


(b) Rendu VPL sans limitation de l'énergie des singularités, 5min26s.



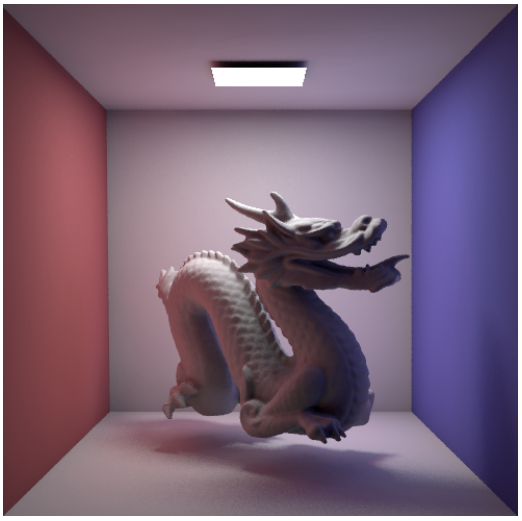
(c) Avec limitaton de l'énergie des singularités, 5min22s.

Figure 4.30 : Rendu avec VPL surfacique avec et sans limitation de l'énergie des singularités. On voit que les images du bas sont légèrement plus sombres (visible dans les angles).



(a) Résultat de référence avec rendu *path tracing*, 1min5s.

(b) Rendu avec VPL sans limitation de l'énergie des singularités, 9min23s.



(c) Avec limitaton de l'énergie des singularités, 9min30s.

Figure 4.31 : Rendu VPL surfacique avec et sans limitation de l'énergie des singularités. On voit que les images du bas sont légèrement plus sombres (visible dans les angles).

Ici aussi la contribution de l'illumination indirecte se calcule en évaluant de nombreux émetteurs. Comme pour l'algorithme des faisceaux (voir la section 4.5) vu précédemment, il utilise le chemin des photons au complet plutôt que seulement les positions aux interactions de diffusion.

Dérivation : Une suite continue de VPLs avec un espacement qui tend vers zéro correspond à une VRL. Pour un point x_u dans un médium, en utilisant l'équation 4.17 de la luminance incidente totale d'une VPL en un point que l'on intègre le long d'une VRL (voir la figure 4.32(a)) on obtient :

$$L_i(x_u \rightarrow \vec{\omega}_u) \approx \Phi \int_0^t \frac{\sigma_s(x_v)\rho(x_v, \vec{\omega}_{vrl} \rightarrow \vec{\omega}_{u,v})\rho(x_u, \vec{\omega}_{u,v} \rightarrow \vec{\omega}_u)T_r(d_{v,u})T_r(v)V(x_v, x_u)}{d^2{}_{u,v}}\, dv, \tag{4.21}$$

avec Φ comme puissance du photon, d la distance, V la visibilité, $\rho(x_v, \vec{\omega}_{vrl} \rightarrow \vec{\omega}_{u,v})$ la fonction de phase à la position x_v sur la VRL (ce point est une VPL), $\rho(x_u, \vec{\omega}_{u,v} \rightarrow \vec{\omega}_u)$ la fonction de phase au point d'évaluation x_u, $T_r(d_{v,u})$ la transmittance entre les deux points x_u et x_v et $T_r(v)$ la transmittance le long de la VRL (ce terme disparaît pour les rayons courts, voir la section 4.5).

Pour la luminance réfléchie en un point x_u sur une surface, il suffit de remplacer la fonction de phase par la BRDF au point x_u de la surface, ce qui donne :

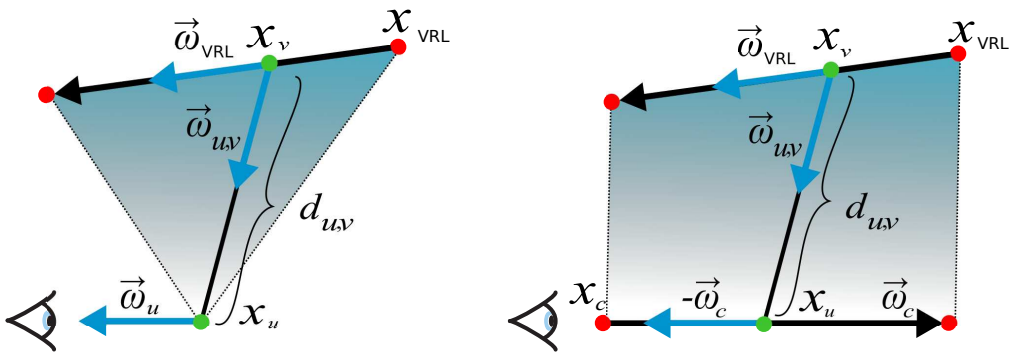
$$L_s(x_u \rightarrow \vec{\omega}_u) \approx \Phi \int_0^t \frac{\sigma_s(x_v)\rho(x_v, \vec{\omega}_{vrl} \rightarrow \vec{\omega}_{u,v})f_r(x_u, \vec{\omega}_{u,v} \leftrightarrow \vec{\omega}_u)T_r(d_{u,v})T_r(v)V(X_v, X_u)}{d^2{}_{u,v}}\, dv. \tag{4.22}$$

De même, en substituant l'équation 4.21 dans l'équation 2.33 on trouve la luminance incidente totale accumulée le long du rayon de la caméra $-\vec{\omega}_c$. Voir la figure 4.32(b) pour la représentation géométrique :

$$L_m(X_c \leftarrow \vec{\omega}_c) \approx \\ \Phi \int_0^s \int_0^t \frac{\sigma_s(x_v)\sigma_s(x_u)\rho(x_v, \vec{\omega}_{vrl} \rightarrow \vec{\omega}_{v,u})\rho(x_u, \vec{\omega}_{u,v} \rightarrow -\vec{\omega}_c)T_r(d_{u,v})T_r(v)T_r(u)V(x_v, x_u)}{d^2{}_{u,v}}\, dv\, du. \tag{4.23}$$

L'équation 4.23 qui définit une intégrale selon l'axe u le long du rayon de la caméra et v le long de la VRL, n'a pas de solution analytique. Nous utiliserons donc la méthode de Monte Carlo pour l'estimer, en posant $g(u,v)$ égale à l'équation 4.23 et un $pdf(u_i, v_i)$ égale à la probabilité de choisir un point (u_i, v_i). L'estimateur sera :

$$L_m(X_c \leftarrow \vec{\omega}_c) \approx \frac{1}{N} \sum_{i=1}^{N} \frac{g(u_i, v_i)}{pdf(u_i, v_i)}. \tag{4.24}$$



(a) Illustration géometrique de la luminance incidente d'une VRL pour un point X_u.

(b) Illustration géometrique de la luminance incidente d'une VRL sur le rayon de la camera $-\vec{\omega}_c$.

Figure 4.32 : Contribution d'une VRL pour un point (gauche) et la contribution accumulée d'une VRL pour un rayon (droite).

Échantillonner u_i et v_i de façon uniforme sera sous-performant, car cela va nécessiter un grand nombre d'évaluations de notre intégrale. La meilleure solution consiste à utiliser un échantillonnage préférentiel pour les points sur le rayon de la caméra et sur la VRL comme nous avons vu pour les VPLs. En sélectionnant les points de plus forte importance, nous obtiendrons convergence plus rapide.

Échantillonnage préférentiel : Novák et al. [26] ont développé des PDFs pour les cas de diffusion isotrope et anisotrope. Nous n'allons présenter ici que le 1er cas que nous avons implémenté. C'est la distance inverse $d(u,v)^{-2}$ qui est la principale source de variation, une PDF adéquate serait donc $pdf(u_i,v_i) \propto d(u,v)^{-2}$. Malheureusement, l'inversion de cette PDF n'a pas de solution analytique. À la place, il est proposé une solution en deux étapes qui premièrement échantillonne un point v_i le long de la VRL avec une PDF marginale, puis un second point u_i le long de la caméra avec une PDF conditionnelle basée sur la distance $(u_i,v_i)^{-2}$. Il faut premièrement effectuer un changement de variable en utilisant les deux points les plus proches u_h, v_h tel que $\hat{u} = u - u_h$ et $\hat{v} = v - v_h$. On définit $\hat{u}_0, \hat{u}_1, \hat{v}_0, \hat{v}_1$, qui sont la translation des points de départ et d'arrivée de la caméra et de la VRL (voir la figure 4.33). Nous aurons ainsi un échantillonnage qui se concentre aux alentours des deux points les plus proches. Par la loi des cosinus, on définit la distance au carré entre deux points tel que $d(\hat{u},\hat{v},h,\theta)^2 = h^2 + \hat{u}^2 + \hat{v}^2 - 2\hat{u}\hat{v}\cos\theta$ avec $\cos\theta$ le produit vectoriel des directions de la caméra et de la VRL. La PDF marginale est maintenant définie par :

$$pdf_v(\hat{v},\hat{v}_0,\hat{v}_1) = \frac{\int_{\hat{u}_0}^{\hat{u}_1} d(\hat{u},\hat{v},h,\theta)^{-2}\, d\hat{u}}{\int_{\hat{v}_0}^{\hat{v}_1}\int_{\hat{u}_0}^{\hat{u}_1} d(\hat{u},\hat{v},h,\theta)^{-2}\, d\hat{u}\, d\hat{v}}. \tag{4.25}$$

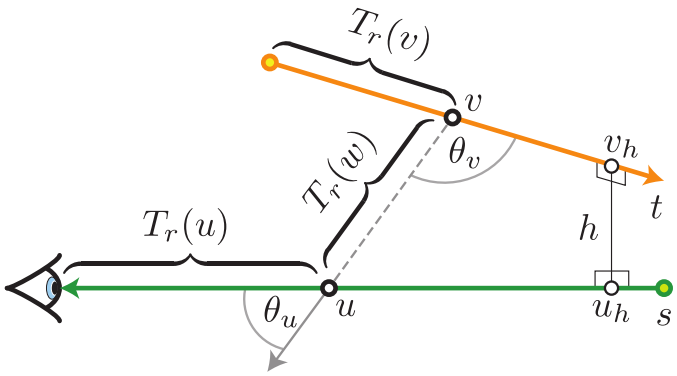


Figure 4.33 : Représentation géométrique des termes utilisés pour l'échantillonnage préférentiel des points u, v sur la VRL et le rayon de la caméra, Novák et al. [26, p.3].

Il n'y a pas de solution analytique connue pour l'équation 4.25, mais il est possible toutefois de procéder à une simplification en considérant que le rayon de la caméra est de longueur infinie. On obtient finalement après résolution la PDF marginale suivante :

$$pdf_v(\hat{v},\hat{v}_0,\hat{v}_1) = \frac{\int_{-\infty}^{\infty} d(\hat{u},\hat{v},h,\theta)^{-2}\,d\hat{u}}{\int_{\hat{v}_0}^{\hat{v}_1}\int_{-\infty}^{\infty} d(\hat{u},\hat{v},h,\theta)^{-2}\,d\hat{u}\,d\hat{v}} = \frac{\frac{\pi}{\sqrt{h^2+\hat{v}^2\sin^2\theta}}}{\pi\frac{A(\hat{v}_1)-A(\hat{v}_0)}{\sin\theta}}, \tag{4.26}$$

avec $A(x) = \sinh^{-1}(\frac{x}{h}\sin\theta)$. Par intégration de l'équation 4.26, on obtient le CDF marginale :

$$cdf_v(\hat{v},\hat{v}_0,\hat{v}_1) = \frac{A(\hat{v}_0)-A(\hat{v})}{A(\hat{v}_0)-A(\hat{v}_1)}. \tag{4.27}$$

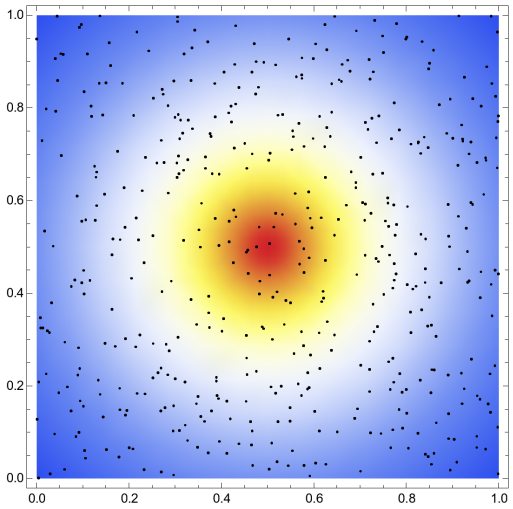
On obtient ainsi la CDF inverse :

$$cdf_v^{-1}(\xi,\hat{v}_0,\hat{v}_1) = \frac{h\sinh(lerp(A(\hat{v}_0),A(\hat{v}_1),\xi))}{\sin\theta}. \tag{4.28}$$

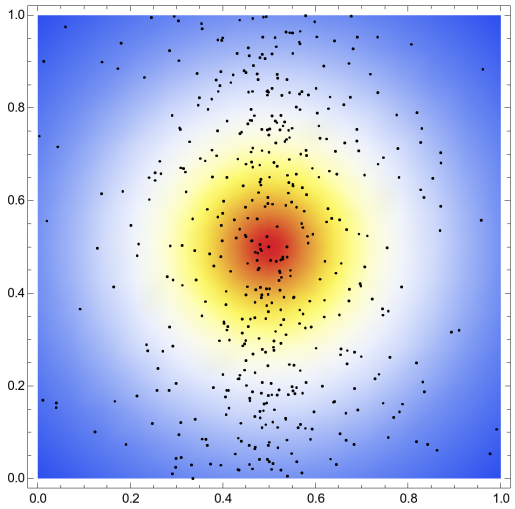
La CDF inverse permet maintenant d'échantillonner une position v_i sur la VRL en utilisant un nombre aléatoire $\xi_1 \in [0,1)$. Puis, à l'aide d'un nouveau nombre aléatoire $\xi_2 \in [0,1)$ et de l'échantillonnage équiangulaire (voir la section 4.6), on échantillonne une position u_i sur le rayon de la caméra. La PDF finale est le produit des deux PDFs utilisées pour l'échantillonnage. La figure 4.34 compare différentes méthodes d'échantillonnages, on peut voir que la méthode présentée est la plus efficace.

Algorithme : Pour implémenter l'algorithme, il faut suivre les étapes suivantes :

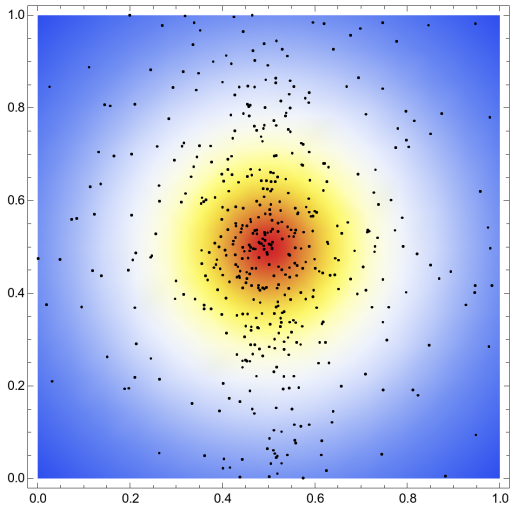
1ère étape :
Cette étape est identique à la 1ère étape du *photon beams* (voir la section 4.5). Mais ici nous n'utilisons pas de structure de données accélératrice et nous n'avons pas besoin de



(a) Échantillonnage aléatoire sur la VRL et le rayon de la caméra. On voit que les points sont uniformément distribués, l'échantillonnage est donc sous-performant.



(b) Échantillonnage aléatoire sur la VRL et préférentiel avec la méthode de Kulla et al. [22] sur le rayon de la caméra. On voit que les points sur le rayon de la caméra sont plus proches des points sur la VRL.



(c) Échantillonnage préférentiel sur la VRL avec la méthode présentée (voir la section 4.7) et la méthode de Kulla et al. [22] sur le rayon de la caméra. On voit que les points sur le rayon de la caméra et sur la VRL sont très proches.

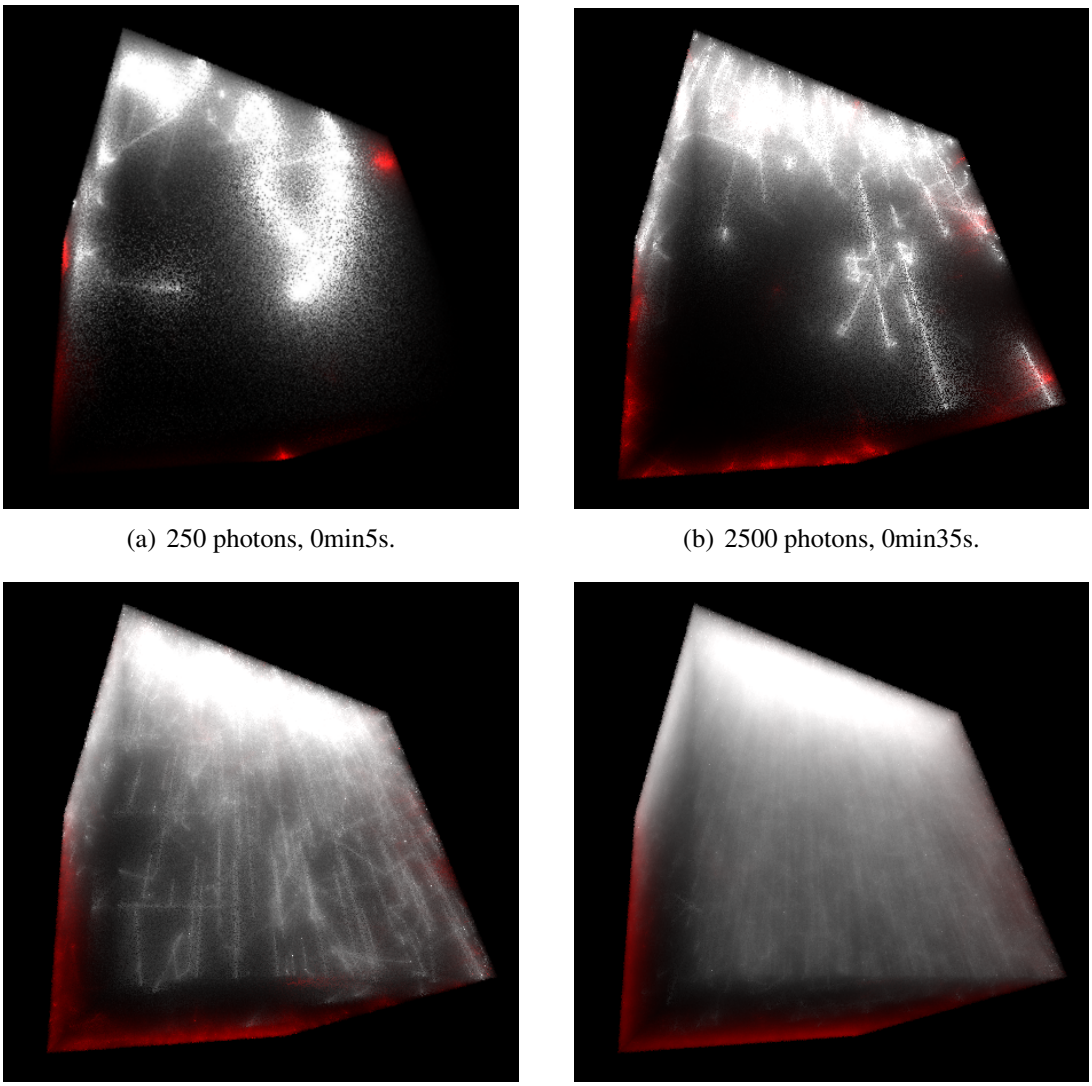
Figure 4.34 : Illustration de trois méthodes d'échantillonnage des positions sur une VRL (axe vertical) et sur le rayon de la caméra (axe horizontal). La couleur du fond représente la distance $(u,v)^2$. Les points les plus proches sont à la position $u = 0.5$ et $v = 0.5$. On voit que la troisième technique échantillonne des paires de points beaucoup plus proches.

calculer des rayons.

2ème étape :
Cette étape est le rendu, alors on va lancer un rayon par pixel et évaluer l'équation 4.23 en appliquant ce que nous avons vu dans la section d'échantillonnage préférentiel pour chacune des VRLs.

Conclusion : Les VRLs étant beaucoup plus denses que les VPLs, on obtient un résultat avec moins de singularités et une meilleure estimation. Comme pour l'algorithme des *faisceaux* (voir la section 4.5), il est aussi possible d'utiliser des segments courts ou longs (voir la section 4.5), mais comme nous ne calculons pas ici de densité locale et qu'il faut évaluer toutes les VRLs, cela n'apporte rien. Comme tous les algorithmes du type *many lights*, il faut utiliser une autre technique pour calculer l'illumination directe. L'estimateur étant non biaisé, il est possible de faire un rendu progressif en moyennant chacune des passes de rendu. Comme pour les VPLs, des techniques de regroupement (*cluster*) ont été développées mais il y a très peu de publications à ce sujet, les travaux les plus intéressants étant ceux de Frederick et al. [7] ainsi que ceux de Mortensen et Stokholm [32].

Résultats : Comme pour les faisceaux de photons (voir la section 4.5), une version progressive avec des faisceaux courts (voir la figure 4.35) et longs (voir la figure 4.36) a été implémentée.



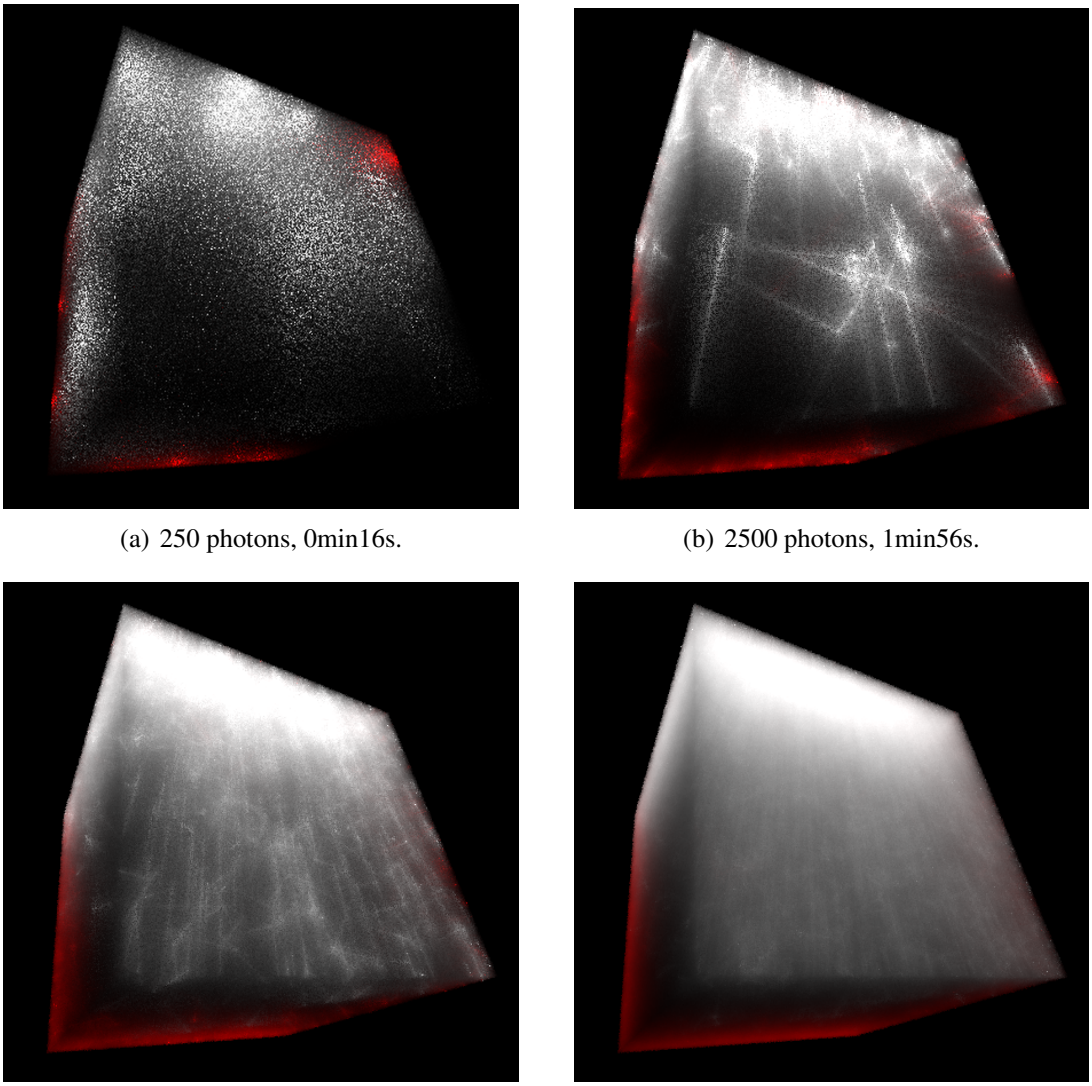
(a) 250 photons, 0min5s.

(b) 2500 photons, 0min35s.

(c) 25000 photons, 5min30s.

(d) 250000 photons, 42min25s.

Figure 4.35 : Rendu de l'illumination indirecte par VRLs avec des rayons courts et un milieu homogène.



(a) 250 photons, 0min16s.

(b) 2500 photons, 1min56s.

(c) 25000 photons, 9min25s.

(d) 250000 photons, 4h5min8s.

Figure 4.36 : Rendu de l'illumination indirecte par VRLs avec des rayons longs et un milieu homogène.

4.8 Conclusion

Dans ce chapitre, plusieurs algorithmes pour simuler la diffusion de la lumière dans les milieux participatifs ont été présentés.

On a commencé par une technique purement stochastique basée sur l'intégration de Monte Carlo : le *path tracing* (voir la section 4.1). Elle est simple à implémenter et non biaisée, ses résultats ont servi de référence pour les autres algorithmes. Mais sa variance est grande, ce qui crée un bruit très visible dans l'image. Sa convergence étant lente, cela justifie le développement de nouvelles techniques.

Le *ray marching* (voir la section 4.2) est assez lent, ce qui rend son utilisation peu performante pour la diffusion multiple. Mais cette technique d'intégration numérique fonctionne bien quand il n'y a pas de récursion par exemple. C'est le cas pour le calcul de l'illumination directe avec des milieux participatifs hétérogènes ou pour l'intégration le long du rayon de la caméra avec le *photon mapping*.

Ensuite, trois techniques utilisant une estimation de densité pour le calcul de la luminance ont été détaillés :

La première est le *photon mapping* (voir la section 4.3). C'est un algorithme bi-directionnel très performant qui fonctionne en deux passes. Il est biaisé dans sa version non progressive, ce biais apparaît sous forme de flou. Dans le cas des milieux participatifs, l'algorithme utilise le *ray marching* pour trouver les photons le long du rayon de la caméra. Il est généralement utilisé pour l'illumination indirecte et permet aussi de tracer des chemins qui sont difficiles pour les techniques stochastiques comme les caustiques. Il est nécessaire de le coupler à un lancé de rayons pour les surfaces ayant une composante spéculaire.

Le *beam radiance estimate* (voir la section 4.4) est une variante du calcul de l'estimation de densité utilisée par le *photon mapping* pour les milieux participatifs. Elle permet de trouver les photons le long du rayon de la caméra sans utiliser le *ray marching* ce qui permet d'obtenir de meilleures performances.

L'algorithme suivant, le *photon beams* (voir la section 4.5), est une variante du *photon mapping* qui utilise une représentation de l'éclairement énergétique continue plutôt que ponctuel comme le fait le *photon mapping*. Ce changement permet d'avoir une meilleure convergence, car la densité d'information locale est plus grande.

Ensuite, on a montré deux techniques qui fonctionnent aussi en deux passes et qui évaluent l'illumination indirecte en accumulant l'illumination directe de nombreux émetteurs virtuels :

Les *virtual point lights* (voir la section 4.6) qui utilisent des lumières ponctuelles, ce qui crée des singularités aux abords des émetteurs (voir les figures 4.31(b) et 4.29). Et finalement les *virtual ray lights* (voir la section 4.7), qui utilisent des segments comme émetteurs.

CHAPITRE 5

CONCLUSION

La diffusion de la lumière dans les milieux participatifs est responsable de nombreux phénomènes visuels importants. La simulation de ses effets engendre généralement de très coûteux calculs qui varient avec la densité, la diffusion (avec l'albédo) et le tropisme en générant différentes interactions avec la lumière.

Nous avons vu trois types de simulations du transport de la lumière qui arrivent à reproduire des effets volumétriques complexes. La première solution présentée est basée sur l'intégration de Monte Carlo. Les autres, quant à elles, sont des techniques en deux passes, bidirectionnelles, utilisant des particules, avec une évaluation par estimation de densité ou bien par calcul de l'illumination directe.

On peut se demander quelles sont les meilleures méthodes.

Il est en fait difficile pour plusieurs raisons d'élaborer une méthode de comparaison des techniques que nous avons vues. Nous aurions pu simplement calculer la variance pour un seul pixel et conclure, mais ce n'est pas aussi simple. Il existe un grand nombre de paramètres qui influent sur les performances et donc sur les résultats de tests comparatifs.

Premièrement, les algorithmes qui calculent une estimation de densité utilisent de façon intensive une ou plusieurs structures de données spatiales pour organiser les informations. Ces structures sont très importantes car sans elles, ces algorithmes seraient extrêmement lents pour localiser les informations nécessaires aux calculs.

Dans nos algorithmes, nous avons utilisé les structures suggérées par les auteurs avec des implémentations uniquement sur CPU, mais il n'est pas certain que ces choix soient les plus adéquats. Nous savons que même avec une implémentation parallélisée, propre et robuste, nous n'obtiendrons pas les performances optimums. Il faut en pratique tenir compte du fonctionnement de la machine à un niveau plus bas, ce qui complexifie la tâche de programmation. On pense particulièrement ici à la vectorisation du code avec les fonctions intrinsèques telles que SSE et AVX, à la gestion de la cache et des pénalités (*cache misses*), aux allocations dynamiques de mémoire (préallocation et alignement, etc.), etc.

Avec l'avènement des GPUs on pourrait penser que le problème est réglé, mais ce n'est pas du tout le cas. Le problème est même encore plus complexe. Il y a une multitude de versions de GPUs plus ou moins compatibles et des transferts mémoires pénalisant.

Au jour d'aujourd'hui, cette problématique liée à l'efficacité des structures de données spatiales et à leur implémentation est un domaine de recherche à part entière. Un très bon exemple est la bibliothèque *open source* **Embree** développée par **Intel**, que nous utilisons et elle est extrêmement performante pour calculer l'intersection avec le monde virtuel, un grand nombre d'applications commerciales l'ont incorporée (V-RAY, Cinema 4D, etc.). Ses performances sont sans commune mesure avec ce que nous pourrions programmer avec nos moyens.

Deuxièmement, nous n'avons pas implémenté toutes les versions des algorithmes. Si nous prenons l'exemple de l'algorithme des faisceaux de photons (voir la section 4.5), Jarosz et al. [14] font état de neuf variations pour l'estimation de la radiance et c'est sans compter la possibilité d'utiliser un rayon constant ou variable. De plus, les nombreux paramètres tels que la taille du noyau utilisé par l'estimation ont aussi un impact (voir la figure 5.1).

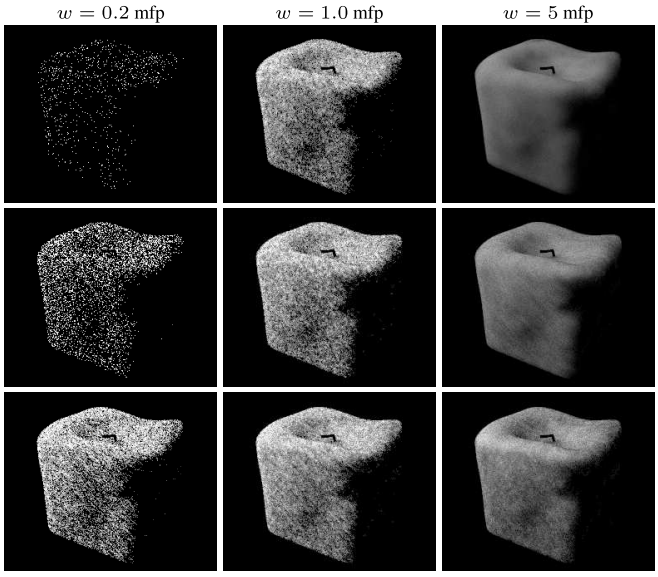


Figure 5.1 : Comparaison des résultats pour trois différentes estimations de densités [23]. Verticalement de haut en bas, rendu avec, *photon mapping* (voir la section 4.3), *beam radiance estimate* (voir la section 4.4) et *photon beams* (voir la section 4.5). L'axe horizontal *mfp* (*mean free paths*) est la largeur du noyau, son unité est l'inverse du libre parcours moyen ($mfp \equiv \sigma_t^{-1}$).

Troisièmement, tous les algorithmes ne sont pas capables de reproduire tous les chemins de la lumière. Si nous souhaitons par exemple simuler des effets de caustiques volumiques (voir la figure 5.1), l'algorithme du *path tracing* que nous avons vu aura énormément de difficulté à les reproduire. Ces chemins du type LS+D (notation de Heckbert [10]) sont par contre très faciles à réaliser avec les techniques utilisant des photons

avec une *photon map* spécialisée. Certains algorithmes sont aussi plus ou moins efficaces suivant les propriétés des milieux participatifs.

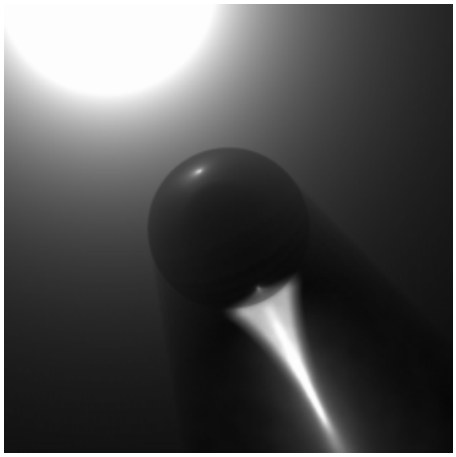


Figure 5.2 : Rendu de caustique volumique avec 500,000 photons avec la technique du *photon mapping* (Jensen [17]).

Quatrièmement, il y a un point important qui n'a pas été abordé et qui est celui de la visibilité. Certains algorithmes ont besoin de faire de nombreux tests d'occultation. Si nous prenons les exemples des VRLs et des VPLs, il est nécessaire au moment du rendu pour chacune des évaluations, de savoir s'il y a un objet opaque bloquant ou pas entre le point évalué et le point sur l'émetteur. Pour des raisons de simplification, nous n'en avons pas tenu compte dans nos implémentations et nous avons aussi choisi notre scène de test de façon à ne pas avoir d'occultation.

Comme le montre la récente publication UPBP [21] (*Unifying Points, Beams, and Paths in Volumetric Light Transport Simulation*), il est aussi possible de combiner les techniques d'estimation de densité et d'intégration avec Monte Carlo dans un même algorithme afin d'obtenir une nouvelle technique.

Concernant les techniques de *lumière multiple* VRL (voir la section 4.7) et VPL (voir la section 4.6) que nous avons vues, nous savons qu'il existe des pistes d'amélioration. Une idée prometteuse est l'utilisation des *lightcuts* [35]. Cette approche est déjà connue pour fonctionner avec les VPLs et les rendus surfaciques. Il serait tout à fait envisageable de l'utiliser avec des milieux participatifs, de même qu'il n'existe aucune publication concernant les VBLs [25] (faisceaux de lumière virtuels) et les *lightcuts*.

On peut conclure qu'il semble plus pertinent de s'orienter vers une optimisation des performances de chacune des techniques et de travailler à trouver des solutions pour les combiner plutôt que de les opposer.

BIBLIOGRAPHIE

[1] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[2] Philippe Blasi, Bertrand Saec et Christophe Schlick. A rendering algorithm for discrete volume density objects. Dans *Computer Graphics Forum*, volume 12, pages 201–210. Wiley Online Library, 1993.

[3] Leo Breiman, William Meisel et Edward Purcell. Variable kernel estimates of multivariate densities. *Technometrics*, 19(2):135–144, 1977.

[4] K.M. Case et P.F. Zweifel. *Linear Transport Theory*. Addison-Wesley Series in Nuclear Engineering. Addison-Wesley, 1967. URL `https://books.google.ca/books?id=pvxQAAAAMAAJ`.

[5] Subrahmanyan Chandrasekhar. *Radiative transfer*. Dover publications, New York, 1960. ISBN 0-486-60590-6. URL `http://opac.inria.fr/record=b1078136`. Unabridged and slightly revised version of the work first published in 1950.

[6] D Deirmendjian. Electromagnetic scattering on spherical polydispersions american elsevier publ. *Comp., New York*, 1969.

[7] Roald Frederickx, Pieterjan Bartels et Philip Dutré. Adaptive LightSlice for Virtual Ray Lights. Dans B. Bickel et T. Ritschel, éditeurs, *EG 2015 – Short Papers*, pages 61–64. The Eurographics Association, 2015.

[8] Toshiya Hachisuka, Shinji Ogaki et Henrik Wann Jensen. Progressive photon mapping. Dans *ACM SIGGRAPH Asia 2008 Papers*, SIGGRAPH Asia '08, pages 130 :1–130 :8, New York, NY, USA, 2008. ACM. ISBN 978-1-4503-1831-0. URL `http://doi.acm.org/10.1145/1457515.1409083`.

[9] Miloš Hašan, Jaroslav Křivánek, Bruce Walter et Kavita Bala. Virtual spherical lights for many-light rendering of glossy scenes. Dans *ACM Transactions on Graphics (TOG)*, volume 28, page 143. ACM, 2009.

[10] Paul S Heckbert. Adaptive radiosity textures for bidirectional ray tracing. *ACM SIGGRAPH Computer Graphics*, 24(4):145–154, 1990.

[11] Louis G Henyey et Jesse Leonard Greenstein. Diffuse radiation in the galaxy. *The Astrophysical Journal*, 93:70–83, 1941.

[12] Hendrik Christoffel Hulst et Hendrik C van de Hulst. *Light scattering by small particles*. Courier Corporation, 1957.

[13] Wenzel Jakob. Mitsuba renderer, 2010. http ://www.mitsuba-renderer.org.

[14] Wojciech Jarosz, Derek Nowrouzezahrai, Iman Sadeghi et Henrik Wann Jensen. A comprehensive theory of volumetric radiance estimation using photon points and beams. *ACM Transactions on Graphics (Presented at SIGGRAPH)*, 30(1):5 :1–5 :19, janvier 2011.

[15] Wojciech Jarosz, Matthias Zwicker et Henrik Wann Jensen. The beam radiance estimate for volumetric photon mapping. *Computer Graphics Forum (Proceedings of Eurographics)*, 27(2):557–566, avril 2008.

[16] Henrik Wann Jensen. *Realistic Image Synthesis Using Photon Mapping*. A. K. Peters, Ltd., Natick, MA, USA, 2001. ISBN 1-56881-147-0.

[17] Henrik Wann Jensen et Per H. Christensen. Efficient simulation of light transport in scenes with participating media using photon maps. Dans *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 311–320, New York, NY, USA, 1998. ACM. ISBN 0-89791-999-8. URL `http://doi.acm.org/10.1145/280814.280925`.

[18] James T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, août 1986. ISSN 0097-8930. URL `http://doi.acm.org/10.1145/15886.15902`.

[19] Alexander Keller. Instant radiosity. Dans *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 49–56. ACM Press/Addison-Wesley Publishing Co., 1997.

[20] Thomas Kollig et Alexander Keller. Illumination in the presence of weak singularities. Dans *Monte Carlo and Quasi-Monte Carlo Methods 2004*, pages 245–257. Springer, 2006.

[21] Jaroslav Křiv'anek, Iliyan Georgiev, Toshiya Hachisuka, Petr V'evoda, Martin Šik, Derek Nowrouzezahrai et Wojciech Jarosz. Unifying points, beams, and paths in volumetric light transport simulation. *ACM Trans. Graph.*, 33(4):1–13, août 2014. ISSN 0730-0301.

[22] Christopher D Kulla et Marcos Fajardo. Importance sampling of area lights in participating media. Dans *SIGGRAPH Talks*, page 55, 2011.

[23] Jaroslav Křivánek, Iliyan Georgiev, Toshiya Hachisuka, Petr Vévoda, Martin Šik, Derek Nowrouzezahrai et Wojciech Jarosz. Unifying points, beams, and paths in volumetric light transport simulation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 33(4), juillet 2014.

[24] KN Liou. An introduction to atmospheric radiation academic. *New York*, 19802: 392, 1980.

[25] Jan Novák, Derek Nowrouzezahrai, Carsten Dachsbacher et Wojciech Jarosz. Progressive virtual beam lights. *Computer Graphics Forum (Proceedings of EGSR)*, 31(4), juin 2012.

[26] Jan Novák, Derek Nowrouzezahrai, Carsten Dachsbacher et Wojciech Jarosz. Virtual ray lights for rendering scenes with participating media. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4), juillet 2012.

[27] Jan Novák, Andrew Selle et Wojciech Jarosz. Residual ratio tracking for estimating attenuation in participating media. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 33(6), novembre 2014.

[28] Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

[29] J Spanier et EM Gelbard. Monte carlo principles and neutron transport problems. 1969.

[30] Julius Adams Stratton. Electromagnetic theory. international series in pure and applied physics, 1941.

[31] Lszl Szirmay-Kalos. *Monte Carlo Methods in Global Illumination-Photo-realistic Rendering with Randomization*. VDM Verlag, 2008.

[32] Heine Stokholm Troels Mortensen. Clustering of Virtual Ray Lights. Mémoire de maîtrise, Aarhus University, Danemark, 2014.

[33] Eric Veach. *Robust monte carlo methods for light transport simulation*. Thèse de doctorat, Stanford University, 1997.

[34] Bruce Walter, Adam Arbree, Kavita Bala et Donald P Greenberg. Multidimensional lightcuts. Dans *ACM Transactions on Graphics (TOG)*, volume 25, pages 1081–1088. ACM, 2006.

[35] Bruce Walter, Sebastian Fernandez, Adam Arbree, Kavita Bala, Michael Donikian et Donald P Greenberg. Lightcuts : a scalable approach to illumination. Dans *ACM Transactions on Graphics (TOG)*, volume 24, pages 1098–1107. ACM, 2005.

[36] Gregory J. Ward, Francis M. Rubinstein et Robert D. Clear. A ray tracing solution for diffuse interreflection. Dans *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '88, pages 85–92, New York, NY, USA, 1988. ACM. ISBN 0-89791-275-6. URL `http://doi.acm.org/10.1145/54852.378490`.

[37] Wikipedia. Correction gamma — Wikipedia, the free encyclopedia, 2016. URL https://fr.wikipedia.org/wiki/Correction_gamma.

[38] Wikipedia. Standard rgb — Wikipedia, the free encyclopedia, 2016. URL https://fr.wikipedia.org/wiki/SRGB.

[39] Wikipedia. Transfert de rayonnement — Wikipedia, the free encyclopedia, 2016. URL https://fr.wikipedia.org/wiki/Transfert_de_rayonnement#Fonction_source_du_rayonnement. [Online ; accessed 4-Avril-2016].