


Rapport de Stage
Diplôme d'Études Approfondies

Numérisation et Visualisation 3D de Surfaces Réelles

Larue Frédéric
Année 2003 / 2004

Encadrement:
J.M. Dischler
P. Tellier
J.P. Chambard

Université Louis Pasteur
4 rue Blaise Pascal
67000 Strasbourg

Remerciements

Je tiens à remercier pour leurs contributions :

- mes encadrants, Jean-Michel DISCHLER et Pierre TELLIER, pour avoir veillé au bon déroulement de ce stage,

- Jean-Pierre CHAMBARD pour son acceuil au sein de la société HOLO3 (Saint-Louis, 68), et pour avoir mis à notre disposition le matériel nécessaire à la réalisation de ce projet,

- Eric COLON de la société HOLO3 pour son support technique.

Table des matières

Chapitre 1

Introduction

Historique de la synthèse d'images

L'informatique graphique est né d'une demande de certains secteurs industriels et scientifique (CAO, simulation, ...) pour lesquels il est tout aussi primordiale de pouvoir visualiser des données que de faire des calculs. Les représentations sous forme de graphiques et de courbes sont couramment utilisées et presque indissociables de domaines comme l'économie, ou l'analyse statistique par exemple. Dans un contexte plus scientifique, l'utilisation de modèles géométriques en trois dimensions pour caractériser les données peut être très utile : en biologie ou en chimie par exemple, une molécule peut être modélisée à l'aide d'un squelette soumis à des contraintes d'angle et de distance.
Les premiers algorithmes développés dans le domaine de la synthèse d'images proposaient l'affichage et le rendu de quelques primitives géométriques de base. L'algorithme de tracé de droites (*Bresenham*, 1962) permettait alors un affichage sous la forme d'une armature en fils de fer (rendu filaire). L'utilisation des volumes en synthèse d'images a introduit la notion de **représentation par les bords** (B-Rep), qui ne tient compte en fait que de la partie visible de l'objet. Bien que des modèles de surfaces paramétriques aient été introduit très tôt (*Pierre Bezier*, 1970), les algorithmes de **remplissage de polygônes** ont contribué à favoriser les modèles à base de facettes (**maillages**).
L'une des composantes essentielles dans la génération d'images est la couleur. Initialement réduites à des affichages bi-chromatiques, les machines ont évolué pour proposer petit à petit un panel de couleurs de plus en plus important. Travailler sur la couleur peut contribuer à améliorer considérablement le rendu d'objets tri-dimensionnels. Dans la réalité, nos deux yeux distinguent chacun une image différente et notre cerveau les recompose pour nous permettre de voir dans les trois dimensions. Mais contrairement à la perception que nous avons du monde qui nous entoure, une image calculée sur un écran 2D d'ordinateur ne présente aucune notion de profondeur. L'éclairage est alors un bon moyen d'apporter une impression de volume et de relief. Les variations de couleur observées nous renseignent au premier coup d'œil sur la forme et la disposition des différents objets d'une scène (voir fig.1.1).

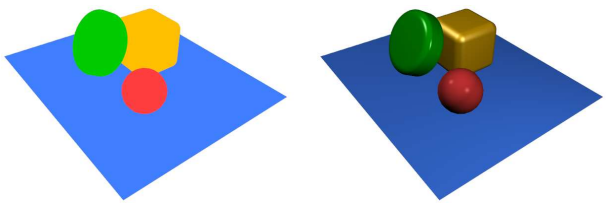


Fig. 1.1 – A gauche : couleur uniforme par objet. **A droite** : application d'un modèle d'illumination. L'impression de volume ressort bien.

L'éclairage se calcul à l'aide d'un **modèle d'illumination** qui consiste a se donner des sources de lumière et un modèle d'interaction lumière / matière. Le premier modèle d'illumination, basé sur la loi de *Lambert*, calcule une quantité de lumière réémise pour chaque face du volume en considérant son orientation par rapport à la source. En 1971, *Henri Gouraud* introduit une technique de lissage qui calcule cette fois l'illumination pour chaque sommet et applique ensuite un dégradé de couleur sur les faces par interpolation bi-linéaire. Ce lissage est d'ailleurs encore aujourd'hui très largement utilisé, et fait partie depuis longtemps des fonctionnalités élémentaires des cartes graphiques. *Bui-Tuong Phong* propose, en 1975, un autre modèle, défini de manière empirique, qui tient compte de la refléction spéculaire, c'est à dire la composante de la lumière qui varie en fonction de la position de l'observateur (les reflets lumineux sur les objets plastique par exemple). Il propose également sa propre méthode de lissage, plus adapté à la restitution de ces reflets.
Le **photo-réalisme** constitue l'un des principaux axes de recherche en synthèse d'images. Il vise à générer des images dont la qualité visuelle est si proche d'une photographies que l'œil humain n'est pas capable de les discerner l'une de l'autre. Les applications sont multiples.
Dans le milieu automobile par exemple, l'élaboration d'un prototype coûte cher. L'outil informatique est de plus en plus utilisé pour réduire ce coût en simulant la résistance des matériaux, le comportement du véhicule, la sécurité de l'habitacle, etc. Mais outre ces considérations techniques, le design est aussi un point très important, qui contribue grandement à l'attrait suscité chez le futur client. Avant d'entrer réellement dans la phase de réalisation, il est plus sûr de proposer un modèle numérique du véhicule sur lequel débattre de ce qui est bon et de ce qu'il faudrait retravailler. La qualité visuelle de ce modèle doit bien rendre compte de ce que sera le véhicule à la sortie de l'usine. Son apparence doit être réaliste. D'autres secteurs s'intéressent également au rendu réaliste, comme l'aéronautique avec les simulateurs de vol, qui doivent paraître suffisament réels pour procurer les même sensations que dans un appareil situé à dix mille mètres d'altitude.
Le **lancé de rayons**, développé en 1980, est la première méthode à introduire le concept de photo-réalisme. Elle constitue en quelque sorte une simulation du comportement de la lumière, en suivant sa trajectoire à travers la scène. Cette méthode a permis de représenter un grand nombre de phénomènes observables dans la nature, comme les ombres portées, les effets de miroir (réflexion des objets les uns dans les autres) ou la réfraction des matériaux transparents (déviation du rayon lumineux due à un changement de milieu).
Toute la compléxité du photo-réalisme réside dans la modélisation des interactions entre la lumière et la matière. Ces interactions se manifestent à différents niveaux d'échelle. Au niveau microscopique (de l'ordre du nanomètre), une surface n'est jamais parfaitement plane. Elle présente toujours des aspérités, inperceptibles à l'œil nu ou au touché, mais qui déterminent la **réflectance** de cette surface, c'est à dire la manière dont les rayons lumineux incidents sont réfléchis. D'un point de vue physique, la reflectance est essentiellement fonction de la position de l'observateur et de la source. Elle doit vérifier certaines propriétés, comme la consevation de l'énergie ou la réciprocité (interchangeabilité du point de vue et de la source lumineuse). La quête du réalisme ne peut pas se baser sur des modèles qui ne respectent pas ces propriétés, comme le modèle de *Phong*, par exemple, pour lequel la conservation n'est pas vérifiée. C'est dans cette optique qu'ont été élaborés des modèles basés sur la physique, comme celui de *Cook-Torrance* (1981) qui utilise une distribution de micro-facettes pour représenter cette micro-géométrie.
Au niveau macroscopique, d'autres phénomènes apparaissent, comme les ombres, générées par occlusion, et les interréflexions, résultats de la contribution des sources secondaires aux échanges globaux d'énergie lumineuse. Ces effets, qui constituent ce que l'on appelle **l'illumination globale**, sont typiquement ce que l'on peut observer à l'échelle de la scène toute entière. Ils ont d'ailleurs très tôt suscité l'intérêt des chercheurs. Les techniques de **radiosité**, apparues pour la première fois en 1984, décomposent la scène en petits éléments de surface, et calculent les

interactions lumineuses entre chacun d'eux, en tenant compte de la quantité d'énergie qu'ils réémettent à travers la scène. Une autre technique, le **photon mapping**, basée sur le lancé de rayons, bombarde la scène d'un grand nombre de particules lumineuses (photons) partant des sources et transportant chacune une quantité d'énergie qui va être absorbée, au fur et à mesure, par rebonds successifs.
On retrouve également ces effets à une échelle intermédiare, au niveau des fins détails géométriques encore perceptibles à l'œil nu, comme les mailles d'un tricot, ou les aspérités d'une éponge, par exemple. Cette méso-structure, comme on l'appelle, a longtemps été ignorée pour être reléguée au rang de simple détail pictural, representée alors comme de simples images collés sur l'objet par des algorithmes de **placage de textures**. On sait pourtant aujourd'hui qu'elle joue un rôle prépondérant dans le réalisme des images de synthèse. Pour rendre compte de ce fin relief, certaines techniques ont vu le jour : le **bump mapping**, introduit la première fois en 1978, modifie l'illumination de la surface en changeant l'orientation des normales. Plus récemment, les **Bidirectional Texture Functions** (BTF) représentent dans un espace à six dimensions la variation d'une surface sous différentes combinaisons d'illumination et d'observation, en tenant compte de tous les phénomènes d'occlusions et d'interréflexions.
Malgré toutes les techniques existantes, simuler tout ce qui se passe en terme d'interactions lumineuses reste extrèmement difficile. Il faudrait en théorie descendre jusqu'à l'échelle atomique pour rendre compte de tous les phénomènes. Face à tant de complexité, les chercheurs ont envisagé d'utiliser la réalité comme une source d'information en plus de s'en servir comme référentiel pour quantifier le réalisme. Différents dispositifs d'acquisition ont été développés dans ce but. Certains permettent de récupérer la géométrie d'un objet et d'autres, travaillant avec des éclairages contrôlés, proposent d'en mesurer les propriétés colorimétriques et de réflectance. Ainsi, les interactions lumière / matière, plutôt que d'être simulées, sont capturées à partir d'échantillons réels et utilisées par les logiciels pour la synthèse d'images photo-réalistes. Mais les modèles analytiques ne sont pas abandonnés pour autant. Ils ont l'avantage d'offrir une représentation compacte, à partir d'un nombre réduit de paramètres. Le modèle de *Lafortune* (1997), par exemple, proposé pour représenter une fonction de réflectance par une approximation non-linéaire, est très utilisé dans les travaux ayant trait à la visualisation d'objets numérisés, car ces paramètres peuvent être facilement déterminer à partir d'échantillons issus de mesures réelles. Face à une efficacité probante, les techniques d'acquisition suscitent aujourd'hui un intérêt grandissant au sein de la communauté scientifique, et sont devenues un enjeu important dans le domaine de la visualisation réaliste.

Cadre et objectifs du travail

Le projet *Archivage et Micro Identification 3 Dimensions* (AMI3D), réunissant autour de la société HOLO3, centre de transfert technologique basé à Saint-Louis (68), le Laboratoire des Sciences de l'Image, de l'Informatique et de la Télédétection (LSIIT) de l'Université Louis Pasteur de Strasbourg (67) et le Laboratoire d'Informatique Graphique et d'Ingénierie de la Vision (LIGIV) de l'Université Jean Monnet de Saint-Etienne (42), a pour objectif la réalisation d'un dispositif destiné à établir une empreinte numérique fiable d'œvres d'art en vue d'un archivage. Les paramètres à considérer pour cette empreinte sont sa forme (géométrie en trois dimensions), sa couleur et la texture (granularité) de sa surface. Ce dispositif devra être automatisé pour limiter autant que possible les interventions humaines, ou au moins les réduire à des manipulations simples, accessibles à des non-spécialistes. A un stade plus avancé, le dispositif devra être capable d'établir une empreinte suffisament précise pour déterminer, de manière automatique toujours, l'authenticité d'une l'œuvre.
Pur produit des nouvelles technologiques de l'imagerie, le matériel de numérisation 3D mis à disposition par HOLO3 fonctionne sur le principe de la lumière structurée, qui consiste à analyser

la déformation de franges lumineuses projetées sur un objet. Cette méthode permet de calculer les coordonnées de dizaines de milliers de points d'une surface en quelques secondes seulement. Mais ce dispositif ne s'occupe en fait que de l'acquisition géométrique. L'équipe *d'Informatique Géométrique et Graphique* (IGG) du LSIIT a été sollicitée pour proposer une solution concernant l'acquisition des autres informations. Dans le cadre de ce stage, nous allons donc tenter de mettre au point une méthode pour capturer et restituer l'apparence d'objets réels sur écrans d'ordinateurs.

Plan du rapport

La visualisation d'objets numérisés est un domaine vaste. Dans un premier temps, nous passons en revue dans le chapitre 2 certains travaux qui ont déjà été fait dans le cadre de la numérisation et de la visualisation. Nous voyons d'abords dans la section 2.1 le fonctionnement du dispositif de numérisation 3D fourni par HOLO3, et discutons de techniques relatives au problème de l'acquisition géométrique (reconstruction de maillages incomplets, recalage d'informations chromatiques, etc). Nous voyons ensuite différentes techniques de visualisation, certaines destinées à l'affichage en temps réel de modèles complexes (section 2.2), et d'autres plutôt axées sur la visualisation photo-réaliste (section 2.3).
Nous abordons ensuite, dans le chapitre 3, les choix qui ont été fait pour atteindre le but fixé. L'acquisition d'un objet réel nécessite de capturer sa géométrie ainsi que son apparence (en terme de couleur ou de texture). L'apparence est généralement mesurée à partir d'images. Nous devons donc nous pencher sur le problème de faire coïncider une information en deux dimensions avec la géométrie, en modifiant éventuellement le processus d'acquisition (recalage, section 3.3).
Le résultat de ce type de mesure constitue souvent une masse de données très importante. Cela pose un cetain nombre de problèmes, à commencer par l'espace mémoire nécessaire pour la stocker. Nous définissons (section 3.4) les structures mises en œuvre pour représenter cette information de manière à la rendre exploitable par un outil de visualisation interactif. La représentation doit être compacte, et doit surtout nous permette de reconstruire sans trop de complexité l'apparence qui a été mesurée sans la dénaturer.
Enfin, nous voyons au chapitre 4 les résultats obtenus, discutons de la robustesse de la méthode, et concluons sur les perspectives envisageables pour ce projet.

Chapitre 2

Etude bibliographique

2.1 Acquisition

Nous discutons ici des techniques en lien, plus ou moins direct, avec l'acquisition de données 3D. Dans le cadre du projet AMI3D, la société HOLO3 nous propose un dispositif de numérisation dont le fonctionnement diffère des techniques plus connues d'acquisition au laser. Nous détaillons le principe de la projection de franges qui permet d'acquérir un objet en quelques secondes seulement. Nous voyons aussi quelques techniques de post-traitement des données, comme la complétion de la surface dans le cas d'un maillage reconstruit de manière incomplète ou des recompositions de textures à partir de clichés réels.

2.1.1 Acquisition de formes 3D par projection de franges

Le dispositif de numérisation est composé de deux appareils : un projecteur envoyant des raies de lumière dont la variation d'intensité décrit une fonction sinusoïde, et une caméra numérique prenant des clichés de la scène. Les deux appareils sont placés de telle sorte qu'il y ait un décalage d'angle θ entre les directions de projection et d'observation. Grâce à ce décalage, le capteur observe des raies de lumière déformées par la géométrie de l'objet, comme on peut le voir sur la figure 2.1. C'est en analysant cette déformation que l'on va pouvoir déterminer l'information de profondeur de la scène [BILLARD98].

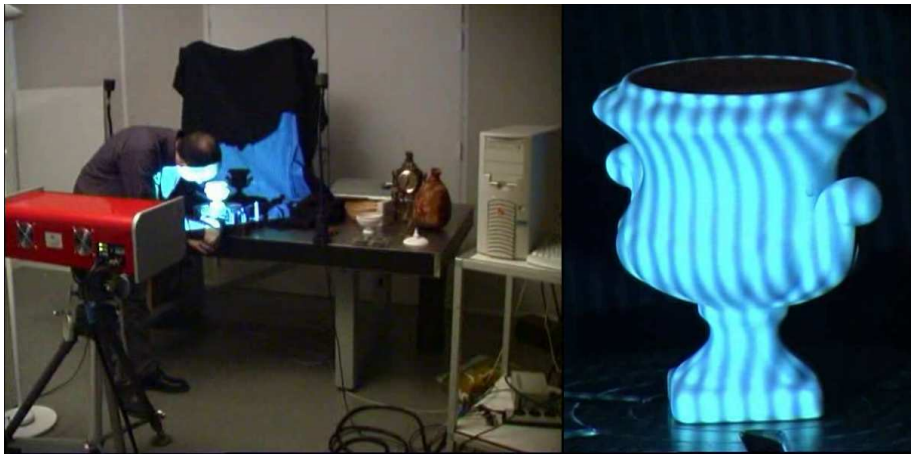


Fig. 2.1 – A gauche : un technicien place l'objet devant le dispositif de numérisation. **A droite** : les franges observées par le capteur sont déformées par la géométrie de l'objet.

Relation profondeur / phase

Une sinusoïde est une fonction périodique définie par :

$$f(x) = A\cos(Fx + \varphi) \tag{2.1}$$

avec : A : amplitude du signal,
F : sa fréquence,
φ : sa phase.

Le schéma ci-dessous met en évidence la relation qui existe entre la profondeur et la phase des franges (sinusoïde) observées. Plus précisément, on calcule un déphasage par rapport à une phase calculée lors du calibrage du dispositif pour un plan de référence dont la distance à la caméra est connue. Les différentes relations géométriques du système nous montrent que le décalage de

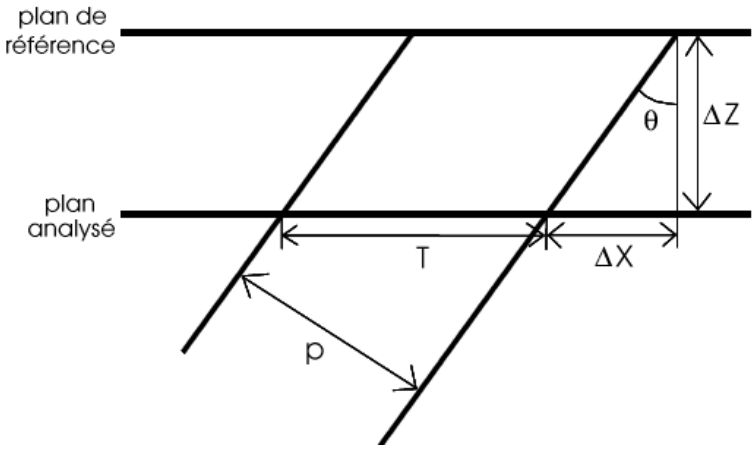


Fig. 2.2 – Relation entre profondeur et décalage de phase.

phase $\Delta\varphi$ est égal à :

$$\Delta\varphi = \frac{2\pi\Delta X}{T} = \frac{2\pi\ \cos\theta}{p}\Delta X \tag{2.2}$$

avec p : période de la sinusoïde projetée,
T : période de la sinusoïde observée.

et comme on a $\tan\theta = \frac{\Delta X}{\Delta Z}$, on peut déduire la relation suivante entre la profondeur et la phase observée :

$$\Delta Z = \frac{\Delta X}{\tan\theta} = \frac{p}{2\pi\sin\theta}\Delta\varphi \tag{2.3}$$

Calcul de la phase

L'intensité lumineuse observée peut être décrite par l'équation suivante :

$$I = I_0\ (1 + m\cos\varphi) \tag{2.4}$$

avec I_0 : intensité moyenne,
m : contraste des franges,
φ : phase.

Pour résoudre cette équation à trois inconnues (I_0, m, φ), il nous faut au moins un système de trois équations. On peut obtenir un tel système par acquisition de plusieurs images dans lesquelles on introduit un décalage de phase connu. Avec trois images I_1, I_2, I_3 obtenues avec des déphasages respectifs de 0, $\frac{2\pi}{3}$, $\frac{4\pi}{3}$ on résout un tel système par la formule suivante :

$$\varphi = \arctan \frac{\sqrt{3}(I_3 - I_2)}{2I_1 - I_2 - I_3} \tag{2.5}$$

L'acquisition par caméra numérique introduit automatiquement du bruit que l'on peut réduire en calculant la phase à partir d'un plus grand nombre d'images. Cette autre formule permet de faire ce calcul pour un nombre n d'images, chacune ayant subit un décalage de $\frac{2\pi}{n}$:

$$\varphi = \arctan \frac{-\sum_{i=1}^{n} I_i \sin\left[\frac{2k\pi}{n}\ (i-1)\right]}{\sum_{i=1}^{n} I_i \cos\left[\frac{2k\pi}{n}\ (i-1)\right]} \tag{2.6}$$

Démodulation

Si l'on projette une sinusoïdale dont la période ne couvre pas l'intégralité du champ de vision, la phase se répète sur l'image modulo 2π. On cherche donc à démoduler cette phase pour obtenir la carte de phase finale, le problème étant de déterminer si un changement brutal correspond à un saut de phase ou à une discontinuité dans la géométrie de l'objet.
En projetant un réseau dont le pas est assez grand pour que la phase observée soit comprise entre 0 et 2π, on élimine les sauts de phase. Malheureusement, on observe un bruit important dû au nombre limité de niveaux de gris utilisés pour projeter les franges.
Pour obtenir une plus grande précision, il faut projeter des réseaux de pas plus petits. Lors de la démodulation, pour déterminer si un changement brutal correspond à un saut de phase ou non, on consulte une image de phase calculée pour un réseau de pas plus grand.

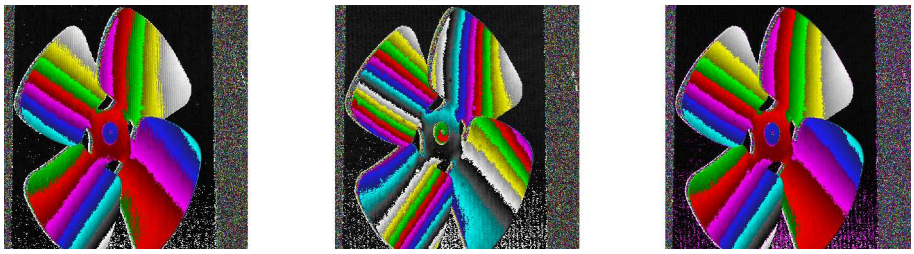


Fig. 2.3 – A droite **(a)** : 40 raies projetées, au centre **(b)** : 80 raies projetées, à gauche **(c)** : démodulation de **(b)** par **(a)**. La phase calculée en **(c)** est moins bruitée qu'en **(a)**.

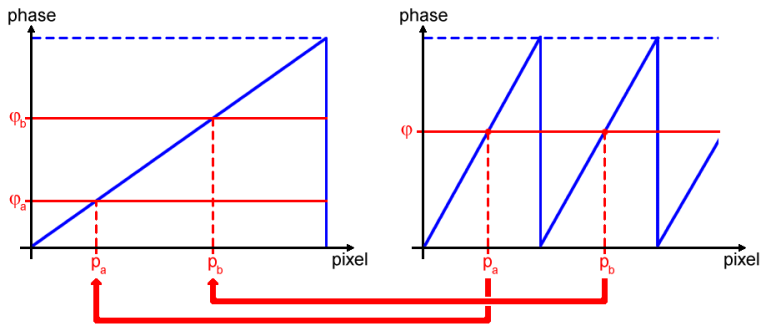


Fig. 2.4 – La phase de droite est démodulée à l'aide d'un réseau de pas plus grand.

Calcul de géométrie

En disposant de deux cartes de phases, l'une pour un plan de référence et l'autre où l'on a introduit l'objet à numériser, on calcule le décalage de phase par soustraction des deux images. La profondeur de l'objet peut alors être déterminée à partir de l'équation 2.3. Le résultat obtenu est une image de points géométriques où chaque pixel contient un triplet de coordonnées (x, y, z).

2.1.2 Complétion de surface

Les données calculés par les dispositifs de numérisation 3D consistent en un nuage de points très dense. Des techniques permettent de reconstruire un maillage à partir de ces points, comme l'algorithme de *ball pivoting* [BERNARDINI99]. Mais la surface reconstruite à partir de données numérisées est souvent incomplète. Des trous peuvent apparaître dans certaines régions inaccessibles, et dont la géométrie est trop complexe pour les combler avec une simple triangulation (polygones qui s'entrecoupent). Certaines applications nécessitent d'avoir un maillage bien formé, comme des simulations physiques par exemple pour lesquelles l'intérieur et l'extérieur de la surface doivent être clairement définis.
L'algorithme décrit dans [DAVIS02] utilise des techniques de traitement d'images pour reconstruire proprement la surface et de manière cohérente.

Principe

La surface est d'abord discrétisée dans une grille de voxels. En utilisant une fonction de distance d_s définie par :

$$d_s \begin{cases} = 0 \text{ si le voxel est sur la surface,} \\ < 0 \text{ si il est en dessous de la surface,} \\ > 0 \text{ si il est au-dessus de la surface.} \end{cases}$$

On peut représenter le maillage par l'isosurface $d_s = 0$. Initialement d_s n'est calculée que pour une fine bande autour de la surface, en prenant comme valeur 0, 1 ou -1 selon la position du voxel.
Lorsque l'on se rapproche d'un trou, d_s perd en exactitude du fait de sa valeur relative au maillage manquant. On introduit alors un indice de confiance $w_s > 0$, valant 1 autour de la surface, et décroissant selon la proximité d'un trou. On va maintenant chercher à étendre cette fonction à toute la grille en diffusant sa valeur de proche en proche.

Diffusion

L'algorithme utilise deux volumes de voxels : le volume source (d_s, w_s) défini précédemment, et un volume de diffusion (d_i, v_i) où d_i est la distance calculée après i itérations, et v_i un ensemble de booléens indiquant la zone de validité (ampleur de la diffusion à l'étape i). L'algorithme est le suivant :

ALGORITHME - Completion de maillages par diffusion.

Fonction `Initialisation()`
$d_0 = d_s$
$v_0 = \begin{cases} TRUE \ si \ w_s > 0, \\ FALSE \ sinon. \end{cases}$
FinFonction

Fonction `Convolution(`d_i, v_i`)`

$(\hat{d}_i, v_i) = h * (d_{i-1}, v_{i-1})$
$d_i = w_s d_s + (1 - w_s)\hat{d}_i$
FinFonction

Fig. 2.5 – Différentes étapes de la diffusion. La ligne rouge réprésente le zéro de la fonction de distance

Lors de la convolution, les voxels non valides sont ignorés, et les valeurs du filtre h sont normalisées en conséquence. La convolution sur v_i est en fait une dilatation par élément structurant h. Une fois le processus de diffusion achevé, le maillage est reconstruit par marching-cube sur l'isosurface $d = 0$.

Amélioration

Connaissant le dispositif d'acquisition, la ligne de visée reliant l'observateur à chaque point du modèle est connue, ce qui permet d'ajouter une contrainte supplémentaire au processus de diffusion afin qu'il tienne compte du fait que certaines régions doivent rester visibles et ne peuvent être comblée. On force ainsi la cohérence du maillage reconstruit. Cette contrainte se traduit lors de la diffusion par une convolution supplémentaire dont le filtre est défini par la contrainte elle-même.

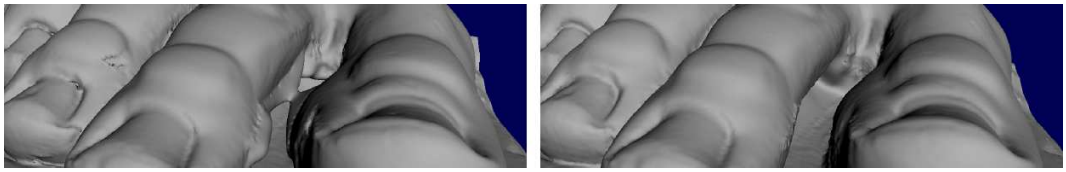


Fig. 2.6 – **A gauche** : sans les contraintes de lignes d'observation. Des «ponts» apparaissent entre les deux orteils. **A droite** : avec les contraintes. Le maillage reconstruit est plus cohérent

Avantages et inconvénients

L'utilisation de la diffusion volumétrique est bien adaptée à la reconstruction de parties manquantes d'un maillage. La méthode est simple et construit toujours un maillage bien formé. Le problème de la méthode se situe au niveau des temps de calcul, pouvant varier de manière importante selon les valeurs affectées aux nombreux paramètres dont l'alorithme dépend. Les tests ont été effectués sur certaines parties du modèle numérisé à partir de la statue de David lors du projet Michelangelo [LEVOY00]. L'appliquer sur toute sa surface demanderait des semaines, à moins de détecter préalablement les zones trouées pour pouvoir mieux cibler la diffusion.

2.1.3 Acquisition de textures

Acquérir la géométrie d'un objet, même à un niveau très fin, ne suffit pas à rendre compte de sont apparence réelle. Il manque un certain nombre d'informations comme la réflectance du matériau ou les détails chromatiques de l'objet. Ce sont ces détails picturaux que l'on souhaite

acquérir dans le cadre de ce stage.
L'algorithme [ROCCHINI99] s'exécute indépendamment du processus d'acquisition géométrique. Il s'agit en fait d'un post-traîtement qui travaille à partir du maillage numérisé et d'un certain nombre de clichés de l'objet réel. Une texture va être générée pour le maillage en recollant de manière cohérente ces clichés sur sa surface en tenant compte des distorsions induites par chaque point de vue (au niveau des angles rasant en particulier). Pour se faire, la méthode se décompose en trois étapes.

Assignation d'une image à chaque sommet

Pour chaque sommet v du maillage, on détermine l'ensemble des images valides (*valid image set*), c'est à dire l'ensemble des clichés dans lesquels v est visible et où il n'est pas un point de la silhouette. La position de l'observateur et les caractéristiques du capteur utilisé pour l'acquisition des clichés sont parfaitement connus. On peut donc déterminer la transformation à appliquer au maillage pour le projeter dans le plan de l'image. Cette projection couplée avec un tampon de profondeur nous permet de vérifier simplement que les deux conditions sont bien remplies.
A partir de cet ensemble, on détermine une mage cible (*target image*) pour le point v : il s'agit de celle pour laquelle l'angle entre la normale et la direction d'observation est minimal (afin de limiter au mieux les effets de distortions). Chaque sommet du maillage doit se voir assigner une *target image*. Son *valid image set* ne peut donc pas être vide, et c'est pour cette raison que les clichés doivent être pris de manière à ce que chaque région de la surface soit visible d'au moins un point de vue.

Classification des faces

Les faces sont alors réparties en deux groupes. Une face f est dite :
- ***interne*** si ses trois sommets (en supposant que le maillage soit triangulaire) ont la même *target image*. Dans ce cas, le cliché concerné est directement utilisé comme image de texture, et les coordonnées de texture de f sont déterminées par une simple projection de ses sommets dans le plan de l'image.
- ***frontière*** si au contraire au moins deux de ses sommets ont une *target image* différente. Ces faces définissent les jonctions entre plusieurs clichés. Il faut donc les traîter avec beaucoup de soin si l'on veut éviter d'obtenir des artéfacts néfastes lors de la visualisation.

Traîtement des *faces frontières*

Les *faces frontières* étant délicates à traîter, on va d'abord chercher à en minimiser le nombre. Pour se faire, un algorithme glouton choisit pour un sommet s une nouvelle *target image* parmi son *valid image set*. Si cette modification diminue le nombre de *faces frontières*, elle est conservée. Pour les *faces frontières* restantes, la texture est reconstruite par balayage du polygone à partir des trois clichés I_1, I_2, I_3 associés respectivement à ses trois sommets v_1, v_2 et v_3. Pour un point p de coordonnées barycentriques $p = \alpha v_1 + \beta v_2 + \gamma v_3$, la couleur dans la nouvelle texture est obtenue par $c_p = \alpha I_1(p) + \beta I_2(p) + \gamma I_3(p)$. En d'autres termes, une interpolation linéaire est effectuée entre les différents clichés.

Avantages et inconvénients

Cette technique, en faite assez intuitive, propose un moyen semi-automatique de reconstruire une texture pour un modèle à partir de données réelles. La reconstruction demande un nombre assez réduit de clichés (8 pour le vase), mais il revient à l'utilisateur de bien les déterminer afin qu'ils couvrent toute la surface de l'objet. Utiliser trop peu de vues peut entraîner des

Fig. 2.7 – A gauche : photographie du vase. **A droite** : modèle numérisé avec reconstruction de la texture. L'image souffre d'un manque flagrant de réalisme dû, entre autre, au fait que le processus ne tient pas compte des reflets spéculaires.

distortions très importantes pour les angles rasants. La reprojection dans l'espace image nécessite une correction sans laquelle les détails chromatiques ne coïncident pas parfaitement d'une image à l'autre pour un même polygone. Mais la méthode à pour principal défaut de considérer de façon systématique tous les objets comme des surfaces lambertiennes. Elle ignore totalement les effets d'illuminations dépendant de l'observateurs (reflets spéculaires) et les modèles reconstruit manquent cruellement de réalisme.

2.2 Visualisation interactive

La quantité de données accessibles en informatique est très importante, et la taille des écrans et des imprimantes ne permet pas aujourd'hui de toutes les représenter. Il devient de plus en plus difficile d'analyser ces données et d'en avoir une compréhension, à quelque niveau que ce soit. La visualisation interactive vise à trouver une représentation dans laquelle il est possible de naviguer et d'interagir avec ces données pour mieux les comprendre.
La clé de l'interactivité en synthèse d'images, c'est avant tout la vitesse. Pouvoir manipuler un objet ou naviguer dans une scène demande que ceux-ci s'affichent en temps réel, à savoir un minimum de 30 images par seconde. Nous voyons ici quelques méthodes qui ont été proposées dans ce but.

2.2.1 Point Sample Rendering

La numérisation de formes 3D est un processus qui nous fournit une information précise de la surface d'un objet, correspondant souvent à un nuage de points très dense. Le développement de ces techniques d'acquisition a incité à mettre au point des méthodes de rendu n'utilisant plus le polygone comme primitive, mais directement le point. Ce type de méthode présente au moins deux avantages : premièrement, il n'est pas nécessaire de reconstruire la surface (voir 2.1.2), et deuxièmement l'affichage est souvent plus rapide qu'un processus complet de remplissage de polygone.
Malheureusement, d'autres problèmes apparaissent : lors de la projection à l'écran (perspective ou orthographique) si l'ensemble des données n'est pas assez dense, deux points adjacents peuvent atteindre des pixels qui ne le sont pas forcément, introduisant ainsi des discontinuités (trous) dans l'image finale.

Cas de la projection orthographique

Il suffit ici que les données soient correctement échantillonnées. On considère un maillage de triangles équilatéraux construit à partir de notre ensemble de points. Idéalement, il faudrait que tous les pixels appartenant au maillage soit atteints par au moins un sommet. On utilise alors la condition suivante, garantie comme suffisante : la longueur des côtés des triangles dans l'espace image doit être inférieure à la dimension d'un pixel. Mais si l'on souhaite changer de point de vue, cette condition de dimension restreinte à l'espace image n'est plus suffisante.

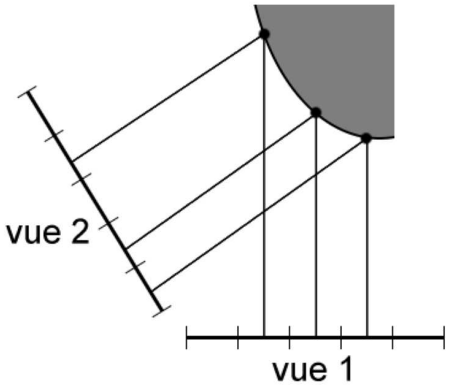


Fig. 2.8 – Echantillonnage dans l'espace image. Problème de discontinuité lorsque l'on change de point de vue.

En tenant compte de l'orientation des polygones par rapport au point de vue à partir duquel ils sont échantillonnés, on peut étendre la contrainte de manière à produire un maillage dont la taille des triangles **sur la surface** soit inférieure à la dimension d'un pixel.
Les problèmes de discontinuité sont alors résolus pour n'importe quel point de vue.

Cas de la projection perspective

Le problème est plus délicat, car l'écart entre les projetés de deux points dépend de leur distance au plan de projection. Echantillonner correctement les données ne suffit plus à prévenir l'apparition de trous. Il faut donc détecter ces trous pour pouvoir les combler.

Détection des trous

On construit un tampon de profondeur hiérarchique, en divisant les dimensions par deux à chaque niveau. Il faut maintenant déterminer dans quelles proportions chaque tampon recouvre les pixels de l'image.
A chaque tampon, on associe un maillage carré dont les sommets sont centrés sur les pixels. Pour un pixel p de l'image, on détermine les coefficients de recouvrement de chaque sommet du carré dans lequel il se trouve de la manière suivante : si le sommet est devant p, $C_k = 1$, sinon $C_k = 0$. Le coefficient de recouvrement de p est obtenu par interpolation linéaire. Le coefficient de recouvrement global C est calculé en faisant la somme (majorée par 1) des coefficients pour chacun des k tampons.
On définit alors le poids d'un pixel par $W = 1 - C$. Plus le poids sera faible, plus le pixel aura de chances d'appartenir au fond et non à la figure, et inversement.

Remplissage des trous

Dans une première étape, on commence par créer une hiérarchie d'images en divisant les dimensions par deux. La couleur du nouveau pixel est la moyenne des couleurs des quatre pixels

du niveau précédent, et le nouveau poids est la somme de leurs poids.
Les images sont recalculées successivement de la plus petite résolution à la plus grande, en descendant dans la hiérarchie. Un pixel p est reconstruit à partir des trois pixels p_1, p_2, p_3 les plus proches dans l'image de niveau supérieur.

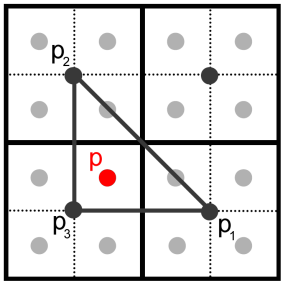


Fig. 2.9 – Calcul de la nouvelle couleur à partir des trois pixels les plus proches dans la résolution supérieure.

On calcule d'abord une couleur interpolée pour p à partir de p_1, p_2 et p_3 :

$$C_i = \frac{1}{4}C_1 + \frac{1}{4}C_2 + \frac{1}{2}C_3$$

et la nouvelle couleur de p est une moyenne pondérée de cette couleur et de sa couleur initiale :

$$C = W_p C_p + (1 - W_p)C_i$$

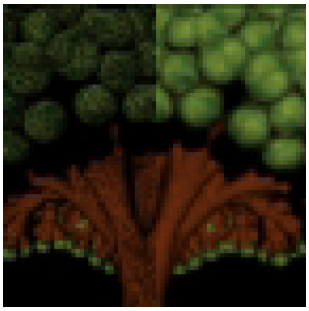


Fig. 2.10 – Avant (à gauche) et après (à droite) complétion des trous

Tests de visibilité

Pour accélérer l'affichage, on souhaite mettre en place un test suffisamment intelligent pour qu'il élimine rapidement un grand nombre de points.
On les regroupe de manière à découper la surface de l'objet en un ensemble de blocs. On considère pour chacun de ces blocs le demi-espace intérieur de l'objet.
Si l'observateur se trouve dans ce demi-espace, aucun point du bloc n'est visible. On peut simplifier cette région par le plus grand cône qu'elle puisse contenir, définissant ainsi pour chaque bloc un «cône de non visibilité».
Pour effectuer un *backface culling* rapide sur les blocs, on utilise des masques de visibilité. La sphère d'observation est divisée en 128 triangles, et chaque bloc B_i contient un masque M^i_{bloc} dont les 128 bits correspondent à chacun des triangles. Un bit est activé si le bloc est visible à partir du triangle associé.

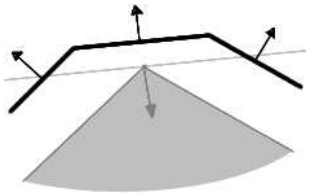
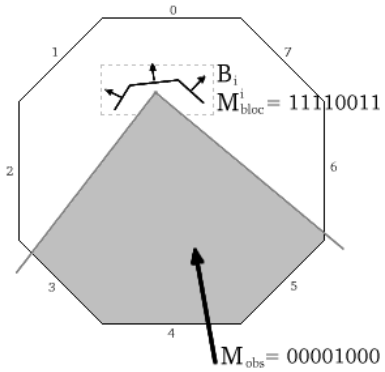


Fig. 2.11 – Demi-espace intérieur et cone de non visibilité.



On utilise un masque similaire M_{obs} pour indiquer dans quelle région de la sphère se trouve l'observateur. Le bloc B_i est alors éliminé si

$$M^i_{bloc} \ \& \ M_{obs} = 0$$

Pour plus d'efficacité, la zone écran est divisée en $2^n \times 2^n$ régions ayant chacune leur propre masque M^j_{obs}. Les blocs sont répartis dans zones écran à l'aide d'un arbre BSP.

Fig. 2.12 – Exemples de rendu basé points.

Avantages et inconvénients

Le rendu basé points permet de générer des images de grande qualité. La rapidité de l'algorithme vient essentiellement du fait qu'il fait appelle à un ensemble de procédures simples. La méthode présente en fait de gros problèmes lorsqu'il s'agit d'échantillonner des objets géométriques plus fins qu'un pixel, ce qui produit des images avec un crénelage important. Un autre problème est que, contrairement aux techniques à base de maillages, il n'est pas facile d'appliquer un quelconque type de lissage. Le *point sample rendering* produit souvent des images présentant des arêtes rugueuse, à moins d'utiliser des méthodes brutales de sur-échantillonnage.

2.2.2 QSplats

Les techniques de numérisation 3D produisent un très grand nombre de données qu'il faut être capable de restituer en un temps raisonnable si l'on souhaite développer un outil de visualisation

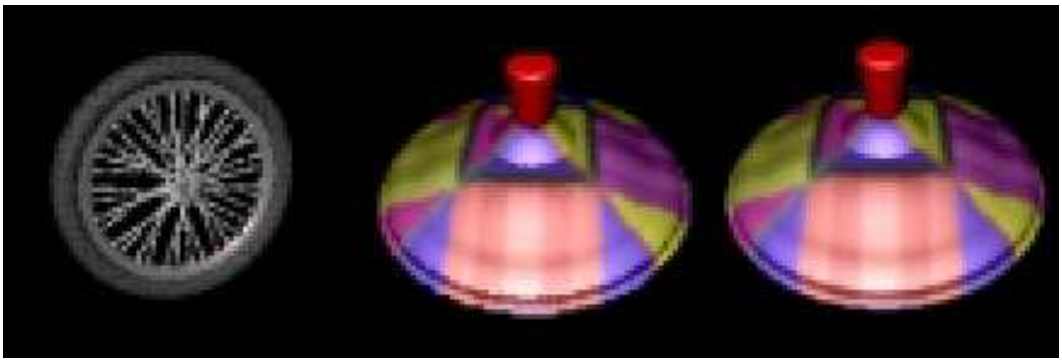


Fig. 2.13 – A gauche : problème d'*aliazing* lors de la représentation de primitives plus fine que la largeur d'un pixel. **Au centre** : problème d'arrêtes rugueuses rencontré dans le rendu basé points. **A droite** : lissage par une méthode brutale de sur-échantillonnage (×2).

interactif. Les niveaux de détails constituent un échantillonnage des données à différentes échelles, et le processus de rendu s'adapte en passant de l'une à l'autre selon les contraintes de performance fixées par l'utilisateur.
Comme nous en avons déjà parlé, travailler à partir d'un nuage de points a l'inconvénient de produire des discontinuité à l'écran. La méthode de rendu utilisée ici consiste à afficher, en chaque point, un motif couvrant, appelé **splat**, suffisamment grand pour combler ces discontinuités. L'algorithme des QSplat [RUSINKIEWICZ00] utilise ces splats pour l'affichage en les basant sur un modèle hierarchique. Il est très adapté à la visualisation de nuages de points très denses. Il a deux objectifs :

- reconstituer la surface lors de l'affichage pour éviter les discontinuités lorsque la densité de points à l'écran devient trop faible (problème de la projection perspective),
- proposer une méthode adaptative basée sur les niveaux de détails pour permettre la visualisation d'un grand nombre de données.

La méthode fonctionne sur une hiérarchie de sphères englobantes, avec comme information une position, un rayon, un cône de normales (direction + angle d'ouverture) et une couleur. Au plus bas niveau, chaque point du nuage a sa propre sphère dont le rayon est suffisamment grand pour ne pas laisser de trou entre deux sphères adjacentes.
Le reste de la hiérarchie est construit de manière récursive : les sphères sont regroupées par quatre pour créer de nouvelles sphères plus grandes dont les attributs sont calculés en fonction de ceux des fils.

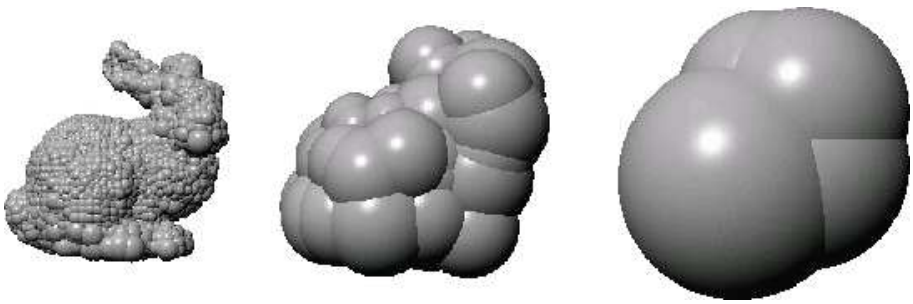


Fig. 2.14 – Le modèle du lapin affiché pour différents niveaux de détails (différentes profondeurs dans la hiérarchie).

Tests de visibilité

Pour accélérer le processus de rendu, deux tests sont effectués :

- *frustum culling* : la position de la sphère est testée par rapport à la pyramide de vision. Si elle est entièrement en dehors, le processus s'arrête pour cette branche de l'arborescence (élagage). Si au contraire elle est entièrement incluse dans la pyramide, le sous-arbre issu de cette sphère n'aura plus à être testé par la suite car il sera automatiquement visible.
- *backface culling* : de la même manière, si le cône de normales fait entièrement dos à l'observateur, on élague. Si au contraire il lui fait entièrement face, il est inutile poursuivre le test sur les fils.

Niveaux de détails

La problématique est ici de maintenir une fréquence d'affichage suffisante pour que l'application reste interactive. Il faut donc fixer une condition qui nous permette de décider quand stopper la récursivité lorsque le processus de rendu parcourt la hiérarchie. On considère le rayon de la sphère projetée dans le plan de la caméra. Lorsqu'il passe en dessous d'un certain seuil, on affiche une splat, et dans le cas contraire on parcourt les fils récursivement. Ce seuil dépend de la vitesse d'affichage observée par rapport à celle que l'on souhaite obtenir.

ALGORITHME - Affichage récursif de la hierarchie de sphères englobantes.

```
Fonction AfficherHierarchieSphere(b)
   Si noeud = feuille Alors
      b.afficherSplat()
   Sinon
      b.testVisibilité()
      Si ¬b.visible() Alors
         RetourFonction
      SinonSi b.rayon_sphère ≤ seuil Alors
         b.afficherSplat()
      Sinon
         AfficherHierarchieSphere(b.children)
      FinSi
   FinSi
FinFonction
```

Choix du motif

Plusieurs motifs peuvent être utilisés pour afficher les splats :

- **carré** : c'est le motif le plus simple et le plus rapide à afficher, mais pour les zones où la densité de points est plus faible ou pour celles à fort contraste on observe une pixélisation importante.
- **disque** : les contours et arêtes saillantes semblent moins bruités.
- **disque gaussien** : la surface est globalement adoucie, mais l'affichage est nettement plus lent. Cela nécessite en effet l'affichage de polygones texturés avec la transparence activée.
- **ellipse** : on affiche une ellipse verticale ou horizontale selon l'orientation de la normale, ou un disque si elle pointe vers l'observateur. Cette technique a pour avantage de ne pas trop déformer la silhouette de l'objet, mais elle risque de créer des discontinuités étant donné que l'un des rayons de l'ellipse est inférieur à celui de la sphère englobante.

Compression

Les QSplats [RUSINKIEWICZ00] proposent également une méthode de compression permettant de réduire l'espace de stockage nécessaire par une décompression à la volée. La position, le

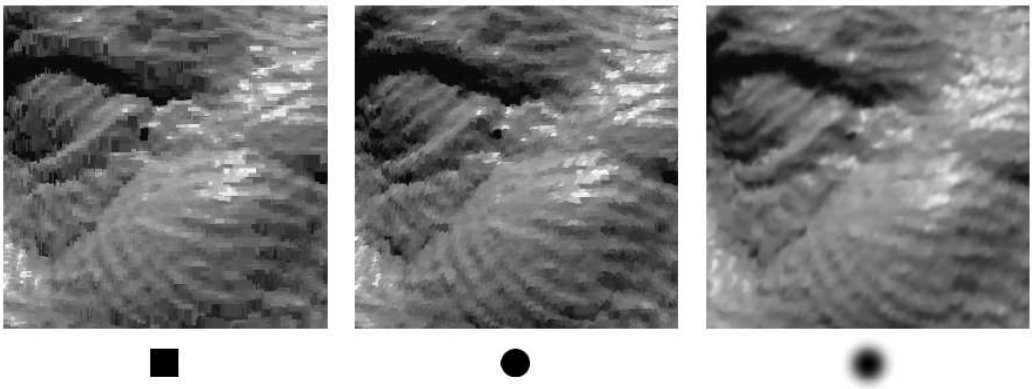


Fig. 2.15 – Affichages obtenus pour différents motifs de splats.

rayon, le vecteur normal et l'angle d'ouverture du cône sont codés sur un total de 32 bits. 48 bits sont nécessaires si on y ajoute la couleur, stockée alors sous la forme d'un triplet (R, V, B) avec 5 bits pour le rouge, 6 bits pour le vert et 5 bits pour le bleu.
Le rayon est codé sur 13 valeurs par rapport à celui du noeud père, c'est à dire qu'il peut varier entre $\frac{1}{13}$ et $\frac{13}{13}$ de sa valeur. De même pour la position, elle est codée comme un décalage par rapport à celle du père entre $\frac{1}{13}$ et $\frac{13}{13}$ de son rayon sur chacune des coordonnées x, y et z. Cette représentation permet de stocker ensemble la position et le rayon sur seulement 13 bits.
Les six faces d'un cube sont utilisées pour coder la normale. Chaque face est un tableau de 50×50. l'intersection de la normale avec le cube est calculée pour récupérer l'indice dans le tableau. La valeur des indices peut varier entre 1 et $6 \times 50 \times 50 = 15000$ et ne nécessite donc que 14 bits pour être représentée. Enfin, l'angle d'ouverture du cône est simplifié à quatre valeurs possibles pour pouvoir être stocké sur 2 bits, et 2 bit supplémentaires sont encore nécessaires pour décrire la structure de l'arbre (le nombre de fils).

Avantages et inconvénients

L'avantage de la hiérarchie QSplat est qu'elle propose un visualisation interactive de données très denses. L'utilisation de niveau de détails permet de régler très précisément la vitesse d'affichage, en utilisant une version plus grossière du modèle original. Les inconvénients sont typiquement ceux que l'on peut retrouver dans un système de rendu basé points. Pour une géométrie très découpée (haute fréquence), l'utilisation d'un maillage donne de meilleurs résultats. De même, lorsque l'observateur se rapproche trop de l'objet, la surface semble très crénelée, ce qui n'apparaitrait pas avec un maillage. Une méthode hybride de rendu basé points lié à un modèle polygonale pourrait être très efficace.

2.2.3 Maillages progressifs

Généralement, les niveaux de détails correspondent à un échantillonnage des données à différentes étapes d'une simplification. Les transitions d'un niveau à un autre peuvent parfois être brutales, et donc visuellement moins agréables car perceptibles.
Les *Progressive Meshes* (PM) [HOPPE00] sont un moyen efficace de contrôler les niveaux de détails d'un modèle. Elles définissent une chaîne de transformations, qui ajoutent ou retirent un seul sommet à la fois tout en conservant l'aspect global de l'objet. La transformation utilisée est définie de telle sorte qu'il est possible de créer un géomorphe (transition douce) pour passer d'un état à l'autre. Le raffinement du maillage est donc bien plus subtil.

Simplification de maillage - *edge collapse (ecol)* et *vertex split (vsplit)*

La simplification du maillage est basée sur une opération élémentaire : *ecol*, qui rapproche les deux sommets d'une arête jusqu'à l'avoir complètement écrasée. Chacune de ces opérations élimine donc du maillage un sommet et deux triangles. *ecol* est inversible, et l'on définit son inverse par l'opération *vsplit* qui divise un sommet en deux, recréant ainsi l'arête qui les joint et leurs deux triangles adjacents. Ces deux transformations sont illustrées sur la figure 2.16.

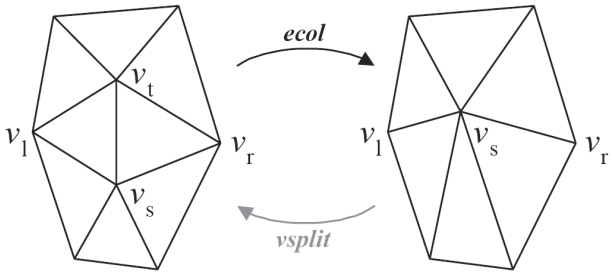


Fig. 2.16 – l'opération élémentaire de simplification *ecol* et son inverse *vsplit*

Une *PM* est construite en appliquant au maillage initial M_n une succession d'opérations *ecol* jusqu'à simplification en un maillage minimum M_0. Elle est ensuite stockée sous la forme d'un couple $(M_0, \{vsplit_1, ..., vsplit_N\})$, définissant en quelque sorte un maillage minimal adjoint des opérateurs nécessaires à sa décompression. Il est donc possible de choisir la qualité du maillage au nombre de sommets près.
Pour déterminer quelle transformation choisir parmi toutes celles possibles à chaque étape de la simplification, le maillage est identifié à un système énergétique et la transformation choisie est celle qui engendre la perte d'énergie la plus faible (minimisation de l'erreur d'une étape à l'autre).

Géomorphe

La succession de transformations définit une suite de maillages M_i. L'opérateur géomorphe correspond en quelque sorte à une interpolation linéaire entre deux maillages successifs M_i et M_{i+1}. Le géomorphe défini sur une seule opération peut aussi l'être sur un ensemble de transformations pour permettre une transition douce entre deux maillages quelconques M_i et M_j de la chaîne. On dispose donc d'un outil de contrôle du niveau de détail très précis.

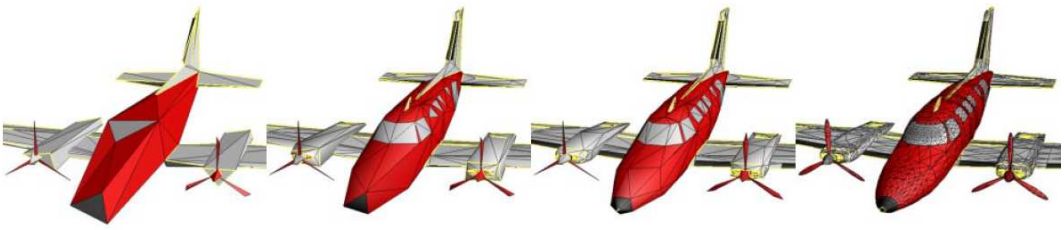


Fig. 2.17 – Exemple de simplification de maillage. **A droite :** le modèle original, composé de 13546 faces. **A gauche :** maillages simplifiés composés de 150, 500 et 1000 faces.

2.3 Visualisation photo-réaliste

Le photo-réalisme, comme nous l'avons déjà défini, consiste à produire des images d'une qualité visuelle proche d'une photographie. La visualisation photo-réaliste propose des méthodes

pour restituer à l'écran certaines quantités physiques, parfois issues de mesures réelles, comme la réflectance d'une surface ou les aspérités de sa méso-structure.
Les méthodes présentées ici nous offrent quelques solutions, en introduisant, par la même occasion, un certain nombre de notions fondamentales en photo-réalisme.

2.3.1 Acquisition de BRDF basée images

La BRDF (Bidirectionnal Reflectance Distribution Function) représente la réflectance (voir annexe B) du matériau, c'est à dire la quantité d'énergie lumineuse renvoyée en une direction donnée et pour un éclairage donné. La mesurer à partir d'objets réels est un processus souvent très long qui nécessite d'échantillonner, pour une multitude de points de vue, la couleur observée en un point avec un éclairage changeant. Des méthodes d'approximation permettent ensuite de reconstruire la fonction pour tout couple de directions d'incidence et d'observation à partir de ces seuls échantillons. *Marschner*, *Westin*, *Lafortune*, *Torrance* et *Greenberg* [MARSCHNER99] proposent une méthode permettant d'extraire d'un seul cliché les valeurs d'une BRDF pour plusieurs conditions d'éclairage et d'observation différentes. Ils exploitent la connaissance de la géométrie de l'objet pour déterminer la position et l'orientation de la lumière et de la caméra dans le repère local de chaque point de la surface.

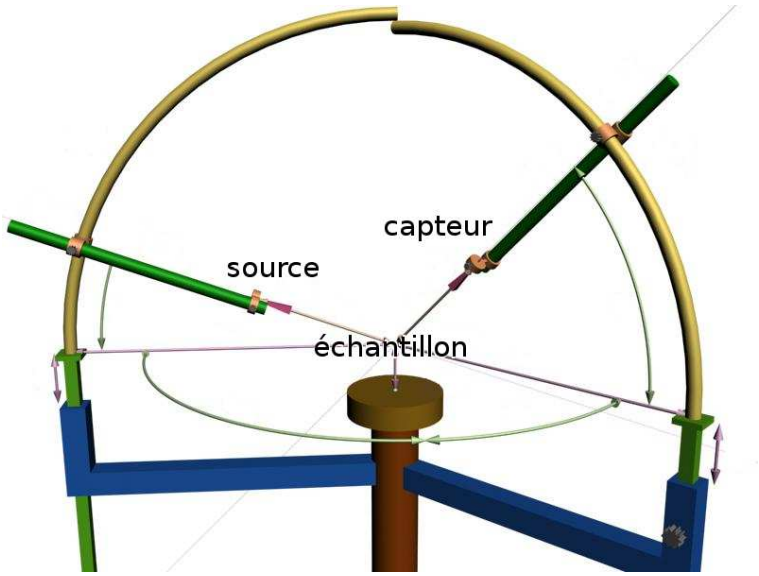


Fig. 2.18 – Un gonioréflectomètre est un outil permettant d'effectuer des mesures de réflectance. L'échantillon à analyser est placé au centre de l'appareil. Les deux bras mobiles peuvent parcourir tout l'hémisphère. L'un porte une source lumineuse ponctuelle, et l'autre un capteur. La réflectance est analysée pour différentes combinaisons d'éclairage et d'observation.

Simplification de la fonction de réflectance

Une BRDF quelconque s'exprime sous la forme d'une fonction à cinq dimensions :

$$f_r = (\theta_i, \phi_i, \theta_e, \phi_e, \lambda)$$

avec (θ_i, ϕ_i) : direction d'incidence de la lumière,
(θ_e, ϕ_e) : direction d'observation,
λ : longueur d'onde.

On se place dans le cas des surfaces isotropes. La BRDF est alors invariante par rotation sur la surface, c'est à dire que si la source lumineuse et l'observateur subissent une rotation du même

angle par rapport à la normale, la valeur de la réflectance ne change pas. La BRDF n'est donc sensible qu'a l'angle relatif $\Delta\phi = \phi_e - \phi_i$.
La longueur d'onde λ est représentée par les composantes tri-chromatiques (R, V, B). La fonction de réflectance est donc réduite à une fonction $f_r = (\theta_i, \theta_e, \Delta\phi)$ pour chacun des trois canaux, et peut donc être représentée à l'aide d'un ensemble de trois vecteurs à trois dimensions.

Acquisition

La méthode traditionnelle consiste à analyser une surface plane à l'aide d'une caméra parcourant l'hémisphère, et en répétant cette opération pour différentes directions d'illumination. En utilisant un objet de géométrie connue, avec une certaine courbure et des propriétés de réflectance uniformes, chaque cliché peut potentiellement nous fournir plus d'informations. En connaissant le repère local de chaque point, on peut déterminer pour chaque pixel les conditions d'éclairage et d'observation par rapport à la surface.
On peut utiliser des objets de géométrie simples comme le cylindre ou la sphère. Le programme intepétant les image peut alors les représenter de manière analytique pour déterminer les caractéristiques géométriques. Pour des objets réels plus compliqués, cette géométrie ne peut être connue sans une numérisation préalable. La fonction de réflectance est alors reconstruite à partir de quelques clichés où seul l'éclairage change, tout en gardant l'objet et la caméra fixes.

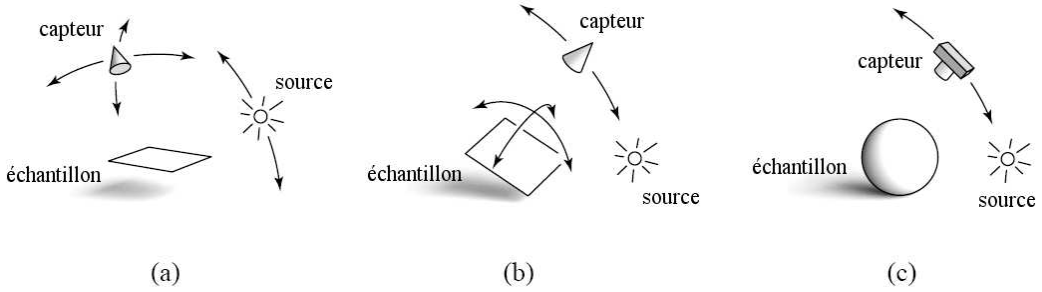


Fig. 2.19 – a) et **b)** : les techniques habituelles d'acquisition de BRDF nécessitent un échantillonnage selon au moins 3 degrés de liberté. **c)** : l'acquisition basée image utilise la connaissance de la géométrie pour réduire ce degré, et donc le nombre d'échantillons.

Avantages et inconvénients

La technique décrite ici permet la mesure de BRDFs bien plus rapidement qu'avec un appareillage spécifique (gonioréflectomètre), et à moindre coût. Le nombre limité de clichés nécessaires réduit considérablement le temps d'acquisition. En contrepartie, les objets utilisés doivent impérativement avoir un matériau uniforme sur toute leur surface et présenter une courbure convexe pour éviter les effets d'interréflexion.

2.3.2 Approximation non-linéaire de fonctions de réflectance

Le modèle de Phong (déscription en annexe A) a été largement utilisé pour le calcul d'illumination. Il est pourtant incorrect. En effet, il ne s'inspire d'aucune loi physique, mais uniquement d'observations. L'un des principaux problèmes qui font que ce modèle n'est pas cohérent est la non conservation de l'énergie.
Le modèle des lobes de cosinus, basé sur le modèle de Phong s'exprime par l'équation suivante :

$$f_r(u,v) = \rho_s C_s \cos^n \alpha = \rho_s C_s (v.u_r)^n = \rho_s C_s (u^T(2nn^T - I)v))^n$$

avec u : direction d'incidence,
v : direction d'observation (sortie),
u_r : vecteur u réfléchi,
α : angle entre u_r et v,
C_s : facteur de normalisation (conservation de l'énergie),
ρ_s : albedo maximale (entre 0 et 1).

Mais ce modèle pose des problèmes : plus l'angle d'observation devient rasant, plus la partie spéculaire disparait, alors que dans la réalité elle semble au contraire devenir plus présente. On comprend le phénomène en analysant l'allure du lobe selon l'angle d'observation.

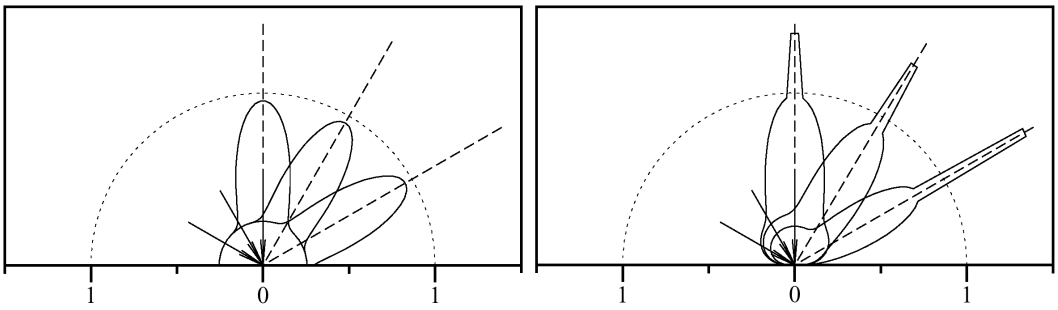


Fig. 2.20 – A gauche : lobes de cosinus standard. Une partie de la composante diffuse disparait sous la surface. **A droite :** réflectance exprimée avec les lobes de cosinus généralisés.

On voit clairement que, pour les angles rasants, une grande partie du lobe disparait sous la surface, ce qui correspond effectivement à une perte d'énergie.
Lafortune [LAFORTUNE97] propose une reformulation du modèle qui corrige ce défaut. L'équation précédente peut être reformulée ainsi :

$$f_r(u,v) = \rho_s(u^T M v)^n$$

où M est la matrice de Householder [1] multipliée par le facteur de normalisation C_s. Comme le modèle doit rester physiquement cohérent, la fonction de réflectance doit vérifier le principe de réciprocité, c'est à dire que le fait d'interchanger l'observateur et la source lumineuse ne change pas la valeur calculée.
Pour ce faire, il faut que la matrice M soit telle que $M = M^T$. On peut la diagonaliser :

$$M = Q^T D Q$$

avec :

$$D = \begin{bmatrix} D_x & 0 & 0 \\ 0 & D_y & 0 \\ 0 & 0 & D_z \end{bmatrix}$$

La matrice Q nous place dans un repère local à la surface, orienté selon la normale la principale direction d'anisotropie (si le matériau est anisotrope).
On peut alors formuler l'équation dans ce nouveau repère avec les valeurs de la matrice D :

$$f_r(u,v) = \rho_s(D_x u_x v_x + D_y u_y v_y + D_y u_y v_y)^n$$

L'allure de la fonction dépend des paramètres D_x, D_y, D_z et n. On peut par exemple retrouver le modèle original des lobes de cosinus en prenant $-D_x = -D_y = D_z = \sqrt[n]{C_s}$. Dans le cas de

[1] matrice symétrique et orthogonale de la forme $M = I - 2uu^T$, où u est un vecteur unitaire appelé vecteur de Householder. Exprime ici la transformation calculant le vecteur réfléchi par rapport à une normale.

surfaces isotropes, on a $D_x = D_y$. La réflexion diffuse peut s'exprimer en prenant $D_x = D_y = 0$: $f_r(u, v) = \rho_d C_d (u_z v_z)^n$.
Des fonctions de réflectances plus complexes, avec un terme diffus, un terme diffus directionnel et un terme spéculaire peuvent être représentées en combinant plusieurs lobes :

$$f_r(u, v) = \sum_i (D_{x,i} u_x v_x + D_{y,i} u_y v_y + D_{z,i} u_z v_z)^{n_i}.$$

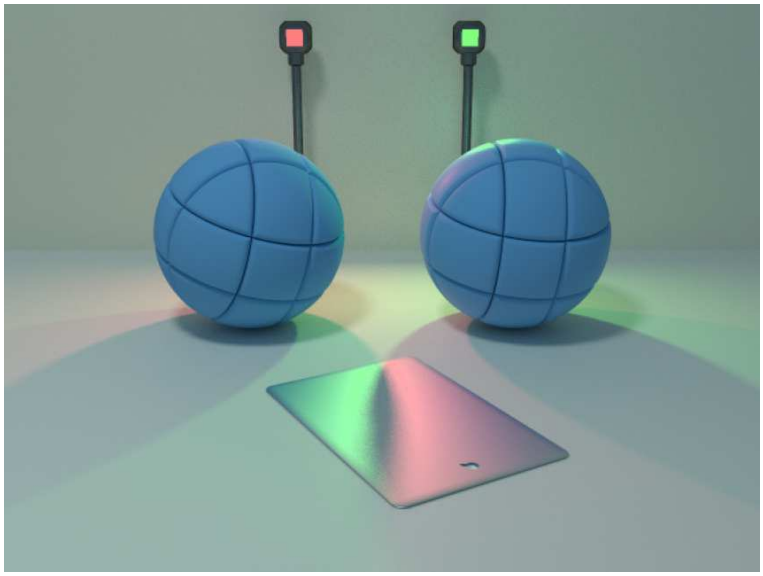


Fig. 2.21 – Scène calculée en utilisant le modèle de Lafortune pour représenter les fonctions de réflectance.

2.3.3 Extraction de matériaux à partir d'images d'objets réels

Capturer l'apparence d'objets réels nécessite en général l'utilisation d'un matériel coûteux (gonioréflectomètre). Pour cette raison, certains chercheurs tentent aujourd'hui de trouver des solutions pour effectuer ces mesures à partir d'un nombre limité de vues.
C'est dans cette optique que *Erkut Erdem, Aykut Erdem* et *Atalay* [ERKUT03] nous proposent une technique basée images permettant de capturer l'apparence d'un objet et de la représenter selon un modèle simple. Les composantes diffuse et spéculaire (voir annexe A) sont extraites séparément. La composante diffuse est stockée comme simple couleur dans une texture globale alors que la composante spéculaire est approximée par un modèle analytique de fonction de réflectance.

Extraction de la composante diffuse

Isoler la composante diffuse ρ_d se fait en deux temps : il faut d'abord supprimer les effets du à l'illumination (ombres et reflets spéculaires), puis déterminer le coefficients diffus.
Pour supprimer les effets d'illumination, six sources lumineuses ponctuelles sont placées autour de la caméra et six images sont prises pour chaque vue en allumant à tour de rôle chacune des six lampes. On capture donc pour chaque vue six intensités différentes $(c_1, \ldots, c_6)$ par pixel. Un valeur trop faible signifie que le pixel se trouve dans une zone d'ombre, alors qu'une valeur plus saturée correspond à un reflet spéculaire. Les intensités ne vérifiant aucun de ces deux cas sont considérées comme de la réflection diffuse pure. En posant l'hypothèse que la surface analysée

est lambertienne, on peut considérer que chacune d'elles vérifie l'équation :

$$\rho_d(\vec{l_i} \cdot \vec{n}) = c_i$$

avec $\vec{l_i}$: vecteur d'incidence pour l'image i,
$\vec{n}$: normale à la surface pour le point considéré,
c_i : intensité observée dans l'image i.

Ce système d'équation linéaire peut être résolu si au moins trois couleurs différentes sont observées pour ce pixel parmis les six images. Pour les pixels ne vérifiant pas cette condition, la composante diffuse ρ_d ne peut pas être correctement reconstruite pour la vue considérée.

Construction de la texture diffuse

La texture diffuse est reconstruite grâce au concept de *particules de surface* (utilisé par *Yilmaz, Mülayim* et *Atalay* dans [YILMAZ02]) pour éviter les discontinuités souvent introduites par la reconstruction à partir de polygônes.
On considère la surface comme un ensemble de particules, possédant chacune les attributs suivants : une position, une normale et une couleur. Chaque particule est associée à un texel [2] de la texture à reconstruire. La couleur est choisie parmi les différentes vues dans lesquelles la particule est visible. C'est celle qui donne le plus petit angle entre la normale de la particule et celle du plan image qui est choisie.

Reconstruction de la réflectance

Il est maintenant possible de calculer pour chaque vue une image résiduelle par différence entre les images initiales et celles de réflexion diffuse pure précédemment définies. Seules les valeurs positives sont conservées, de manière à ne garder que la composante spéculaire.

Fig. 2.22 – A gauche : une vue initiale, immédiatement issus de l'acquisition. **A droite :** image calculée pour supprimer les effets du à l'illumination.

A partir des pixels valides dans les images résiduelles (pixels correspondant à un reflet spéculaire, c'est à dire ceux qui ont une valeur positive), on peut définir un ensemble d'échantillons de luminance. Chacun de ces échantillons possède trois attributs : un vecteur d'observation $\vec{v}$, un vecteur d'incidence $\vec{u}$ et une luminance r déterminée à partir de la couleur observée en tenant compte de l'intensité de la source et de l'atténuation due à son éloignement.
C'est le modèle d'approximation non-linéaire de Lafortune qui à été choisi pour approximer la fonction de réflectance. La forme utilisée est celle définie pour les surfaces isotropes, avec un seul lobe spéculaire :

$$f(u,v) = \rho_d + (C_{x,y}(u_x v_x + u_y v_y) + C_z u_z v_z)^n$$

[2] élément de texture, similaire à un pixel dans une image.

avec u : vecteur d'incidence,
v : vecteur d'observation,
$C_{x,y}, C_z$: coefficients de modulation du produit scalaire,
n : exposant décrivant l'envergure du lobe.

Etant donné que ρ_d à été préalablement calculé, les seuls paramètres restant à définir sont $C_{x,y}, C_z$ et n. On utilise l'algorithme d'optimisation non-linéaire de Levenberg-Marquardt (détaillé en annexe C) pour les approximer à partir de nos échantillons de luminance.

Avantages et inconvénients

Bien que l'idée soit séduisante, la façon dont la méthode est présentée ne fait pas preuve d'une grande rigueur : l'approximation de la composante diffuse se base sur l'hypothèse que la surface est lambertienne, hypothèse assez peu crédible pour le cas d'objets réels. On peut cependant l'accepter du fait que l'inexactitude engendrée n'est pas si grande.
En revanche, l'estimation de la composante spéculaire reconstruit une seule fonction de réflectance pour tout l'objet en prenant des échantillons de luminance à divers endroit de sa surface. Celà pose l'hypothèse, encore moins probable que la première, que les interréflexions et autres phénomènes du même ordre peuvent être ignorés.
De même, on observe souvent des variations spatiales au niveau de l'apparence des objets réels, même s'ils ne sont composés que d'un seul matériau. Cette méthode ne tient pas compte de celà. Une seule BRDF est définie pour tout l'objet et l'on ne peut alors exprimer qu'une brillance uniforme sur toute sa surface. Celà fonctionne peut être bien sur des données synthétisées (comme celà semble être le cas pour les exemples fournis), mais il faudrait pouvoir en observer les effets sur des échantillons réels.

2.3.4 Reconstruction basée images de matériaux à variation spatiale

Pour obtenir un rendu photoréaliste, utiliser une seul BRDF pour représenter un matériau n'est pas toujours suffisant. Dans la réalité, un matériau observé sur une surface n'est jamais parfaitement uniforme en tous points. Typiquement, une surface un peu usée ou salie présente ce genre de variation. C'est ce que l'on appelle *variation spatiale* d'un matériau. Après s'être penché sur la représentation de réflectances, deux types de méthodes ont vu le jour pour tenter de représenter ce phénomène :

- Les *surface light field* (voir section 2.3.6) représentent cette variation spatiale à partir d'une scène en éclairage fixe. Une luminance est déterminée pour chaque point de la surface sous forme d'une texture, et la visualisation est très rapide.
- Les BTF (voir section 2.3.5) permettent, quant à elles, d'exprimer une réflectance en tenant compte des variations spatiales, du vecteur d'observation, et du vecteur d'incidence. La contrainte d'un éclairage fixe est donc levée, mais au prix d'une énorme quantité de données pour une seul petit échantillon de texture.

La méthode décrite par *Lensch, Kautz, Goesele, Heidrich* et *Seidel* [LENSCH01] représente la variation spatiale en définissant, pour chaque matériau, une base de BRDFs à partir de laquelle l'apparence de chaque point sera reconstruite par combinaison linéaire. A partir d'images d'objets réels, les pixels sont répartis dans des *clusters* selon leur probabilité d'appartenance à l'un ou l'autre des matériaux. Ces clusters sont subdivisés récursivement de manière à obtenir un cluster par matériau. Une base de BRDFs est alors calculée pour chaque cluster.

Acquisition et échantillonnage

Dans un premier temps, une acquisition 3D de la géométrie est effectuée et la surface est reconstruite sous la forme d'un maillage triangulaire. On dispose également d'un certain nombre

de clichés de l'objet dont la correspondance avec la géométrie (repère du plan image dans l'espace global) a été établie.
Les triangles sont projetés dans chaque image et l'aire du triangle résultant est calculée en nombre de pixels. I_{best} définie l'image dans laquelle le polygone est le mieux échantillonné et correspond à celle pour laquelle cette aire est la plus grande. Pour chaque pixel de I_{best} appartenant au triangle, on construit un *lumitexel* ℓ.
Un lumitexel est une structure contenant une position $\vec{x}$, une normale $\vec{n}$ et une liste d'échantillons de luminances $\Re_i$, eux-même définis par une luminance r, une direction d'incidence $\vec{u}$ et une direction d'observation $\vec{v}$ dans le repère local.
La position $\vec{x}$ du lumitexel est déterminée par lancé de rayons, et la normale $\vec{n}$ par interpolation des normales aux sommets. La liste des $\Re_i$ est construite à partir des images dans lesquelles le point $\vec{x}$ est visible, et la luminance r de chaque échantillon est calculée par reprojection du point dans l'espace image en tenant compte de la couleur du pixel incident, de l'intensité et de l'atténuation quadratique de la source.

Clustering

Chaque cluster K_i est défini par une BRDF f_i. Le modèle utilisé pour représenter la fonction de réflectance est celui de Lafortune pour les surfaces isotropes, avec un seul lobe spéculaire :

$$f(u,v) = \rho_d + (C_x u_x v_x + C_y u_y v_y + C_z u_z v_z)^n$$

avec u : direction d'incidence,
v : direction d'observation,
ρ_d : terme diffus,
C_x, C_y, C_z : coefficients de modulation du produit scalaire,
n : exposant paramétrant l'envergure du lobe spéculaire.

Le lumitexels sont distribués dans les différents clusters. Etant donné que chaque lumitexel correspond en quelque sorte à un échantillonnage pauvre de BRDF, le cluster choisi est celui pour lequel la fonction f_i donne la meilleure approximation. Le cluster K_i est choisi si f_i minimise l'erreur suivante :

$$E_{f_i}(\ell) = \frac{1}{\mid \ell \mid} \sum_{\Re_j \in \ell} s \cdot I(f_i(\vec{u}_j, \vec{v}_j) u_{j,z}, r_j) + D(f_i(\vec{u}_j, \vec{v}_j) u_{j,z}, r_j)$$

avec $\mid \ell \mid$: nombre d'échantillons de luminance liés au lumitexel ℓ,
$I(r_1, r_2)$: fonction mesurant la différence d'intensité entre r_1 et r_2,
$D(r_1, r_2)$: fonction mesurant la différence de couleur entre r_1 et r_2.

Après répartitions, la BRDF f_i de chaque cluster est ré-estimée à partir des lumitexels de K_i afin de mieux approximer la réflectance du matériau qu'elle est censée représenter. Cette nouvelle estimation de f_i (appelée *fitting*) est effectuée à l'aide de la méthode d'optimisation non-linéaire de Levenberg-Marquardt, détaillée en annexe C.
Initialement, l'algorithme ne travaille qu'avec un seul cluster, et les autres sont obtenus par subdivisions récursives. Le choix du cluster à subdiviser dépend de l'erreur de la fonction f_i par rapport à tous les lumitexels contenus dans K_i :

$$E_i = \sum_{\ell_j \in K_i} E_{f_i}(\ell_j)$$

L'algorithme de Levenberg-Marquart utilisé précédemment pour fitter la BRDF utilise une matrice de covariance [3] pour progresser vers la solution. Le vecteur propre $\vec{e}$ associé à la plus grande

[3] inverse de la matrice hessienne (matrice des dérivées secondes).

valeur propre λ de cette matrice nous donne la direction dans l'espace des paramètres pour laquelle la variance des échantillons est la plus grande. Cette variance permet de définir un plan de coupe séparant très probablement deux matériaux différents regroupés dans un même cluster. On définit alors deux nouvelles BRDFs :

$$f_{i,1}(\vec{a} + \tau\lambda\vec{e}, \vec{u}, \vec{v}) \text{ et } f_{i,2}(\vec{a} - \tau\lambda\vec{e}, \vec{u}, \vec{v})$$

avec : $\vec{a}$: vecteur des paramètres de f_i,
τ : facteur de mise à l'échelle.

Les lumitexels de K_i sont répartis dans les deux nouveaux clusters créés à partir de ces BRDFs, puis $f_{i,1}$ et $f_{i,2}$ sont fittées pour donner une approximation de ces nouveaux matériaux.
Après la création d'un nouveau cluster, une reclusterisation globale est effectuée pour être sûr que les échantillons sont toujours dans le bon groupe après que les paramètres aient changé. Tous les lumitexels sont alors redistribués, et les fonctions f_i sont refittées en conséquence. Le processus est itéré jusqu'à ce que les changements observés soit inférieurs à un certain seuil.
La subdivision s'arrête lorsque le nombre de clusters équivaut à un nombre de matériaux estimé par l'utilisateur lui-même. Tous les lumitexels regroupés dans un même cluster K_i sont constitués du même matériau dont la BRDF moyenne correspond à la fonction de réflectance f_i qui lui est associée.

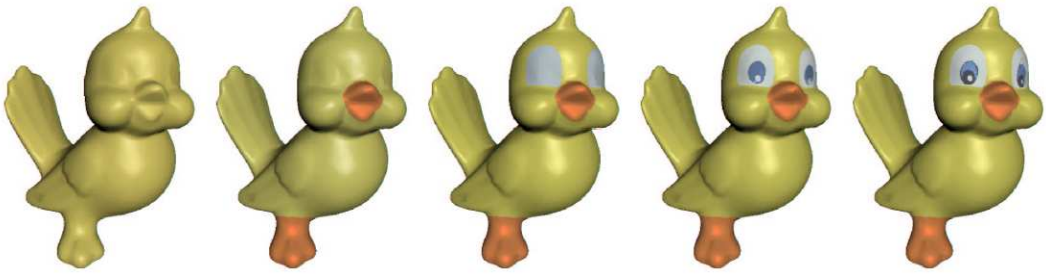


Fig. 2.23 – Résultat de la répartition en cluster. A chaque image, un nouveau cluster est créé. La variation spatiale n'est pas encore représentée, mais les différents matériaux sont séparés de manière bien distincte.

Variation spatiale

La variation spatiale est représentée en définissant pour chaque cluster K une base de plusieurs BRDFs $(f_1, \ldots, f_m)$. La réflectance f_π d'un lumitexel peut alors être exprimée comme une combinaison linéaire de ces fonctions :

$$f_\pi = w_1 f_1 + \ldots + w_m f_m$$

Toute la problématique consiste donc à déterminer cette base, ainsi que les poids w_i.
La base est déterminée en utilisant la méthode d'optimisation non-linéaire pour minimiser la fonction

$$E_\pi(K) = \frac{1}{\mid K \mid} \sum_{\ell_i \in K} E_{f_\pi,i}(\ell_i)$$

C'est de nouveau l'algorithme de Levenberg-Marquardt qui est utilisé, mais cette fois ci pour fitter simultanément les m BRDFs. Les algorithmes d'optimisation non-linéaire sont connus pour être sensibles aux paramètres initiaux. La base initiale sur laquelle est lancé l'algorithme doit être judicieusement choisie. Elle est constituée des BRDFs suivantes :

- la BRDF f du cluster K,
- les BRDFs des clusters voisins, pour permettre la reconstruction des régions de transition entre les différents matériaux,

- les BRDFs des clusters dont le matériau est proche de celui de K,
- deux BRDFs construites à partir de f, l'une en augmentant et l'autre en diminuant le coefficient diffus et l'exposant du lobe spéculaire.

Cette base initiale est, en elle-même, une bonne approximation, et permet une convergence rapide de l'algorithme.
Les poids w_i de la combinaison linéaire associée à un lumitexel ℓ sont déterminés par résolution du système d'équations suivant :

$$\begin{pmatrix} r_1 \\ \vdots \\ r_{|\ell|} \end{pmatrix} = \begin{pmatrix} \tilde{f}_1(\vec{u_1}, \vec{v_1}) & \cdots & \tilde{f_m}(\vec{u_1}, \vec{v_1}) \\ \vdots & \ddots & \vdots \\ \tilde{f}_1(\vec{u_{|\ell|}}, \vec{v_{|\ell|}}) & \cdots & \tilde{f_m}(\vec{u_{|\ell|}}, \vec{v_{|\ell|}}) \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix}$$

où $\tilde{f}(\vec{u}, \vec{v}) = f(\vec{u}, \vec{v})u_z$.
Cette pondération par le produit scalaire u_z entre la direction d'incidence et la normale permet d'exprimer la BRDF comme une fonction par angle solide projeté.

Fig. 2.24 – A gauche : le modèle du canard en plastique après la phase de clustering. Les matériaux sont bien découpés, mais rèstent uniformes sur tout l'objet. **A droite** : le même modèle avec reconstruction de la variation spatiale. Le résultat est très réaliste.

Avantages et inconvénients

Chaque matériau est représenté par une texture contenant les poids w_i associés à chaque lumitexel et une table de BRDFs. On a donc une représentation à la fois conpacte et simple à utiliser et qui, surtout, permet un rendu d'un réalisme rare. Néammoins, les auteurs ne font pas mention de tests de performances. Leurs seuls rendus se limitent à des scènes précalculées par lancé de rayons et jamais à ce qui nous intéresse, à savoir des applications en temps réel.
Lench a utilisé cette technique basée sur la matrice de covariance pour d'autres travaux ayant trait à l'acquisition de données chromatiques. Il l'utilise pour mettre au point un protocole d'acquisition entièrement automatique consistant à receuillir cette information avec le moins de clichés possibles [LENSCH03]. La matrice de covariance est alors utilisée pour déterminer quel doit être le prochain point de vue.

2.3.5 Rendu temps-réel d'une BTF

Une BTF (Bidirectionnal Texture Function) est une fonction décrivant la réflectance d'un objet en chaque point de sa surface. On peut voir une BTF comme une texture 2D contenant, pour chaque texel, une BRDF différente. Une telle fonction permet de représenter des matériaux variant spatialement, en tenant compte de fins détails géométriques comme un *bumpmap* [4] pourrait le faire mais avec un réalisme surprenant. L'acquisition d'une BTF se fait de la même manière que pour une BRDF, avec des surfaces présentant un certain relief. La couleur observée n'est pas capturée qu'en un seul point, mais pour tout l'échantillon de surface. On capture donc, en plus de la réflectance pure, les phénomènes d'inter-reflexion et d'occlusion à un niveau géométrique très fin.
L'inconvénient majeur des BTF est la quantité de données engendrée pour un seul échantillon. Il s'agit en effet de fonctions à 6 dimensions $B(u, v, \theta_i, \phi_i, \theta_e, \phi_e)$ où (u, v) sont les paramètres de surface, (θ_i, ϕ_i) représentent la direction d'incidente lumineuse, et (θ_e, ϕ_e) représentent la direction de d'observation. On fait abstraction de la septième dimension correspondant à la longueur d'onde λ, que l'on représente par un ensemble de valeurs significatives, comme le triplet RVB par exemple. Une texture BTF de 256×256 texels, échantillonnée avec 64×64 directions d'observation et d'incidence différentes, demande une capacité de stockage de 768Mo. Le matériel graphique actuel peut difficilement traîter de telles masses de données et celà pour plusieurs raisons : premièrement, la quantité de mémoire graphique n'est pas toujours suffisante pour pouvoir tout stocker, et deuxièmement, il n'existe pas encore d'implémentation hardware directe permettant de manipuler des textures 6D.
L'enjeu actuel dans ce domaine consiste à trouver des techniques de compression et de représentation de ces données permettant à la carte graphique de les traîter en temps réel. C'est ce que se proposent de faire *Meseth*, *Müller* et *Klein* [MESETH03] en analysant dans un premier temps le comportement d'une fonction de réflectance face aux variations de chacune des deux directions d'incidence et de sortie. De plus, leur méthode semble donner de meilleurs résultats lorsque le relief de la surface présente des variations de profondeur plus importantes.

Fig. 2.25 – Un modèle synthétisé de siège de voiture. **A gauche** : la BRDF reconstruite restitue de manière réaliste l'apparence du velour. **A droite** : le même objet restitué avec un simple bumpmapping. Les reflets spéculaires pour les angles rasants ont presque complétement disparus.

Analyse préliminaire

Disposant d'un échantillon de BTF $B(u, v, \theta_i, \phi_i, \theta_e, \phi_e)$, on extrait un *surface light field* $LF_{(\theta_i,\phi_i)}(u, v, \theta_e, \phi_e)$ en gardant un éclairage fixe, et un *reflectance field* $RF_{(\theta_e,\phi_e)}(u, v, \theta_i, \phi_i)$ en gardant l'observateur fixe. Après étude, il se trouve que $RF_{(\theta_e,\phi_e)}$ présente moins de discontinuités que $LF_{(\theta_i,\phi_i)}$, ce qui signifie que changer la direction d'éclairage provoque des modifications

[4] méthode consistant à perturber localement la normale à partir d'une texture. Les changements se répercutent sur le calcul d'illumination et donne l'illusion d'un relief.

moins brutales que lorsque l'on change la direction d'observation.
La fonction $RF_{(\theta_e,\phi_e)}$ semble donc plus appropriée à la compression.

Algorithme de rendu

Un $RF_{(\theta_e,\phi_e)}$ est mesuré pour plusieurs directions d'observation, puis compressé à l'aide des lobes de *Lafortune* [LAFORTUNE97], en séparant la couleur de la reflectance comme celà est fait pour les *Polynomial Texture Maps* [MALZBENDER96]. Chaque $RF_{(\theta_e,\phi_e)}$ est ensuite stocké sous la forme d'une texture 2D de coefficients C_x,C_y,C_z et n pour chaque lobe de *Lafortune* (les trois paramètres de la matrice diagonale et l'exposant). Toutes ces textures obtenues pour l'ensemble des $RF_{(\theta_e,\phi_e)}$ sont ensuite empilées dans une texture 3D. La couleur préalablement séparée est stockée de la même manière.
Chaque RF est indexé par (θ_i, ϕ_i), la direction d'incidence de la lumière. Pour une direction d'observation arbitraire, un *cube map* est utilisé pour déterminer les quatre RF correspondant aux échantillons dont les directions d'observation sont les plus proches. La couleur finale est alors obtenue par interpolation de ces quatres valeurs.

2.3.6 Light Field Mapping

Un light field est une fonction à quatre dimensions $f(r, s, \theta, \phi)$ décrivant la luminance en chaque point (r, s) de la surface et pour chaque direction d'observation (θ, ϕ). Une telle fonction est obtenue par une succession de clichés réalisés en éclairage fixe, échantillonnant l'hémisphère d'observation. Une approximation de la fonction est obtenue par reconstruction. Il s'agit souvent d'un très grand nombre de données et il faut trouver des solutions pour les traîter efficacement. *Chen et al.* [CHEN02] proposent de prétraîter les données pour les mettre sous forme de textures facilement exploitables par le matériel graphique actuel et garantissant un affichage en un temps correct.

Partitionnement des données

Il faut commencer par découper les données de manière à pouvoir les traîter localement. Disposant d'un maillage triangulaire, on pourrait penser instinctivement à découper selon les polygones, de manière à pouvoir les traîter indépendamment les uns des autres. Malheureusement un tel découpage introduit des discontinuités le long des arêtes.
Chen présente un autre partitionnement qui ne produit pas de discontinuité. Pour chaque sommet v_j, on considère le cercle des triangles qui lui sont adjacents. Pour chaque point de ce cercle, la valeur du light field f est pondérée par Λ^{v_j}, le coeficient barycentrique associé à v_j. Pour un point p appartenant à un triangle $v_1v_2v_3$, on a :

$$\sum_{i=1}^{3} \Lambda^{v_i} = 1$$

ce qui garantit que le light field est correctement reconstruit le long de la surface à partir de fonctions définies localement autour des sommets. On appelle ces fonctions les *vertex light field*, que l'on note f^{v_j} pour un sommet v_j.

Approximation du light field

On voudrait décomposer le light field en une approximation basée sur le produit de deux fonctions :

$$f(r, s, \theta, \phi) = \sum_{k=1}^{K} g_k(r, s) h_k(\theta, \phi)$$

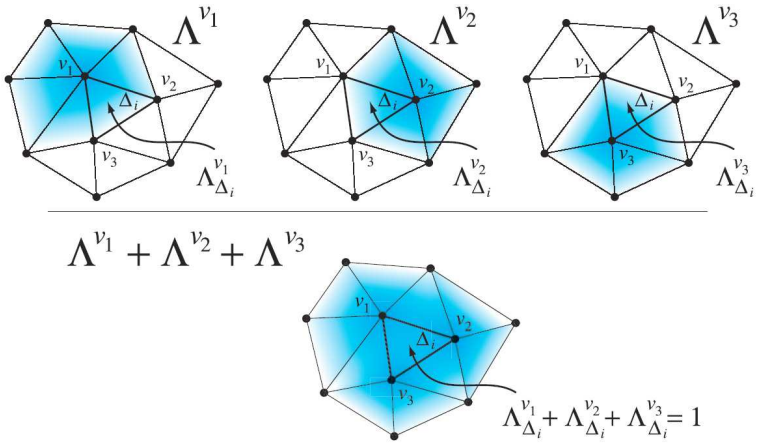


Fig. 2.26 – reconstruction du light field en un point à partir des trois *vertex light field* adjacents.

où la fonction g ne dépend que des paramètres de surface, et la fonction h ne dépend que des paramètres d'observation. Chaque fonction f^{v_j} peut s'exprimer sous la forme d'une matrice :

$$F^{v_j} = \begin{bmatrix} f^{v_j}(r_1, s_1, \theta_1, \phi_1) & \cdots & f^{v_j}(r_1, s_1, \theta_M, \phi_M) \\ \vdots & \ddots & \vdots \\ f^{v_j}(r_N, s_N, \theta_1, \phi_1) & \cdots & f^{v_j}(r_N, s_N, \theta_M, \phi_M) \end{bmatrix}$$

Par un algorithme de factorisation, on peut obtenir une approximation d'ordre K de F^{v_j} :

$$\tilde{F^{v_j}} = \sum_{k=1}^{K} u_k v_k^T$$

où les vecteurs u_k et v_k représentent respectivement les fonctions $g_k^{v_j}(r, s)$ et $h_k^{v_j}(\theta, \phi)$.

Rendu

Les fonctions g et h peuvent être représentées de manière à être correctement utilisées pour profiter pleinement de l'accélération matérielle. Pour g le problème ne se pose a priori pas puisque les paramètres de surface sont interpolés par le matériel graphique. Les paramètres d'observation en revanche nécessitent d'être transformés. Soit d le vecteur d'observation. En projetant d dans le plan tangent à la surface, on obtient des coordonnées (X, Y) dans une texture représentant la fonction h.

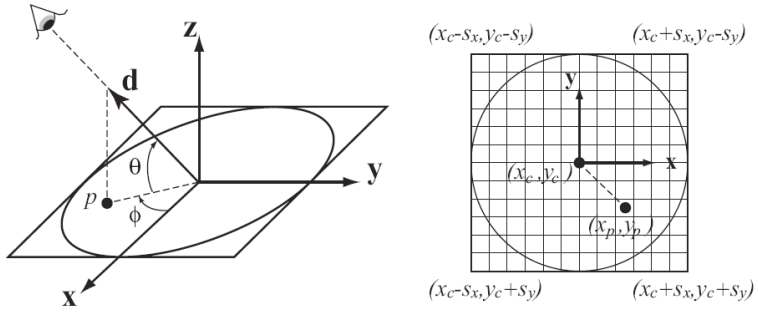


Fig. 2.27 – Les coordonnées de texture de la *view map* calculées à partir du vecteur d'observation.

La procédure de rendu est alors la suivante : les deux textures (g et h) sont appliquées et multipliées entre elles pixel par pixel. On évalue de cette manière plusieurs termes d'approximation (pour différentes valeurs de k), et la somme est effectuée via le tampon d'accumulation.

Acquisition et pré-traîtement

Le protocole d'acquisition nous fournit des données brutes. Pour des objets de petite taille, l'information géométrique est obtenue à partir d'un scanner à lumière structurée, et pour des objets de plus grande taille, c'est la technologie par balayage laser qui est utilisée. L'information de luminance en revanche est récupérée de la même manière dans les deux cas, c'est à dire par une succession de clichés couvrant grossièrement l'hémisphère d'observation.
On souhaiterait traîter ces données pour construire la fonction $f(r, s, \theta, \phi)$ du light field. La surface étant observée selon différents points de vue, on veut «normaliser» les éléments de surfaces disponibles dans chaque image, de manière à ce qu'ils aient tous la même taille. On commence par déterminer pour chaque sommet v_j l'ensemble des clichés à partir desquels il est visible. Les éléments de texture associés à v_j dans chaque vue sont alors redimensionnés en prenant le plus grand comme référence. Ces éléments de textures sont alors pondérés par le coefficient barycentrique Λ^{v_j} associé à v_j.
Après avoir uniformisé les échantillons, la fonction de light field est reconstruite sur tout l'hémisphère par interpolation des informations receuillies dans les quelques vues mesurées. Les vecteurs direction associés aux images initiales sont projetés dans le plan horizontal. Une triangulation de Delaunay sur les points obtenus construit, en quelque sorte, un maillage sur l'hémisphère. En interpolant les valeurs aux sommets, on peut reconstruire le light field pour d'autres points de vue.

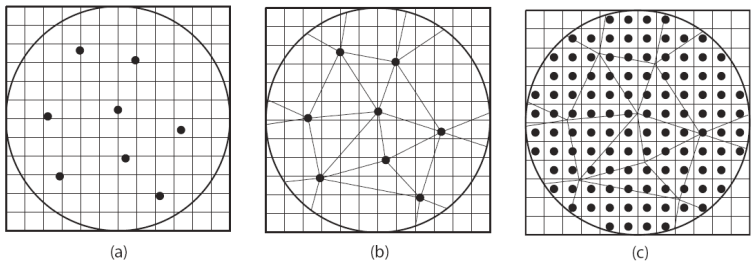


Fig. 2.28 – le light field est reconstruit pour tout l'hémisphère d'observation par interpolation. **(a)** projection des vecteurs d'observation initiaux, **(b)** triangulation de Delaunay sur les projetés, **(c)** interpolation le long d'une grille régulière.

Autre technique

La fonction de light field peut être représentée et exploitée de bien des manières, comme par exemple dans le *light field rendering* [LEVOY96] où cette fois les images sont utilisées telles qu'elles. Dans ce cas là, le calibrage du dispositif d'acquisition doit être plus précis. Le clichés sont souvent pris selon un échantillonnage régulier. Lors du processus de rendu, on reconstruit en quelque sorte les clichés pour les points de vue qui nous intéressent. Le principal intérêt réside dans le fait qu'il n'est pas nécessaire de connaître la géométrie de l'objet et on évite donc tous les problèmes éventuels qui accompagnent souvent le processus de numérisation 3D. En contrepartie, il est nécessaire de garder une trace de toutes les images initiales, ce qui requiert une capacité de stockage beaucoup plus importante. Ce sont bien sûr deux approches totalement différentes et qui dépendent largement de l'utilisation que l'on en fait.

2.3.7 Pré-calcul des transferts de luminance

Sloan, *Kautz* et *Snyder* [SLOAN02] présentent une méthode permettant de capturer l'illumination de la scène (en tenant compte des occlusions et inter-réflexions) pour tout l'hémisphère d'incidence. Les échantillons mesurés (ou synthétisés) sont projetés dans la base des harmoniques

sphériques (SH) qui, à l'instar des séries de Fourier pour les fonctions périodiques, permet d'approximer des fonctions sphériques par un certain nombre de coefficients. Cette projection, en plus de simplifier la représentation de l'hémisphère, offre des propriétés permettant de minimiser considérablement les calculs lors du processus de rendu.

Base SH

La base des harmoniques sphériques (SH) est formée d'un ensemble de fonctions sphériques élémentaires y_l^m permettant d'approximer des fonctions sphériques plus complexes. Soit f une fonction sphérique. Faire une projection de f d'ordre n revient à calculer les coefficients de f pour les n premières bandes de la base :

$$f_l^m = \int f(s) y_l^m(s) ds$$

L'approximation $\tilde{f}$ de f est obtenue par :

$$\tilde{f}(s) = \sum_{l=0}^{n-1} \sum_{m=-l}^{l} f_l^m y_l^m(s)$$

Une projection d'ordre n donne n^2 coefficients. Plus l'ordre est élevé, meilleure est l'approximation.

Calcul de luminance pour une surface diffuse

L'équation d'illumination est la suivante :

$$L_e = \int B(s) G(s) V(s) L_i(s) ds$$

avec B : fonction de réflectance (terme dépendant de l'observateur), constante pour le cas diffus,
G : relation géométrique (orientation de la surface par rapport à la source),
V : coefficient de visibilité (1 si le point est visible de la source, 0 sinon),
L_i : luminance incidente,
L_e : luminance réfléchie.

La fonction de transfert regroupe les coefficients géométriques et de visibilité. On a donc :

$$L_e = \int T(s) L_i(s) ds$$

L'une des propriétés intéressantes de la base SH est l'orthonormalité qui se traduit par :

$$\int \tilde{a}(s) \tilde{b}(s) ds = \sum_{i=1}^{n^2} a_i b_i$$

c'est à dire que l'intégration du produit de deux fonctions est égale au produit scalaire de leurs coefficients de projection. Or T est une fonction sphérique qui, à chaque direction d'incidence, associe la proportion d'énergie lumineuse réémise. En considérant un éclairage placé à l'infini, on peut également représenter L_i par une fonction sphérique. En projetant ces deux fonctions dans la base, on obtient deux vecteurs de coefficients $\vec{t}$ et $\vec{l}$, et le calcul de la luminance peut se simplifier à $L_e = \vec{t}.\vec{l}$

Calcul de luminance pour une surface brillante

L'équation d'illumination doit cette fois tenir compte de la direction d'observation :

$$L_e(\phi) = \int B(\phi, s)T(s)L_i(s)ds$$

avec $B(\phi)$: fonction de réflectance (BRDF),
$L_e(\phi)$: luminance sortante pour une direction d'observation ϕ.

$B(\phi)$ est projetée dans la base pour donner un vecteur $\vec{b_\phi}$ de coefficients. Pour transformer l'intégration en produit scalaire comme précédemment, il faut utiliser non plus un vecteur mais une matrice T de coefficients pour la fonction de transfert.
L'équation devient alors :

$$L_e(\phi) = \vec{b_\phi}.(T\ \vec{l_\phi})$$

Précalcul de la fonction de transfert

Un vecteur / matrice de transfert est calculé pour tous les sommets du maillage. On souhaite y ajouter en plus les phénomènes d'interréflexion. Il va donc falloir procéder en deux étapes :

- une première passe calcule l'illumination en chaque sommet, en tenant compte de la visibilité pour capturer les effets d'ombrage,
- une deuxième passe utilise ces données pour déterminer la contribution apportée en un point par son voisinage.

Le tout est compressé par projection d'ordre 5 dans la base SH (suffisant dans le cas de fonction basse fréquence), générant une matrice 25×25 pour chaque sommet.

Rendu

Le calcul des produits scalaires (ou des produits matrice/vecteur) et l'interpolation linéaire des résultats peuvent être effectués par la carte graphique elle-même à l'aide des *pixel shader* [5]. Cet algorithme a d'ailleurs été implémenté dans DirectX 9.

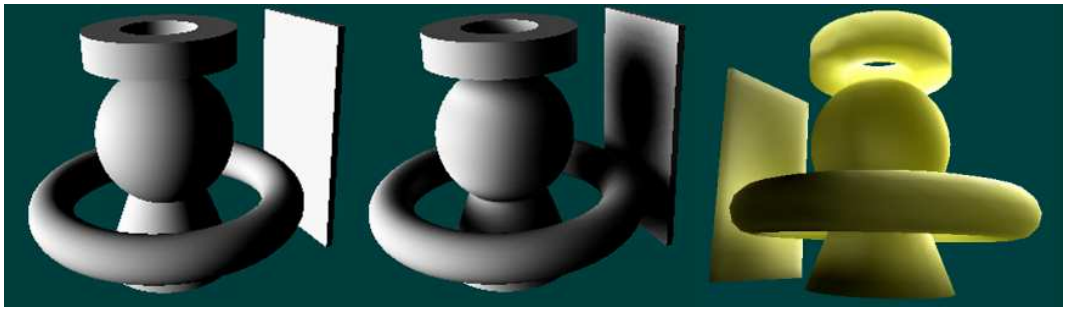


Fig. 2.29 – A gauche : un objet éclairé avec le modèle d'illumination d'OpenGL. **Au centre** : illumination utilisant les harmoniques sphériques. **A droite** : simulation des transferts de luminance sur un objet translucide avec les harmoniques sphériques.

Extension

Dans un autre article [KAUTZ02], les auteurs proposent une amélioration de leur technique permettant d'utiliser n'importe quelle BRDF. A partir de l'équation d'illumination :

$$L_e(\phi) = \int B(\phi, s)G(s)L_i(s)ds$$

[5] unité de la carte graphique effectuant les calculs d'illumination par pixel. Peut être redéfinie par programmation directe du processeur graphique à l'aide d'un langage de bas niveau (type assembleur).

on projette le produit de la BRDF avec le coefficient de relation géométrique dans la base SH :

$$c_i(\phi) = \int y_i(s)B(\phi, s)G(s)ds$$

Une fois de plus, pour une fonction sphérique basse fréquence, une projection d'ordre 5 (donnant 25 coefficients) est suffisante. Ces coefficients sont stockés dans une texture indexée par ϕ.
La luminance incidente est ensuite projetée dans la base, soit sous la forme d'un unique vecteur dans le cas d'un éclairage placé à l'infini, soit à raison d'un vecteur $L_{i,p}$ par sommet dans le cas contraire. Selon que l'éclairage soit fixe ou non, on peut déterminer ce(s) vecteur(s) par un précalcul ou bien à la volée lors du processus de rendu.
Pour chaque sommet p, l'algoritme de rendu procède alors de la manière suivante :

- une rotation en base SH est appliquée au vecteur $L_{i,p}$ pour obtenir les coefficients de la luminance incidente dans le repère tangentiel du sommet p ;
- une rotation est appliquée à ϕ pour obtenir également la direction d'observation ϕ_p dans ce même repère ;
- en indexant la texture avec ϕ_p, on récupère les coefficients $c(\phi_p)$ de la BRDF pour la direction d'observation considérée ;
- le produit scalaire $L_{i,p} \cdot c(\phi_p)$ nous donne le résultat de l'intégration de l'équation d'illumination.

Représenter la BRDF de cette manière nous permet d'utiliser des fonctions de réflectances complexes (issues de mesures par exemple) avec la technique du PRT. Pour fusionner les deux techniques, la matrice de tranfert est appliquée au vecteur $L_{i,p}$ en la couplant à la matrice de rotation utilisée au moment où la luminance incidente est placée dans le repère tangentiel.
Le PRT présente une technique de rendu bidirectionnel mais sans réellement préciser de quelle manière exprimer la fonction de réflectance du matériau. La façon de faire décrite ici se présente donc plutôt comme un complément.

2.4 Appréciations

Nous savons d'ores et déjà que les données à traîter sont volumineuses. Après la phase d'acquisition, la cohérence entre l'information géométrique et les données de réflectance peut être rétablie en utilisant des projections dans le plan image, comme cela est fait pour le *Multiple Texture Stitching* [ROCCHINI99]. Cependant, nous utiliserons certainement une propriété inhérente au dispositif d'acquisition par lumière sutructurée, en utilisant la carte de phase calculée pour définir en chaque point une valeur unique quelque soit le point de vue, ce qui nous permettrait alors de faire coïncider les pixels des différents clichés.
Souhaitant garder l'application interactive, les techniques de rendu adaptatif semblent être une bonne alternative : on ne peut en effet pas traîter à chaque fois tous les points. Il faut forcément passer par une étape de simplification, d'autant plus qu'à long terme nous souhaiterions pouvoir les visualiser avec un rendu directionnel [6], voir bidirectionnel [7], ce qui implique un traîtement plus coûteux. Les QSplat [RUSINKIEWICZ00] permettent de faire abstraction du maillage, puisque la surface n'est pas réellement reconstruite. Mais l'aspect de la scène perd considérablement en réalisme lorsque l'affichage s'adapte aux manipulations de l'utilisateur. Un maillage est certes plus coûteux à traîter que les splats, mais en exploitant correctement l'application de texture, il serait possible de recréer l'information manquante. Avec une telle méthode, un maillage progressif pourrait être plus approprié, bien que le problème de garder la cohérence entre une texture complexe (bidirectionnelle) et le maillage lors des transitions entre les niveaux de détails est un problème ardu.
Un autre problème délicat concerne la compression de l'information de réflectance. Elle aussi constitue un ensemble important de données, souvent plus important encore que les données géométriques, et elle ne peut être traîtée directement sans poser de problème de stockage. Les harmoniques sphériques [SLOAN02] compressent efficacement les données, puisque toute la sphère d'incidence peut être représentée par 5^2, 6^2 ou 7^2 coefficients. Mais d'autres techniques de compression peuvent être plus adaptées à notre problème. Le sujet traîté par *Chen et al.* [CHEN02] semble très proche de notre problème, mais le rendu est-il suffisamment rapide pour être exploité en temps réel ? De plus, la scène se restreint à un éclairage fixe et l'étendre à un rendu bidirectionnel risque de s'avérer très difficile. Néammoins, les techniques de compression mises en œuvre dans cet article méritent certainement que l'on s'y intéresse.

[6] illumination variant selon la position de l'observateur.
[7] illumination variant selon la position de l'observateur et de la source (ou de l'environement lumineux).

Chapitre 3

Vers une numérisation et visualisation d'objets réels

3.1 Problématique

Articulé autour d'un matériel de numérisation 3D fourni par la société HOLO3, basée à Saint-Louis (68), ce stage a pour but de réaliser un outil de visualisation interactive de données issues d'acquisition d'objets réels. Le logiciel de pilotage du scanner, dispositif a priori déstiné à ne faire que l'acquisition de la géométrie, ne permet qu'une visualisation très basique, soit sous la forme d'une image où chaque pixel correspond à un point dont la distance est représentée par un niveau de gris, soit sous la forme du nuage de points flanqué d'une couleur blanche uniforme, sans même le moindre éclairage. On comprend bien que dans ces conditions, il est difficile d'apprécier la qualité de l'information numérisée.

Nous disposons donc initialement d'une information géométrique qu'il va falloir traîter de manière convenable afin de reconstruire la surface de l'objet (ou au moins d'en simuler la reconstruction), afin d'éviter les effets désagréables induits par l'affichage d'un nuage de points, comme l'apparition de discontinuités irrégulières lorsque l'objet devient trop proche de l'observateur.

Une visualisation réaliste nécessite également de reproduire le plus fidèlement possible les propriétés des matériaux qui constituent l'objet. On peut adopter deux types de représentations : une réprésentation directionnelle qui restitue la luminance de chaque point, c'est à dire travailler en éclairage fixe et ne prendre alors en compte que les variations d'intensités par rapport au déplacement de l'observateur, ou une représentation bidirectionnelle qui capture la réflectance même du matériau, ce qui nous donne pour n'importe quelle direction d'illumination la quantité d'énergie renvoyée vers une direction d'observation quelconque.
Bien évidemment, cette seconde information est bien plus complète que la première, puisqu'elle nous renseigne sur le comportement global de l'objet face à un environnement lumineux donné. En d'autres termes, celà nous permettrait de restituer son apparence dans n'importe quelles conditions, pour le placer par exemple dans une scène entièrement virtuelle.
L'information directionnelle quant à elle ne peut pas prévoir le comportement du matériau face aux changements apportés à l'éclairage environnant. On ne peut donc pas simuler n'importe quels effets. Si l'objet est placé dans une scène entièrement synthétisée, celle-ci doit être cohérente vis-à-vis des conditions observées lors de l'acquisition, sans quoi elle perdrait tout crédit.

La mesure de ces deux types d'informations (directionnelle et bidirectionnelle) passe par le même processus d'acquisition. Une caméra prend une succession de clichés de manière à couvrir

du mieux possible la sphère d'observation. Dans le cas directionnel, la source lumineuse reste statique. Dans le cas bidirectionnel, pour chaque direction d'observation on prend plusieurs clichés dont chacun correspond à une direction sur la sphère d'incidence lumineuse.
Il nous faut alors faire coïncider toutes ces images avec la géométrie. Nous devrons donc définir une méthode de recalage pour déterminer quels pixels correspondent aux points géométriques de notre modèle. Il existe déjà un certain nombre de méthodes couramment utilisées dans ce domaine, mais nous avons choisi d'en développer une nouvelle qui tire parti d'une caractéristique propre au dispositif de numérisation que nous utilisons. Recueillir une information précise nécessite un échantillonnage dense, ce qui implique un très grand nombre de données. Par exemple, un échantillonnage de la sphère d'observation de 64×64 pour un nuage de 100000 points en couleur RVB nécessite déjà plus d'un gigaoctet de mémoire pour être stocké.

Nous avons commencé par implémenter une méthode très basique pour pouvoir vérifier que notre algorithme de recalage fonctionnait correctement. En considérant le point de vue courant, on détermine les quatre échantillons correspondant aux points de vue les plus proches. Une simple interpolation linéaire entre ces valeurs nous donne une bonne approximation de la couleur observée. Lorsque l'échantillonnage est trop pauvre, on constate par endroit la disparition de reflets spéculaires. L'interpolation ne peut bien évidemment pas reconstituer une information manquante. Les mesures doivent donc impérativement être très précises, très denses. Des méthodes aussi basiques que celle que nous venons de citer deviennent alors inexploitables du fait du volume trop important des données. Souhaitant garder l'application interactive, il nous est fortement recommandé d'exploiter au mieux le matériel graphique pour accélérer le processus d'affichage, mais la capacité mémoire des cartes actuelles n'est malheureusement pas en mesure de stocker de telles quantités de données.
C'est là le dernier point que nous traîterons : il faut trouver une représentation qui permette de compresser l'information, tout en nous laissant une certaine flexibilité au niveau de son utilisation. Si la méthode compresse bien, mais recquiert trop de temps de calcul, elle n'est pas intéressante dans la mesure où l'application doit pouvoir tourner en temps réel. De même, on ne peut pas se permettre de choisir une compression qui altère les données initiales de manière trop significative : nous recherchons autant le réalisme que l'interactivité.

3.2 Traîtements préliminaires

Le dispositif d'acquisition est constitué d'un projecteur de franges lumineuses et d'une caméra numérique comme capteur. Le logiciel fourni par la société HOLO3 pour piloter le processus d'acquisition reconstruit l'information géométrique de la scène par une analyse des images recueillies. Après traitement, cette information se présente sous la forme d'une cartographie de coordonnées (x, y, z) correspondant à la grille de pixels de la caméra. La précision de la numérisation dépend donc directement de la résolution du capteur.
Comme l'explique *Philippe Billard* [BILLARD98], le calcul de la géométrie à partir de clichés passe d'abord par le calcul de la phase de la sinusoïde projetée. En stoppant le procesus avant la conversion finale, on peut également récupérer la cartographie des phases. Nous discuterons plus tard de l'intérêt de cette information, dans la section 3.3 : *Recalage des données chromatiques.*
Enfin, un simple cliché nous fournit l'information chromatique associée à la scène pour un certain point de vue, sous la forme d'une grille (image) de triplets (R, V, B).

3.2.1 Adjacence implicite

Cette représentation cartographique des données permet de simplifier certains calculs en utilisant des algorithmes proches du traîtement d'image. Elle a pour principal avantage de nous

donner implicitement une relation d'adjacence sur l'ensemble des points du nuage. On évite ainsi un lourd processus de reconstruction de la surface. Etant donné qu'une carte ne considère l'objet que sous un seul point de vue, il est possible que cette adjacence ne soit pas correcte à cause de phénomènes d'occlusion. Le problème non trivial du recalage de données géométriques obtenues pour différents points de vue n'est pas le sujet qui nous intéresse, et nous ne disposons donc pas de l'information nécessaire pour traîter ces zones occludées.

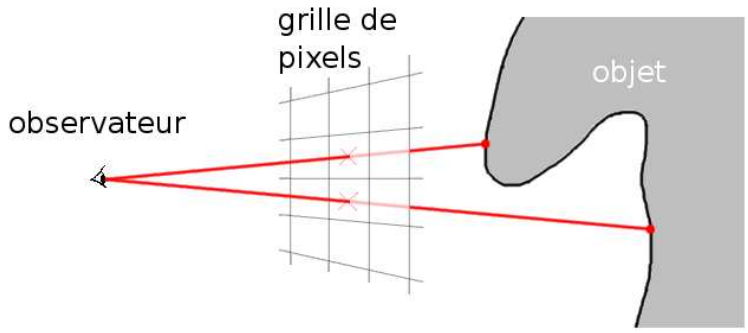


Fig. 3.1 – En ne considérant qu'une seule vue, deux pixels adjacents dans la cartographie géométrique ne représentent pas forcément deux points proches sur la surface de l'objet. La surface ne doit pas être reconstruite n'importe comment.

Pour tenir compte au mieux de cette inexactitude, nous dirons que si la distance qui sépare deux points correspondant à deux pixels adjacents de la carte est supérieure à un seuil d_{max}, alors ces deux points ne seront pas considérés comme connectés, malgré leur voisinage implicite sur la carte.

3.2.2 Coïncidence des cartes

Un tel format de données permet également de faire coïncider plusieurs cartes acquises selon le même point de vue et contenant des informations différentes. Deux pixels ayant la même position sur deux cartes différentes fournissent deux informations relatives au même point géométrique de la surface. On peut ainsi naturellement associer une valeur chromatique ou une valeur de phase à chaque sommet du modèle.

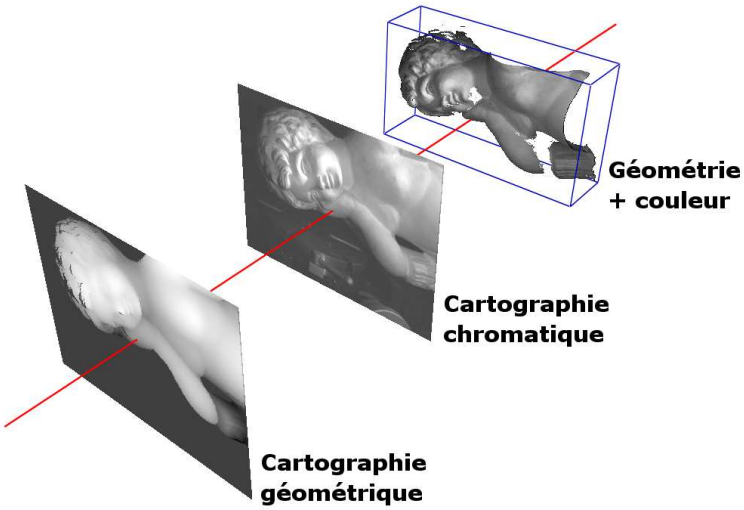


Fig. 3.2 – Association de l'information chromatique à la géométrie par cohérence des cartes.

3.2.3 Reconstruction des normales

Pour des tests de visibilité ou des calculs d'illumination, nous aurons besoin de connaître le vecteur normal à la surface. La relation d'adjacence donnée implicitement par la cartographie géométrique nous permet de construire autour de chaque point un disque de polygones triangulaires. En considérant dans le 8-voisinage les pixels qui sont valides et qui satisfont la contrainte de distance ennoncée précédemment, deux pixels successifs en tournant dans le sens trigonométrique définissent un triangle dont un peut calculer la normale par simple produit vectoriel. La normale du point est ensuite calculée en moyennant les vecteurs obtenus pour tous les polygones adjacents.

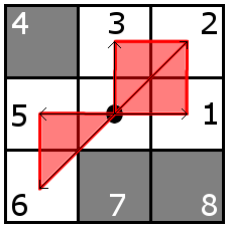


Fig. 3.3 – La normale d'un point est calculée à partir de la cartographie géométrique, en considérant les polygones triangulaires qui peuvent être construit avec ses 8-voisins valides.

ALGORITHME - Reconstruction des normales.

```
Pour p Dans carte_géométrique Faire
   Si p.estValide() Alors
       p.normale := Vecteur(0.0,0.0,0.0)
       Pour i De 1 À 8 Faire
           Si p.voisin[i].estValide() Et p.voisin[succ(i)].estValide() Alors
               a := vecteurEntrePoints(p,p.voisin[i])
               b := vecteurEntrePoints(p,p.voisin[succ(i)])
               p.normale := p.normale + produitVectoriel(a,b)
           FinSi
       FinPour
       p.normal.normaliserVecteur()
   FinSi
FinPour
```

3.3 Recalage des données chromatiques

On souhaite obtenir une information sur la réflectance de notre modèle c'est à dire connaître la manière dont il renvoie la lumière pour un éclairage donné. Plus précisément, on souhaite connaître la quantité d'énergie lumineuse renvoyée pour chaque direction d'observation. On souhaite visualiser des objets issus de modèles réels. Rendre compte de leur aspect à partir de techniques d'illumination existantes (Phong, Cook & Torrance, ...) n'est pas chose aisée. Pour reproduire les propriétés du matériaux de manière fidèle, on peut en capturer l'aspect sous différentes conditions d'éclairage et d'observation par une succession de clichés.
Capturer la luminance signifie associer à chaque point la couleur perçue pour chacune des directions d'observation mesurées. Cette corrélation peut être faite de plusieurs manières. Nous présenterons ici deux techniques.
La première est couramment utilisée, mais nécessite de connaître précisément les caractéristiques du dispositif. C'est bien souvent le cas puisque ce genre de mesures s'effectuent en général à l'aide

d'un *gonioreflectomètre*.
La deuxième utilise une propriété inhérente à la méthode de numérisation par projection de franges. Elle n'a besoin d'aucune information sur les caractéristiques optiques de la caméra, ni sur sa position relative à l'objet. C'est pour cette raison que nous l'avons développé, puisque nous ne disposions pas d'outil précis lors de nos premières mesures.

3.3.1 Première approche - Projection dans l'espace image

Cette méthode est utilisée entre autre par *Rocchini*, *Cignoni* et *Montani* [ROCCHINI99] pour reconstruire une texture sur toute la surface d'un objet à partir de clichés réels. Connaissant précisément la position de la caméra ainsi que la géométrie de l'objet, on peut en déduire la transformation permettant de passer du repère global à l'espace image associé à chaque cliché. L'utilisation éventuelle d'un tampon de profondeur lors de cette reprojection nous permet de savoir quels points ne sont pas visibles de ce point de vue (backface ou occlusion), bien que celà nécessite de projeter un maillage complet du fait des discontinuités qui peuvent être engendrées par un nuage de points. Le pixel dans lequel le point se projette nous donne la nouvelle couleur. On peut aussi utiliser une combinaison linéaire des quatre pixels les plus proches pour approximer de manière plus exacte la valeur réelle que l'on devrait obtenir.
L'utilisation d'un goniomètre bien calibré nous donne les informations nécessaires pour exploiter cette méthode. Néammoins, nous souhaitons nous pencher sur une technique n'étant pas soumise à de telles contraintes de précision. Au moment où nous avions besoin de recaler ces données chromatiques, nous ne disposions pas encore du matériel nécessaire. De plus, si à plus long terme le dispositif est destiné à faire l'acquisition d'objets de grande taille et à circuler de musée en musée, le problème de précision risque de s'avérer plus difficile à résoudre pour un appareillage imposant sans moyen de fixation.

3.3.2 Approche adoptée - Coïncidence des couples de phases

Comme nous l'avons déjà expliqué, la numérisation par lumière structurée est un procédé qui projette sur l'objet des franges lumineuses dont la variation d'intensité décrit une courbe sinusoïdale. Le capteur, légèrement décalé par rapport au projecteur, perçoit une courbe dont la phase est décalée en fonction de la profondeur de chaque point. C'est en calculant ce déphasage par rapport à un plan de référence qu'il est possible de déterminer la géométrie de l'objet.
Le logiciel de pilotage du scanner a été modifé pour nous permettre de récupérer la cartographie des phases avant qu'elle ne soit transformée en carte de coordonnées 3D.

Unicité des couples de phases

La projection en bandes de la sinusoïde se traduit à la surface de l'objet par une succession de lignes parallèles à l'intérieur desquelles la phase reste constante. La carte de phases une fois démodulée nous donne une fonction croissante dans la direction perpendiculaire à celle des franges, ce qui nous garantit que la valeur de chaque ligne est unique, c'est à dire qu'on ne retrouvera cette phase dans aucune autre région de la surface.
En considérant pour un même objet deux cartes de phases obtenues avec des franges d'orientations différentes, le couple de phases associé à chaque point de la surface correspond à l'intersection de deux lignes d'équi-phase, et ce couple est donc unique.
Le système d'équations calculant la phase de la sinusoïde à partir de plusieures images est totalement indépendant des caractéristiques optiques et de la localisation de la caméra. Celà signifie que si nous utilisons deux caméras comme capteurs lors du processus d'acquisition, elles calculeront une même valeur de phase pour un point donné de la surface. Le calibrage du dispositif n'intervient en effet que lors de la conversion en données 3D.

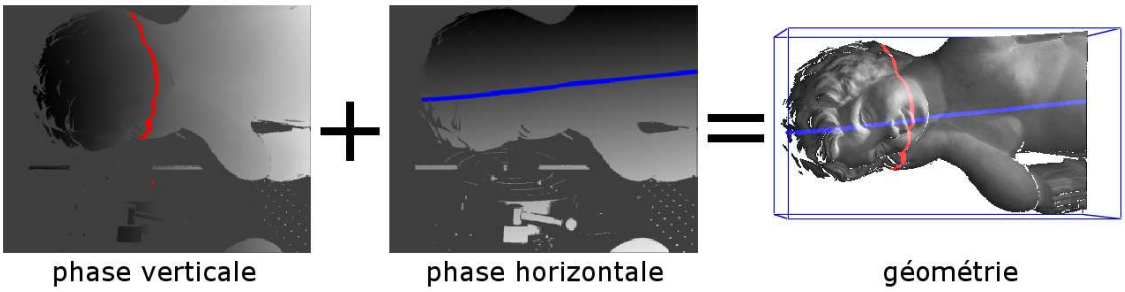


Fig. 3.4 – Utiliser deux cartes de phases nous donne un couple de valeurs uniques pour chaque point de la surface.

Nous pouvons alors établir un protocole d'acquisition permettant d'associer à l'information géométrique une information chromatique prise sous plusieurs points de vue :

- juste après l'acquisition géométrique, faire deux acquisitions de phases avec des orientations de franges différentes, et prendre un cliché de la scène.
- faire à nouveau deux acquisitions de phase et prendre un cliché avec une caméra annexe.
- pour tous les pixels de la carte géométrique correspondant à des points valides, mémoriser le couple de phases et la couleur.
- rechercher dans la seconde vue le pixel ayant le même couple de phases, et mémoriser la nouvelle couleur. Si un tel pixel n'existe pas, soit le point concerné est hors-champs, soit il est occludé par une autre région de la scène.

Localisation d'un couple

Le problème de recalage se résume donc à localiser pour chaque point du modèle un couple dans une carte que nous appellerons « carte biphase ». Pour un rendu directionnel ou bidirectionnel, il faut généralement prendre plusieurs centaines de clichés pour obtenir un résultat acceptable. Il faut donc que la localisation soit rapide afin de réduire les temps de prétraîtement. De plus, la finesse de la numérisation étant liée à la résolution de la caméra, la taille des cartes générées peut devenir conséquente en fonction des critères de qualité exigés par l'utilisateur. Les données que nous traîtons actuellement dépassent le million de pixels par image, ce qui est tout juste suffisant par rapport à la taille des objets numérisés. Si à plus long terme le dispositif est destiné à numériser des objets de plus grande taille, la résolution du capteur augmentera en conséquence si l'on souhaite garder une précision suffisante. Cette opération doit donc être rapide aux vues de la masse importante de données à traîter.
Pour chacune des vues (principale et secondaire) nous faisons une acquisition de phases avec des franges verticales et une autre avec des franges horizontales. Nous utilisons un KD-Tree de dimension deux pour partitionner l'espace biphase alternativement selon chacune des deux phases, et chaque feuille de l'arbre contient la position du pixel atteint. Comme les cartes biphase et chromatique coïncident, on récupère également la nouvelle couleur.

Post-traîtement

Après quelques tests, on peut constater que la méthode fonctionne bien. Cependant, le processus de numérisation introduit inévitablement une erreur. Deux caméras placées en deux points de vue différents ne calculeront jamais exactement les mêmes couples de phases : le volume observé étant discrétisé en une grille de pixels, la probabilité pour que le point géométrique correspondant au centre d'un pixel ait un équivalent dans la seconde vue est très faible. Visuellement parlant, celà se traduit par un léger bruit au niveau des arêtes et des régions à fort contraste. Pour résoudre le problème, on applique un filtre linéaire pour lisser la couleur de chaque point

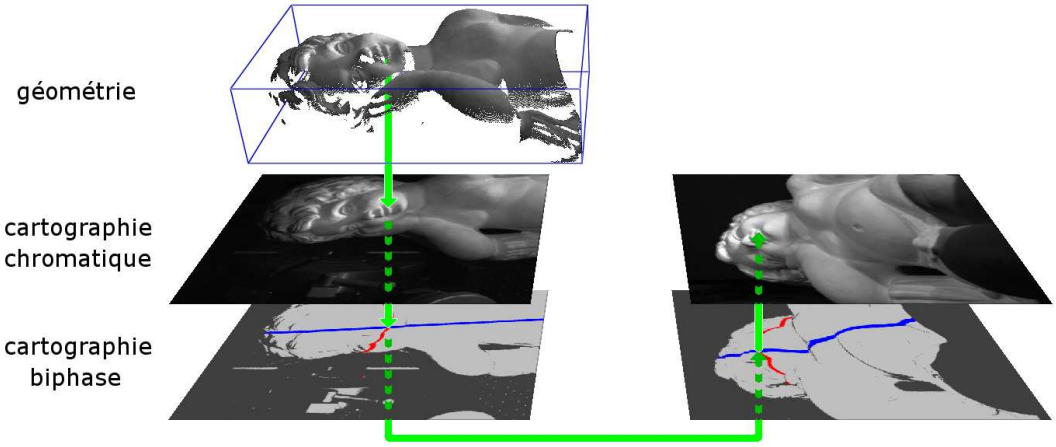


Fig. 3.5 – Recalage de la couleur par cohérence entre carte biphase et carte chromatique.

en tenant compte du voisinnage du pixel correspondant dans la cartographie chromatique. Le filtre est appliqué sur une fenêtre de taille 5×5. Ces dimensions ont été déterminées de manière empirique. L'utilisation d'une fenêtre plus grande altère le résultat de manière beaucoup trop significative.

3.3.3 Résultats

La méthode de recalage par cartes biphases permet de travailler sur des données pour lesquelles le protocole d'acquisition n'a pas besoin de faire preuve d'une grande précision. Il est même tout à fait envisageable d'utiliser ce principe pour recaler des données géométriques, c'est à dire faire coïncider plusieurs nuages de points correspondant à différentes vues du même objet pour en reconstruire toute la surface.
Malheureusement, cette méthode fonctionne mieux dans la théorie que dans la pratique. Elle souffre en effet de quelques défauts. Récupérer une carte de phase revient à faire une numérisation. On observe donc les mêmes problèmes que lors d'un processus d'acquisition géométrique : un objet brillant renverra une lumière trop saturée, et un objet fortement contrasté absorbe trop la lumière dans les régions sombres. Dans les deux cas, l'algorithme de calcul de phase ennoncé par [BILLARD98] renvoie une valeur éronnée, et considère que le calcul a échoué. On obtient dans ce cas des cartes incomplètes. Il faut souvent jouer sur le temps d'exposition de la caméra et faire ainsi plusieurs acquisitions pour un même point de vue, puis combiner les différentes cartes obtenues pour combler au mieux les trous.
Ce procédé d'acquisitions multiples n'est pas une solution en soi. Il peut difficilement être automatisé, car les temps d'expositions adequats sont fortement dépendants de la nature du matériau analysé. Il faut dire que la numérisation 3D est un domaine délicat dans lequel il existe encore beaucoup de problèmes. Le plus gênant dans tout ça, c'est que les objets qui nous intéressent sont justement ceux qui passent difficilement à la numérisation.

3.4 Visualisation

Cette section se décompose en deux parties : tout d'abord, nous décrirons quelle méthode nous avons choisie pour afficher le nuage de points issus de la numérisation. Ensuite, nous nous intéresserons aux méthodes possibles pour représenter l'information chromatique et ses variations. Nous avons choisi de nous limiter au cas directionnel. En fait la capture d'une information bidirectionnelle demande un outil de mesure précis et disposant de suffisament de degrés de liberté

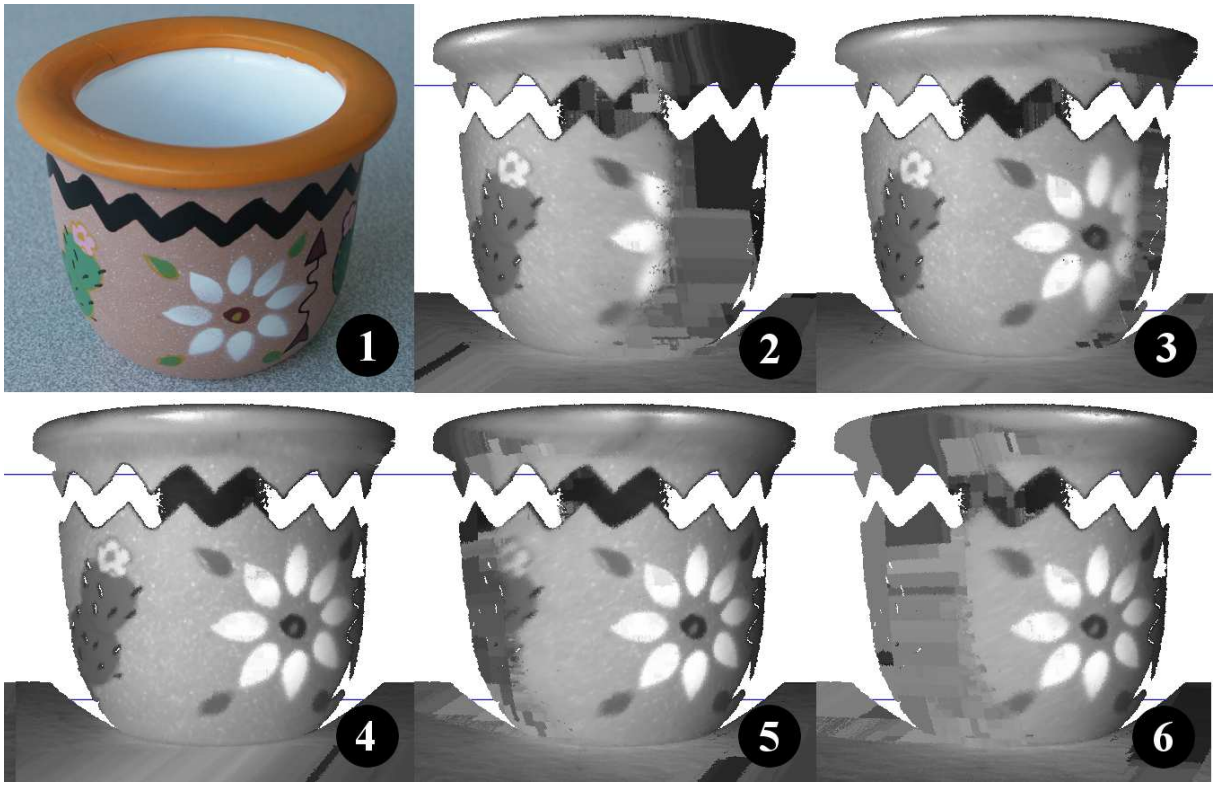


Fig. 3.6 – Vignette 1 : le modèle original du pot à cactus. **Vignette 2 à 6** : la couleur de 5 vues différentes recalées sur la géométrie : -60^o, -30^o, 0^o, 30^o et 60^o (acquisition en noir et blanc). Pour certaines vues, on remarque des zones sur les bords de l'objet où la couleur est complétement altérée. Ce sont les régions qui n'étaient pas visibles par la caméra annexe lors de l'acquisition (occlusion ou backface). On remarque aussi le déplacement du reflet spéculaire sur le haut du pot, ce qui correspond exactement au phénomène que l'on souhaite capturer.

pour que la source lumineuse et la caméra puissent parcourir en même temps tout l'hémisphère. Or, comme nous l'avons déjà dit, au moment de l'implémentation nous ne disposions pas d'un tel matériel. Il n'était pas réellement possible de travailler dans des conditions autres qu'en éclairage fixe. Une première raison à celà : une mesure de réflectance demande une source lumineuse ponctuelle (ou quasi-ponctuelle), et nous ne disposions que de simples lampes de bureau. La deuxième raison est le fait qu'il est impossible d'obtenir une mesure précise avec un placement manuel et approximatif des outils. Nous nous sommes donc concentrés sur la visualisation d'un *light field* (point de vue variable, éclairage fixe). Nous détaillerons les deux approches envisagées.

3.4.1 Méthode d'affichage

Rendu basé points

Nous souhaitions développer un outil de visulisation hautement interactif. Pouvoir choisir un compromis entre la qualité visuelle et la rapidité d'affichage semblait donc primordial. Nous avons vu dans les articles [HOPPE00] et [RUSINKIEWICZ00] deux techniques très différentes de contrôle du niveau de détail, l'une travaillant à partir d'un maillage et l'autre directement à base de points. Bien que les cartographies géométriques fournies par le dispositifs d'acquisition nous permettent d'établir une relation d'adjacence, et donc une définition immédiate de maillage, nous avons choisi les QSplats comme modèle adaptatif. Tout d'abord, cette technique se prête bien au rendu de nuage de points, ce qui correspond tout juste aux données brutes récupérées après la numérisation, d'autant plus que ces nuages sont souvent très denses et il ne présente pas grand intérêt à afficher un polygone dont la taille ne dépasse pas un pixel. Ensuite, les maillages progressifs sont plus délicats à manipuler du fait qu'il s'agit de chaînes de transformations. Le parcours en profondeur d'une hiérarchie est quant à lui très simple et permet en plus d'éliminer par élagage un grand nombre de points lors par exemple des tests de visibilité. Cette souplesse d'utilisation en fait donc un excellent candidat.

Construction de la hierarchie

La hierarchie utilisée est la même que celle décrite dans [RUSINKIEWICZ00]. Chaque noeud possède quatre fils, et un certain nombre d'attributs : une position, un cône de normales (spécifié par une direction et un angle d'ouverture), le rayon de la sphère englobante et une information chromatique (dont nous préciserons la nature plus tard). Ces attributs sont calculés de manière à tenir compte des fils : le rayon et la position doivent pemettre de créer une sphère qui les englobe et le cône de normales doit être suffisament large pour contenir leurs cônes respectifs.
Le niveau le plus bas de la hiérarchie correspond aux points eux-mêmes. Leurs positions sont connues et nous avons déterminé leurs normales par l'algorithme décrit précédemment. L'angle d'ouverture du cône est bien sûr nul. On peut se servir de l'adjacence implicite décrite par la cartographie géométrique pour déterminer le rayon de la sphère, mais comme nous l'avons déjà dit cette description de l'adjacence n'est pas forcément correcte. A cause des phénomènes d'occlusion lors de l'acquisition, deux pixels adjacents dans la carte peuvent décrire deux points de la surface très éloignés l'un de l'autre. Etant donné que le rayon doit être suffisamment grand pour combler les discontinuités avec les points voisins, il faudrait en théorie considérer la distance par rapport au voisin le plus éloigné. Mais si l'un des pixels voisins correspond en effet à un point non adjacent sur la surface de l'objet, le rayon calculé sera beaucoup trop grand et le résultat visuellement désagréable, comme illustré sur la figure 3.7 (apparition de plaques).
Nous avons donc choisi de fixer un seuil d_{max} au dessus duquel la distance d'un voisin ne sera pas prise en compte. Si l'on disposait d'un maillage de notre modèle, celà reviendrait à dire que ce seuil est la plus grande longueur autorisée pour une arête. Nous avons testé plusieurs valeurs. Disposant actuellement de modèles numérisés avec deux scanners de résolutions différentes, l'écart

entre les points n'est pas le même et le seuil apparamment adapté à l'un des objets semble soit trop grand soit trop petit pour l'autre, et vice-versa. Nous avons finalement automatisé ce calcul pour qu'il s'adapte de lui même au modèle en considérant d_{max} comme une valeur dépendante de la distance moyenne entre deux points adjacents : $d_{max} = Kd_{moy}$. Nous obtenons de bons résultats pour les deux résolutions avec $K = 3$ (figure 3.7 : figure de droite).

Fig. 3.7 – **Gauche** : le nuage de points initial, avec la couleur. L'observateur est suffisament proche pour que les discontinuités apparaissent. **Centre** : reconstitution de la surface par splats. Le rayon des sphères englobantes est calculé directement à partir de l'adjacence implicite de la carte géométrique, sans tenir compte des occlusions. Sur les bords de l'objet, les splats sont trop grossières. **Droite** : le calcul du rayon a été corrigé. On observe de ce fait plus de trous, mais ceux-ci correspondent à des zones non-visibles par le dispositif d'acquisition et sont donc cohérents.

Contrôle du niveau de détails

C'est l'étape primordiale qui nous permet de contrôler le temps nécessaire au processus d'affichage pour opérer. La hiérarchie est traversée en profondeur. Dans un premier temps, on peut stopper le parcour lorsque le diamètre de la sphère englobante en un nœud devient inférieur à un pixel. Une précision plus fine n'est pas intéressante : les nœuds fils se projetant tous dans le même pixel, il vont écraser l'un après l'autre la valeur précédemment calculée. Ainsi, on réduit déjà considérablement le nombre de points à traîter.

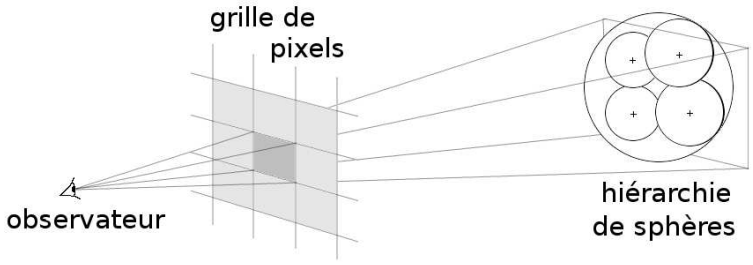


Fig. 3.8 – La sphère englobante se projette en un seul pixel. Il n'est donc pas nécessaire de traîter ses fils. La taille de projection à l'écran fait office de condition d'arrêt pour l'algorithme récursif de rendu.

Malgré tout, le temps de traîtement nécessaire peut encore être trop élevé. Il faut dans ce cas stopper le parcours lorsque la taille de la sphère devient supérieure à un certain seuil, d'un nombre plus important de pixels par exemple. Ce seuil est alors recalculé à chaque étape en tenant compte des contraintes de performance fixées par l'utilisateur. Le seuil $S_{splat,i}$ à l'étape i est calculé à partir de l'étape précédente par :

$$S_{splat,i} = \sqrt{\frac{T_{i-1}}{T_{max}}} S_{splat,i-1}$$

avec T_{i-1} : temps nécessaire pour afficher la frame précédente,
T_{max} : temps maximum demandé pour l'affichage d'une frame,
$S_{splat,i-1}$: valeur du seuil à l'étape précédente.

Prenons un exemple. Supposons que l'utilisateur fixe un taux de rafraichissement minimum FPS_{min} de 20 images par seconde. On a alors $T_{max} = \frac{1}{FPS_{min}} = 0,05$. Le temps taux de rafraichissement observé est de 8 images par seconde pour un seuil de 1 pixel.
On peut observer l'évolution de ce seuil dans le temps, et constater qu'il suffit de peu d'étapes pour ajuster le seuil de manière à respecter la contrainte :

étape (i)	$S_{splat,i}$	FPS_i	T_i	$\sqrt{\frac{T_i}{T_{max}}}$
0	1,000	9,241	0,108	1,471
1	1,471	11,614	0,086	1,312
2	1,930	16,144	0,061	1,113
3	2,149	24,126	0,042	0,910
4	1,956	18,427	0,054	1,042
5	2,038	21,165	0,047	0,972
6	**1,981**	**19,914**	**0,050**	**1,002**

Compression

De la méthode initiale [RUSINKIEWICZ00], nous n'avons pour l'instant gardé que la compression des normales. La représentation de l'information de couleur sur 48 bits ne nous concerne pas : l'information de texture que nous souhaitons capturer ne s'apparente pas en un simple triplet (R, V, B). Nous discutons dans une autre section de la représentation adoptée (3.4.2), et d'une technique de compression évenntuelle (3.4.4).

3.4.2 Représentation de la luminance

Nous avons choisi une représentation directionnelle de l'information photométrique, c'est à dire en effectuant les mesures en déplaçant l'observateur tout en gardant un éclairage fixe. Comme nous l'avons déjà dit à plusieurs reprises, il est impossible d'utiliser les données directement issues de la mesure : les clichés sont en trop grand nombre pour être utilisés tels quels avec une simple méthode d'interpolation linéaire. Il faut utiliser une représentation qui nous permette de compresser cette information tout en lui laissant une certaine flexibilité. En effet, il ne s'agit pas d'une compression pure : un des intérêts est bien sûr de réduire la quantité de données mais surtout de permettre une évaluation simple pour garder un temps de calcul acceptable lors de l'exécution.
Nous avons abordé deux manières de faire. La première utilise la méthode d'approximation non-linéaire exposée par Lafortune [LAFORTUNE97]. Elle a l'avantage comme nous allons le voir de représenter une luminance (ou même une réflectance) par un nombre très limité de valeurs. Le temps d'évaluation est discutable, mais son principal défaut se situe au niveau des contraintes imposées sur l'illumination de la scène lors des mesures.
Nous verrons en second lieu une méthode d'approximation utilisant la base des harmoniques sphériques qui nous permet de représenter une fonction sphérique (typiquement, une information de luminance) par un vecteur de coefficients. Par rapport à la méthode précédente, nous verrons que le nombre de coefficients nécessaires à une bonne approximation est bien plus important. En contrepartie, la manière dont on les utilise peut rendre l'évaluation très rapide. Mais ce qui importe le plus c'est que la luminance est enregistrée pour toute toute la sphères d'observation, ce qui veut dire qu'il n'y a cette fois aucune contrainte au niveau des mesures : l'environnement lumineux peut être totalement arbitraire. Le résultat est donc évidemment bien plus réaliste.

3.4.3 Approximation par les lobes de Lafortune

Description de la méthode

Le modèle non-linéaire de Lafortune pour approximer la fonction de réflectance étend le modèle de Phong de manière à le rendre physiquement correct. Placé dans un repère local à chaque point, le modèle peut être exprimé de la manière suivante, sous la forme d'une somme de k lobes :

$$L_e(u,v) = \sum_{i=1}^{k}(C_{i,x}u_xv_x + C_{i,y}u_yv_y + C_{i,z}u_zv_z)^{n_i}$$

avec u : vecteur d'incidence lumineuse,
v : vecteur d'observation,
L_e : luminance observée pour les directions u et v,
C_x, C_y, C_z : coefficients de modulation du produit scalaire,
n : exposant paramétrant l'envergure du lobe.

Le modèle de Lafortune est souvent utilisé avec un nombre limité et fixé de lobes dont l'un, considéré comme constant, représente la composante diffuse (non-directionnelle). Les autres servent à représenter toute la partie spéculaire. On utilise alors la formule suivante :

$$L_e(u,v) = \rho_d + \sum_{i=1}^{k}(C_{i,x}u_xv_x + C_{i,y}u_yv_y + C_{i,z}u_zv_z)^{n_i}$$

avec ρ_d : composante constante,
C_x, C_y, C_z et n : les paramètres décrivant l'allure du lobe spéculaire.

En fait, la méthode que l'on souhaite mettre en place est similaire à celles décrites par *Lensch* [LENSCH03], [LENSCH01] et *Erkut Erdem* [ERKUT03] pour représenter une BRDF. Nous voulons ici décrire les variations d'un *light field* induites par un changement de point de vue. L'éclairage restant fixe, le vecteur u peut être considéré comme constant pour chaque échantillon et être de ce fait intégré aux coefficients C_i pour réduire encore le modèle :

$$L_e(v) = \rho_d + \sum_{i=1}^{k}(\kappa_i \cdot v)^{n_i}$$

Il suffit juste que la direction d'incidence lumineuse soit connue lors de l'acquisition. Cette représentation de la luminance nécessite $4k+1$ valeurs. Pour un éclairage contrôlé, composé d'une source lumineuse ponctuelle, on peut se limiter à un seul lobe spéculaire. Nous avons donc pour chaque point cinq coefficients : $\rho_d, \kappa_x, \kappa_y, \kappa_z$ et n, ce qui offre une représentation très compacte. Toute la difficulté de la méthode consiste à déterminer ces 5 coefficients à partir des échantillons récupérés lors de l'acquisition (*fitting*). L'algorithme d'optimisation non-linéaire de Levenberg-Marquardt permet d'approximer ces valeurs à une erreur près. La convergence est assez rapide. C'est la méthode utilisée par Lafortune lui-même pour approximer des BRDF de matériaux réels. Pour plus de détails sur cette technique, se référer à l'annexe C.

Problèmes et limitations

Ce modèle est simple et semble plutôt approprié à notre problème. Sa représentation se prête bien à l'exploitation du matériel graphique : le nombre limité de coefficients peut aisément être stocké sous forme de texture et les calculs nécessaires ne demandent qu'un nombre reduit d'opérations, ce qui permet une programmation simple de la carte graphique à l'aide des *pixels shaders*. Nous n'avons malheureusement pas pu tester la méthode par manque de matériel : en effet,

l'estimation des coefficients par l'algorithme de Levenberg-Marquardt recquiert une précision importante au niveau de l'échantillonnage.
Nous aurions bien voulu observer le comportement de cette méthode face à un environnment lumineux plus complexe. Plus de sources nécessiteraient certainement plus de lobes, et le nombre de coefficients augmenterait en conséquence. Nous ferons certainement cette étude lors de prochains travaux. Malgré tout, nous pouvons dire que le problème consistant à déterminer le nombre adéquat de lobes n'est pas trivial. L'utilisation d'un algorithme génétique pourrait être plus avantageux qu'une méthode d'optimisation linéaire, en utilisant comme population initiale un ensemble de solutions avec un nombre variable de lobes. De même, le calcul de l'exposant sur plusieurs lobes serait-il encore assez rapide pour une application en temps-réel ?
En somme, la technique à été mise au point et implémentée sans pouvoir être concrètement testée. Sa validité semble évidente sous les contraintes d'éclairage énoncées plus haut, puisque le modèle reste identique au modèle original de Lafortune, en utilisant simplement un vecteur u constant. On peut éventuellement observer une précision faible pour des matériaux plus complexes du fait du nombre limité de lobes utilisés mais le résultat resterait visuellement cohérent. Mais sous d'autres conditions, seule une expérimentation pourrait nous dire ce qu'elle vaut. Nous nous sommes donc penchés sur une autre approche.

3.4.4 Approximation par les harmoniques sphériques

Définition mathématique

Les harmoniques sphériques définissent une base orthonormale (dite *base SH*) sur la sphère. Ce sont des fonctions sur $\mathbb{C}$ définies par :

$$Y_l^m(\theta,\phi) = K_l^m e^{im\phi} P_l^{|m|}(\cos\theta), l \in \mathbb{N}, -l \leq m \leq l$$

où P_l^m est le polynôme de Legendre associé, et K_l^m est un facteur de normalisation :

$$K_l^m = \sqrt{\frac{(2l+1)}{4\pi}\frac{(l- \mid m \mid)!}{(l+ \mid m \mid)!}}$$

On peut décomposer les fonctions Y_l^m en séparant les parties réelle et imaginaire pour obtenir une base de fonctions y_l^m réelles :

$$y_l^m(\theta,\phi) = \begin{cases} \sqrt{2}\ Re(Y_l^m(\theta,\phi)), & m > 0 \\ \sqrt{2}\ Im(Y_l^m(\theta,\phi)), & m < 0 \\ Y_l^0(\theta,\phi), & m = 0 \end{cases} = \begin{cases} \sqrt{2}\ K_l^m \cos(m\phi) P_l^m(\cos\theta), & m > 0 \\ \sqrt{2}\ K_l^m \sin(-m\phi) P_l^{-m}(\cos\theta), & m < 0 \\ K_l^0 P_l^0(\cos\theta), & m = 0 \end{cases}$$

Projection

Le fait que la base SH soit orthonormale nous permet de définir pour une fonction sphérique quelconque f un ensemble de coefficients f_l^m associés à chaque harmonique :

$$f_l^m = \int_S f(s) y_l^m(s) ds$$

Chaque valeur de l définit une *bande* de la base, contentant les $2l+1$ harmoniques $y_l^{-l}, \ldots, y_l^l$. Une reconstruction d'ordre n utilise les n premières bandes (ie. n^2 coefficients) pour approximer la fonction initiale f en évaluant le polynôme de fonctions :

$$\tilde{f}(\theta,\phi) = \sum_{l=0}^{n-1} \sum_{m=-l}^{l} f_l^m y_l^m(\theta,\phi)$$

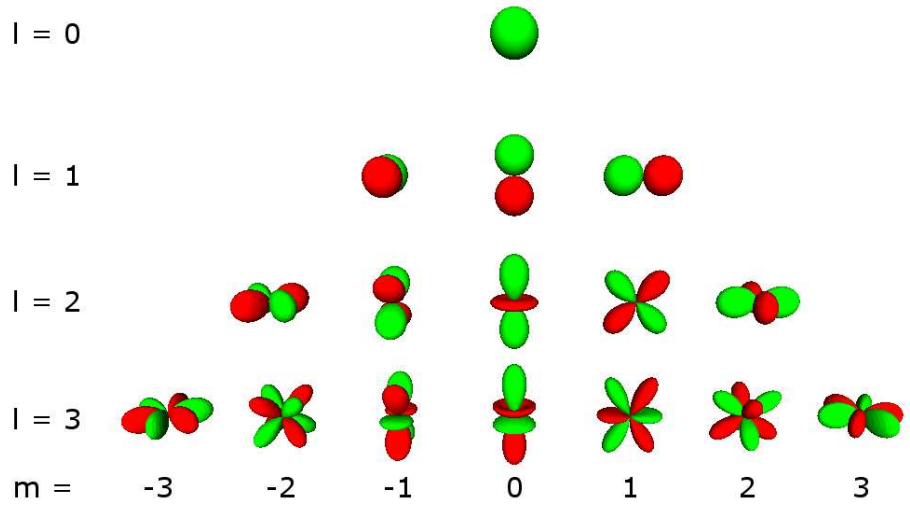


Fig. 3.9 – Les quatre premières bandes d'harmoniques $y_l^m (l = 0, \ldots, 3,\ m = -l, \ldots, l)$. En vert : les valeurs positives, en rouge : les valeurs négatives.

Pour simplifier la notation, on pose $y_k = y_l^m, k = l(l+1)+m+1$. Le polynôme d'approximation peut être reformulé :

$$\tilde{f}(\theta, \phi) = \sum_{i=1}^{n^2} f_i y_i(\theta, \phi)$$

La qualité de l'approximation dépend de l'ordre de reconstruction choisi. Les premières bandes de la base correspondent à des fonctions basse fréquence et une approximation de faible ordre va tendre à adoucir les aspérités de la fonction projetée. Une fonction haute fréquence nécessite donc un ordre de reconstruction plus élevé pour obtenir une approximation correcte.

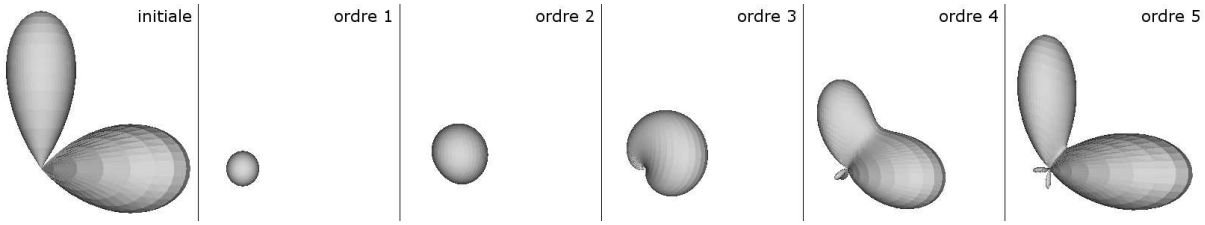


Fig. 3.10 – **A gauche :** fonction sphérique projetée. **A droite :** les reconstructions d'ordre 1 à 5.

Propriétés

La base SH présente des propriétés intéressantes. Disposant de deux fonctions sphériques f et g projetées respectivement en deux vecteurs de coefficients $(f_1, \ldots, f_N)$ et $(g_1, \ldots, g_N)$, l'orthonormalité nous garantit l'identité suivante :

$$\sum_{i=1}^{N} f_i g_i = \int_S f(s) g(s) ds$$

c'est à dire que le produit scalaire des deux vecteurs de coefficients est égale à l'intégration sur toute la sphère du produit des fonctions f et g.
Une deuxième propriété intéressante est l'invariance par rotation. Soient f et g deux fonctions sphériques telles que $g(s) = R(f(s))$, où R est une rotation, et $\tilde{f}$ et $\tilde{g}$ leurs reconstructions respectives après projection en base SH. On a alors $\tilde{g}(s) = R(\tilde{f}(s))$. En d'autres termes, la

rotation n'introduit pas de modification sur la fonction reconstruite. Il existe d'ailleurs une rotation définie dans la base des harmoniques sphériques et exprimée sous la forme d'une matrice opérant directement sur les coefficients de projection.
Enfin, voici la dernière propriété susceptible de nous intéresser. Soient deux fonctions sphériques f et g, et leurs vecteurs respectifs de coefficients en base SH $\vec{f_c}$ et $\vec{g_c}$. La reconstruction d'une fonction à partir d'une interpolation linéaire entre $\vec{f_c}$ et $\vec{g_c}$ nous donne le même résultat que si l'on avait d'abord fait la reconstruction des deux fonctions suivie de l'interpolation. Par ce principe, un certain nombre d'opérations, telles que la moyenne ou la somme de deux fonctions, peut être effectué en agissant directement sur les vecteurs. Cette propriété nous intéresse tout particulierement pour l'intégration au modèle hierarchique (voir section 3.4.4).

Implémentation

La luminance mesurée en un point est typiquement le genre d'information que l'on peut représenter sous la forme d'une fonction sphérique. Contrairement à une représentation analytique permettant une génération au vol, l'utilisation de données mesurées nécessite un échantillonnage dense de la sphère d'observation. Le nombre de valeurs à stocker en chaque point devient vite très important.
Après l'acquisition, nous utilisons donc les harmoniques sphériques pour compresser cette information. En utilisant l'intégration de Monte-Carlo, on peut évaluer l'intégrale de l'équation de projection pour pouvoir la résoudre numériquement :

$$f_k = \frac{4\pi}{N} \sum_{i=1}^{N} f(s_i) y_k(s_i)$$

Cette nouvelle équation nécessite un échantillonnage uniforme sur la sphère, en utilisant les équations de Monte-Carlo. Ce n'est pas forcément le cas pour nos mesures. Nous pouvons reconstruire un maillage entre les directions connues de manière à pouvoir interpoler la luminance sur toute la sphère à partir des trois points de vue les plus proches. Il est donc possible de générer de nouveaux échantillons de manière à ce que leur distribution soit uniforme sur la sphère.
Les échantillons peuvent être projetés en considérant deux approches. Dans un premier cas, la distance entre l'observateur et l'objet est fixée, alors que dans l'autre elle peut varier.
Pour le premier cas, le plus simple, l'outil de visualisation est réglé pour que ses caractéristiques soient identiques à celles du dispositif d'acquisition, c'est à dire les mêmes caractéristiques optiques et la même distance à l'objet que lors des mesures. Lors de l'acquisition, un cliché pris pour une direction d'observation donne des échantillons de luminance différents pour chaque point : l'angle d'observation dans le repère local de chacun n'est pas le même. Mais vu que la distance à l'objet ne varie pas, on peut considérer que chaque cliché capture non pas l'aspect de chaque point pour des conditions locales, mais l'aspect de l'objet lui même pour un angle d'observation donné. Les échantillons peuvent alors être utilisés tels quels, en projetant les échantillons avec la direction par rapport à l'objet et non dans le repère local. Celà revient à définir un repère local qui soit identique pour chaque point (et ne tenant donc pas compte de l'orientation de la surface), et considérer un observateur placé à l'infini.
Evaluer la luminance en un point veut dire reconstruire la fonction sphérique puis récupérer sa valeur pour une direction donnée sur la sphère, c'est à dire résoudre l'équation suivante :

$$L_e(\theta_{obs}, \phi_{obs}) = \sum_{i=1}^{N} f_i y_i(\theta_{obs}, \phi_{obs})$$

Le vecteur $V_{SH} = (y_1(\theta_{obs}, \phi_{obs}), \ldots, y_N(\theta_{obs}, \phi_{obs}))$ correspond à la projection en base SH de la direction d'observation. Sous les hypothèses que nous venons de poser, la direction d'observation

$(\theta_{obs}, \phi_{obs})$ est la même pour tous les points du modèle. Ce vecteur n'a donc besoin d'être évalué qu'une seule fois pour tout l'objet. La luminance est alors calculée par un simple produit scalaire :

$$L_e(\theta_{obs}, \phi_{obs}) = (f_1, \ldots, f_N) \cdot V_{SH}$$

Ce cas a l'avantage de simplifier considérablement les calculs mais bien entendu, le principal défaut vient justement de la contrainte que nous lui imposons : la distance entre l'objet et la caméra ne doit pas changer. En calculant les angles d'observation dans le repère tangentiel à chaque point, l'échantillonnage nous permet de projeter une fonction qui représente de manière réelle la luminance observée pour ces conditions d'éclairage. On ne mesure plus cette fois l'apparence de l'objet tout entier pour un point de vue donné, mais bien une quantité physique relative à chaque point de la surface. La direction d'observation doit alors être évaluée en chaque point en fonction de la position de la caméra, ce qui peut être coûteux. Bien que ce calcul puisse être fait par un *pixel shader*, nous avons préféré utiliser un *cube map* [1] pour discrétiser la sphère d'observation en stockant un ensemble de vecteurs V_{SH}. Il vaut mieux utiliser une résolution assez importante pour cette texture car les harmoniques des bandes 6 ou supérieures sont des fonctions à très haute fréquence. L'interpolation linéaire sur cette grille discrète pourrait donner une reconstruction trop imprécise. Le processus de rendu est alors assez simple :

- Les coordonnées du vecteur d'observation sont calculées dans le repère local du point.
- Ce vecteur est projeté sur le *cube map*. L'interpolation matérielle nous donne une bonne approximation du vecteur V_{SH} local.
- Le calcul final reste le même :

$$L_e(\theta_{obs}, \phi_{obs}) = (f_1, \ldots, f_N) \cdot V_{SH}$$

Nous avons choisi la première méthode pour les raisons suivantes : d'abord, elle est plus simple à implémenter, surtout lorsqu'il s'agit de travailler sur des données relativement peu précises. Ensuite, l'outil que nous avons développé permet de visualiser de petits objets, bien cadrés à l'écran. Rapprocher ou éloigner la caméra ne présente pas un grand intérêt par rapport au fait de changer l'angle d'observation.

Intégration au modèle hiérarchique

Nous avons opté précédemment pour une technique d'affichage à base de points, travaillant à partir d'une hiérarchie pour contrôler du niveau de détail. Nous n'avions alors pas parlé de l'information chromatique puisque sa nature restait encore à préciser.
Les données issus de l'acquisition nous ont permises de calculer une luminance en chaque point du nuage. Il faut maintenant remonter cette information dans la hierarchie pour définir une luminance à tous les niveaux. Nous avons choisi de moyenner cette information comme cela est fait pour la couleur dans le modèle original des QSplats. En toute rigueur, cette façon de faire n'est pas physiquement correcte, puisque nous devrions tenir compte de la visibilité de chacun des fils. Nous travaillons ici sur des fonctions sphériques, et la projection en base SH est un processus très couteux en terme de temps de calcul. Nous allons voir comment il est possible d'en réduire le nombre en ne projetant que les fonctions contenues dans les feuilles. Le reste de la hiérarchie peut en être déduit.
Nous avons parlé un peu plus haut des propriétés inhérentes aux harmoniques sphériques, et plus particulièrement du fait que les coefficients peuvent être interpolés pour exprimer l'interpolation entre les fonctions qu'ils représentent. Celà nous permet de définir notre moyenne, non pas sur les échantillons de fonctions, mais sur les coefficients de projection eux-mêmes.

[1] texture représentée par les six faces d'un cube permettant de discrétiser une sphère de directions. Souvent utilisée pour des textures environnementales.

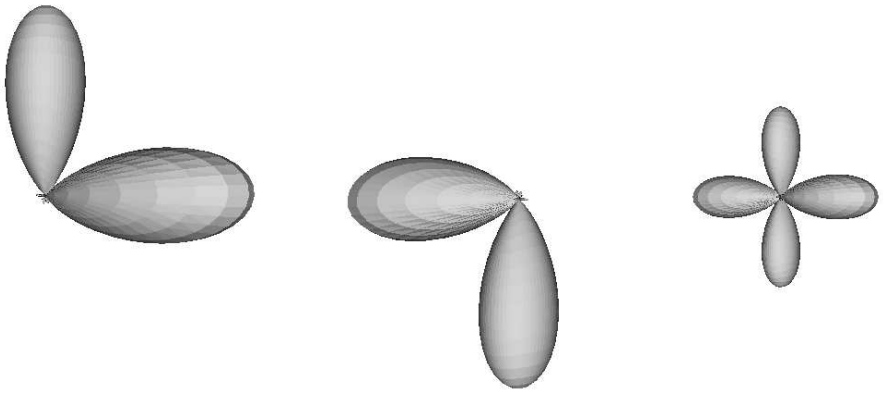


Fig. 3.11 – Droite et centre : deux fonctions sphériques projetées en base SH. **A droite** : la même fonction est obtenue, que l'on fasse une moyenne avant la projection, ou directement sur les coefficients.

Nous nous sommes également penché sur un moyen de compresser les coefficents eux-mêmes. Nous utilisons une projection d'ordre 6, ce qui représente 36 coefficients pour chaque fonction de luminance. Le modèle du pot à cactus est constitué de plus de 400 000 points, ce qui donne une hiérarchie de presque 800 000 nœuds. Les coefficients de projection demandent à eux seuls une capacité de stockage d'environ 110Mo. Nous avons divisé cette quantité par quatre en quantifiant l'intervalle de valeurs de manière à stocker chaque coefficient sur un seul octet.
Dans une première phase, nous recherchons le coefficient le plus grand (en valeur absolue), et nous l'utilisons comme diviseur pour normaliser les autres coefficients. Nous réduisons ainsi l'intervalle de valeurs à $[-1, 1]$, que nous discrétisons en l'intervalle entier $[-127, 127]$, stockable sur un octet. En représentant les coefficients du vecteur d'observation dans la base SH de la même manière, une implémentation purement logicielle peut travailler directement avec ces valeurs entières pour accélérer le processus d'affichage. Dans l'absolue, il est tout de même préférable de faire faire ces calculs par la carte graphique. Dans ce cas, cette méthode de compression permet au moins de limiter la taille des fichiers généré pour chaque modèle sur le disque dur. Il vaut mieux, pour une implémentation matérielle, se pencher sur les extensions OpenGL concernant les compressions de texture, qui permettent une décompression hardware des données.

Chapitre 4

Conclusion et perspectives

Nous avons détaillé ici une méthode de rendu basé points permettant une visualisation réaliste à partir d'un processus d'acquisition tenant compte des caractéristiques du matériau. Nous avons utilisé pour l'aquisition un scanner 3D par lumière structurée qui nous a été prêté par la société HOLO3. Construit sur un modèle hierarchique de sphères englobantes, le modèle peut être visualisé en temps réel grâce à un affichage adaptatif simulant une reconstruction de la surface au moment même du rendu.
L'acquisition se fait en deux temps : d'abord une numérisation 3D de l'objet nous permet d'obtenir sa géométrie, puis l'information chromatique est récupérée à l'aide d'une caméra annexe échantillonant du mieux possible la sphère d'observation. Les méthodes traditionnelles de recalage par reprojection dans l'espace image nécessitent une précision lors des mesures que nous ne pouvions malheureusement pas fournir par manque de moyen. Nous avons donc envisagé une nouvelle approche pour faire coïncider les données chromatiques avec la géométrie en exploitant une caractéristique du dispositif d'acquisition. La numérisation par lumière structurée projette une sinusoïdale sur l'objet sous forme de franges lumineuses. Le calcul de la géométrie passe d'abord par le calcul de la phase de cette sinusoïde. Nous avons exploité cette donnée pour définir une quantité unique pour chaque point de la surface et dont la valeur est indépendante du capteur utilisé. Cette propriété nous a permis de faire coïncider les différentes informations receuillies et de constuire un échantillonnage de la luminance sur toute la surface.
Ces données représentent en général une masse importante et leur exploitation brute pose souvent des problèmes de stockage en mémoire. Nous avons alors étudié deux méthodes permettant de représenter cette luminance d'une manière à la fois compacte et maniable. La première méthode, basée sur le modèle d'approximation non-linéaire de Lafortune [LAFORTUNE97], semble donner de très bons résultats ([LENSCH03], [LENSCH01]). En général, le modèle est utilisé pour décrire la fonction de réflectance, tenant compte des directions d'observation et d'illumination. Nous l'avons adapté à notre problème en fixant le vecteur d'incidence, ce qui implique d'utiliser une source lumineuse nécessairement ponctuelle. La représentation obtenue est très compacte, puisqu'elle ne nécessite que cinq coefficients par point.
Cependant, voulant travailler à partir d'un environnement lumineux quelconque, nous avons proposé une autre technique originale utilisant les harmoniques sphériques pour représenter notre fonction de luminance. La qualité de l'approximation dépend du nombre de coefficients utilisés pour reconstruire la fonction. Nous utilisons 36 coefficients (soit une projection d'ordre 6) ce qui nous offre une reconstruction suffisamment précise pour nos données. De plus, nous avons simplifié le modèle de manière à réduire cette reconstruction à un simple produit scalaire. L'évaluation est donc très rapide.

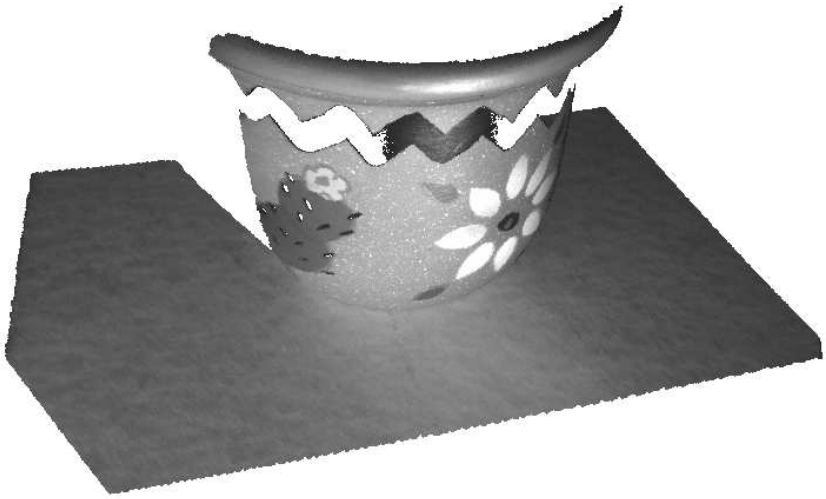
Ce logiciel de visualisation sera sujet à l'avenir à un certain nombre d'améliorations. La simplification apportée au modèle des harmoniques sphériques pour en accélérer l'évaluation impose

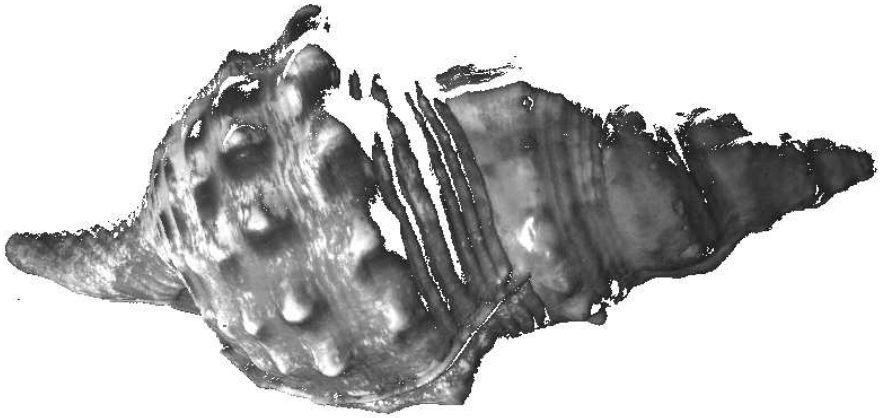
que la distance entre le point de vue et l'objet reste constante. Or nous avons exposé une autre façon de faire permettant de lever cette contrainte et donc de positionner l'observateur où bon nous semble. Nous mettrons en œuvre cette variante lorsque nous aurons la possibilité de faire des mesures précises, sans quoi celà ne présente pas grand intérêt. De même, nous n'avons pour le moment proposé qu'une implémentation purement logiciel, alors que les deux techniques (Lafortune et harmoniques) se prêtent tout particulierement à une exploitation du matériel graphique par programmation directe de la GPU (processeur graphique).
Le modèle hiérarchique peut être amélioré. Initialement, la luminance que nous avons mesurée correspond à celle des feuilles et nous avons moyenné ces fonctions pour reconstruire la luminance des nœuds internes. C'est en fait physiquement incorrect, puisque nous ne tenons pas compte de la visibilité de chaque sous-élément. Il faudra certainement modifier la hierarchie : les fonctions dans les niveaux supérieurs risquent d'être à très haute fréquence et l'ordre de projection nécessaire à une bonne reconstruction devra être augmenter en conséquence. Il faudra travailler sur un modèle permettant de moduler le nombre de coefficients en fonction de la profondeur d'un nœud lors du parcours de l'arbre.
Nous travaillerons également sur une approche bi-directionnelle. L'approximation par les lobes de Lafortune a fait ses preuves dans ce domaine ([LENSCH03], [LENSCH01]). Le modèle est simple et notre implémentation pour l'utiliser en éclairage fixe ne pose que quelques hypothèses supplémentaires sans y apporter de modification radicale. Les principaux changements se feront surtout au niveau des mesures. L'utilisation des harmoniques sphériques pour représenter une réflectance n'a quant à elle rien à voir avec l'approche adoptée ici pour le cas directionnel. Il faut utiliser non pas un vecteur mais une matrice de coefficients, ce qui augmente encore l'espace mémoire nécessaire et la méthode d'évaluation change elle aussi totalement ([SLOAN02],[KAUTZ02]).

Nous avons pu constater lors de ce projet à quel point la visualisation de données réelles n'est pas un problème trivial. La numérisation est en soit un processus délicat, soumis encore aujourd'hui à de nombreuses contraintes. Les objets trop brillants ou trop contrastés sont particulièrement difficiles à gérer. Mais la numérisation 3D n'est qu'une petite partie du processus d'acquisition d'objets réels. La géométrie n'est rien si l'apparence de l'objet n'est pas elle aussi capturée.
On sait aujourd'hui synthétiser des images d'un réalisme époustouflant, mais souvent au prix de temps de calcul très élevés. Les méthodes permettant une visualisation réaliste en temps réel sont aujourd'hui encore très limitées. Celles que nous avons illustrées ici sont les plus couramment utilisées car ce sont à ce jour les plus concluantes. D'autres méthodes, mises au point dans le même but, offrent soit une représentation très compacte, soit un calcul rapide, mais peinent souvent à rallier les deux.
Les techniques d'affichage font aussi l'objet d'évolutions intéressantes. L'utilisation d'un maillage est de moins en moins courant dans le domaine de la visualisation et dans les travaux concernant la numérisation 3D en particulier. Les données mesurées sont tellement denses que l'utilisation du point comme primitive s'impose presque comme une évidence, d'autant plus que, contrairement à d'autres domaines comme celui de la modélisation, le maillage n'est pas une donnée fondamentale. Le rendu basé points permet d'en faire abstraction en simulant une reconstruction de la surface au moment même de l'affichage.
Nous avons tenté ici d'unifier ces différentes technologies dans un seul et même but : restituer de la manière la plus fidèle qui soit l'apparence d'objets numérisés. Mais le problème est ardu et les choix à faire pour le résoudre sont multiples. Nous avons choisi la voie qui nous semblait la plus adaptée compte tenu des moyens mis à notre disposition. Les objectifs sont globalement atteints.

s

, Roll = -54.000, Distance = 395.000





Bibliographie

[BERNARDINI99] Bernardini, Mittleman, Rushmeier, Silva, Taubin, *The Ball-Pivoting Algorithm for Surface Reconstruction*, IEEE Transactions on Visualization and Computer Graphics, pp. 349-359, 1999.

[BILLARD98] Billard, *Mesure de formes 3D*, Diplôme de Recherche Technologique en Instrumentation, Télécommunication et Traitements d'Images, Institut Universitaire d'Ingénierie de la Vision, Saint Etienne, 1998.

[CHEN02] Chen, Bouguet, Chu, Grzeszczuk, *Light Field Mapping : Efficient Representation and Hardware Rendering of Surface Light Fields*, Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, pp. 447-456, San Antonio, Texas, 2002.

[CULA01] Cula, Dana, *Compact Representation of Bidirectional Texture Functions*, IEEE Conference on ComputerVision and Pattern Recognition, December 2001.

[DAVIS02] Davis, Marschner, Garr, Levoy, *Filling Holes in Complex Surfaces using Volumetric Diffusion*, 1st International Symposium on 3D Data Processing, Visualization, Transmission, Padova, Italy, 2002.

[ERKUT03] Erkut Erdem, Aykut Erdem, Atalay, *Image-Based Extraction of Material Reflectance Properties of a 3D Rigid Object*, EUROGRAPHICS, 2003.

[GREEN04] Green, *Spherical Harmonic Lighting : The Gritty Details*, R&D Programmer, Sony Computer Entertainment America, 2004.

[GROSSMAN98] Grossman, Dally, *Point Sample Rendering*, Proceedings of the 9th EUROGRAPHICS Workshop on Rendering, pp.181-192, 1998.

[HOPPE00] Hoppe, *Progressive Meshes*, SIGGRAPH 1996.

[KAUTZ02] Kautz, Sloan, Snyder, *Fast, Arbitrary BRDF Shading for Low-Frequency Lighting Using Spherical Harmonics*, Proceedings of the 13th EUROGRAPHICS Workshop on Rendering, pp. 291-296, Pisa, Italy, 2002.

[KOUDELKA03] Koudelka, Magda, Belhumeur, Kriegman, *Acquisition, Compression, and Synthesis of Bidirectional Texture Functions*, TEXTURE 2003, 3rd International Workshop on Texture Analysis and Synthesis, 2003.

[LAFORTUNE97] Lafortune, Foo, Torrance, Greenberg, *Non-Linear Approximation of Reflectance Functions*, 24th annual conference on Computer graphics and Interactive Techniques, SIGGRAPH, 1997.

[LENSCH01] Lensch, Kautz, Goesele, Heidrich, Seidel, *Image-Based Reconstruction of Spatially Varying Materials*, 12th EUROGRAPHICS Workshop on Rendering, 2001.

[LENSCH03] Lensch, Lang, Sá, Seidel, *Planned Sampling of Spatially Varying BRDFS*, EUROGRAPHICS Computer Graphics Forum, pp. 473-482, 2003.

[LEVOY96] Levoy, Hanrahan, *Light Field Rendering*, Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, pp. 31-42, SIGGRAPH, 1996.

[LEVOY00] Levoy, Pulli, Curless, Rusinkiewicz, Koller, Pereira, Ginzton, Anderson, Davis, Ginsberg, Shade, FulkLevoy, *The Digital Michelangelo Project : 3D Scanning of Large Statues*, Proceedings of the 27th annual Conference on Computer graphics and Interactive Techniques, SIGGRAPH, 2000.

[MALZBENDER96] Malzbender, Gelb, Wolters, *Polynomial Texture Maps*, SIGGRAPH 2001.

[MARSCHNER99] Marschner, Westin, Lafortune, Torrance, Greenberg, *Image-Based BRDF Measurement including Human Skin*, 10th EUROGRAPHICS Workshop on Rendering, 1999.

[MESETH03] Meseth, Müller, Klein, *Preserving Realism in Real-Time Rendering of Bidirectional Texture Functions*, OpenSG Symposium, 2003.

[ROCCHINI99] Rocchini, Cignoni, Montani, *Multiple Texture Stitching and Blending on 3D Objects*, Proceedings of the 10th EUROGRAPHICS Workshop on Rendering, pp. 127-138, Granada, 1999.

[RUSINKIEWICZ00] Rusinkiewicz, Levoy, *QSplat : a Multiresolution Point Rendering System for Large Meshes*, Proceedings of the 27th Annual Conference on Computer Graphics and Interactive techniques, pp. 343-352, SIGGRAPH, 2000.

[SLOAN02] Sloan, Kautz, Snyder, *Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments*, Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, pp. 527-536, SIGGRAPH, 2002.

[TONG02] Tong, Zhang, Liu, Wang, Guo, Shum, *Synthesis of Bidirectional Texture Functions on Arbitrary Surfaces*, SIGGRAPH 2002.

[YILMAZ02] Yilmaz, Mülayim, Atalay, *Reconstruction of Three Dimensional Models from Real Images*, International Symposium on 3D Data Processing, Visualization and Transmission, 2002.

[ZHANG97] Zhang, Hoff III, *Fast Backface Culling Using Normal Masks*, Proceedings of the Symposium on Interactive 3D graphics, 1997.

Annexe A

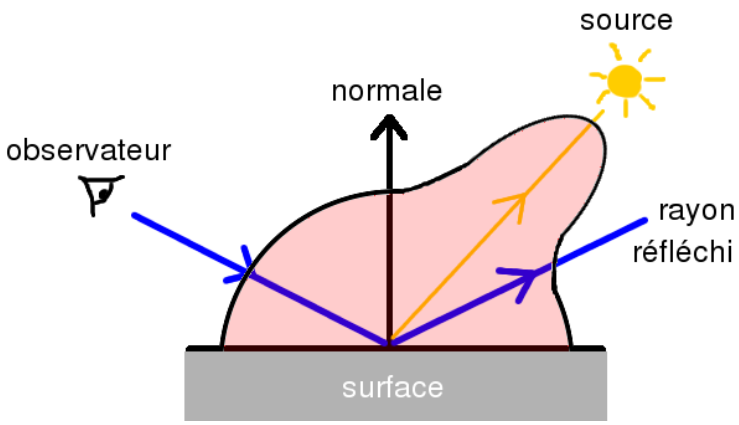
Modèle de Phong

Le modèle de Phong est un modèle d'illumination proposé en 1975 par le chercheur américain *Bui-Tuong Phong*, qui propose un lissage de la surface ainsi que la représentation des reflets spéculaires.

A.1 Le modèle

Ce modèle décompose l'illumination en un point en trois composantes :
- un terme ambient correspondant à une lumière résiduelle présente partout dans la scène,
- un terme diffus uniquement dépendant de la position de la source (souvent appelé *éclairage* dans le langage courant),
- un terme spéculaire représentant toute la partie directionnelle, c'est à dire dépendante de la position de l'observateur et de la source (appelé dans le langage courant *reflet*).

Soient $\vec{n}$ la normale à la surface, $\vec{l}$ le vecteur orienté vers la source, $\vec{o}$ le vecteur orienté vers l'observateur et $\vec{o_r}$ le vecteur d'observation réfléchi.



Le terme ambient est généralement considéré comme constant dans toute la scène. On le représente par L_a.
Le calcul du terme diffus est basé sur la loi de Lambert. Il est proportionnel au cosinus de l'angle formé entre les vecteurs $\vec{n}$ et $\vec{l}$, et s'exprime donc par :

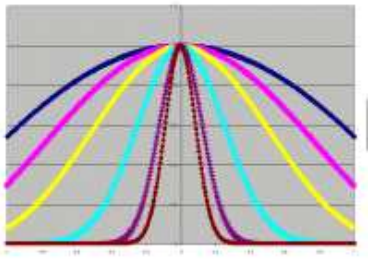
$$L_d(\vec{n} \cdot \vec{l})$$

Pour le calcul du terme spéculaire, Phong se base sur une observation : l'intensité perçue augmente lorsque la direction du rayon réfléchi devient proche de celle de la source. On utilise cette

fois le cosinus de l'angle entre les vecteurs $\vec{o_r}$ et $\vec{l}$:

$$L_s(\vec{o_r} \cdot \vec{l})^S$$

L'exposant S permet d'influencer la brillance du matériau (c'est à dire moduler la largeur du lobe spéculaire). On peut voir le comportement d'un lobe de cosinus lorsqu'il est élevé à une puissance :



L'illumination L en un point de la surface est donc calculé par la formule suivante :

$$L = L_a K_a + L_d K_d(\vec{n} \cdot \vec{l}) + L_s K_s(\vec{o_r} \cdot \vec{l})^S$$

avec L_a : l'intensité ambiente de la scène,
L_d, L_s : les intensités lumineuses émises par la source considérée pour les terme diffus et spéculaire,
K_a, K_d, K_s : les coefficients de réflexion ambiente, diffuse et spéculaire qui caractérise le matériau considé

A.2 Lissage

Le lissage de *Gouraud* (1971) calcul l'illumination pour chaque sommet et effectue un interpolation bi-linéaire (dégradé de couleur) sur le polygone. Phong propose d'interpoler la normale et de recalculer l'illumination en chaque pixel.

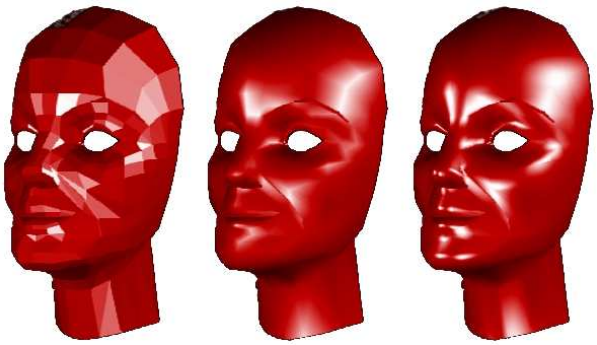


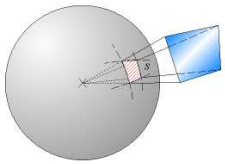
Fig. A.1 – A gauche : modèle à facette, sans lissage. **Au centre** : lissage de Gouraud, les reflets spéculaires sont considérablement altérés. **A droite** : lissage de Phong, qui réstitue bien l'illumination sur toute la surface.

Il est intéressant de noter que les modèles d'illumination sont souvent séparés en un composante diffuse et une composante spéculaire.

Annexe B

Glossaire de la lumière

Un angle solide est l'analogue tridimensionnel de l'angle plan. Au lieu de deux lignes se réunissant à un sommet, on considère un cône se réduisant à un point.
Pour trouver l'angle solide couvert par un objet, on considère une sphère centrée au point d'intersection de l'objet. Ensuite, on mesure la superficie de la partie de la sphère qui est contenue dans l'objet, et on la divise par la surface totale de la sphère.
Un angle solide se mesure par rapport à l'aire S de la surface d'une sphère de rayon unité (l'aire de cette surface vaut 4). Si le rayon R de la sphère est quelconque, on prend pour mesure $\frac{S}{R}$.
L'unité de mesure est le steradian (sr).



Le **flux lumineux**, Φ, désigne la lumière émise par une source en calculant le nombre de photons traversant une section pendant une seconde. Il s'agit d'une puissance mesurée en Watts (W).

L'**éclairement** (terme anglais : *irradiance*), E, est la puissance (flux) reçue par unité de surface :

$$E = \frac{\Phi}{S}$$

avec S : aire de la surface.
Exprimé en Watt / m^2.

La **luminance** (terme anglais : *radiance*), L, est un flux émis par angle solide et par unité de surface apparente (prise en compte de son orientation) :

$$L = \frac{\Phi}{S\ \omega \cos\theta}$$

Exprimé en $Watt.m^{-2}.sr^{-1}$.

La **réflectance**, ρ, est le rapport entre la luminance et l'éclairement :

$$\rho(u, v) = \frac{L(v)}{E(u)}$$

avec u : direction d'incidence,
v : direction de sortie.

Annexe C

Méthode d'optimisation linéaire Levenberg-Marquardt

C.1 Le problème

L'algorithme de Levenberg-Marquardt est une méthode permettant de minimiser au sens des moindres carrés une fonction d'erreur χ^2 de la forme :

$$\chi^2(x,\omega) = \frac{1}{2}\sum_{i=1}^{m} r_i^2(x,\omega)$$

où chaque $r_i : \mathbb{R}^n \to \mathbb{R}, n \leq m$, est appelé *fonction résiduelle*. χ^2 est fonction des variables x mais dépend d'un ensemble de paramètres ω fixés. Nous souhaitons déterminer ces paramètres de manière à ce que χ^2 soit minimale. Chaque fonction résiduelle r_i peut être associée à un échantillon pour lequel l'ensemble de variables x a été fixé. La méthode de résolution travaille donc en considérant χ^2 et r_i comme des fonctions de ω.

C.2 Méthode de Gauss-Newton

La méthode la plus simple et la plus intuitive pour trouver le minimum d'une fonction est certainement la *descente du gradient*, qui met à jour à chaque étape le vecteur de paramètres en suivant la pente de la fonction, et selon un pas λ :

$$\omega_{i+1} = \omega_i - \lambda \bigtriangledown \chi^2$$

Cette méthode souffre malheureusement de gros problèmes de convergence. Pour améliorer cette méthode, nous voudrions un pas adaptatif, qui soit grand lorsque la pente est douce et petit lorsqu'elle devient plus raide, pour être sûr de ne pas manquer le minimum.
La méthode de Gauss-Newton utilise la courbure de l'espace d'erreur (donnée par les dérivés secondes) pour influencer l'amplitude du gradient. On veut résoudre l'équation $\bigtriangledown\chi^2 = 0$. En se basant sur l'hypothèse que χ^2 est quadratique autour d'un point ω_0, on peut donné un développement de Taylor pour un ordre limité autour de ω_0 :

$$\bigtriangledown\chi^2(\omega) = \bigtriangledown\chi^2(\omega_0) + (\omega - \omega_0) \bigtriangledown^2 \chi^2(\omega_0)$$

En posant $\bigtriangledown\chi^2(\omega) = 0$,

$$\bigtriangledown\chi^2(\omega_0) + (\omega - \omega_0) \bigtriangledown^2 \chi^2(\omega_0) = 0 \tag{C.1}$$

$$(\omega - \omega_0) \bigtriangledown^2 \chi^2(\omega_0) = - \bigtriangledown \chi^2(\omega_0) \tag{C.2}$$

$$(\omega - \omega_0) = -\frac{\bigtriangledown\chi^2(\omega_0)}{\bigtriangledown^2\chi^2(\omega_0)} \tag{C.3}$$

On obtient alors la règle de mise à jour pour la méthode de Gauss-Newton :

$$\omega_{i+1} = \omega_i - \frac{\bigtriangledown\chi^2}{\bigtriangledown^2\chi^2}$$

C.3 Algorithme de Levenberg-Marquardt

La méthode exposée précédemment à l'avantage de converger rapidement, mais peut aussi très mal fonctionner si les paramètres initiaux ont été mal choisis. Levenberg-Marquardt combine les avantages de la descente du gradient simple et de Gauss-Newton, en utilisant la règle de mise à jour suivante :

$$\omega_{i+1} = \omega_i - \frac{\bigtriangledown\chi^2}{(\bigtriangledown^2\chi^2 + \lambda I)}$$

Si, à la suite d'une actualisation, l'erreur diminue, celà veut dire que l'hypothèse précédente de quadraticité locale sur χ^2 est correcte, et dans ce cas on diminue λ pour réduire l'influence de la descente du gradient. Si, au contraire, l'erreur augmente, l'hypothèse n'est plus justifiée, et on augmente λ pour suivre d'avantage le gradient, et réduire l'influence de Gauss-Newton.
On peut observer que si λ devient très grand, l'influence du Hessien (dérivés seconde) devient quasi-nulle. On peut pourtant s'en servir pour influencer le gradient selon la courbure. Cette amélioration, apportée par Marquardt, remplace la matrice identité de l'équation précédente par la diagonale de la matrice hessienne :

$$\omega_{i+1} = \omega_i - \frac{\bigtriangledown\chi^2}{(\bigtriangledown^2\chi^2 + \lambda diag(\bigtriangledown^2\chi^2))}$$

C.4 Calcul des dérivés

Pour implémenter l'algorithme de Levenberg-Marquardt, il faut calculer les dérivés premières et secondes de la fonction $\chi^2(x,\omega) = \frac{1}{2}\sum_{i=1}^{m} r_i^2(x,\omega)$ par rapport à ω.
Posons $r = (r_1, \ldots, r_m)$.
Pour les dérivés premières :

$$\begin{aligned}
\bigtriangledown\chi^2(\omega) &= \frac{1}{2}\sum_{i=1}^{m}\bigtriangledown(r_i^2(\omega)) = \sum_{i=1}^{m} r_i(\omega)\bigtriangledown r_i(\omega) && \text{(C.4)}\\
&= r_1(\omega)\begin{pmatrix} \frac{\partial r_1}{\partial\omega_1} \\ \vdots \\ \frac{\partial r_1}{\partial\omega_n} \end{pmatrix} + \cdots + r_m(\omega)\begin{pmatrix} \frac{\partial r_m}{\partial\omega_1} \\ \vdots \\ \frac{\partial r_m}{\partial\omega_n} \end{pmatrix} && \text{(C.5)}\\
&= \begin{pmatrix} \frac{\partial r_1}{\partial\omega_1} & \cdots & \frac{\partial r_m}{\partial\omega_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_1}{\partial\omega_n} & \cdots & \frac{\partial r_m}{\partial\omega_n} \end{pmatrix}\begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix} && \text{(C.6)}\\
&= J_r^T(\omega)r(\omega) && \text{(C.7)}
\end{aligned}$$

où J_r est le Jacobien du vecteur de fonctions r par rapport à ω.
Pour les dérivées secondes :

$$\bigtriangledown^2\chi^2(\omega) = J_r^T(w)J_r(w) + \sum_{i=1}^{m} r_i(\omega)\bigtriangledown^2 r_i(\omega)$$

Si les fonctions résiduelles r_i sont petites, ou si elles peuvent être approximées par des fonctions linéaires ($\bigtriangledown r_i(\omega)$ petits), on peut simplifier le calcul de la matrice hessienne par :

$$\bigtriangledown^2\chi^2(\omega) = J_r^T(\omega)J_r(\omega) \tag{C.8}$$

$$= \begin{pmatrix} \frac{\partial r_1}{\partial \omega_1} & \cdots & \frac{\partial r_m}{\partial \omega_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_1}{\partial \omega_n} & \cdots & \frac{\partial r_m}{\partial \omega_n} \end{pmatrix} \begin{pmatrix} \frac{\partial r_1}{\partial \omega_1} & \cdots & \frac{\partial r_1}{\partial \omega_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial \omega_1} & \cdots & \frac{\partial r_m}{\partial \omega_n} \end{pmatrix} \tag{C.9}$$

$$= \begin{pmatrix} \sum \frac{\partial r_i}{\partial \omega_1}\frac{\partial r_i}{\partial \omega_1} & \cdots & \sum \frac{\partial r_i}{\partial \omega_1}\frac{\partial r_i}{\partial \omega_n} \\ \vdots & \ddots & \vdots \\ \sum \frac{\partial r_i}{\partial \omega_1}\frac{\partial r_i}{\partial \omega_n} & \cdots & \sum \frac{\partial r_i}{\partial \omega_n}\frac{\partial r_i}{\partial \omega_n} \end{pmatrix} \tag{C.10}$$

C.5 Application au modèle de Lafortune

Le modèle non-linéaire de Lafortune pour approximer la fonction de réflectance est généralement utilisé avec deux lobes, dont l'un est considéré comme constant. On le représente alors sous la forme :

$$L_e(u,v) = \rho_d + (C_x u_x v_x + C_y u_y v_y + C_z u_z v_z)^n$$

Les paramètres à déterminer sont alors $\omega = (\rho_d, C_x, C_y, C_z, n)$, et les fonctions résiduelles sont définies par :

$$r_i(\omega) = L_e(u_i, v_i) - L_{m,i}$$

avec u_i : direction d'incidence lumineuse pour le i-ème échantillon,
v_i : direction d'observation pour le i-ème échantillon,
$L_{m,i}$: luminance observée pour cet échantillon.

Pour calculer les dérivés de χ^2, il nous faut connaître les dérivés partielles de r_i par rapport à ω. $L_{m,i}$ n'est pas un terme dépendant de ω, et on a donc $\frac{\partial r_i}{\partial \omega} = \frac{\partial L_e}{\partial \omega}(u_i, v_i)$. Il nous suffit donc de connaître les dérivés partielles de L_e :

$$\frac{\partial L_e}{\partial \rho_d} = 1 \tag{C.11}$$

$$\frac{\partial L_e}{\partial C_x} = n(C_x u_x v_x + C_y u_y v_y + C_z u_z v_z)^{n-1} u_x v_x \tag{C.12}$$

$$\frac{\partial L_e}{\partial C_y} = n(C_x u_x v_x + C_y u_y v_y + C_z u_z v_z)^{n-1} u_y v_y \tag{C.13}$$

$$\frac{\partial L_e}{\partial C_z} = n(C_x u_x v_x + C_y u_y v_y + C_z u_z v_z)^{n-1} u_z v_z \tag{C.14}$$

$$\frac{\partial L_e}{\partial n} = n(C_x u_x v_x + C_y u_y v_y + C_z u_z v_z)^{n} \log(C_x u_x v_x + C_y u_y v_y + C_z u_z v_z) \tag{C.15}$$