Un Modèle Général et Multiéchelle de Textures Volumiques

Fabrice Neyret

INRIA Rocquencourt - Projet SYNTIM   BP 105, 78153 Le Chesnay Cedex

Fabrice.Neyret@inria.fr

Mots-clés : réalisme, texture, géométries complexes, modèle de réflectance, niveaux de détail

Résumé : *L* es textures volumiques permettent de modéliser les géométries répétitives complexes comme l'herbe, la fourrure, le feuillage, etc, en stockant des données de type voxel et un modèle de réflectance spécifique. Dans ce papier, nous introduisons une primitive pour la réflectance qui est relativement générique, mais surtout 'filtrable'. Générique signifie qu'un large spectre de données peut être représenté directement ; filtrable signifie que l'on peut définir une fonction de lissage, capable de convertir le modèle de réflectance aux diverses échelles. Cela permet de transformer la méthode des textures volumiques en un outils général et efficace, disposant d'un rendu adaptatif s'appuyant sur des données préfiltrées (d'où faible coût et faible aliasing). En particulier, le coût est maintenant lié à la complexité apparente (ce qui est vu), et non plus à la complexité des données initiales (ce qui est connu).

1 Introduction

Les géométries complexes répétitives comme l'herbe, les cheveux, le feuillage, la fourrure, la forêt, etc, sont des composantes majeures du monde naturel, très importantes pour le réalisme des images de synthèse, mais difficiles à traiter en terme de géométrie explicite. Il existe de nombreuses méthodes pour les modéliser, qui introduisent plus ou moins d'effets 3D en fonction des buts et des moyens de l'utilisateur : plaquage de couleur ou de transparence, cartes de perturbation des normales [3, 12, 8], cartes d'élévation [10, 11, 19], cartes de réflectances filtrables [6], systèmes de particules [15, 16], textures volumiques [7, 18] ...

Toutes ces méthodes ont à traiter deux problèmes : modéliser des effets 3D sans disposer de toute l'information 3D, et donner des résultats valables à diverses échelles tout en évitant l'aliasing (dû à la grande quantité d'information visible derrière chaque pixel de l'image). Ces effets 3D sont l'illumination locale et les effets de blocage (l'occlusion et l'ombrage).
Deux autres problèmes plus généraux résident dans la façon de déterminer les paramètres du modèle qui approximent des données réelles, et dans le fait de pouvoir les utiliser sans programmation additionnelle.

Les textures volumiques constituent une approche texturelle qui utilise une façon 3D de simuler la géométrie, et sont ainsi capables de gérer correctement les effets 3D : la géométrie complexe à modéliser recouvre une surface à la manière d'une peau épaisse (par exemple une prairie recouvrant une colline, ou de la fourrure sur un animal). Un échantillon de cette géométrie est construit dans un volume de référence cubique, qui est déformé et plaqué sur la surface à la manière d'une texture 2D. Habituellement, les copies déformées du volume de référence (appelées *texels*) s'appuient exactement sur un élément de surface bilinéaire, et les quatre sommets supérieurs sont associés aux normales des quatre coins de la face (ces normales pouvant bien sûr être perturbées). Ainsi, le volume déformé est un 'patch' trilinéaire. Le volume de référence stocke une région de l'espace échantillonnée en *voxels*, à une résolution donnée dans les trois dimensions. Deux informations différentes sont stockées dans chaque voxel : une densité (plus précisement une *présence*), qui est la proportion de l'espace du voxel occupée par la géométrie, et une 'mémoire' du comportement de la réflectance de la géométrie dans cette zone. Ce comportement est codé par un modèle de réflectance, commun à tout le volume de référence, et par des paramètres locaux, stockés dans chaque voxel (c'est l'objet de ce papier). La répartition de la densité parmi les voxels représente ainsi l'information spatiale, tandis que la réflectance représente ce qu'il se passe localement dans un voxel, sans notion de position ni de dimension. Le rendu est effectué par lancer de rayon, se transformant en rendu volumique dans les zones de la scène occupées par des texels. Un morçeau de code spécifique implémente le calcul de l'illumination associé au modèle de référence.

Un état de l'art des textures volumiques est présenté au chapitre 2, incluant quelques techniques pour modéliser la réflectance. Nous nous proposons, en partant de l'approche de Kajiya, de construire une représentation plus générale, avec de bonnes propriétés quant au coût de calcul et à la qualité visuelle. Nous adoptons un schéma multiéchelle, qui permet d'économiser les calculs tout en évitant l'aliasing. Il existe déjà des algorithmes multiéchelles pour les textures 2D, qui permettent de précalculer l'aspect local de la texture à diverses échelles, ces précalculs pouvant être utilisés lors du rendu. On se dispense alors de suréchantillonnage, sans pour autant obtenir d'aliasing. Cette amélioration, quand on l'applique aux textures volumiques, implique un modèle de réflectance suffisamment général, et un moyen de le lisser. Nous décrivons ce modèle au chapitre 3.

2 Travaux antérieurs

Les textures volumiques (ou *texels*) ont été introduites par Kajiya en 1989 [7], dans le but de modéliser la fourrure. Dans son implémentation le modèle de réflectance est celui d'un cylindre, c'est à dire que la matière contenu dans un voxel est supposée représenter localement un cylindre de densité donné, dont seul l'axe est stocké dans le voxel (cet axe constitue ainsi un paramètre de réflectance). Une procédure de construction stocke la densité (présence) et les paramètres de réflectance dans chaque voxel du volume de référence.
Lors du rendu, lorsque les rayons intersectent une surface recouverte de texels, la partie du rayon incluse dans le texel est reportée dans l'espace du volume de référence, où un lancer de rayon volumique particulier est utilisé [1].

Voici les grandes lignes du calcul du rendu :

- Un lancer de rayon parcourt la scène à l'aide de rayons qui intersectent les surfaces.
- Lorsqu'une surface est recouverte par un texel, le rayon commute en 'mode volumique' [2]. Ceci est effectué dans le volume de référence, après avoir calculé les points d'entrée et de sortie correspondants à ceux du texel déformé. La trajectoire dans le volume de réference est approximé par une droite (ce qui suppose de n'avoir que de légères déformations).
- le rendu volumique traverse le volume d'avant en arrière ; il multiplie les transparences et additionne les intensités (pondérées par la transparence cumulée). Un échantillonnage stochastique est effectué le long du rayon afin d'éviter de traiter tous les voxels. Une zone de densité ρ traversée sur une épaisseur L a une transparence $e^{-\tau\rho.L}$, où la constante τ convertie la densité en atténuation (cf [7]). L'intensité finale collectée le long d'une large zone inhomogène est donc $I = \sum_{avant}^{arrière}(illu. \prod_{avant}^{courant} e^{-\tau\rho.dL})$.
- L'illumination locale *illu* est le produit de l'albedo (indiquant la quantité de lumière réfléchie en fonction de la densité locale) [3], de la réflectance intrinsèque, et de la lumière (ou de l'ombre) reçue. La réflectivité est obtenue à partir du modèle de réflectance (ici un rendu de cylindre) en fonction des paramètres stockés (l'axe), appliqués aux directions de la lumière et du point de vue.
- La lumière reçue est estimée en lançant un rayon d'ombrage vers la source, qui ne prend en compte que l'atténuation entre la source de lumière et le point courant (l'hypothèse de faible albedo néglige les réflections multiples) [4].

Pour modéliser la fourrure d'un ours en peluche, Kajiya choisit une réflectance cylindrique. Mais supposer que le comportement de la réflectance soit constant dans tout le volume (même si les paramètres sont stockés dans les voxels) limite l'applicabilité à certaines classes d'objets réguliers. Pire, la méthode est chère en stockage, en rendu et en suréchantillonnage.

Shinya a introduit dans [18] quelques idées et extensions (corrélation entre les contenus des texels, rayons coniques pour parcourir le volume), et en particulier la nécessité de filtrer les données afin de précalculer les informations à toutes les échelles. Le filtrage est facteur d'efficacité pour

1. La particularité de ce lancer de rayon volumique est que l'illumination de chaque voxel peut être calculée, connaissant le modèle de réflectance, la lumière incidente, la direction du point de vue et l'albedo (i.e. la lumière sera réfléchie comme si un cylindre occupait le voxel, ce qui représente plus d'information que le simple gradient de densité utilisé par les rendus volumiques classiques).

2. En fait le texel se trouve sur la surface ; une autre surface est donc construite au dessus des texels lors de la construction, et c'est cette surface que l'on intersecte. Certains auteurs posent les texels sous la surface, afin de simplifier la tâche.

3. Ici, la densité, la transparence, l'opacité, etc, ne sont pas des données physiques mais des informations statistiques : la densité correspond à la présence, l'opacité à l'occlusion. On peut prendre l'albedo égal à l'opacité.

4. cf [17] pour les interactions lumineuses dans le cas général.

deux raisons : il évite le recours au suréchantillonnage contre l'aliasing, et il fournit des données précalculées à la bonne échelle en fonction de la taille apparente à l'image.

Isoler et filtrer l'aspect photométrique d'une forme est lié à la modélisation de l'anisotropie : un voxel contient une densité mais pas de forme, donc la réflectance doit être codée explicitement (soit donnée, soit extraite de la géométrie échantillonnée). En fait, aux petites échelles, les comportements photométriques sont beaucoup plus importants que les formes (c'est pourquoi la séparation des aspects spatiaux et photométriques rend le modèle des texels efficace : la géométrie à représenter sous forme de texture volumique est faiblement échantillonnée, et les réflectances sont paramétrisées de façon compacte). Pour coder un comportement de réflectance arbitraire, une primitive de réflectance peut être modélisée par de la 'micro-géométrie', que l'on peut se représenter comme une sorte de 'cristallisation' (la forme est trop petite pour être vue, sauf au travers de son comportement photométrique), comme des sphères [2], ou des cylindres [14]. Mais une primitive de réflectance particulière n'autorise que des objets particuliers (des cylindres, par exemple), et plus encore interdit le filtrage si la 'somme' d'un ensemble de ces primitives ne peut être approximé par la même primitive. La réflectance peut aussi être modélisée par la *BDRF* complète (*Bi-Directional Reflectance Function*, qui indique la quantité de lumière venant d'une direction donnée et réfléchie vers une direction donnée) [4], ou par une approximation de la fonction locale de répartition des normales [6] (Fournier 'filtre la géométrie' codée en carte de perturbation de normales).

Notre approche a des points communs avec toutes ces méthodes, dans la mesure où nous avons choisis une micro-primitive déformable pour approximer la fonction locale de répartition des normales, ce qui permet de modéliser des formes quelconques, et surtout de filtrer ces primitives. La filtrabilité est un point crucial pour nous, dans la mesure où le multiéchelle n'est envisageable que si l'on sait filtrer les données.

3 Extension Multiéchelle des Textures Volumiques

La structure de donnée multiéchelle naturelle pour stocker des voxels est l'octree. Chaque niveau d'un tel arbre correspond à une résolution, et la zone d'un voxel 'père' à un niveau donné est associée au niveau au dessus à huit voxels 'fils' qui se partagent cette zone. Les informations gardées par les voxels à bas niveau doivent ainsi représenter le comportement précalculé de ce qu'il se passe dans cette zone, dont on a une information plus précise stockée dans les fils (jusqu'à une certaine profondeur, correspondant à la résolution de l'octree). Ceci est obtenu en 'filtrant' les informations en un sens à définir. Un effet de bord de la représentation est l'économie de stockage[5] : les zones vides ou constantes n'ont pas besoin d'être représentées à la résolution maximale, cela peut être fait au niveau où les voxels correspondent à la taille de la région en question.

La façon la plus simple d'effectuer le calcul du rendu sur une structure de donnée multiéchelle avec un lancer de rayon consiste à utiliser une approche 'rayons coniques' simplifiée. On connait alors l'épaisseur des rayons durant le parcours, ce qui permet de choisir les données précalculées de résolution correspondante. Une reconstruction correcte de l'information est ainsi obtenue en utilisant un unique rayon par pixel[6]. Ceci est un point essentiel : le coût est maintenant lié à la complexité visible, et non plus à la complexité des données.

Voici les diverses tâches à accomplir lors de la phase de construction :

- Comme dans l'implémentation de Kajiya, les surfaces se composent de faces bilinéaires sur lesquelles les texels seront plaqués (les texels sont donc une déformation trilinéaire du volume de référence). Pour chaque face bilinéaire, un pointeur de texel indique quel motif (i.e. quel volume de référence) utiliser ; les matières de la surface et du texel sont définies selon le modèle de Phong (couleurs spéculaire, diffuse et ambiante). On indique également si le texel doit être posé sur ou sous la face.

- Les formes à modéliser sont 'peintes' dans le volume de référence pour chaque type de texture volumique, soit par un graphiste, soit automatiquement par un échantillonneur de géométrie, en stockant dans chaque voxel la densité (ou présence) et la micro-primitive locale (qui modélise la fonction locale de répartition des normales). On peut utiliser de nombreuse méthodes à cette fin, nous en dirons un mot un peu plus bas[7].

5. Les octrees ont d'abord été créés à cette fin, en fait...

6. En fait les choses ne sont pas si simples, dans la mesure ou les informations préfiltrées sont cubiques, alors que les régions couvertes par un rayon conique ne sont pas aussi régulières ; il reste donc un léger aliasing.

7. Nous utilisons essentiellement un script qui décrit les primitives, qui sont définies par leur fonction de distance.

- Les niveaux successifs de l'octree sont calculés par filtrages itérés. On moyenne les densités, tandis que les primitives qui modèlent la réflectance sont additionnées comme décrit plus loin [8].

- L'octree est compressé : les zones vides ou constantes étant correctement représentées par quelques voxels pères à basse résolution, on détruit les voxels fils identiques (ce qui économise à la fois la mémoire et les calculs). Les informations peu contrastées sont simplifiées selon le même principe.

Voici les grandes lignes du calcul du rendu :

- Un lancer de rayon parcourt la scène à l'aide de rayons qui intersectent les surfaces.

- Comme pour l'implémentation de Kajiya, quand une surface est recouverte par un texel, le rayon doit y pénétrer [9] et commuter en 'mode volumique'. cette opération est effectuée dans le volume de référence, après avoir calculé la portion de trajectoire correspondante. Les directions de la lumière et du point de vue sont également reportées dans l'espace du volume de référence.
  On a en outre besoin de l'épaisseur du rayon, afin d'estimer le niveau de l'octree à considérer (le lancer de rayon devient alors un lancer de cônes simplifié). Cette estimation constitue un problème à part entière ; quand les déformations sont légères, nous choisissons la taille de voxel qui coïncide grossièrement avec l'épaisseur du rayon.

- Le rendu volumique est effectué récursivement le long du rayon, jusqu'à ce que l'on atteigne la taille minimum de voxel (en fonction de l'épaisseur du rayon et de la résolution des données), sans échantillonnage stochastique.

- On calcule l'illumination locale : la réflectivité est obtenue à partir des caractéristiques de la primitive locale et des directions de la lumière et du point de vue (nous décrivons plus loin comment) ; l'occlusion locale, qui modifie l'opacité locale, est obtenue de la même façon.

- La lumière incidente est estimée en lançant un rayon d'ombrage, qui ne tient compte que de l'atténuation entre la source de lumière et le point courant (cependant les primitives locales modulent l'atténuation au passage).

Comme on l'a dit plus haut, le multiéchelle implique le filtrage : le modèle de réflectance doit être suffisamment générique pour rendre compte de l'aspect moyen d'un ensemble d'éléments par un élément du même type (cette généricité entraine au passage que l'on peut modéliser une grande variété de formes), mais il ne doit pas être trop complexe à cause du coût de stockage (les paramètres sont stockés dans chaque voxel !).

Dans notre approche, le lancer de rayons coniques, le plaquage de texture, la déformation trilinéaire, la traversée de l'octree et le rendu volumique sont implémentés de façon relativement habituelle, c'est pourquoi on s'intéresse ici essentiellement à l'amélioration du modèle de réflectance, qui conditionne l'implémentation de toute l'approche multiéchelle.

Il nous reste donc à :

- Spécifier une bonne primitive en terme de généralité, calculabilité, et propension au filtrage. cf chapitre 3.1.

- Rendre cette primitive : cela consiste à évaluer la quantité totale de lumière réfléchie vers l'observateur par la primitive. cf chapitre 3.2.

- 'Filtrer' la primitive : cela revient à 'résumer' la géométrie à une échelle donnée afin de construire les niveaux de représentation plus grossiers (ceci ne peut qu'être une approximation, dans la mesure où le filtrage rigoureux de la géométrie est impossible, à cause des effets d'occlusion et d'ombrage que l'on néglige). cf chapitre 3.3.

La méthode en elle-même est maintenant décrite. Mais pour en faire un outil utilisable, la construction du volume et le plaquage des texels doivent être décrits. Dans la mesure où l'approche

---

Les primitives sont peintes récursivement dans l'octree, ce qui permet de n'utiliser que la mémoire nécessaire (les zones constantes à l'intérieur où à l'extérieur des formes sont fortement compressées).

8. Les voxels peuvent être marqués une fois calculés : cela permet un recalcul partiel si l'on modifie ultérieurement une petite partie du volume, par exemple dans un but d'animation.

9. ceci dans le cas où le texel est sous la surface, sinon c'est la surface qui a été spécialement construite à cet effet au dessus des texels que l'on intersecte d'abord.

est suffisamment générique pour s'appliquer à toutes sortes d'applications, de nombreuses directions doivent être explorées : images 3D type tomographie, systèmes de particules, scripts descriptifs, échantillonnage de géométrie, fonctions de bruit procédural [8] et hypertextures [13], etc . Pour ce qui concerne le plaquage, nous nous ramenons comme Kajiya aux faces bilinéaires, ce qui est plutôt limité (mais il existe de nombreuses autres méthodes de plaquage). Une autre voie consiste à remplir directement l'espace sans déformation, c'est à dire à utiliser les texels comme des 'compacteurs de géométries' plutôt que comme un modèle de matériau à peau épaisse.
Accessoirement, il faut aussi s'occuper des perturbations des texels, si l'on souhaite éviter la répétition exacte du motif 3D. On peut modifier la taille et l'orientation des normales ; on peut aussi (et c'est complémentaire) agrandir, déplacer ou faire tourner le volume de référence pour chaque texel [10].

3.1 Choix de la micro-primitive

Conserver toute la BDRF n'est pas utile ici, dans la mesure où l'anisotropie de la réflectance est due aux variations locales de la surface dans la zone d'un voxel (ce qui revient à dire qu'on peut la représenter par la fonction de répartition des normales). Une telle fonction doit elle-même être condensée en peu de paramètres, à cause du stockage dans chaque voxel. La décomposition en pics de Phong proposée par Fournier [6] est efficace, mais encore trop chère pour des volumes. A l'opposé, les modèles de sphère ou de cylindre sont trop spécifiques (et un ensemble de deux cylindres ne se comporte pas avec la lumière comme un unique 'cylindre moyen'). Nous avons donc choisis une primitive géométrique, l'ellipsoïde [11], ce qui est moins général qu'une approximation aux moindres carrés de la répartition des normales, mais dispose de suffisamment de degrés de liberté : avec six paramètres, elle est capable d'approximer a minima une sphère, un cylindre (un ellipsoïde allongé), un bout de plan (un ellipsoïde aplati), et toutes les formes intermédiaires.
Nous verrons plus loin qu'un ensemble d'ellipsoïdes peut être approximé par un ellipsoïde du point de vue du comportement photométrique, ce qui revient à dire qu'il est filtrable.

En fait, on a besoin de moins de six paramètres [12] : comme on l'a dit plus haut, la 'micro-primitive' est une sorte de 'cristallisation', une forme sans dimension ni position, dont le seul but est de réfléchir la lumière. Dans tous les calculs de rendu ou de filtrage, une 'normalisation' est donc nécessaire afin de traiter les différentes formes avec le même poids. Nous avons choisi la surface apparente moyenne (nous sommes dans un contexte de visibilité, la largeur ou le volume ne sont donc pas significatifs), évaluée dans la direction des trois axes.

Il y a plusieurs façon de choisir la nature de ces paramètres : Deux représentations utiles sont une base avec trois longueurs, et les coefficients d'une forme quadratique. Nous avons choisi de stocker la première, qui est plus commode lors de la construction, et peut rapidement conduire à la seconde représentation.

On doit garder en tête que l'on manipule deux niveaux de formes lors du dessin dans le volume (sans compter celle de la surface sur laquelle les texels sont plaqués) : la forme globale (qui peut elle-même être un cylindre ou un ellipsoïde), et la micro-primitive locale stockée dans chaque voxel. Pour les objets ordinaires, la dernière représente la contribution géométrique de la forme pour chaque voxel. Mais on peut aussi bien modéliser les objets anisotropes, en prenant la forme locale (la cristallisation) plus ou moins indépendante de la forme globale (cf figure 1).

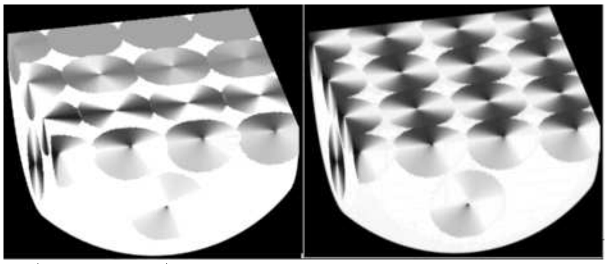

FIG. 1 - *aluminium brossé* (un seul texel). Les primitives locales sont des cylindres concentriques se transformant progressivement en sphères (à gauche), ou des cylindres radiaux (à droite). Les 'cylindres' sont modélisés par des ellipsoïdes allongés.

10. Ceci n'est pas possible si un objet est à cheval entre deux texels ! (Ce qui arrive pour les motifs 'raccord', qui sont dessinés dans un espace cyclique.)

11. Il faut retenir que cet ellipsoïde n'est pas utilisé directement pour indiquer la forme de la fonction de répartition des normales (comme le ferait un pic de Phong), mais comme une approximation locale de la surface du point de vue de la répartition des normales.

12. On stocke néanmoins les six paramètres pour ne pas avoir à refaire de calcul.

3.2 Rendu des primitives

On doit calculer la réflectance d'une micro-primitive, c'est à dire le ratio global d'énergie reçue des sources de lumières (en fonction de l'atténuation) et réfléchi (selon le modèle de Phong) vers l'observateur par tout l'ellipsoïde. Les interactions avec l'environnement sont résolues par le rendu volumique, qui évalue la quantité de lumière incidente, et cumule l'illumination[13] et l'opacité le long de chaque rayon (une primitive locale modifie également l'opacité, en fonction de sa surface apparente dans la direction du rayon[14]). Le problème consite donc essentiellement à évaluer la réflection globale sur un ellipsoïde, connaissant sa forme et la position de la source de lumière et de l'observateur (c'est à dire à intégrer la BDRF).

L'intégration à la surface des coniques a rarement une formulation exacte. Nous utilisons une intégration numérique optimisée[15], en échantillonnant la surface apparente afin d'obtenir la réflectance. L'optimisation consiste à sélectionner directement les échantillons qui tombent sur l'ellipsoïde, en évaluant le rectangle circonscrit à l'ellipse apparente (on l'obtient à l'aide de la forme quadratique). De plus, on n'échantillonne que la moitié de l'ellipse, les normales symétriques étant obtenues en utilisant les propriétés de l'ellipsoïde.

Pour évaluer la lumière incidente (ou l'ombrage, ce qui revient au même), un rayon d'ombrage doit être lancé vers les sources de lumière afin de tester les occultations et de collecter l'atténuation. Nous adoptons l'hypothèse de faible albedo[16] ; on ne s'occupe donc que de l'opacité le long de cette trajectoire, dans la mesure où les réflections secondaires sont omises[17].

3.3 Filtrage des primitives

3.3.1 Notion intuitive de filtrage

Les considérations sur les ellipsoïdes doivent être menées dans l'espace dual de distribution des normales plutôt que dans l'espace géométrique, dans la mesure où les primitives locales se manifestent par leur comportement en réflectance plutôt que par leur forme. On doit garder en tête que les comportements qui semblent corrects du point de vue de la géométrie conduisent souvent à des comportements incorrects dans l'espace des normales, et réciproquement.

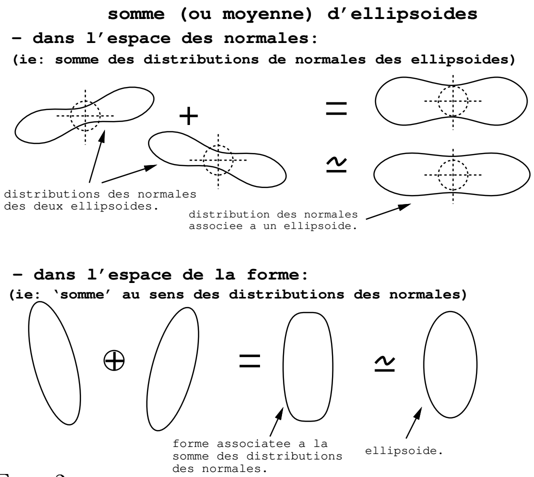

FIG. 2 - Construction de la *somme* de deux ellipsoïdes.

Comme on l'a dit lors du choix de cette primitive, la somme de la répartition des normales de deux ellipsoïdes *ressemble* à la répartion des normales d'un ellipsoïde, l'approximation étant la

13. L'illumination locale vaut lumière×albedo×réflectance, où l'albedo est lié à l'opacité, lui-même lié à la densité comme vu plus haut.

14. ce qui est un aspect de l'anisotropie: un cylindre fin qui occupe un voxel à densité fixée bloque moins de lumière dans la direction de l'axe que dans la direction orthogonale.

15. L'analyse de l'exécution nous a montré qu'en dépit du coût important du rendu local, d'autres opérations comme le parcours du volume par les rayons et d'autres tâches de base consomment une plus grande proportion de temps (sans parler du coût d'intersection des faces bilinéaires).

16. L'hypothèse de faible albedo est commode est fréquemment choisie, mais un peu faible pour rendre compte des phénomènes naturels.

17. Le calcul va d'autant plus vite que les formes sont opaques. Seule la surface d'un objet dense est visible, et donc même un objet fractal se présente à l'observateur comme une surface (éventuellement discontinue). A l'opposé, chaque point d'un volume translucide est vu ; les rayons le traversent complètement, ce qui occasionne nombre de rayons d'ombrage. En conclusion, les formes (opaques) complexes sont les bienvenues.

plus mauvaise pour deux cylindres orthogonaux, mais assez bonne pour des primitives voisines. Cela se représente aisément en 2D (cf figure 2).

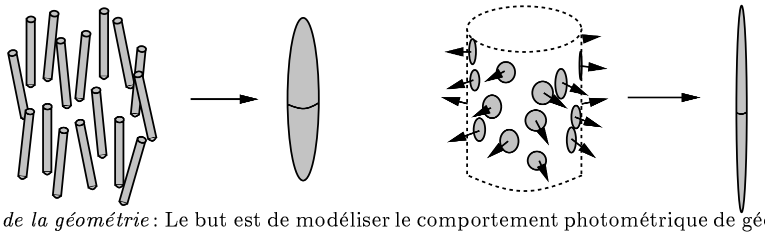

FIG. 3 - *Filtrage de la géométrie* : Le but est de modéliser le comportement photométrique de géométries complexes vues de loin. Pour déterminer le comportement en réflectance d'un voxel associé à une région de l'espace, l'approche consite à trouver une forme simple dont la distribution des normales (qui contrôle les réflections) approxime celle de la géométrie originale. Ainsi, la distribution moyenne des normales d'un vaste ensemble de cylindres dont la direction fluctue est proche de la répartition des normales d'un ellipsoïde, et la distribution moyenne des normales d'un vaste ensemble d'éléments de surface sur un cylindre est précisemment celle du cylindre, qui peut être approximé par un ellipsoïde allongé (car les distributions de normale sont voisines).

Pour les volumes remplis de façon homogène, les choses se passent bien : le filtrage réitéré d'une prairie (l'herbe étant modélisée par des cylindres avec une orientation qui fluctue) donne, au niveau le plus grossier (quand toute la prairie tient dans un seul voxel), un ellipsoïde orienté selon la direction moyenne, et d'autant plus fin que les variations sont faibles. Le filtrage réitéré d'un vaste cylindre (dont la section occupe plusieurs voxels, avec des primitives locales plates) donne, au niveau le plus grossier (quand le voxel devient assez large pour englober toute la section), une primitive quasi-cylindrique (cf figure 3).

Pour les volumes inhomogènes, l'approximation est réelle : filtrer avec cette méthode une zone qui contient deux cylindres orthogonaux conduit à une réflectance isotrope : l'orientation moyenne de deux axes orthogonaux est une absence d'orientation, il y a donc bien une perte d'information entre la somme et la 'moyenne'.

Nous avons vu en introduction que les effets de blocage (occultation et ombrage) étaient cruciaux car typiquement 3D, ce qui est perdu avec les méthodes de texturage ordinaires. Ici, les effets de blocage sont corrects dans la mesure où l'on dispose d'une information réellement 3D, mais les choses dégénèrent au filtrage[18] : quand on additionne les micro-primitives d'une région, les parties cachées deviennent en partie visibles (la somme étant purement géométrique, sans notion de point de vue). A nouveau, les choses se passent bien pour les volumes homogènes, dans la mesure où l'auto-similarité résiste au filtrage. Faute de quoi il y a une approximation significative : il ne peut y avoir d'effet de blocage en dessous de la taille du voxel, où la photométrie remplace la géométrie[19]. Un autre point concerne les ombres : quand la densité stockée dans les voxels décroît trop au cours du filtrage, les ombres se comportent comme dans un brouillard diffus plus que comme sur un objet compact. Bien qu'il faille prendre garde à ces effets, en pratique les choses se passent relativement bien.

3.3.2 Définition du filtrage

Comme suggéré plus haut, *additionner* les ellipsoïdes revient à choisir la forme dont la fonction de répartition des normales est la plus proche de celle associée à la somme des fonctions de répartitions de normales des ellipsoïdes à fusionner. Pour éviter une longue optimisation aux moindres carrés, nous avons choisi une façon directe de calculer cette 'somme' (ce qui est raisonnable dans la mesure où la primitive est simple).

Plusieurs méthodes peuvent êtres utilisées, mais n'ont pas la même qualité : on peut par exemple interpoler séparemment les bases des ellipsoïdes (à l'aide des quaternions) et les longueurs. Mais cela donne de mauvais résultats, parceque les longueurs devraient intervenir dans l'interpolation des axes, et parcequ'on introduit des informations parasites en codant l'ellipsoïde par une base : la direction des vecteurs n'a pas de sens, et peut conduire à des résultat faux si elle intervient dans le calcul (ainsi, la somme de deux ellipsoïdes identiques modélisés par deux bases opposées devrait être conservative).

18. Les techniques de filtrage de Fournier sont précisemment faites pour traiter correctement le problème du filtrage, avec des effets de blocage corrects.

19. L'approximation significative apparaît au niveau de l'octree où deux objets différents modélisés dans le texel sont fusionnés en un seul voxel. Ceci est lié au fait que la fonction de répartition des normales perd l'information de position, et oublie ainsi qu'une seule normale peut correspondre à deux zones différentes d'un objet non convexe, qui peuvent se cacher l'une l'autre.

A l'aide de la forme quadratique Q associée aux ellipsoïdes, il est possible d'expliciter la fonction de répartition des normales (qui peut être interprétée comme la densité de probabilité f d'avoir une normale N dans une direction donnée). Celle-ci est égale au Jacobien de la bijection entre la surface de l'ellipsoïde et l'espace dual des normales (sur la sphère de Gauss) :
$f_Q(N) = det(Q^{-1})/(N.Q^{-1}.N)^2$
le problème consiste à trouver Q tel que f_Q soit le plus proche possible de $f_{Q_1} + f_{Q_2}$. L'étude de Q montre que les choses sont 'quasi-additives' en Q^{-1} : si l'on pose $g_{Q^{-1}} = f_Q$, on a $g_{\lambda.Q^{-1}} = \lambda.g_{Q^{-1}}$, la somme de deux formes identiques est donc conservative (le facteur λ disparaîtra à la normalisation). Si l'on écrit le développement limité de $g_{Q_1^{-1}} + g_{Q_2^{-1}}$ pour deux ellipsoïde identiques tournés de θ l'un par rapport à l'autre, on trouve $g_{Q_1^{-1}+Q_2^{-1}} + O(sin^2(\theta))$ (on sait que les choses se passent mal pour les angles élevés entre ellipsoïdes).

Nous avons donc choisi comme ellipsoïde moyen celui dont la forme quadratique inverse est la moyenne des formes quadratiques inverses des ellipsoïdes à fusionner, les formes quadratiques s'obtenant aisément à partir des bases et des longueurs[20].

La figure 4 illustre le filtrage successif des primitives locales : la série d'images a l'apparence d'opérations de lissage successives de l'image originale, alors qu'elles sont juste le rendu de géométries successivement filtrées lors de la phase de construction.

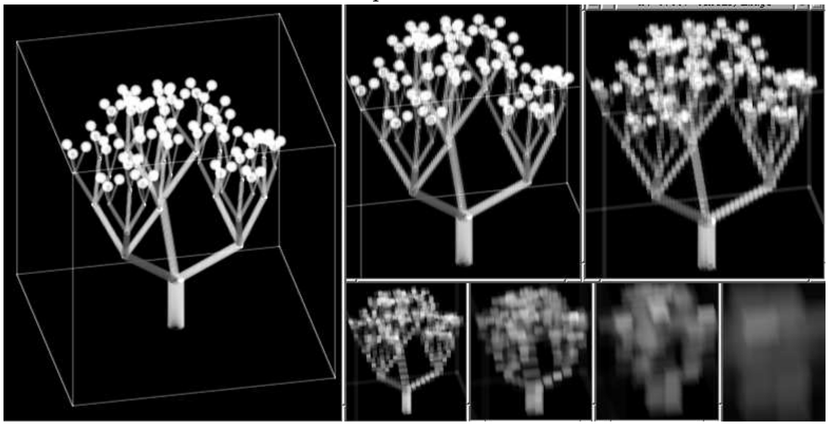

FIG. 4 - Volume 256^3 et niveaux inférieurs de l'octree, jusqu'à 4^3. Sur une Indigo², la construction dure 8 s, et le rendu 20 s pour la première image, 12 et 10 pour les deux suivantes, à la résolution 444x444. Le point important est que les images de bas niveau ont l'apparence du lissage de l'image complète, alors qu'elles sont obtenues directement : le coût de rendu est maintenant lié à la complexité visible (ce qui est vu), et non plus à la complexité géométrique (ce qui est connu).

4 Résultats

Afin d'illustrer l''illusion géométrique' créée par les texels, la figure 5 présente des données complexes dans un texel unique.

L'étrange image de buissons et de sphères sur la figure 6 montre le comportement du filtrage progressif : les texels de buisson contiennent 2000 feuilles avec une résolution 256^3 ; ils pavent directement l'espace dans un volume de 50x500x1 texels. Bien qu'il n'y ait qu'un seul rayon par pixel, il n'y a pas d'aliasing. Le calcul à résolution 444x444 prend 14 minutes sur une station de travail Silicon Graphics *Indigo*².

Avec l'image de simili velour en figure 8, on voit apparaître les effet d'anisotropie liés à la répétition de petits objets : chaque poil obéit au modèle de Phong, mais selon les orientations on peut voir l'accumulation du haut des poils illuminés (en haut de la bosse), ou l'ombre à la base des poils (à droite de la bosse). L'ombre sur la gauche est plus ordinaire, la lumière venant de droite.

Les trois derniers couples d'images montrent divers types de texels à résolution 128^3, et leur plaquage sur des géométries comprenant 100 à 1000 'patches' bilinéaires (quand les texels sont trop près, les voxels sont parfois visibles individuellement). A la résolution vidéo (768x576), le calcul prend de 5 à 20 minutes, le coût d'intersection géométrique en représentant une large part. Dans l'image de la forêt (figure 9), on peut remarquer que le niveau de détail utilisé pour le texel peut juste correspondre à la distance minimale voulue par l'utilisateur. Sur l'image des buildings (figure 11), le volume de référence montre que l'information de réflectance peut donner l'illusion

20. Bien sûr, les ellipsoïdes à additionner sont pondérés par leur densité.

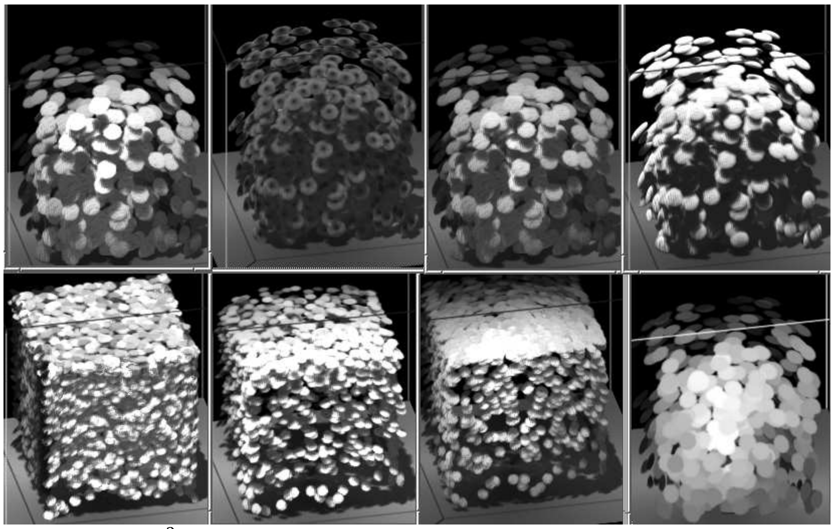

FIG. 5 - buissons taillés (256^3 voxels, jusqu'à 2000 feuilles), avec divers types de primitives locales. Le temps de calcul sur une *Indigo*^{2} varie entre 30 et 40 secondes. (Le texel est entouré d'un cadre, dont on voit l'ombre sur le sol, lui-même légèrement anisotrope.)

d'une forte résolution (cependant les détails inférieurs au voxel tendent à devenir transparents). Le 'tore à fourrure' de la figure 10 utilise un texel cyclique. Sur l'image texturée, il reste un peu d'aliasing (les texels sont très déformés afin de 'coiffer' les poils).

5 Conclusion

Dans ce papier, nous nous sommes interessés à l'amélioration du modèle de réflectance dans l'approche des textures volumiques de Kajiya. Nous utilisons une primitive locale paramétrée, l'ellipsoïde, qui est capable de modéliser de nombreux types de formes (i.e. de nombreux types de réflections), mais surtout qui autorise le filtrage, et ainsi la représentation multiéchelle en octree de données volumiques.

Cette seule propriété étend largement le cadre applicatif de la technique :

- Avec un rendu adaptatif et ainsi un coût faible, les texels permettent de fournir les informations à diverses échelles[21], et peuvent être utilisés partout comme une économie pour la géométrie répétitive. Plus que l'accélération, cela rend accessible le calcul d'animations de scènes très complexes.
- Comme la primitive de réflectance est relativement générique, une grande catégorie de données peut être modélisée.
- En évitant le plus gros de l'aliasing, les texels peuvent être vus comme un moyen de rendre correctement les petites géométries. De plus, cela est fait à faible coût.
- Le concept de plaquer de la géométrie 3D sur de la géométrie 3D est en soit une façon commode de construire des scènes complexes.
- L'anisotropie est disponible, par effet de bord de la méthode.

Il y a néanmoins quelques défauts :

- L'approche plaquage s'applique à des motifs répétitifs, et suppose que l'on soit capable de définir un plaquage subissant des déformations raisonnables.
- Il peut sembler naturel d'imbriquer deux 'textures géométriques', ou de mettre un objet réel et un texel au même endroit, mais cela est assez difficile à implémenter.
- Le texturage suppose qu'il n'y ait pas de mouvements dans le motif lui-même, faute de quoi le motif de référence doit être recalculé chaque fois (ce qui peut toutefois être acceptable).

21. Depuis l'échelle où tout le texel se projette en un seul pixel jusqu'à l'échelle où un simple voxel apparaît comme un cube.

- Comme pour les textures classiques, on ne peut se ramener exactement aux données précalculées : il faut choisir entre le flou et l'aliasing (ou suréchantillonner) dans les mauvais cas (quoi qu'il en soit, les choses se passent infiniment mieux qu'avec de la géométrie répétitive conventionnelle !).
- Le filtrage ne traite pas parfaitement des effets de blocage, ce qui peut limiter l'échelle d'applicabilité dans certains cas.

Dans les travaux à venir, nous souhaitons améliorer la technique de filtrage, et accélérer le rendu en évaluant la réflection sans intégration numérique. De plus, il faut pouvoir estimer finement la taille de voxel à utiliser lors du rendu afin de supprimer l'aliasing résiduel sans flouter les données. Et bien sûr, pour obtenir un outil réellement productif, il faut compléter l'implémentation par plusieurs méthodes de peinture du volume (dont l'échantillonnage automatique de la géométrie), et par des méthodes de plaquage indépendantes des faces, comme les textures paramétriques usuelles.

FIG. 6 - Les buissons et les sphères pavent directement l'espace sur une région 50x500x1. On peut remarquer qu'il n'y a aucun aliasing à l'horizon, alors qu'on n'utilise qu'un seul rayon par pixel (le coût de rendu est de 14 minutes). La résolution est ici de 444x444 ; toutes les images suivantes sont à la résolution vidéo (768x576).

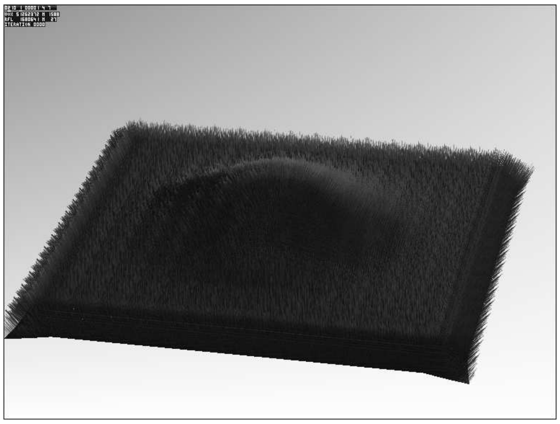

FIG. 7 - *(à gauche)* Une prairie sur 1404 patches bilinéaires. Le texel contient 16 brins d'herbe à section en 'V' décroissante, dessinés dans un volume cyclique ; sa résolution est de 128x128x128 (compression 91%).

FIG. 8 - *(à droite)* Sorte de velour. Les cylindres sont tous orthogonaux à la surface, ce qui entraine une anisotropie à grande échelle.

FIG. 9 - *à gauche*: un seul texel at 128x128x128 (compressé à 92%), destiné à être vu de loin. *à droite*: plaquage sur une colline de 578 faces bilinéaires (23 minutes de rendu, dont la moitié pour l'intersection des patches).

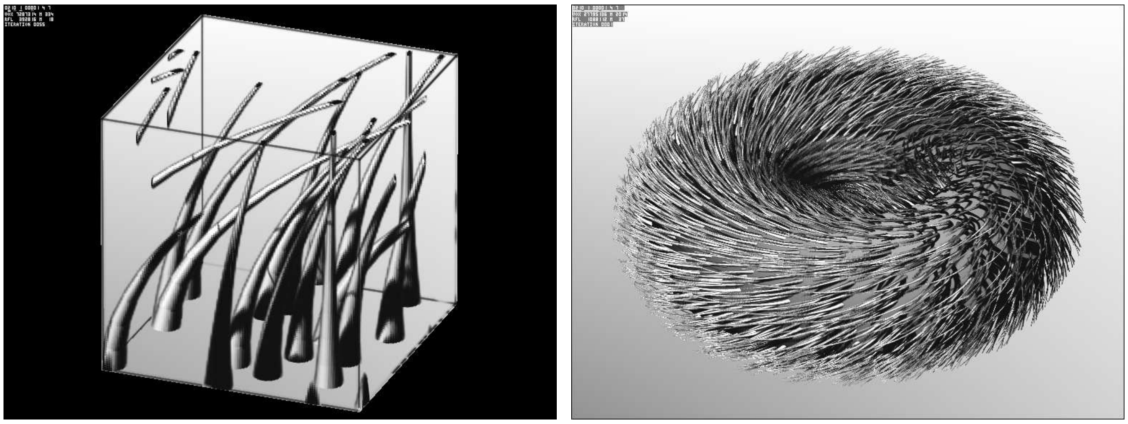

FIG. 10 - *à gauche*: poils dessinés cycliquements dans un volume128x128x128 (compressé à 93%). Le rendu du texel seul prend 3.5 minutes. *à droite*: plaquage sur un tore composé de 240 patches bilinéaires (12 minutes).

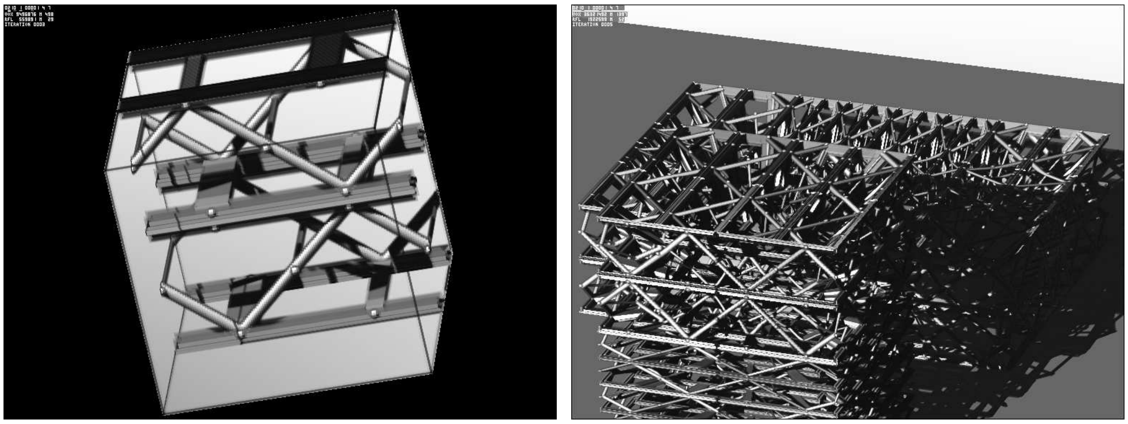

FIG. 11 - *à gauche*: texel contenant des éléments de construction à résolution 128x128x128 compressé à 92% (les éléments sont parfois plus fins qu'un voxel, et tendent alors à devenir transparents). *à droite*: plaquage sur des objets cubiques (81 faces bilinéaires, 14 minutes).

Références

[1] John Amanatides and Andrew Woo. A fast voxel traversal algorithm for ray tracing. In G. Marechal, editor, *Eurographics '87*, pages 3–10. North-Holland, August 1987.

[2] J. F. Blinn. Light reflection functions for simulation of clouds and dusty surfaces. In *Computer Graphics (SIGGRAPH '82 Proceedings)*, volume 16(3), pages 21–29, July 1982.

[3] James F. Blinn. Simulation of wrinkled surfaces. In *Computer Graphics (SIGGRAPH '78 Proceedings)*, volume 12(3), pages 286–292, August 1978.

[4] Brian Cabral, Nelson Max, and Rebecca Springmeyer. Bidirectional reflection functions from surface bump maps. In Maureen C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21(4), pages 273–281, July 1987.

[5] Phillippe de Reffye, Claude Edelin, Jean Francon, Marc Jaeger, and Claude Puech. Plant models faithful to botanical structure and development. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22(4), pages 151–158, August 1988.

[6] Alain Fournier. Normal distribution functions and multiple surfaces. In *Graphics Interface '92 Workshop on Local Illumination*, pages 45–52, May 1992.

[7] James T. Kajiya and Timothy L. Kay. Rendering fur with three dimensional textures. In Jeffrey Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23(3), pages 271–280, July 1989.

[8] John-Peter Lewis. Algorithms for solid noise synthesis. In Jeffrey Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23(3), pages 263–270, July 1989.

[9] Gavin S. P. Miller. From wire-frames to furry animals. In *Proceedings of Graphics Interface '88*, pages 138–145, June 1988.

[10] F. Kenton Musgrave, Craig E. Kolb, and Robert S. Mace. The synthesis and rendering of eroded fractal terrains. In Jeffrey Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23(3), pages 41–50, July 1989.

[11] J. W. Patterson, S. G. Hoggar, and J. R. Logie. Inverse displacement mapping. *Computer Graphics Forum*, 10(2):129–139, June 1991.

[12] Ken Perlin. An image synthesizer. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19(3), pages 287–296, July 1985.

[13] Ken Perlin and Eric M. Hoffert. Hypertexture. In Jeffrey Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23(3), pages 253–262, July 1989.

[14] Pierre Poulin and Alain Fournier. A model for anisotropic reflection. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24(4), pages 273–282, August 1990.

[15] W. T. Reeves. Particle systems – a technique for modeling a class of fuzzy objects. *ACM Trans. Graphics*, 2:91–108, April 1983.

[16] William T. Reeves and Ricki Blau. Approximate and probabilistic algorithms for shading and rendering structured particle systems. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19(3), pages 313–322, July 1985.

[17] Holly E. Rushmeier and Kenneth E. Torrance. The zonal method for calculating light intensities in the presence of a participating medium. In Maureen C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21(4), pages 293–302, July 1987.

[18] Mikio Shinya. Hierarchical 3D texture. In *Graphics Interface '92 Workshop on Local Illumination*, pages 61–67, May 1992.

[19] Frédéric Taillefer. Fast inverse displacement mapping and shading in shadow. In *Graphics Interface '92 Workshop on Local Illumination*, pages 53–60, May 1992.