Etude en vue de la multirésolution de l'apparence

Julien Hadim

N° d'ordre : 3794

THÈSE

PRÉSENTÉE À

L'UNIVERSITÉ BORDEAUX 1

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Par **Julien Hadim**

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : INFORMATIQUE

Étude en vue de la multirésolution de l'apparence

Soutenue le : 11 Mai 2009

Après avis des rapporteurs :

Kadi Bouatouch	Professeur
Jean-Michel Dischler	Professeur

Devant la commission d'examen composée de :

Bruno Jobard	Maître de Conférence	Examinateur
Kadi Bouatouch	Professeur	Rapporteur
Christophe Schlick ..	Professeur	Directeur de Thèse
Xavier Granier	Chargé de Recherche	Co-Directeur de Thèse
Jean-Michel Dischler	Professeur	Président du Jury

– 2009 –

Remerciements

Les travaux présentés dans cette thèse n'auraient pas été possible sans l'aide ni les encouragements de bon nombre de gens. Je tiens tout d'abord à remercier mes deux directeurs de thèse, Christophe Schlick et Xavier Granier, pour la confiance qu'ils ont su m'accorder durant ces trois années de thèse et les deux années qui l'ont précédée. Je remercie Christophe Schlick, sans qui je n'aurais sans doute jamais démarré cette thèse, pour sa gentillesse infinitésimale et ses qualités pédagogiques et scientifiques. Je remercie Xavier Granier, sans qui cette thèse n'aurait peut-être jamais abouti, pour avoir accepté de travailler avec moi, sa gentillesse, ses encouragements, ses qualités scientifique et sa disponibilité sans faille.

Je tiens à remercier mes deux rapporteurs de thèse, Kadi Bouatouch et Jean-Michel Dischler, pour leur interêt envers les travaux présentés dans ce document, pour leur participation à la relecture et l'expertise qu'ils en ont fournit. Je remercie Bruno Jobard pour son interêt envers les travaux présentés et pour avoir accepté d'être examinateur lors de la soutenance. Enfin, je les remercie d'avoir bien voulu se déplacer à Bordeaux pour faire partie du jury lors de la soutenance. Je tiens aussi à remercier encore une fois Jean-Michel Dischler pour avoir présidé ce jury.

Je remercie la Région Aquitaine et l'INRIA, les organismes qui ont financé ces travaux.

Je remercie Pascal Guitton pour m'avoir accepté sans conditions au sein de l'équipe-projet IPARLA et de son soutien dans l'obtention de finance pour ces travaux. Je remercie aussi en général tous les membres de l'équipe-projet IPARLA où travail scientifique et bonne humeur se marient parfaitement. Plus particulièrement, je remercie Patrick Reuter, Martin Hachet, Carole Blanc, Salvatore Spinello et Joachim Pouderoux pour leurs encouragements et les discussions que nous avons pu avoir.

Plus particulièrement, je tiens à remercier Tamy Boubekeur pour son amitié, pour toutes ses années joviales partagée que ce soit en journée, en soirée ou encore à l'étranger mais aussi pour ses encouragements, nos collaborations, nos discussions scientifiques ou non. Je remercie Florian Levet et Michaël Raynaud pour nos collaborations dans certains travaux. Je remercie Romain Pacanowski pour son amitié, son interêt pour mes travaux, ses encouragements et sa motivation à toute épreuve.

Enfin, je remercie tous mes amis qui m'ont soutenu et encouragé durant ces trois années et demi. Xavier Caubet pour ses réveils matin et la préparation du café, Xavier Hanna pour ses encouragements et son aide durant la rédaction de ce mémoire, Elisabeth Brunet pour la compétition de rédaction du mémoire et ses encouragements, mais aussi tous ceux que j'oublie, excusez-moi. Je tiens à remercier ma famille qui m'a toujours soutenu moralement et financièrement dans mes études, surtout ma petite maman et mes grands-parents.

Étude en vue de la multirésolution de l'apparence

Résumé :

Les fonctions de texture directionnelle (« *Bidirectional Texture Function* » ou BTF) ont rencontré un certain succès ces dernières années dans le contexte de la synthèse d'images en temps-réel grâce à la fois au réalisme qu'elles apportent et au faible coût de calcul nécessaire. Cependant, un inconvénient de cette approche reste la taille gigantesque des données et de nombreuses méthodes ont été proposées afin de les compresser. Dans ce document, nous proposons une nouvelle représentation des BTFs qui améliore la cohérence des données et qui permet ainsi une compression plus efficace de celles-ci.

Dans un premier temps, nous étudions les méthodes d'acquisition et de génération des BTFs et plus particulièrement, les méthodes de compression adaptées à une utilisation sur cartes graphiques. Nous réalisons ensuite une étude à l'aide de notre logiciel *BTFInspect* afin de déterminer parmi les différents phénomènes visuels mesurés dans les BTFs, ceux qui influencent majoritairement la cohérence des données par pixel.

Dans un deuxième temps, nous proposons une nouvelle représentation pour les BTFs, appelées « *Flat Bidirectional Texture Function* » Flat-BTFs, qui améliore la cohérence des données d'une BTF et donc la compression des données. Cette nouvelle représentation est alors implémentée avec des BTFs synthétiques afin de valider sa mise en œuvre. Dans l'analyse des résultats obtenus, nous montrons statistiquement et visuellement le gain de cohérence obtenu ainsi que l'absence d'une perte significative de qualité en comparaison avec la représentation d'origine.

Enfin, dans un troisième temps, nous validons l'utilisation de notre nouvelle représentation dans des applications de rendu en temps-réel sur cartes graphiques. Puis, nous proposons une compression de l'apparence grâce à une méthode de quantification adaptée et présentée dans le cadre d'une application de diffusion de données 3D entre un serveur contenant des modèles 3D et un client désirant visualiser ces données.

Discipline : Informatique

Mots-clés : Synthèse d'images, rendu en temps-réel, apparence réaliste, multirésolution, BTF, méso-structure, GPU, diffusion de données 3D.

LaBRI - INRIA Bordeaux Sud-Ouest - Région Aquitaine
Université Bordeaux I
351 Cours de la Libération
33405 Talence Cedex (FRANCE)

Study for Multiresolution Appearance

Abstract :

In recent years, *Bidirectional Texture Function* (BTF) has emerged as a flexible solution for realistic and real-time rendering of material with complex appearance and low cost computing. However one drawback of this approach is the resulting huge amount of data : several methods have been proposed in order to compress and manage this data. In this document, we propose a new BTF representation that improves data coherency and allows thus a better data compression.

In a first part, we study acquisition and digital generation methods of BTFs and more particularly, compression methods suitable for GPU rendering. Then, We realise a study with our software *BTFInspect* in order to determine among the different visual phenomenons present in BTF which ones induce mainly the data coherence per texel.

In a second part, we propose a new BTF representation, named *Flat Bidirectional Texture Function* (Flat-BTF), which improves data coherency and thus, their compression. The analysis of results show statistically and visually the gain in coherency as well as the absence of a noticeable loss of quality compared to the original representation.

In a third and last part, we demonstrate how our new representation may be used for realtime rendering applications on GPUs. Then, we introduce a compression of the appearance thanks to a quantification method on GPU which is presented in the context of a 3D data streaming between a server of 3D data and a client which want visualize them.

Discipline : Computer science

Keywords : Computer graphics, realtime 3D rendering, realistic appearance, multiresolution, BTF, mesostructure, GPU, 3D data streaming.

LaBRI - INRIA Bordeaux Sud-Ouest - Région Aquitaine
Université Bordeaux I
351 Cours de la Libération
33405 Talence Cedex (FRANCE)

Table des matières

Introduction **1**
Contexte . . . 1
Motivations . . . 2
Contributions . . . 3
Organisation du document . . . 3

I Bidirectional Texture Function **5**

1 L'apparence des objets **7**
1.1 L'apparence . . . 8
1.2 Réflectance en un point d'une surface . . . 9
1.3 Variation de la réflectance pour une surface plane . . . 11
1.4 Variation de la réflectance pour une surface arbitraire . . . 12
1.5 Les BTFs . . . 13
1.6 Les domaines de recherche autour des BTFs . . . 14

2 Création des BTFs **17**
2.1 L'acquisition de BTFs . . . 17
2.1.1 Les systèmes à composants mobiles . . . 17
2.1.2 Les systèmes optiques . . . 19
2.1.3 Les systèmes à composants fixes . . . 19
2.1.4 Autres systèmes . . . 21
2.1.5 Discussion . . . 21
2.2 La génération de BTFs . . . 22
2.3 Discussion . . . 23

3 Représentations compactes de BTFs **25**
3.1 Minimisation de modèles paramétriques . . . 26
3.1.1 Minimisation par texel . . . 26
3.1.2 Minimisation par texel et par vue . . . 27
3.2 Réduction de dimensionnalité . . . 27
3.2.1 Réduction sur données complètes . . . 27
3.2.2 Réduction par texel . . . 28
3.2.3 Réduction par texel et par vue . . . 28
3.2.4 Réduction par partitions . . . 28
3.3 Discussion . . . 30

4 BTFInspect : Un outil d'analyse et d'étude de BTFs **33**
4.1 Manipulation des BTFs . . . 34
4.1.1 Motivations . . . 34
4.1.2 Structure de données . . . 34
4.2 Étude du phénomène de parallaxe . . . 36
4.2.1 Mise en valeur de la méso-structure sous-jacente . . . 36
4.2.2 Observation dans les images . . . 36
4.3 Étude des ABRDFs . . . 37
4.3.1 Vue en image . . . 37
4.3.1.1 Construction . . . 37
4.3.1.2 Interprétation des images . . . 38
4.3.2 Vue en 3D . . . 39
4.3.2.1 Construction . . . 40
4.3.2.2 Exemple d'utilisation avec les textures polynomiales (PTM) . . . 40

Bilan de notre étude **43**

II Nouvelle représentation des BTFs : décorrélation de la parallaxe **45**

5 Une nouvelle représentation des BTFs : les Flat-BTFs **47**
5.1 Reconstruction de la méso-structure des BTFs . . . 48
5.1.1 Reconstruction par stéréovision . . . 48
5.1.2 Reconstruction par photométrie . . . 48
5.1.3 Mesure de la méso-structure . . . 49
5.1.4 Discussions . . . 49
5.2 Rendu de méso-structures . . . 50
5.3 Définition de la représentation Flat-BTF . . . 52
5.4 Conversion d'une BTF en Flat-BTF . . . 54
5.5 Discussion . . . 54

6 Génération de Flat-BTFs **57**
6.1 Modélisation de méso-structures . . . 58
6.2 Paramétrisation globale . . . 59
6.3 Texturation de la méso-structure et BRDF . . . 59
6.4 Synthèse de BTFs classiques . . . 60
6.5 Synthèse de la fonction Flat-BTF . . . 61
6.5.1 Images géométriques d'une méso-structure . . . 61
6.5.2 Synthèse de l'apparence . . . 62
6.5.3 Traitement des texels indéfinis . . . 63
6.6 Synthèse des cartes d'indirections : la fonction VDIM . . . 63
6.7 Discussion . . . 64

7 Résultats expérimentaux et analyse **65**
7.1 Jeu de données . . . 65
7.2 Mesure de la cohérence de l'apparence . . . 66
7.3 Etude visuelle des ABRDFs . . . 67
7.4 Erreur de reconstruction . . . 68
7.5 Analyse globale . . . 70

III Mise en œuvre : encodages pour un rendu efficace 73

8 Implémentation et utilisation des Flat-BTFs sur GPU 75
8.1 Utilisation des Flat-BTFs sur GPU . . . 76
8.2 Encodage GPU de la fonction VDIM . . . 77
8.3 Erreur d'échantillonnage . . . 78
8.4 Implémentation de la fonction VDIM sur GPU . . . 78
8.5 Résultats : performance et qualité visuelle . . . 79
8.6 Discussion . . . 80

9 Quantification de l'apparence 85
9.1 Contexte d'application . . . 85
9.2 Système client/serveur . . . 87
9.3 Quantification de l'apparence adaptée au GPU . . . 87
9.3.1 Quantification des normales . . . 89
9.3.2 Déquantification des normales . . . 90
9.3.3 Quantification/déquantification des couleurs . . . 90
9.4 Implémentation . . . 91
9.4.1 Encodage . . . 91
9.4.2 Optimisation du transfert pour les entrées/sorties . . . 91
9.4.3 Décodage . . . 91
9.5 Expérimentations . . . 93
9.5.1 Temps de quantification . . . 93
9.5.2 Performance de la diffusion des données . . . 93
9.5.3 Erreur de quantification . . . 94
9.6 Conclusion . . . 94

Conclusion générale 97
Résumé des contributions . . . 97
Travaux futurs et perspectives . . . 98

Liste des figures 101

Liste des tableaux 107

Bibliographie 107

Introduction

Contexte

Depuis son apparition, la synthèse d'image 3D connaît une croissance supportée par celle des ordinateurs en termes de puissance de calcul et de capacité mémoire. Issus du monde de la CAO[1] et de la simulation scientifique, les secteurs de l'industrie du film et du jeu vidéo constituent désormais les principaux moteurs de son évolution. Que ce soit pour créer des effets spéciaux dans des films traditionnels ou bien la réalisation complète de films d'animation, les images générées par ordinateur ont atteint un réalisme capable de tromper les spectateurs les plus avertis. Ce réalisme demande des temps de calcul qui de nos jours peuvent encore durer plusieurs semaines à grand renfort de fermes de calcul comprenant plusieurs centaines d'ordinateurs. Le secteur du jeu vidéo ne bénéficie pas encore du même niveau de réalisme car les interactivités avec le monde virtuel requièrent des temps de calcul de moins d'$^1/_{25^{ème}}$ de seconde pour créer une image en temps-réel. Au fil des progrès technologiques des cartes graphiques et de l'évolution des techniques de synthèse, la qualité du rendu en temps-réel et particulièrement dans les jeux vidéo ne cesse de se rapprocher du réalisme obtenu par un rendu pré-calculé pour les œuvres cinématographiques. Ces deux mondes distincts qui s'opposent l'un par sa contrainte de temps illimité et sa rigidité, l'autre par son interactivité et des temps de calculs réduits à l'extrême, semblent au contraire converger dans une même direction au travers d'une course dont le but final est la reproduction virtuelle, réaliste et interactive d'un monde réel ou imaginaire.

Dans cette course au réalisme, le domaine de l'acquisition s'est fortement développé ces dernières années au point de prendre une place très importante parmi les thèmes classiques de la synthèse d'image 3D que sont la modélisation, la synthèse et l'animation. Les avancées technologiques des scanners 3D, des appareils photos numériques ou de tout autre dispositif de mesure permettent aujourd'hui de capturer la forme et l'apparence des entités du monde réel en partant de petits objets jusqu'aux rues entières d'une grande ville avec ses bâtiments. Des imprimantes 3D permettent même de reconstruire des copies réelles pour des petits objets comme on numérise un document papier avant d'en imprimer plusieurs copies. Cette capture de la réalité donne une représentation numérique dense, précise et directe de la réalité, elle évite le recours à des processus coûteux de simulation physique censés reproduire les formes et leurs apparences.

En parallèle, la miniaturisation et la portabilité des ordinateurs progressent tout autant. Ainsi les téléphones portables d'aujourd'hui sont assimilables à de vrais ordinateurs en miniatures. Les écrans qui désormais peuvent tenir dans une main deviennent de plus en plus précis, la capacité mémoire augmente et le tout avec une gestion plus efficace de l'énergie. La synthèse d'image 3D se retrouve aussi sur ce type de support au travers d'applications 3D interactives comme par exemple des outils de géolocalisation et de navigation, de la visualisation d'objet 3D ou encore des jeux vidéos. Les futurs

[1] CAO pour Conception Assisté par Ordinateur : rassemble les techniques permettant de modéliser et de concevoir virtuellement des produits manufacturés.

développements en synthèse d'images doivent aussi prendre en compte l'évolution de ces périphériques et tout particulièrement le fait que leurs performances sont en retrait de quelques générations par rapport aux ordinateurs fixes.

Motivations

Dans ce contexte, les méthodes permettant de définir l'apparence des objets 3D jouent un rôle majeur dans la synthèse d'images réalistes parce qu'elles permettent de déterminer les interactions entre la lumière d'un environnement et la matière des objets. De nos jours, l'ensemble des problèmes physiques relatifs à l'obtention d'une apparence réaliste a été identifié. Si des solutions satisfaisantes existent dans le cadre de la synthèse d'images fixes, ce n'est pas encore le cas pour la synthèse d'images en temps-réel. À l'heure actuelle, les méthodes qui définissent une apparence réaliste en temps-réel progressent au fil des avancées technologiques des cartes graphiques.

De nouvelles approches voient aussi le jour et tout particulièrement avec la démocratisation de nouveaux matériels d'acquisition tels que les appareils photos numériques. Ces dernières années, des dispositifs spécifiques ont été développés pour mesurer l'apparence des objets réels qui une fois stockée sous une forme numérique peut être utilisée directement pour une synthèse d'image réaliste en temps-réel. L'innovation de cette approche est d'apporter une représentation réaliste de l'apparence des objets ne nécessitant pas l'utilisation de modèles physiques destinés à simuler la réalité au prix de calculs complexes.

La notion de multi-résolution est une propriété importante pour les méthodes définissant l'apparence des objets 3D. Une méthode est dite multi-résolution si elle peut fournir différents niveaux de détails d'une même apparence en fonction des conditions de visualisation des objets. Par exemple, en fonction de la taille selon laquelle un objet apparaît dans une image, une méthode multi-résolution doit pouvoir fournir le niveau de détails qui correspond le mieux au compromis entre la qualité de l'apparence et le temps imparti pour une synthèse en temps-réel. Il est inutile de consommer trop de ressources pour un objet qui ne sera représenté au final que par quelques pixels à l'écran.

D'une manière générale, nous nous sommes intéressés durant ces travaux de thèse aux méthodes qui définissent une apparence multi-résolution pour les objets 3D dans le cadre de la synthèse d'images en temps-réel. Après une étude approfondie des méthodes existantes et émergentes, nous nous sommes focalisés plus particulièrement sur des textures à six dimensions obtenues par un procédé d'acquisition sur des matériaux réels : les « *Bidirectional Texture Fonctions* » (BTF).

Contributions

Dans ce document, nous présentons une étude sur les « *Bidirectional Texture Fonctions* » (BTF) réalisée à l'aide du logiciel *BTFInspect* qui a été développé à cet effet. Au cours de cette étude, nous détaillons et analysons les différents phénomènes visuels et lumineux qui entrent en jeu dans les BTFs et montrons que ce sont les effets visuels se rapportant au relief qui perturbent le plus les méthodes de compression des BTFs en terme de variance des données par texel.

Nous proposons ensuite une nouvelle représentation des BTFs, appelée « *Flat Bidirectional Texture Functions* » (Flat-BTF), dont la principale particularité est de stocker séparément les effets de parallaxe des autres effets d'illuminations présents dans les BTFs. Nous démontrons à l'aide de BTFs de synthèse que cette nouvelle représentation améliore la cohérence des données par texel ce qui en favorise la compression. L'utilisation des Flat-BTFs reste compatible avec les utilisations usuelles des BTFs sur cartes graphiques dans leur représentation standard. Ces travaux sont en cours de soumission [HGS08].

Enfin dans le contexte d'une représentation de l'apparence plus simple et classique, encodée par une normale et une couleur, nous proposons un schéma de compression et de quantification de cette apparence au sein d'une application de diffusion de données 3D entre un serveur contenant des modèles 3D et un client désirant visualiser ces données. Ces travaux ont été publiés [HBR+07].

Organisation du document

La suite de ce document est organisée en trois grandes parties :

La première partie intitulée « *Bidirectional Textures Funtions* » (BTF) commence avec un tour d'horizon des méthodes existantes pour définir l'apparence des objets dans la synthèse d'images en temps-réel (Chap. 1). Ensuite nous présentons un état de l'art sur les BTFs et plus particulièrement sur l'acquisition (Chap. 2) et les représentations compactes (Chap. 3). Puis, nous exposons les différentes fonctionnalités de notre logiciel *BTFInspect* suivi d'une étude des données de différentes BTFs concernant les différents effets capturés ainsi que leur influence au niveau des méthodes de compression (Chap. 4). Cette partie se conclut par une analyse et une discussion des résultats de notre étude (Chap. 4.3.2.2).

La deuxième partie intitulée « *Flat Bidirectional Texture Functions* » introduit la nouvelle représentation que nous proposons pour les BTFs (Chap. 5). Nous détaillons ensuite les différentes étapes nécessaires à son implémentation pour des BTFs de synthèse (Chap. 6) suivi d'une analyse des avantages de cette nouvelle représentation pour la cohérence des données et la préservation de qualité des données en comparaison avec la représentation standard (Chap. 7).

Enfin, la troisième et dernière partie intitulée « Mise en œuvre » présente une méthode de rendu en temps-réel utilisant des données de la représentation Flat-BTF (Chap. 8). Nous proposons une méthode de compression et de quantification de l'apparence dans le cadre d'une application client/serveur pour de la diffusion de données 3D (Chap. 9).

Première partie

Bidirectional Texture Function

Chapitre 1

L'apparence des objets

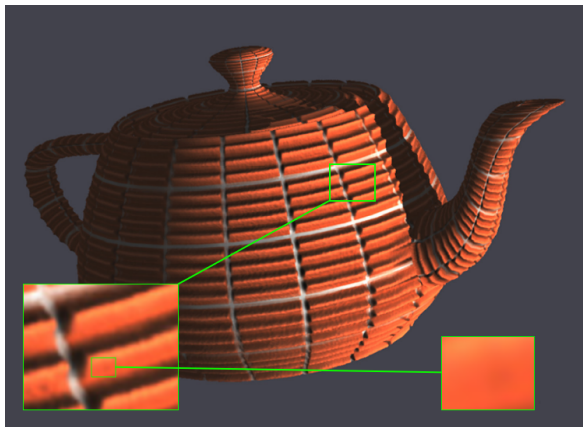


Fig. 1.1: *À gauche, un rendu réaliste en temps-réel pour une scène représentant l'intérieur d'une maison avec ses meubles. L'apparence réaliste des objets est ici définie par les SVBRDFs de McAllister et al. [MLH02] (introduites dans la Section 1.3). À droite, un objet en forme de théière issu des travaux de Wang et al. [WTS+05] et dont l'apparence de la surface reproduit un mur de briques à l'aide de BTFs (définies dans la Section 1.5).*

La synthèse d'images réalistes reproduit, modélise et simule des éléments du monde réel dans le but d'en obtenir une représentation virtuelle sur ordinateur. L'ensemble de ces éléments numériques constitue alors une scène dont les principaux composants rappellent ceux que l'on retrouverait sur la scène de tournage d'un film :

- des sources lumineuses éclairant la scène et constituant un environnement lumineux ;
- des objets, animés ou non, représentant un acteur ou participant au décor de la scène ;
- des caméras pour prendre des prises de vue de la scène.

En synthèse d'images, le terme de « rendu » désigne le procédé qui consiste à construire une image en fonction de ces différents composants. La majeure partie des calculs contribue à résoudre l'ensemble des échanges énergétiques et photométriques qui ont lieu dans la scène à un instant donné entre la lumière des différentes sources lumineuses et la matière des objets constituant la scène. Cette information lumineuse capturée par les caméras virtuelles permet de créer les images de synthèse. Le rendu en temps-réel repose sur le fait de pouvoir calculer au minimum 25 images par seconde. En dessous, le rendu est interactif jusqu'à quelques images par seconde. Le rendu en temps-réel est traditionnellement obtenu par rastérisation car les algorithmes relatifs à cette méthode étaient plus adaptés pour une accélération matérielle à l'époque des premières cartes graphiques 3D. D'une manière générale, plus les composants d'une scène sont détaillés et fidèles à la réalité, et plus le rendu de la scène pourra être réaliste.

Les objets 3D d'une scène sont caractérisés par leur forme et l'apparence de leur surface. Pour simplifier, on peut dire que la forme d'un objet permet de désigner son occupation de l'espace 3D, et donc dans l'image. L'apparence de sa surface permet de définir la couleur de chacun de ses pixels. La

forme et l'apparence de la surface des objets sont en général définies par des méthodes disjointes, cela permet entre autres de définir plusieurs apparences de surface pour des objets à la forme identique. La forme des objets, généralement géométrique, est soit modélisée par des artistes ou bien mesurée à l'aide d'un scanner laser 3D. De même, l'apparence de la surface des objets peut être acquise à partir d'objets réels ou bien définie par des artistes. De nombreuses méthodes existent pour la représenter en fonction de la complexité de l'apparence et du degré de réalisme visé. Dans la suite de ce chapitre, nous passons en revue les différentes méthodes usuelles et plus particulièrement celles profitant des processus d'acquisition pour capturer l'apparence d'objets réels et utilisables dans le cadre d'un rendu en temps-réel.

1.1 L'apparence

L'apparence pour la surface d'un objet nous renvoie à ce que notre système visuel perçoit. Cette apparence correspond donc à l'information lumineuse qui est renvoyée par la surface de l'objet vers notre système visuel en réaction aux différentes interactions entre l'environnement lumineux de la scène et la matière de l'objet. En synthèse d'images, le terme d'apparence pour une surface désigne aussi indirectement les propriétés d'apparence qui caractérisent les différentes interactions lumineuses possibles avec la matière constituant la surface d'un objet. La définition idéale et réaliste des propriétés d'apparence prend en compte l'ensemble des phénomènes physiques et photométriques impliqués [PH04] telles que la nature ondulatoire de la lumière ou bien l'absorption et la réflexion de la lumière par la matière. Ces différents phénomènes sont très complexes à calculer et prennent des formes diverses et variées selon les matériaux (transparence, anisotropie, diffusion différée dans le temps, etc.). Dans le cadre du rendu en temps-réel, les méthodes qui définissent les propriétés d'apparence doivent avoir recours à de nombreuses approximations et sont en général spécialisées à l'aide de multiples critères :

- **le ou les types de matériaux à représenter**. Chaque matériau dispose de ses propres propriétés physiques mais on distingue des propriétés communes en fonction de leur nature (plastique, minéral, végétal, organique, métallique, liquide, etc.) ;
- **la manière dont le ou les matériaux sont répartis ou usinés sur la surface des objets**. Une surface constituée d'une seule matière aura une apparence différente selon qu'elle soit brute, polie, tissée, gravée, vernie, etc. De même, une surface constituée de plusieurs matériaux aura une apparence différente selon leur répartition ;
- **les différents effets lumineux à reproduire selon leur complexité et les temps des calculs qu'ils nécessitent**. Certains phénomènes lumineux demandent énormément de calculs comme les concentrations de lumière (caustique) alors qu'ils sont presque imperceptibles comparés à d'autres effets beaucoup plus simples à reproduire comme l'éclairement direct des sources lumineuses ;
- **la multi-résolution de l'apparence : la ou les échelles à considérer pour la surface de l'objet**. En fonction de l'échelle à laquelle un objet est considéré, que ce soit dans sa représentation géométrique ou sa représentation finale dans l'image, sa surface peut être représentée selon différentes sous-échelles de précision et à chaque niveau peut correspondre des effets lumineux différents.

Pour expliciter ce dernier critère, on peut s'appuyer sur l'exemple de droite de la Figure 1.1 montrant le rendu d'une théière. Si l'on considère l'échelle des différentes briques sur sa surface, et donc plus précise que l'échelle de la forme générale, on constate un phénomène d'ombrage entre les briques qui varie en fonction de la source lumineuse. À une échelle encore plus petite où l'on discerne la granularité de la terre cuite, d'autres phénomènes lumineux apparaissent alors qu'ils ne sont pas forcément visibles à une échelle plus grossière. Dans la pratique, l'aspect multi-résolution intervient surtout dans la forme des objets où plusieurs niveaux de détails géométriques sont définis en fonction de leur taille dans l'image. La définition de l'apparence doit alors aussi présenter plusieurs niveaux de complexité et de détails des surfaces correspondant aux différents niveaux de détails de la forme des objets.

On distingue en général trois niveaux d'échelles géométriques différents pour considérer la sur-

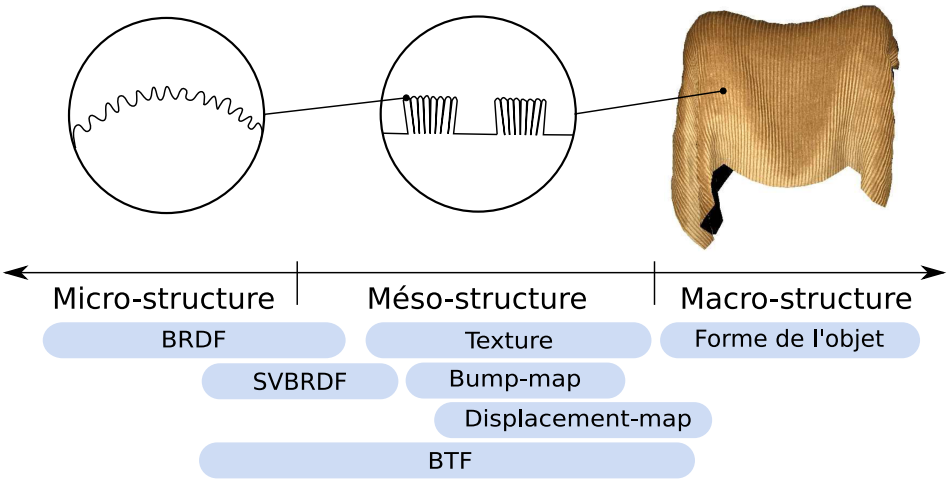


FIG. 1.2: *Les différentes échelles de géométrie entrant en considération dans l'apparence de la surface d'un matériau avec le classement des principales méthodes existantes en fonction des échelles de géométrie considérées.*

face d'un objet : l'échelle microscopique, l'échelle mésoscopique et l'échelle macroscopique. Comme le montre la Figure 1.2, la macro-structure ou macro-géométrie correspond à la forme des objets tandis que la micro-structure et la méso-structure influencent l'apparence de la surface. La micro-structure permet de considérer le comportement intrinsèque de la matière avec la lumière alors que la méso-structure correspond plus aux effets dus au relief et à la variation de différentes matières réparties sur la surface d'un objet. Cette dernière échelle a une influence très importante pour l'apparence réaliste de la surface des matériaux : si l'on reprend l'exemple de la théière de la Figure 1.1, elle représente la géométrie des briques sur la surface de l'objet. Les sections suivantes décrivent les différentes méthodes existantes par ordre croissant de complexité et de richesse d'apparence.

1.2 Réflectance en un point d'une surface

La couleur est la notion la plus simple pour l'apparence d'une surface. La couleur varie bien évidemment selon les matériaux constituant un objet mais, d'un point de vue photométrique, les couleurs résultent de la réflexion de la lumière sur la surface des objets. Les couleurs varient en fonction de la quantité d'énergie réfléchie ou absorbée pour chaque longueur d'onde constituant le spectre de la lumière visible. On parle alors de réflectance de la surface au lieu de couleur. Par commodité, le spectre continu de la lumière est représenté dans l'espace de couleur trichromatique Rouge-Vert-Bleu (RVB). La réflectance en un point d'une surface varie habituellement en fonction de la direction d'incidence de la source de lumière et de la direction d'observation de la surface (point de vue). À l'échelle microscopique, la fonction à quatre dimensions, appelée *Bidirectionnal Reflectance Distribution Function* ou BRDF [Nic70] exprime les propriétés de réflectance en un point d'une surface en fonction des directions incidentes et sortantes (deux angles polaires pour chacune = quatre dimensions) selon l'hémisphère supérieur au point considéré (*cf.* Figure 1.3).

Tout comme la forme des objets, les BRDFs peuvent être mesurées directement sur des objets réels ou bien alors synthétisées par ordinateur sur la base de modèles analytiques existants. Les travaux de Ngan *et al.* [NDM05] donnent un bon aperçu des modèles analytiques de BRDF existants ainsi qu'une comparaison de qualité entre données réelles et données approximées. L'acquisition des BRDFs se fait généralement à l'aide d'un gonio-réflectomètre qui mesure les intensités lumineuses renvoyées par une surface pour un grand nombre de points de vue et de directions d'incidence de lumière. Dans

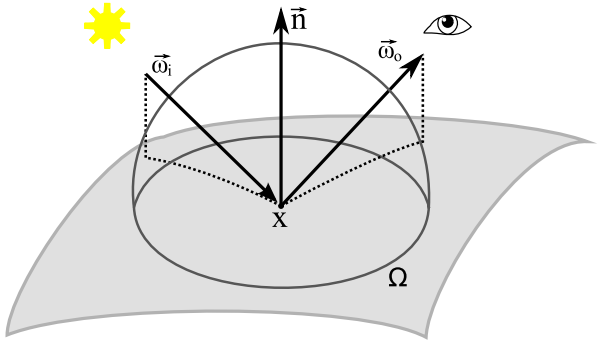


FIG. 1.3: *Configuration géométrique pour la réflexion dans la direction sortante $\vec{\omega_o}$ d'un rayon incident $\vec{\omega_i}$ au point x d'une surface. L'ensemble des directions Ω est représenté par l'hémisphère supérieur au point x et aligné avec la normale au point $\vec{n}$ (schéma de Müller et al. [MMS+04]).*

la pratique, les BRDFs sont surtout représentées par les modèles analytiques dont les paramètres permettent de jouer sur les propriétés de réflectance de la surface. Ces modèles sont plus ou moins réalistes selon les différents phénomènes physiques pris en compte. Ils sont aussi utilisés dans des processus de minimisation pour compresser et approximer les nombreuses données de réflectance des BRDFs mesurées. La plupart des modèles de BRDF existants considère la surface d'un objet à une échelle microscopique et suppose qu'elle est composée d'une multitude de micro-facettes dont chacune renvoie exactement la même quantité d'énergie incidente reçue dans la direction de réflexion par rapport à la normale correspondante de la micro-facette. La distribution des normales de ces micro-facettes est caractéristique des différents matériaux représentables et conduit à différentes réflexions. Comme illustré par la Figure 1.4, lorsque la distribution est telle que l'énergie est réémise uniformément dans toutes les directions, la réflexion est dite *lambertienne* [Cal98]. À l'opposé, la réflexion est *spéculaire pure* lorsque l'ensemble des normales des micro-facettes est parallèle à la normale de la surface. Les réflexions se situant entre ces deux extrêmes sont dites *directionnelles (« glossy »)*. Actuellement, le modèle le plus couramment utilisé par les applications de rendu en temps-réel est le modèle de Blinn-Phong [Bli77], très simple à mettre en œuvre tout en étant visuellement convaincant mais physiquement faux.

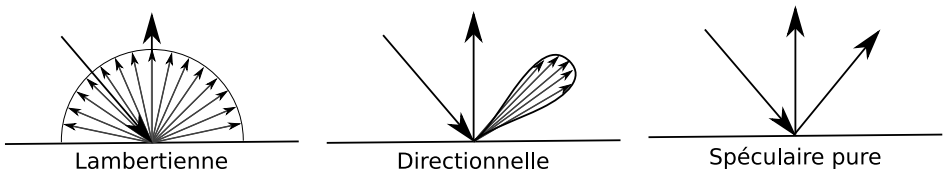


FIG. 1.4: *Les différents types de réflexion en un point d'une surface.*

L'utilisation de la BRDF seule est suffisante pour représenter de manière plus ou moins réaliste des matériaux homogènes et opaques tels que les matières plastiques ou les métaux, à la condition que les principaux effets lumineux soient pris en compte par les modèles analytiques utilisés. Spécifiquement, la BRDF est en fait une approximation de la fonction à huit dimensions nommée *Bidirectionnal Surface Scattering Distribution Function* [NRH+77] (BSSRDF) dont les quatre dimensions supplémentaires servent à distinguer le point d'impact des rayons incidents sur la surface et le point de réémission de l'énergie reçue. Contrairement à la BRDF, la BSSRDF permet de prendre en compte les phénomènes de réfraction et de transmittance où les rayons incidents traversent alors la surface semi-transparente au

point d'impact. Pour représenter des phénomènes plus complexes tels que la dispersion, il faut encore rajouter des dimensions pour prendre en compte les changements de la longueur d'onde des rayons incidents, le temps mis par la lumière pour traverser la matière ou encore la durée de la réflectance. Les principaux paramètres entrant en jeu dans l'interaction lumière/matière sont illustrés sur la Figure 1.5.

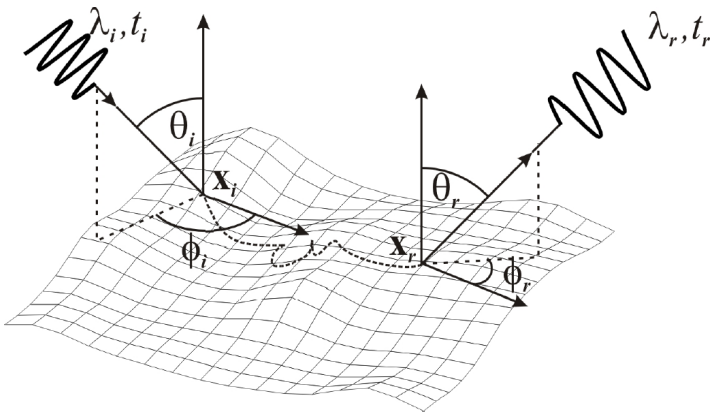


FIG. 1.5: *Les différents paramètres entrant en jeu dans l'interaction de la lumière avec la matière. Soit un photon de longueur d'onde λ_i frappant la surface d'un matériau au temps t_i au point x_i. La direction incidente du photon est définie dans le repère local à la surface au point x_i par le couple d'angles (θ_i, ϕ_i). Le photon traverse la surface et ressort à la position x_r, au temps t_r avec la longueur d'onde λ_r dans la direction (θ_r, ϕ_r).*

La réflectance telle qu'elle est définie sur la Figure 1.5 comme une fonction à douze paramètres est bien trop complexe pour le contexte actuel du rendu d'images en temps-réel. Pour simplifier le problème, les représentations les plus précises de la réflectance se cantonnent généralement à la BSSRDF, en posant les hypothèses suivantes :

- la lumière traverse la matière en un temps infinitésimal $(t_i = t_r)$;
- la durée de réflectance d'une surface est invariante $(t_0 = t_i = t_r)$;
- l'interaction n'affecte pas la longueur d'onde $(\lambda_i = \lambda_r)$;
- la couleur est approximée par trois composantes couleurs rouge, vert et bleu.

En conséquence, les phénomènes comme la dispersion, la phosphorescence[1] et la fluorescence[2] ne sont pas représentés dans la synthèse d'images en temps-réel mais leur rareté et leur faible impact visuel en font des effets négligeables dans les conditions d'éclairement standard. Dans la pratique, l'utilisation de la BRDF est suffisante dans la majorité des applications de rendu 3D en temps-réel, à quelques exceptions près, où les effets de la BSSRDF peuvent être simulés à l'aide de méthodes optimisées et spécifiques à l'application [MKV+03, TJP+03, JSX+08].

1.3 Variation de la réflectance pour une surface plane

Pour une surface plane, la réflectance varie spatialement lorsque la surface est constituée de matériaux hétérogènes (ex : le marbre) ou de plusieurs matériaux différents. En synthèse d'images, la variation des couleurs est généralement représentée avec des textures [Cat74]. Une texture est en fait une simple image que l'on plaque sur la surface d'un objet pour représenter les variations des propriétés de réflectance à l'échelle de la méso-structure. La simplicité de cette méthode comparée à l'apport de détails pour l'apparence des objets en a démocratisé l'usage. Les textures sont soit créées par des artistes soit obtenues à partir de photographies et d'images synthétiques. Dans la pratique,

[1] Accumulation d'énergie et retransmission décalé dans le temps
[2] Transfert d'énergie d'une longueur d'onde à une autre.

les textures sont combinées avec un modèle analytique de BRDF dont elles définissent la variation de la composante diffuse ou d'une manière générale, toute variation spatiale de propriété d'apparence. La combinaison des deux dimensions spatiales des textures et des quatre dimensions angulaires de la BRDF aboutit à une bonne représentation de la réflectance d'une surface pour certains matériaux. Afin d'obtenir une représentation plus réaliste des objets, des dispositifs spécifiques ont été développés pour mesurer une fonction à six dimensions directement sur des objets réels. Debevec *et al.* [DHT+00] ont développé un dispositif pour mesurer la réflectance de visages humains et Malzbender *et al.* [MGW01] se sont inspirés du système pour mesurer la réflectance de certains objets archéologiques. Cependant ces travaux ne mesurent pas la totalité des dimensions de la BRDF car la direction d'observation est fixée. C'est avec l'introduction des *Spatially Varying BRDF* (SVBRDF) par McAllister *et al.* [MLH02] qu'une mesure complète de la réflectance pour des échantillons planaires de matériaux a été rendue possible grâce à un dispositif muni d'un gonio-réflectomètre pour mesurer intégralement et spatialement les BRDFs. La Figure 1.1 illustre une utilisation des SVBRDFs représentées par des lobes du modèle de Lafortune [LFTG97] dont les différents paramètres sont stockés dans des textures. Plus tard, Garner *et al.* [GTHD03] ont mis en œuvre un dispositif pour mesurer rapidement les propriétés diffuses et spéculaires de la surface d'objets planaires. Les propriétés sont mesurées par une simple caméra durant le passage unique d'une source linéaire sur la surface et les propriétés sont minimisées pour correspondre au modèle physique de Ward [War92]. Ces représentations donnent une bonne approximation plus ou moins réaliste de l'apparence pour des surfaces quasi-planes avec des propriétés de réflectance hétérogènes.

1.4 Variation de la réflectance pour une surface arbitraire

Pour représenter les effets de la méso-structure d'une surface présentant du relief, les représentations le plus couramment utilisées sont le *bump mapping* [Bli78] et le *displacement mapping* [Coo84]. Le *bump mapping* simule les effets lumineux produits par la méso-structure d'une surface sans pour autant nécessiter sa modélisation géométrique. L'astuce consiste à perturber la normale en chaque point d'une surface en utilisant une fonction de hauteur encodée dans une texture. La perturbation de la normale modifie l'éclairement local sur la surface des objets et donne une impression de relief sans pour autant modifier la géométrie de la surface. Pour enrichir le réalisme du rendu avec d'autres effets lumineux et visuels induits par la méso-structure, des méthodes d'amélioration ont été proposées pour apporter des effets d'ombrage propre [Max88, SC00, HS99] mais elles sont peu utilisées car elles demandent des calculs complexes et plusieurs passes de rendu sont nécessaires. Pour introduire la visibilité de la méso-structure, la méthode de Kaneko *et al.* [KTI+01] simule les occultations pour des fonctions de hauteur et intègre des effets de parallaxe au rendu. Cependant la méthode ne fonctionne que pour des fonctions d'élévation de faible hauteur donnant des occultations singulières.

Le *displacement mapping* [Coo84] enrichit la macro-géométrie de la surface des objets à l'aide d'une fonction de hauteur. Contrairement au *bump mapping*, cette méthode modélise la géométrie de la méso-structure ce qui permet d'améliorer sensiblement la qualité de la silhouette des objets définie grossièrement par la macro-géométrie. Cet effet qui s'ajoute aux effets d'éclairement local, d'ombrage propre et d'occultation propre produit un impact visuel non négligeable dans l'apparence des objets. Cette approche est d'ailleurs utilisée autant par les méthodes de rendu en temps-réel que celles hors-lignes. La principale difficulté dans le *displacement mapping* pour le rendu en temps-réel est la détermination rapide de la visibilité de la méso-structure déformant la surface des objets. Dans les solutions existantes, la visibilité est en général déterminée soit par lancer de rayons [OBM00, BD06, PO06, MW08] soit par pré-calcul de la visibilité pour un ensemble de directions d'observation prédéterminées [WWT+03, WTL+04].

Ces approches offrent une bonne représentation spatiale de l'apparence et donnent un rendu plus ou moins réaliste mais la représentation de la réflectance en chaque point de la surface reste de faible qualité comparée aux SVBRDFs qui capturent tous les détails angulaires de la réflectance pour des surfaces planes. L'idéal serait une méthode combinant les détails angulaires des SVBRDFs avec les effets associés aux méso-structures. De plus, aucune de ces méthodes ne permet de prendre en

compte l'éclairement indirect qui se produit au sein des méso-structures et qui contribue beaucoup dans l'apparence réaliste d'une surface. À ce sujet, Wong *et al.* [WHON97a] ont introduit la notion de BRDF apparente (ABRDF) pour désigner la complexité de la réflectance en un point d'une méso-structure où sont mêlés à la fois des effets d'éclairement direct, d'éclairement indirect et de parallaxe. Ces effets indirects sont entre autres dus aux occultations propres et aux ombres propres qui se produisent sur une méso-structure en fonction des conditions d'observation et des configurations lumières qui varient. L'étude de Wong montre que les ABRDFs sont des fonctions très complexes et sont difficilement représentables par les modèles analytiques utilisés et développés pour les BRDFs.

Une autre approche basée sur des mesures de l'apparence de matériaux réels apporte une alternative intéressante grâce à la richesse des détails et des effets lumineux qu'elle capture à la fois pour les dimensions spatiales et les dimensions angulaires. Cette dernière approche nommée « Bidirectionnal Texture Function » (BTF) est explorée en détails dans la suite de ce chapitre et dans les Chapitres 2 et 3.

1.5 Les BTFs

FIG. 1.6: *Images obtenues par Müller et al. [MMS+04] utilisant des BTFs pour définir l'apparence des objets. En (a), des BTFs définissent les différentes matériaux constituant l'intérieur d'une Mercedes classe C comme le plastique, le cuir ou le bois. En (b), une BTF est utilisée pour définir l'apparence complexe de la surface d'un tricot.*

Le concept de *Bidirectional Texture Functions* (BTF) a été introduit par Dana *et al.* [DvNK97] en 1997 pour l'acquisition de BTFs et par Dischler [Dis98] en 1998 pour la génération de BTFs de synthèse. Une BTF peut être vue comme une texture 6D qui, comparée à une texture classique, est directionnelle par l'ajout d'une dépendance aux directions de vue et de lumière : elle est alors simplement composée d'un ensemble de textures classiques dont chacune correspond à une direction de vue et une direction d'éclairement de l'échantillon de matériau représenté. Grâce à ces dépendances directionnelles et au fait qu'elles soient construites à partir de mesures sur des matériaux réels, les BTFs permettent de représenter l'ensemble des effets lumineux et visuels dus à la fois aux propriétés de réflectance des surfaces mais aussi à leurs méso-structures. En plus de représenter tous les effets combinés des approches précédentes (*cf.* Section 1.3 et 1.4), tels que l'éclairement direct, les ombres propres et les occultations propres (parallaxe), les BTFs capturent aussi les effets d'éclairement indirect sur la méso-structure ou encore dans une certaine mesure la transmittance. Les BTFs permettent donc une représentation réaliste de l'apparence des matériaux comme en témoigne les images de la Figure 1.6. Müller *et al.* [MMS+04] définissent la BTF d'un matériau comme l'intégrale de la BSSRDF sur une surface S :

$$BTF_{rgb}(x, \theta_i, \phi_i, \theta_r, \phi_r) = \int_S BSSRDF_{rgb}(x_i, x, \theta_i, \phi_i, \theta_r, \phi_r) dx_i \quad (1.1)$$

avec x la position mesurée sur la surface de référence, x_i le point d'impacte de la direction d'incidence de la lumière représentée par ses coordonnées sphériques (θ_i, ϕ_i) et (θ_r, ϕ_r) les coordonnées sphériques de la direction d'observation. En considérant une surface plane selon le plan XY et l'axe Z représentant la normale à la surface pour une direction donnée, l'angle θ représente l'angle que fait la direction avec l'axe Z et l'angle ϕ représente l'angle que fait la direction avec l'axe X dans le plan XY (*cf.* Figure 1.7). Cette convention, illustrée par la Figure 1.7, est celle adoptée pour la suite du document. Les BTFs sont mesurées en général avec une lumière directionnelle blanche et pour des surfaces de référence planes, c'est-à-dire qu'on considère que la surface sur laquelle elles seront plaquées est plane. Il n'y a pas de différence fondamentale au moment du rendu entre l'utilisation d'une BTF et une texture classique si ce n'est le choix, pour chaque texel, de la texture directionnelle correspondant à la direction du point de vue et de lumière incidente données. Dans la pratique et de même que pour les dimensions spatiales des textures traditionnelles, des interpolations linéaires des données sont appliquées entre les directions de vue les plus proches puis entre les directions de lumières les plus proches. De ce fait, les BTFs offrent à l'heure actuelle un des meilleurs compromis apparence réaliste/coût de calcul comparativement aux approches existantes.

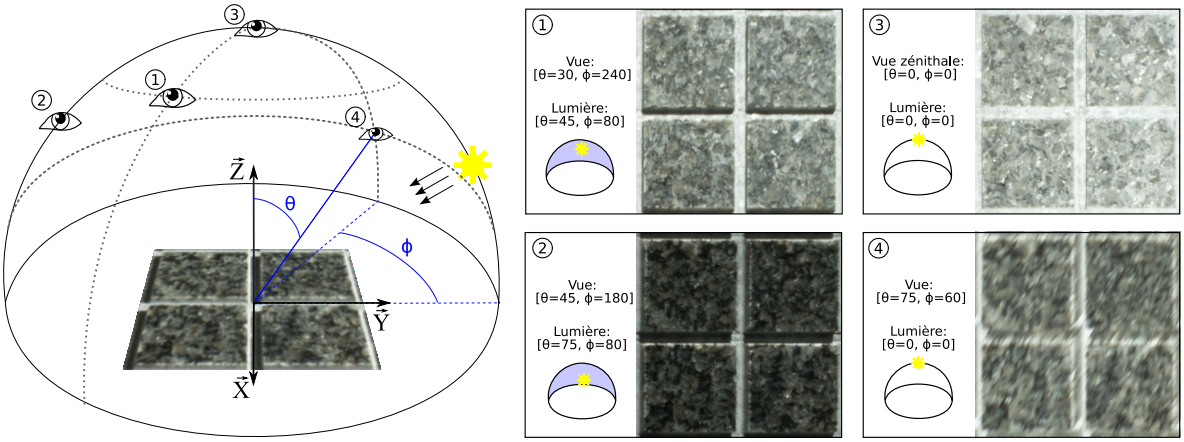


FIG. 1.7: *Une BTF : aperçu d'un échantillon de surface pour quatre différents points de vue (numérotés 1, 2, 3 et 4) et différentes directions de lumière incidente. L'angle θ représente l'angle que fait la direction avec l'axe Z et l'angle ϕ représente l'angle que fait la direction avec l'axe Y dans le plan XY.*

1.6 Les domaines de recherche autour des BTFs

Le diagramme de la Figure 1.8 montre les différentes étapes qui peuvent se succéder dans le processus d'utilisation des BTFs et chacune d'entres elles, correspond à un domaine de recherche bien spécifique. Un état de l'art de Müller *et al.* [MMS+04] donne un bon aperçu des travaux existant en 2004 dans les différents domaines. Le domaine de **l'acquisition** concerne le développement et la mise en œuvre des différents dispositifs expérimentaux permettant de produire des BTFs à partir de l'apparence naturelle des matériaux. Des BTFs peuvent aussi être obtenues en reproduisant virtuellement l'environnement d'acquisition et par la synthèse d'images réalistes de matériaux, on parle alors de **BTFs synthétiques**.

Une BTF peut représenter jusqu'à plusieurs giga-octets de données pour un matériau. Aussi de nombreux travaux dans le domaine de la **compression** proposent des solutions pour réduire

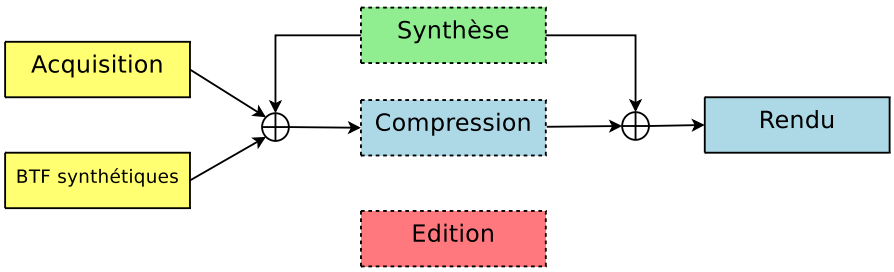


FIG. 1.8: *Les différents domaines impliqués dans l'utilisation des BTFs. L'**acquisition** et les **BTFs synthétiques** sont les deux moyens d'obtenir des BTFs. La **compression** s'occupe de réduire la taille des données qui peuvent ensuite être exploitée par des méthodes de **rendu**. La **synthèse** de textures peut intervenir avant ou après compression des données pour créer des BTFs sans motifs de répétition à partir des BTFs d'origines. L'**édition** des BTFs est un domaine perpendiculaire à tous ces domaines car elle peut intervenir avant ou après chacun des autres domaines.*

considérablement la taille des données afin de rendre raisonnable le stockage de nombreuses BTFs. De plus, la compression des BTFs est indispensable pour la synthèse d'images en temps-réel où plusieurs BTFs sont amenées à tenir dans l'espace mémoire des cartes graphiques. Les méthodes de compression sont alors souvent associées au **rendu** temps-réel des objets dont l'apparence est définie par des BTFs car la vitesse de décompression des données est alors déterminante pour le calcul en temps-réel.

Plusieurs travaux ont adapté les méthodes de **synthèse** de textures aux BTFs [LYS01, TZL+02, KMBK03, LHZ+04, LPF+07] pour générer des BTFs applicables sans répétition de motifs discernables sur des grandes surfaces. **L'édition** des BTFs [KBD07, MSK07] a pour objectif la modification des propriétés de réflectance ou de la méso-structure implicitement représentées dans les BTFs tout en gardant un certain réalisme ainsi qu'une certaine cohérence dans les images.

Dans la suite du mémoire, les différents travaux de recherche relatifs à l'acquisition des BTFs et aux BTFs synthétiques seront présentés dans le Chapitre 2. Ensuite, nous discuterons des différentes méthodes existantes permettant à la fois la compression et le rendu des BTFs dans le Chapitre 3 sous la dénomination de « représentation compacte ».

Chapitre 2

Création des BTFs

Les BTFs sont obtenues soit par la mesure de l'apparence réelle de matériaux soit par la génération informatique de ces mesures en simulant l'apparence par des procédés de synthèse d'images réalistes. Le processus d'acquisition consiste à photographier des échantillons plans de matériaux depuis plusieurs points de vue et ce pour différentes directions d'illumination. D'un autre côté, les BTFs peuvent aussi être synthétisées par la simulation numérique d'un processus d'acquisition idéal. Dans les sections suivantes, nous présentons les différents dispositifs d'acquisition avec leurs atouts et leurs inconvénients ainsi que quelques exemples sur la génération numérique des BTFs.

2.1 L'acquisition de BTFs

L'acquisition des BTFs reste de nos jours un processus complexe et requiert la mise en place d'un protocole qui peut être coûteux dans un environnement dédié. Le principe est d'obtenir des mesures de l'apparence de la surface d'un matériau pour différents angles de vue et pour différentes directions de lumière incidente. Un dispositif d'acquisition de BTFs contient au moins les trois composants de base suivants :

- un support pour les matériaux à mesurer ;
- un appareil de mesure de l'intensité lumineuse sur la surface de l'échantillon (un capteur CCD[1] par exemple) ;
- une source lumineuse directionnelle et blanche.

Des dispositifs d'acquisition variés ont été élaborés depuis les premières expérimentations de Dana *et al.* [DvNK99] et on peut les classifier en fonction de la manière dont les différents composants sont organisés pour capturer les six dimensions des BTFs.

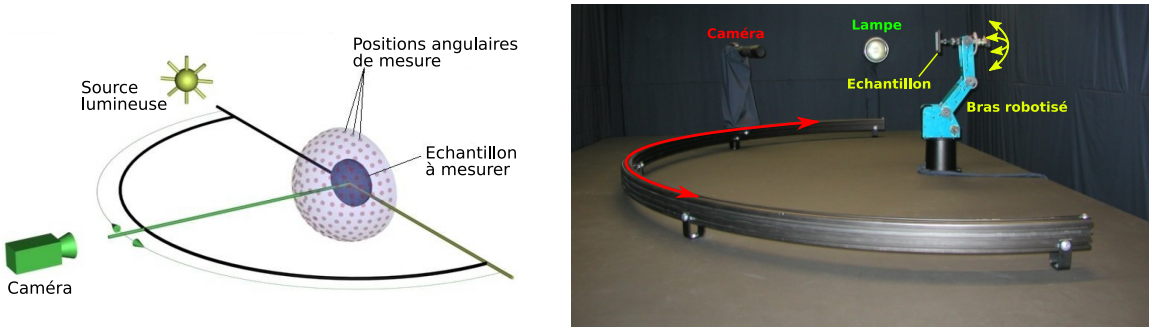
2.1.1 Les systèmes à composants mobiles

Ces systèmes disposent de parties mécaniques mobiles pour permettre les mouvements des trois composants de base. Ces parties mobiles sont robotisées et les déplacements contrôlés par ordinateur. Le premier dispositif permettant l'acquisition de BTFs est le système introduit par Dana *et al.*[DvNK99]. La disposition des éléments reprend une configuration déjà utilisée lors de la mesure de BRDFs [Cor, Nis, War92]. Seul le gonio-réflectomètre[2] a été remplacé par une caméra numérique pour rajouter les dimensions spatiales aux réflectances mesurées. Pour ce dispositif, illustré par la Figure 2.1-(a), la caméra varie selon une seule dimension angulaire en se déplaçant sur un rail tandis que le support

[1] Un capteur CCD (*Charge-Couple Device*) est un capteur électronique photosensible qui convertit le rayonnement lumineux (les photons) en signal électrique grâce à des photodiodes. Ce capteur est utilisé dans les appareils photos numériques et les caméras numériques.

[2] Un gonio-réflectomètre mesure la réflectance en un point précis d'une surface. Cet appareil de mesure est généralement utilisé pour mesurer la BRDF de matériaux homogènes.

du matériau pivote selon deux axes de rotation. Un troisième axe de rotation, parallèle à la normale des surfaces mesurées, permet des mesures anisotropes des surfaces. La source lumineuse quant à elle reste fixe durant tout le processus d'acquisition. Les mesures obtenues sont regroupées dans la base de BRDFs et de BTFs nommée CUReT [CUR]. Müller *et al.* [MMS+05] constatent que la densité de ces données (205 images pour une BTF) n'est pas suffisante pour obtenir un rendu d'images de synthèse de qualité.



(a) (b)

FIG. 2.1: *(a) Schéma du système d'acquisition inspiré des systèmes de mesure de BRDFs [SSK03]. La caméra se déplace sur un rail alors que la source lumineuse est fixée. Le support du matériau peut pivoter sur deux axes. (b) Photographie du système d'acquisition de l'université de Bonn [SSK03]. On distingue au premier plan le rail circulaire supportant la caméra située en arrière plan. Le bras robotisé recevant les échantillons se trouve au centre et la source lumineuse est au bout du rail en arrière plan.*

Le dispositif a été repris par McAllister *et al.* [McA02] ainsi que par Satler *et al.* [SSK03] (*cf.* Figure 2.1-(b)) pour améliorer les résolutions spatiales et angulaires des mesures afin d'obtenir une représentation des matériaux de plus haute qualité lors des rendus. Une autre amélioration concerne la mesure d'échantillons de matériaux raccordables pour que les jointures ne soient pas visibles si la texture est répétée plusieurs fois sur la surface. Les données de Satler *et al.* [SSK03] sont regroupées dans la base de données de BTF Bonn [Bon] et sont accessibles en ligne. Les données présentes sont soit au format RVB 24 bits avec une résolution spatiale de 256×256, soit au format HDR *RADIANCE* [War91] avec une résolution de 800×800. La résolution angulaire est identique pour les directions de vue et de lumière et comprend 81 directions différentes réparties en angles solides uniformes sur un hémisphère. La durée totale de l'acquisition est assez longue car elle dure environ 14 heures pour une seule BTF. Koudelka *et al.* [KMBK03] ont réalisé un système similaire à la seule différence que la caméra devient fixe et la source lumineuse mobile. Le temps d'acquisition d'une BTF est de 10 heures parce que les résolutions spatiales et angulaires utilisées sont plus faibles que Satler *et al.* [SSK03].

Un tout autre concept visible sur la Figure 2.2 et développé par Wang *et al.* [WTS+05], propose un système de deux supports en arc avec au centre le support de l'échantillon à mesurer. Le support du matériau est rotatif selon l'axe de la normale à la surface à mesurer. Un des arcs est muni de huit caméras réparties uniformément alors que le second arc supporte huit lampes réparties également de manière uniforme. L'arc des caméras est fixe alors que l'arc des lumières est mobile autour du support. Le pas de rotation n'est pas précisé, mais la résolution spatiale est de 1024×728 en 24 bits. Ce système dispose aussi d'un laser permettant de reconstruire une fonction de distance dépendante du point de vue. Cette dernière mesure permet de connaître la méso-structure correspondante pour chaque image des BTFs mesurées. Ce système innove par son utilisation de plusieurs caméras et lampes qui accélèrent le processus d'acquisition.

La durée d'acquisition de BTFs avec de tels systèmes demeure assez longue car non seulement le déplacement des éléments mobiles prend beaucoup de temps mais il induit aussi des opérations de calibration des appareils de mesure à chaque déplacement.

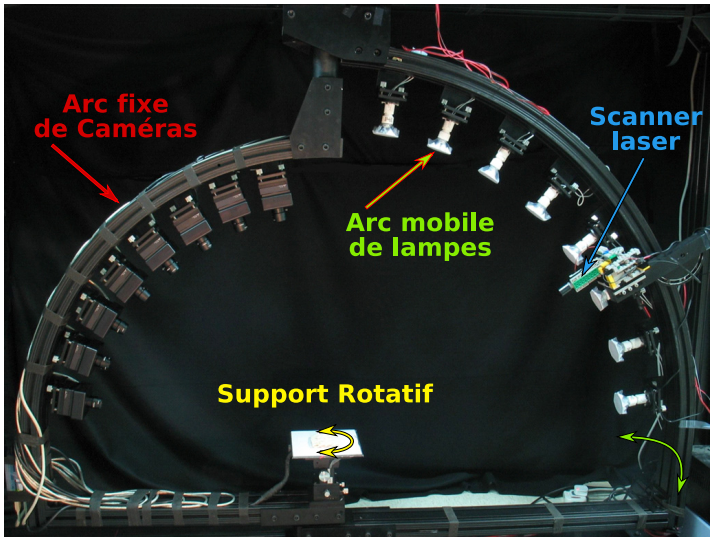


FIG. 2.2: *Le système de Wang et al. [WTS+05] avec ses deux supports en arc dont celui supportant les lampes qui est mobile. Sur ce même arc mobile est fixé le scanner laser permettant l'acquisition de la méso-structure sous la forme de plusieurs fonctions de hauteur. Le support de l'échantillon est rotatif pour apporter une dimension supplémentaire aux différentes prises de vue.*

2.1.2 Les systèmes optiques

Ces systèmes utilisent des procédés optiques pour capturer les six dimensions d'une BTF. Les trois composants de base sont tous fixes, aucun d'entre eux n'est déplacé durant le processus d'acquisition. En 2003, Han et Perlin [HP03] ont utilisé un kaléidoscope pour l'acquisition de BTFs. L'utilisation du kaléidoscope permet de visualiser et d'éclairer les matériaux en différentes positions angulaires sans bouger la caméra ni la source lumineuse. Le kaléidoscope est composé de plusieurs petits miroirs arrangés en facette sur la surface interne d'un hémisphère. Le processus d'acquisition est très rapide mais la résolution spatiale obtenue est assez faible du fait que toutes les vues sont acquises en même temps et réparties sur une seule image. Les données ainsi mesurées peuvent aussi contenir des erreurs dues soit aux imperfections des miroirs, soit au fait qu'une source de lumière peut être réfléchie plusieurs fois, biaisant ainsi l'apparence.

L'année suivante, Dana *et al.*[DW04] ont élaboré un dispositif utilisant un miroir parabolique. Le principe est le même que pour le système de Han et Perlin sauf que la surface du miroir parabolique est continue comparée aux facettes du kaléidoscope. L'utilisation du miroir permet une très haute densité angulaire mais réduit malheureusement la dimension spatiale à un seul point comme pour une mesure de BRDF. Un déplacement planaire des échantillons sous le point de mesure permet de construire spatialement la BTF. Les auteurs vantent une acquisition en temps-réel dans le futur, on peut donc supposer, même si ce n'est pas précisé, que la durée d'acquisition d'un tel système reste assez longue.

2.1.3 Les systèmes à composants fixes

Au lieu de déplacer les composants de base, ces systèmes utilisent une multitude de caméras et de lampes disposées uniformément sur un support hémisphérique. Le processus d'acquisition se fait alors en parallèle, réduisant ainsi grandement sa durée. Chaque position de mesure angulaire est munie d'une caméra et/ou d'une source lumineuse comme le montre la Figure 2.3. La position des appareils de mesure étant fixée, ces dispositifs ont l'avantage de ne pas nécessiter de calibration entre chaque

mesure mais, en revanche, les résolutions angulaires sont fixées définitivement.

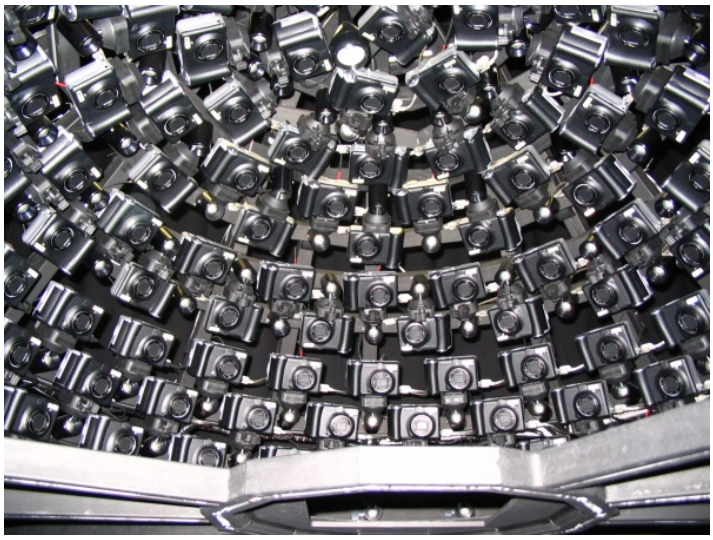


FIG. 2.3: *Système d'acquisition à composants fixes de l'université de Bonn [MBK05]. Le matériau à mesurer est placé sur le support visible en bas à droite. Quand une lampe s'allume, tous les appareils photos prennent un cliché de l'échantillon. Lorsque toutes les lampes ont été allumées, on a bien une BTF avec N photos prises de différents points de vue pour M sources lumineuses.*

Les premiers systèmes à utiliser ce concept de composants multiples fixes sont restreints à l'acquisition de champs ou fonctions de réflectance [DHT+00, MGW01]. En d'autres mots, ces systèmes mesurent les variations de l'apparence selon les changements de directions lumineuses mais uniquement pour une seule direction de vue. Une fonction de réflectance constitue donc une sous-fonction d'une BTF pour une vue donnée. Dans ces dispositifs, le dôme de mesure ne présente qu'un seul appareil photo fixé au pôle et plusieurs lampes réparties uniformément sur la surface hémisphérique. Pour l'acquisition des BTFs selon ce principe, Müller *et al.*[MBK05] ont construit un dôme hémisphérique avec 151 appareils photo et autant de sources lumineuses (*cf.* Figure 2.3). Ce dispositif peut acquérir aussi bien des surfaces planes que des petits objets. Il permet la reconstruction géométrique par des méthodes basées images pour des petits objets en plus de l'acquisition de BTF. Le temps de mesure pour une BTF est assez rapide. Il faut environ 40 minutes pour acquérir $151 \times 151 = 22\,801$ images. La durée des mesures n'est plus limitée par le déplacement des composants mais par la vitesse de transfert des données fournies par le fonctionnement en parallèle des 151 appareils photos.

Neubeck *et al.* [NZG05] utilisent un système hybride avec les systèmes à parties mobiles. Les 169 lampes sont supportées par un dôme hémisphérique tandis qu'une caméra est montée sur un double bras rotatif selon deux axes depuis la base du support. Le système permet d'obtenir 264 vues différentes donnant des BTFs contenant environ 40 000 images pour 25 giga-octets. L'objectif des auteurs n'est pas la rapidité d'acquisition mais une très forte densité angulaire permettant la mise en œuvre de méthodes d'évaluation de la méso-structure des matériaux.

Tout comme les systèmes à composants fixes, une étape de recalage des images consistant à détourer et corriger la perspective des images est nécessaire. Comme le montre la Figure 2.4, la correction de perspective est effectuée par rapport au plan de référence donné par les images prises au zénith du support.

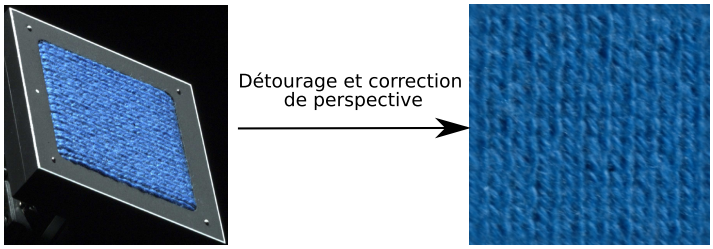


FIG. 2.4: *À gauche, image acquise pour une vue en perspective ($\theta = 60, \phi = 144$) d'un échantillon de tricot sous lumière directionnelle ($\theta = 60, \phi = 18$). À droite, la même image après détourage et correction de perspective ([MMS+04]).*

2.1.4 Autres systèmes

Ngan et Durand [ND06] ont une approche statistique reposant sur l'utilisation d'une unique caméra. Différentes prises de vue caractéristiques du matériau sont obtenues en positionnant les uns à coté des autres plusieurs échantillons orientés différemment. Pendant que la caméra qui est placée au dessus des échantillons filme, une source lumineuse très petite est déplacée à la main tout autour des échantillons. La position de la source lumineuse est retrouvée par une minimisation basée sur la détection du pic spéculaire dans les mesures. Le nombre de vues étant insuffisant pour une BTF de qualité, des données pour des vues intermédiaires sont construites à partir de plusieurs analyses statistiques des histogrammes des images acquises. Le processus d'acquisition est donc assez rapide hormis la durée des post-traitements. La fiabilité et la qualité est discutable pour les images statistiques des vues intermédiaires et non mesurées.

Les travaux de Furukawa *et al.* [FKIS02] et Lench *et al.*[LGK+01] mélangent à la fois l'acquisition de la géométrie de petits objets et l'acquisition de l'apparence paramétrée sur leurs surfaces. La géométrie est capturée à l'aide d'un scanner et plusieurs images de l'apparence sont capturées pour différents angle de vue et de différentes positions de sources lumineuses. Ces travaux acquièrent une apparence dépendante d'une géométrie et les BTFs acquises sont donc spécifiques à un objet.

2.1.5 Discussion

Si l'on veut appliquer le théorème de Nyquist-Shannon sur l'échantillonnage d'un signal, les fréquences de la réflectance comparées aux choix des pas d'échantillonnages angulaires ou spatiaux sont difficilement quantifiables quand on sait que les propriétés de la réflectance sont définies jusqu'à l'échelle microscopique. Alors les systèmes d'acquisition actuels font au mieux pour juger la qualité des mesures, souvent selon des critères visuels. Les pas d'échantillonnage sont avant tout déterminés par des contraintes d'occupation en mémoire des mesures ou de faisabilité du choix des résolutions pour les points de mesure. Une autre propriété primordiale pour un dispositif est bien évidemment la durée d'acquisition d'une BTF.

L'existant : Il semble que les systèmes les plus rapides soient ceux à composants fixes et l'approche de Ngan et Durand [ND06], bien que cette dernière offre une qualité réduite et un échantillonnage angulaire trop faible pour des BTFs destinées à des rendus de haute qualité. À notre connaissance, le meilleur dispositif d'acquisition à ce jour est le dôme de Müller *et al.* [MBK05] mais c'est sans doute aussi le plus onéreux. Il offre une des meilleures résolutions spatiales et angulaires ainsi que des couleurs en haute définition pour un temps d'acquisition de seulement 40 minutes.

Les systèmes à composants mobiles sont de leur côté beaucoup plus lents, et nécessitent des étapes de calibration mais ils permettent cependant une bonne flexibilité pour les pas d'échantillonnages.

La voie des systèmes d'acquisition utilisant des procédés optiques semble prometteuse en termes de

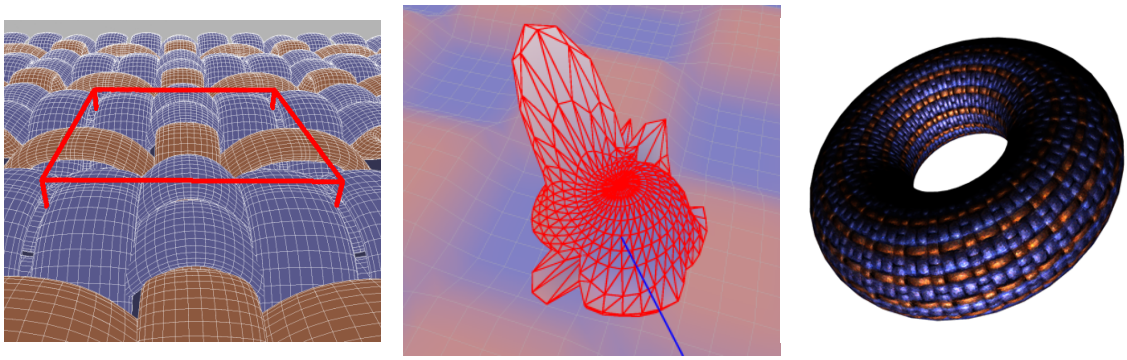
rapidité et de résolution angulaire. Cependant les dispositifs existants ne sont pas assez convaincants en termes de résolution spatiale ou de rapidité d'acquisition.

Perspectives : Les perspectives pour les systèmes d'acquisition futurs résident bien sûr dans l'amélioration des critères décrits ci-dessus mais aussi dans la mobilité et la miniaturisation. La mobilité permettrait de mesurer des BTFs en dehors de l'environnement contrôlé d'un laboratoire. Par exemple, on pourrait imaginer mesurer l'apparence d'objets non déplaçables pour des sites archéologiques. La miniaturisation, qui va de paire avec la mobilité, permettrait de mesurer plus facilement l'apparence sur toutes sortes de surface difficile d'accès pour les dispositifs actuels comme l'apparence de murs ou de plafonds. Récemment Moshe *et al.* [BEWW+08] ont développé un procédé d'acquisition de BRDF grâce à des diodes électroluminescentes qui jouent à la fois le rôle de source lumineuse et de capteur. Ce principe très léger et portatif est très prometteur pour l'acquisition de BTFs à l'aide d'un système peu encombrant et portatif.

Une dernière perspective serait de répondre au problème de l'espace mémoire nécessaire aux BTFs directement à l'acquisition de celles-ci. Les travaux de Ghosh *et al.* [GAHO07] sur l'acquisition de BRDF à l'aide d'un miroir parabolique permettent, en lieu et place d'un échantillonnage, de faire l'acquisition directement dans la base des harmoniques sphériques. Ce concept pourrait s'appliquer aux BTFs à condition de trouver un procédé rapide afin que le système mesure les variations spatiales de l'apparence.

2.2 La génération de BTFs

La génération de BTFs consiste à modéliser la surface des échantillons de matériaux ainsi que leurs propriétés de réflectance afin d'obtenir les images d'une BTF à partir du rendu des matériaux modélisés pour différentes positions de caméra et différentes directions de lumière. Dans la pratique, la surface des matériaux est représentée au niveau de la méso-structure à l'aide d'un maillage et les propriétés de réflectance sont définies par les modèles analytiques standards de BRDF. Le rendu est généralement effectué par lancer de rayons pour une vue orthographique avec un éclairement global pour simuler tous les phénomènes d'un photoréalisme produits par la méso-structure et les propriétés de réflectance du matériau.



(a) (b) (c)

FIG. 2.5: *La génération de BTFs par Suykens et al. [SvLD03]. En (a), la méso-structure modélisée pour représenter les mailles d'un tissu, la zone rouge symbolise le morceau d'échantillon sélectionné et raccordable pour générer les BTFs. En (b), la visualisation d'une ABRDF en un point donné et pour un rayon incident (en bleu) qui résulte de la combinaison des propriétés de réflectance et des effets induits par la méso-structure. En (c), un exemple de rendu de la BTF synthétique correspondante plaquée sur un tore.*

Pour du rendu réaliste de textiles, Daubert *et al.* [DLHS01] ont modélisé des mailles de tricots

à l'aide de surfaces implicites avant de les convertir en maillages. Le rendu des images a été effectué par lancer de rayons couplé avec une méthode d'éclairement global basée sur la cohérence dans le pré-calcul de la visibilité de la source lumineuse [DKS+03]. Suykens *et al.* [SvLD03] ont créé différentes BTFs avec leur propre moteur d'éclairement global. Diverses méso-structures ont été modélisées en formes d'échiquier, d'anneau concentrique ou encore de mailles de tricot, cette dernière est illustrée sur la Figure 2.5. Vasilescu et Terzopoulos [VT04] ont généré des BTFs avec des méso-structures riches représentant par exemple un amas de pièces de monnaie ou encore un épi de maïs. Devant la durée du temps de rendu obtenu sur un logiciel grand public de rendu par lancer de rayons, les BTFs générées disposent d'assez peu de directions de vue (37) et de lumière (21). Kautz *et al.* [KBD07] ont utilisé un système de calcul d'éclairement global par tracé de rayons [PH04] pour créer des BTFs de pelouse et de tissus pour leur logiciel *BTFShop*. Ce logiciel permet d'éditer et de modifier avec cohérence les diverses composantes d'une BTF comme l'éclairement locale ou la méso-structure sous jacente. La validité des outils a pu être prouvée grâce à la comparaison entre les effets escomptés de ces outils sur l'apparence des BTFs mesurées et le résultat équivalent obtenu par génération de BTFs avec les paramètres correspondants et modifiés.

En effet, la génération de BTFs permet la validation d'un système d'acquisition avant sa mise en œuvre. De plus, l'environnement contrôlé de la génération permet de mettre en relation les variations de l'apparence d'un matériau et les paramètres sous-jacents du modèle de matériau. Ces atouts de la génération peuvent permettre par exemple de valider des méthodes d'estimation des propriétés des matériaux à partir de leur apparence. Tout comme l'acquisition, la génération présente une contrainte forte pour la durée de création des BTFs. La durée d'une génération est liée aux résolutions spatiales et angulaires et à la qualité du rendu, en particulier pour le calcul de l'éclairement indirect au sein de la méso-structure.

2.3 Discussion

Il est possible d'obtenir des BTFs en accédant simplement aux bases de données mises gratuitement en ligne par certains des auteurs de systèmes d'acquisition. La réalisation d'un système d'acquisition demande beaucoup d'investissements financiers et de ressources qui ne sont pas l'objectif principal de ce mémoire même si une idée d'un système d'acquisition utilisant des fibres optiques nous a paru très prometteuse en gain de temps et en mobilité par rapport aux systèmes existants.

Nous avons donc, dans un premier temps, comme la plupart des acteurs de la communauté sur les BTFs, travaillé à partir des BTFs déjà acquises et mises en ligne. Nous avons utilisé les BTFs fournies par l'université de Bonn [Bon] et par Magda et Kriegman [MK06]. Ces données nous ont permis de mener les premiers développements et la mise en place de notre logiciel intitulé *BTFInspect* pour la visualisation, l'analyse et la compression des BTFs (*cf.* Chapitre 4). Dans un second temps, nous avons eu recours à la génération de BTFs pour expérimenter et valider notre nouvelle représentation des BTFs (*cf.* Chapitre 5) sur des « BTFs idéales ».

Chapitre 3

Représentations compactes de BTFs

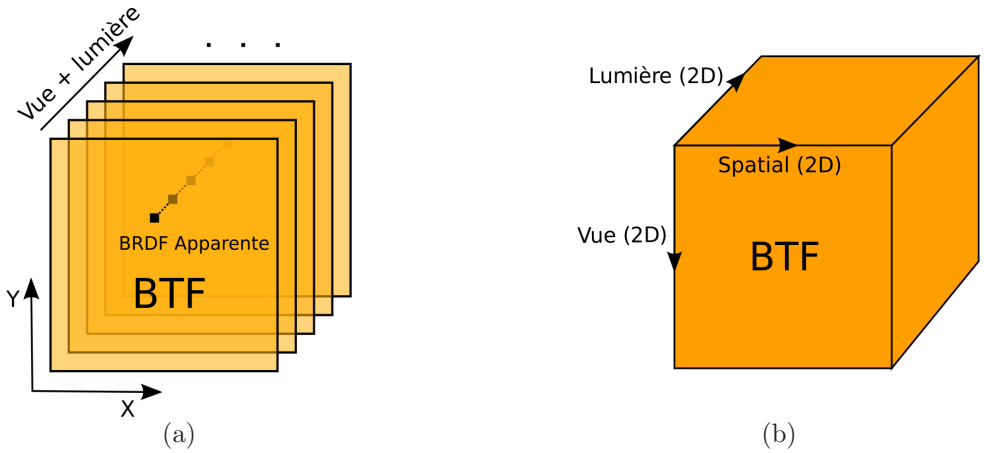


Fig. 3.1: *(a) Vue des BTFs comme des textures dépendantes des directions de vue et des directions de lumière. Une BRDF apparente peut correspondre à chaque texel. (b) Vue des BTFs comme un signal à 6 dimensions.*

À ce jour, les BTFs offrent un des meilleurs compromis pour les applications de synthèse d'images en temps-réel en terme de réalisme/temps de calcul avec une définition réaliste de l'apparence de la surface des objets. Cependant les BTFs à l'état brut sont inutilisables pour la synthèse d'images sur cartes graphiques du fait de leur taille imposante. Par exemple, une BTF de l'université de Bonn (*cf.* Section 2.1.1), avec ses 81 directions de vue et ses 81 directions lumineuses, comprend 6561 images pour un même matériau soit environ 10 giga-octets de données pour 256x256 pixels en 24 bits. La compression des images au format *JPEG* réduit sensiblement la taille des données d'un facteur dix mais cela reste très insuffisant pour charger en mémoire le grand nombre de BTFs nécessaires pour représenter les différents matériaux utilisés dans une scène.

Les BTFs nécessitent donc d'être compressées par des méthodes adaptées afin de réduire leur taille de l'ordre du giga-octets à quelques dizaines de méga-octets par BTF. Selon Müller *et al.* [MMS+05], les propriétés idéales d'une représentation compacte des BTFs sont bien sûr une **préservation du mieux possible de l'apparence** et une **exploitation de la redondance** des données, mais surtout une **décompression en temps-réel** qui soit adaptée aux cartes graphiques.

Dans toutes les méthodes proposées pour compresser les BTFs, on peut distinguer deux approches. La première considère une BTF comme une texture dont les texels varient en fonction du point de vue

et la direction lumineuse (*cf.* Figure 3.1). Cette approche vise en premier lieu à compresser les variations angulaires de chaque texel en les considérant comme des données de BRDFs ou de BRDFs apparentes (ABRDF) [WHON97b] qui contiennent des effets dépendant de la méso-géométrie. On recherche alors à optimiser par texel un modèle analytique pour représenter au mieux la BRDF apparente. La deuxième approche considère les BTFs comme un signal multidimensionnel (six dimensions) comme illustré sur la Figure 3.1 : on peut alors appliquer les méthodes classiques de traitement du signal ou de compression linéaire pour compresser les données. L'exemple le plus classique est une analyse en composantes principales (ACP).

3.1 Minimisation de modèles paramétriques

Les méthodes de minimisation de modèles paramétriques considèrent une BTF comme une texture dont chaque texel referme une ABRDF représentant les effets conjugués des propriétés locales de réflectance et de la méso-structure des surfaces. Ces méthodes cherchent alors à approximer les variations lumineuses de chaque texel avec des modèles analytiques de BRDFs ou d'autres fonctions paramétriques tels que des polynômes biquadratiques. La détermination des paramètres des modèles est faîte par des processus de minimisation linéaire ou non en fonction des modèles.

3.1.1 Minimisation par texel

Les travaux de McAllister *et al.* [MLH02] sur les SVBRDFs ont sans doute fortement inspirés l'utilisation de modèles analytiques de BRDF pour représenter les BTFs. Dans ces travaux, les SVBRDFs sont approximées par texel par des lobes du modèle de Lafortune [LFTG97] et d'ordinaire un ou deux lobes suffisent. Les paramètres des lobes sont déterminés par un processus de minimisation non-linéaire implémentant l'algorithme de Lenvenberg-Marquardt [PFTV88]. La validité des résultats dépend de la phase complexe et manuelle d'initialisation des paramètres et de la recherche non-linaire des solutions qui peut prendre jusqu'à plusieurs heures. Pour le rendu sur carte graphique, les paramètres des lobes sont stockés dans des textures et l'interpolation spatiale est effectuée par les fonctions internes des unités de texturation. Le modèle étant continu dans les dimensions angulaires, aucune interpolation n'est nécessaire pour les directions de vue et de lumière.

Cette méthode à base de lobes de Lafortune a été appliquée aux BTFs par Daubert *et al.* [DLHS01]. Cependant pour mieux représenter les données plus complexes des ABRDFs, le modèle est enrichi d'un facteur de visibilité pour chaque composante couleur et dépendant de la direction de vue ce qui permet de mieux représenter les phénomènes dus à la méso-structure telles que les ombres propres ou les occultations propres. La représentation devient plus dense car il faut stocker des valeurs supplémentaires pour chaque vue et chaque texel. Le processus de rendu est aussi plus complexe parce que, en plus du rendu des lobes, le terme de visibilité stocké dans des textures supplémentaires nécessite une interpolation manuelle avec les vues les plus proches. Cela implique autant d'accès textures supplémentaires que de vues proches considérées (3 ou 4) pour le rendu de chaque fragment.

Sur le même principe d'utilisation de lobes de modèles de BRDF, Ma *et al.* [MCC+04] minimise, toujours de manière non-linéaire, les paramètres du modèle de Blinn-Phong [Bli77] pour les données par texel et par vue d'une BTF. En d'autres termes, un lobe est utilisé pour représenter les données d'une ABRDF réduite à une vue fixée, ce qui permet d'ôter les effets de la parallaxe et donc d'être plus proche du comportement d'une BRDF. Cette méthode innove par la détermination d'un lobe moyen par texel à partir de la moyenne des paramètres de chaque lobe dépendant d'une vue donnée. Pour améliorer la qualité de représentation le résidu obtenu par texel, qui représente l'erreur par rapport aux données originelles, est alors approché par la composante spéculaire de Blinn car il est censé représenter les détails de haute fréquence. Le rendu très similaire aux techniques de *bump mapping* est très performant au détriment de la préservation de la qualité de l'apparence, tout comme l'approche de Daubert, et notamment pour les variations de haute fréquence comme les ombres ou les pics spéculaires.

Dernièrement, Forest et Paulin [FP05] ont proposé une méthode basée aussi sur celle McAllister *et al.* pour compresser les BTFs. Leur approche améliore le modèle de base avec une composante spéculaire analogue à celle de Blinn en vue d'améliorer la qualité. Le modèle reste performant mais

présente les mêmes difficultés que les méthodes précédentes pour bien représenter les effets lumineux liés aux méso-structures et notamment la visibilité qui dépend de la direction de vue.

3.1.2 Minimisation par texel et par vue

Il convient d'introduire ici la notion de *champ de réflectance* qui correspond aux variations de l'apparence d'une surface en fonction des directions lumineuses pour une vue donnée. Une BTF renferme donc autant de champs de réflectance différents que de directions de vue. Pour compresser des champs de réflectance acquis avec leur système, Malzbender *et al.* [MGW01] ont introduit les *polynomial texture maps* (PTMs) qui représentent, par texel et pour une vue donnée, les variations de luminance par un polynôme biquadratique et la chrominance par une couleur moyenne. Les coefficients des polynômes sont obtenus par une minimisation linéaire aux sens des moindres carrés. Ce champ de réflectance possède l'avantage de ne contenir aucun effet de parallaxe et Meseth *et al.* [MMK03a, MMK04] étendent la méthode des PTMs pour les différents champs de réflectance contenus dans une BTF. Cette représentation utilise soit des polynômes biquadratiques, similairement à Malzbender *et al.*, soit une modification du modèle de Lafortune indépendante de la direction de vue. L'évaluation du polynôme sur une carte graphique est triviale et l'interpolation spatiale des coefficients est assurée matériellement. Bien que cette approche améliore la qualité de la compression, l'occupation mémoire est supérieure d'un facteur du nombre de vue par rapport à la plupart des approches « par texel » présentées précédemment (*cf.* Section 3.1.1). De plus, la dépendance par direction de vue implique lors du rendu une recherche des vues les plus proches ainsi qu'une interpolation linéaire manuelle.

Filip et Haindl [FH04] augmentent encore la dépendance aux directions pour leur représentation par texel et approximent chaque champ de réflectance d'une BTF par une somme pondérée de lobes de Lafortune. Les coefficients de pondération dépendent d'un sous-ensemble des directions de lumière. En comparaison avec les approches de Meseth *et al.*, cette approche améliore la qualité pour la préservation de l'apparence mais au prix d'une augmentation de l'occupation en mémoire et d'une augmentation du temps de décompression au moment du rendu. En effet, après l'interpolation des vues les plus proches, il faut encore interpoler les directions de lumière les plus proches. Lors de sa publication, cette représentation ne permettait pas un rendu en temps-réel.

3.2 Réduction de dimensionnalité

Dans cette approche, les différentes méthodes considèrent une BTF comme un signal multidimensionnel et réalisent alors une compression linéaire des données qui se résume généralement à une analyse en composantes principales (ACP). Les méthodes se différencient dans la manière d'aborder les multiples dimensions : certaines méthodes compressent sur la totalité des six dimensions alors que d'autres ne compressent que par texel ou par direction de vue dans l'objectif d'accélérer la décompression au moment du rendu sur carte graphique.

3.2.1 Réduction sur données complètes

En organisant les données d'une BTF en une unique matrice, Koudelka *et al.* [KMBK03] et Liu *et al.* [LHZ$^+$04] compressent la totalité des données avec une ACP. La matrice est construite en arrangeant les ABRDFs d'une BTF en colonne. Chaque ligne de la matrice correspond donc à la texture pour une vue et une direction lumineuse données. Pour une BTF de la base de Bonn, cela représente une matrice de 256×256 soit 65 536 colonnes et 81×81 soit 6 561 lignes pour chaque composante de couleur. Koudelka *et al.* [KMBK03] affirment que pour représenter fidèlement tous les effets lumineux d'une BTF, les 150 premières valeurs principales suffisent. Les vecteurs obtenus sont ensuite compressés au format *JPEG* ce qui rend cette représentation impraticable pour une reconstruction de la BTF en temps-réel pour les cartes graphiques actuelles.

Afin d'améliorer la cohérence de l'analyse par rapport aux données, Vasilescu et Terzopoulos [VT04] proposent de guider l'ACP selon les directions de vue, de lumière ou les variations spatiales. Dans

ce but, une analyse multilinéaire considère les données comme un tenseur à trois modes et une décomposition en valeurs singulières (DVS) est effectuée pour chacun d'eux. La méthode apporte une meilleure réduction de dimensionnalité et la qualité est supérieure à une ACP pour des vecteurs de taille identique. La taille de la matrice, en précision flottante, requise par ces méthodes peut atteindre plusieurs giga-octets en mémoire et le calcul de la décomposition peut être très long. Ainsi ces méthodes ne peuvent s'appliquer qu'à des sous ensembles spatiaux des BTFs. En 2005, Wang *et al.* [WWS+05] améliorent la prise en compte de la cohérence spatiale dans l'analyse multilinéaire de Vasilescu et Terzopoulos et de plus, pour pallier au problème de taille mémoire de la matrice, l'analyse est menée hors mémoire.

3.2.2 Réduction par texel

Comme pour l'approche d'optimisation de la Section 3.1, ces méthodes s'intéressent à la compression des ABRDFs par texel d'une BTF. Suykens *et al.* [SvLD03] ont développé une méthode appelée « factorisation chaînée de matrice » (*Chained Matrix Factorization*) qui s'inspire des travaux précurseurs de McCool *et al.* [MAA01] sur la factorisation homomorphique de BRDFs dont le principe consiste à approximer un signal en quatre dimensions par un produit de fonctions en deux dimensions. Ces fonctions sont déterminées par la projection des fonctions en quatre dimensions sur deux dimensions à l'aide de différentes paramétrisations spécifiques. L'apport de Suykens *et al.* réside dans la combinaison de plusieurs de ces factorisations avec des paramétrisations différentes à chaque fois. Cette formulation permet une meilleure dynamique des coefficients pour la quantification traditionnelle sur 8 bits des textures. L'application de cette méthode sur les ABRDFs d'une BTF donne un taux de compression assez faible et un partitionnement est alors nécessaire pour indexer spatialement les ABRDFs similaires et réduire la taille mémoire. L'algorithme de décompression permet un rendu temps-réel mais la réduction de mémoire est insuffisante pour permettre à la fois d'avoir une bonne qualité et de pouvoir compresser la totalité des ABRDFs d'une BTF.

Une toute autre approche par Ma *et al.* [MCT+05] décompose chaque image d'une BTF en pyramide Laplacienne. Pour chaque niveau, les ABRDFs par texel - ou « BRDFs Laplaciennes » par texel telles que nommées par les auteurs - sont compressées par une ACP, hormis le plus haut niveau qui correspond aux plus basses variations et qui est compressé à l'aide d'un modèle de Phong. Cette méthode est multi-résolution et apporte un taux de compression parmi les meilleurs. Cependant le rendu temps-réel n'est pas garanti selon la densité de texels représentant l'objet à l'écran.

Bien qu'ils ne s'appliquent qu'aux SVBRDFs, on peut aussi citer pour leur originalité les travaux de Lawrence *et al.* [LBAD+06] qui compressent les données par texel de SVBRDFs par une décomposition hiérarchique de fonctions 1D et 2D. La décomposition matricielle utilisée est classique mais l'apport réside dans la factorisation qui est contrainte pour garantir la positivité ou la nullité des coefficients ainsi que la linéarité de leurs combinaisons. Ces trois propriétés permettent une décomposition éditable en temps-réel mais applicable uniquement aux SVBRDFs. La méso-structure des BTFs impliquant une interdépendance des texels, cela rend difficile l'adaptation de cette méthode.

3.2.3 Réduction par texel et par vue

Satler *et al.* [SSK03] effectuent une ACP par texel pour chaque champ de réflectance d'une BTF. La matrice par texel et par vue est plus légère en mémoire et donc plus rapide à analyser. Le nombre de valeurs propres déterminantes pour conserver une bonne qualité est entre 4 et 16 par vue, soit un facteur 10 fois supérieur aux méthodes de réduction sur les données complètes. La taille des données reste importante et la reconstruction est à peine temps-réel entre la recombinaison des valeurs propres et l'interpolation explicite pour les directions de vue et les directions de lumière les plus proches.

3.2.4 Réduction par partitions

Au lieu d'analyser les données par champ de réflectance, Müller *et al.* [MMK03b] partitionnent les données des ABRDFs par texel indépendamment des directions de lumière et de vue. Chaque partition

est ensuite analysée avec une ACP et la méthode recherche itérativement le meilleur partitionnement en se basant sur l'erreur de reconstruction de l'ACP comme critère de sélection. Les auteurs obtiennent une bonne qualité pour un nombre de partitions entre 16 et 32 avec huit valeurs propres pour chacune. Malgré un accès texture supplémentaire pour indexer les partitions par rapport à Satler *et al.* (*cf.* Section 3.2.3), le rendu est sensiblement plus rapide car la méthode nécessite moins de valeurs (32 clusters avec 8 valeurs propres, au lieu de 16 valeurs pour 81 vues pour Satler). Cependant la réduction de données ne se fait pas sans une perte en qualité de reconstruction.

Haindl et Filip [HF07] partitionnent spatialement les données de BTFs et proposent une analyse probabiliste des données guidée par une estimation d'un champ de hauteurs et d'un champ de normales pour chaque partition. Cette approche est développée dans le cadre de la synthèse de texture et le rendu interactif n'est pas praticable.

3.3 Discussion

Travaux	Résolution spatial	Taux de compression	Préservation de la qualité	Vitesse de décompression
Daubert *et al.* [DLHS01]	256×256	$37,5 : 1$	● ○ ○	● ● ●
Ma *et al.* [MCC+04]	256×256	n.c.	● ○ ○	● ● ●
Meseth *et al.* [MMK03a, MMK04]	256×256	$11 : 1$	● ○ ○	● ● ○
Filip et Haindl [FH04]	256×256	n.c.	● ● ○	● ○ ○
Koudelka *et al.*[KMBK03]	128×128	$470 : 1$	● ● ●	○ ○ ○
Liu *et al.* [LHZ+04]	256×256	$\approx 70 : 1$	● ● ●	● ○ ○
Vasilescu et Terzopoulos [VT04]	256×256	n.c.	● ● ●	○ ○ ○
Wang *et al.* [WWS+05]	192×192	$48 : 1$	● ● ●	○ ○ ○
Suykens *et al.*[SvLD03]	256×256	$20 : 1$	● ● ○	● ● ○
Ma *et al.* [MCT+05]	256×256	$44 : 1$	● ● ○	● ● ●
Satler *et al.* [SSK03]	256×256	$10 : 1$	● ● ○	● ● ○
Müller *et al.* [MMK03b]	256×256	$100 : 1$	● ● ○	● ● ●
Haindl et Filip [HF07]	256×256	$\approx 1 \times 10^5 : 1$	● ● ○	○ ○ ○

TAB. 3.1: *Résumé des propriétés en termes de représentation compacte des BTFs pour les différentes méthodes présentées. La note maximale pour la décompression temps-réel/interactive correspond au mieux à environ 30 images par seconde.*

Le tableau 3.1 ci-dessus résume et classifie approximativement les propriétés des différentes méthodes concernant le taux de compression, la préservation de la qualité et la commodité d'une décompression sur carte graphique. D'un point de vue qualitatif, les méthodes de réduction de dimensionnalité sont celles qui donnent les meilleurs résultats pour la préservation de l'apparence originale. Cependant les méthodes offrant les meilleurs taux de compression ne bénéficient pas toujours d'une décompression rapide sur carte graphique. Les résultats des différentes méthodes démontrent que le fait de considérer les données par texel est un facteur déterminant pour une décompression rapide sur les cartes graphiques. En effet, les BTFs restent conceptuellement proches des textures classiques et les méthodes compressant selon les dimensions spatiales ne tirent pas parti de l'accélération matérielle pour l'interpolation linéaire au moment du placage. Les performances de la décompression s'en trouvent alors fortement réduites. Les travaux de Müller *et al.* [MMK03b] et de Ma *et al.* [MCT+05] nous apparaissent comme les meilleures méthodes de compression pour une réduction des données significative, une bonne préservation de la qualité et un rendu temps-réel sur les cartes graphiques actuelles (≈ 30 images/seconde). Les performances restent tout de même limitées au rendu d'objet seul à l'écran et on ne peut pas vraiment envisager à l'heure actuelle une utilisation massive des BTFs pour tous les objets d'une scène. La Figure 3.2, issue de l'état de l'art de Müller *et al.* [MMS+04], montre une comparaison qualitative pour différentes méthodes pour une ABRDF issue d'une BTF représentant un échantillon de béton aggloméré.

De plus, les approches par réduction de dimensionnalité présentent intrinsèquement plusieurs inconvénients pour la décompression sur carte graphique. Quelque soit l'arrangement des données compressées, les méthodes linéaires induisent des accès mémoire aléatoires lors de la décompression, provoquant des incohérences de cache et diminuant les performances des cartes graphiques qui ne sont pas conçues à cet effet. Un autre inconvénient de ces méthodes lors du rendu est que les interpolations pour les directions de vue et les directions de lumière les plus proches doivent être déterminées explicitement. L'interpolation des données n'est pas faite implicitement par la carte graphique comme pour l'interpolation entre texels voisins. Cet inconvénient se retrouve d'ailleurs sur toutes les méthodes qui compressent les données par vue ou par direction lumineuse ce qui correspond à peu près dans toutes

les méthodes présentées précédemment. Seules les méthodes de McAllister *et al.* [MLH02] et Ma *et al.* [MCC+04] utilisent des modèles à base de lobes garantissant une définition continue des données pour les domaines angulaires. Cependant la première est limitée aux SVBRDFs et la seconde fournit une trop faible préservation de la qualité.

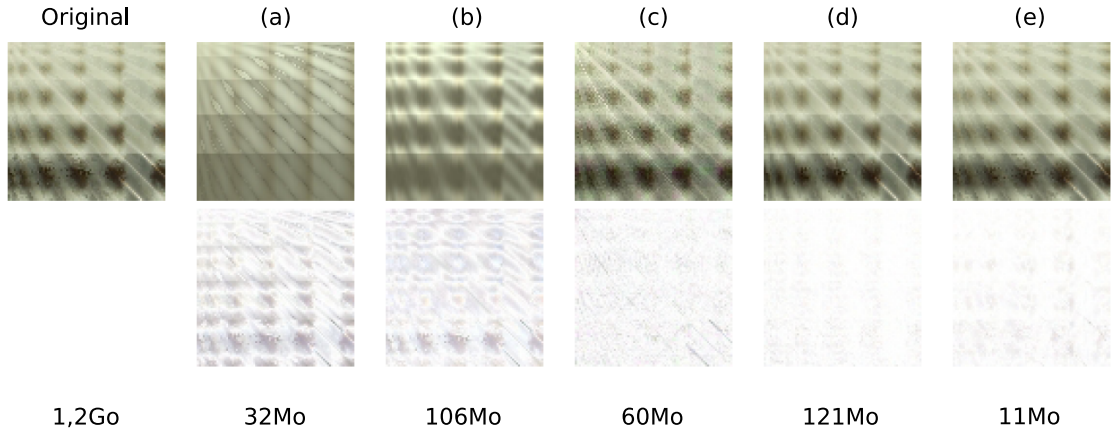


FIG. 3.2: *Comparaison visuelle de la préservation de l'apparence de Müller et al. [MMS+04] pour une sélection des différentes méthodes présentées. Sur la ligne du haut, une ABRDF représentée dans une image (cf. Section 4.3.1 pour plus de détails) de la BTF d'origine et les ABRDFs reconstruites et sur la ligne du bas, les images inversées de la différence avec l'ABRDF originale. De gauche à droite, une ABRDF originale issue d'une BTF de béton aggloméré, en (a) les lobes pondérés de Lafortune de Daubert et al. [DLHS01], en (b) les champs de réflectance par vue de Meseth et al. [MMK03a], en (c) la méthode de factorisation chaînée de Suykens et al. [SvLD03] avec 4 facteurs, en (d) la compression par ACP par vue de Satler et al. [SSK03] avec 8 termes et enfin en (e) la méthode de réduction par partition de Müller et al. [MMK03b] avec 32 clusters et 8 termes.*

D'après notre analyse des atouts et des inconvénients de toutes ces approches, on peut extraire les bonnes propriétés d'un rendu performant pour les représentations compactes des BTFs :

1. La **compression des données par texel** et donc la considération des BTFs comme des textures d'ABRDFs pour pouvoir bénéficier de l'interpolation linéaire sur les cartes graphiques.
2. L'utilisation de méthodes de compression ne produisant **pas d'accès aléatoires** ou fortement non-linéaires lors du processus de reconstruction à partir des données compressées.
3. La garantie d'une **définition continue pour les dimensions angulaires** ce qui induit une interpolation implicite des données.

L'approche qui semble répondre le mieux à ces critères est l'approximation par modèles analytiques de BRDF par texel. Cette approche est la plus naturelle pour les cartes graphiques et vise à compresser les BTFs en quelques dizaines de textures. Les taux de compression de cette approche sont très compétitifs comparés aux méthodes de Müller *et al.* [MMK03b] et de Ma *et al.* [MCT+05]. Malheureusement, l'inconvénient majeur de cette approche est une faible préservation de la qualité de l'apparence due à la complexité des phénomènes entrant en jeu dans les ABRDFs à représenter. Une des raisons est sans aucun doute le fait que ces modèles n'ont pas été développés à l'origine pour représenter de telles variabilités dans les données.

En visant une amélioration de la préservation de la qualité pour cette approche, nous avons effectué nos recherches dans le but d'améliorer ou créer des représentations compactes en utilisant des modèles analytiques qui permettent une bonne représentation des ABRDFs. La première étape consiste à comprendre la complexité des ABRDFs pour entrevoir un moyen de les simplifier et/ou de les compresser. Dans le chapitre suivant, nous présentons notre logiciel *BTFInspect* pour la manipulation et l'étude

des BTFs. Nous analysons et mettons en évidence les différents phénomènes entrant en jeu ainsi que leur importance dans la constitution des ABRDFs.

Chapitre 4

BTFInspect : Un outil d'analyse et d'étude de BTFs

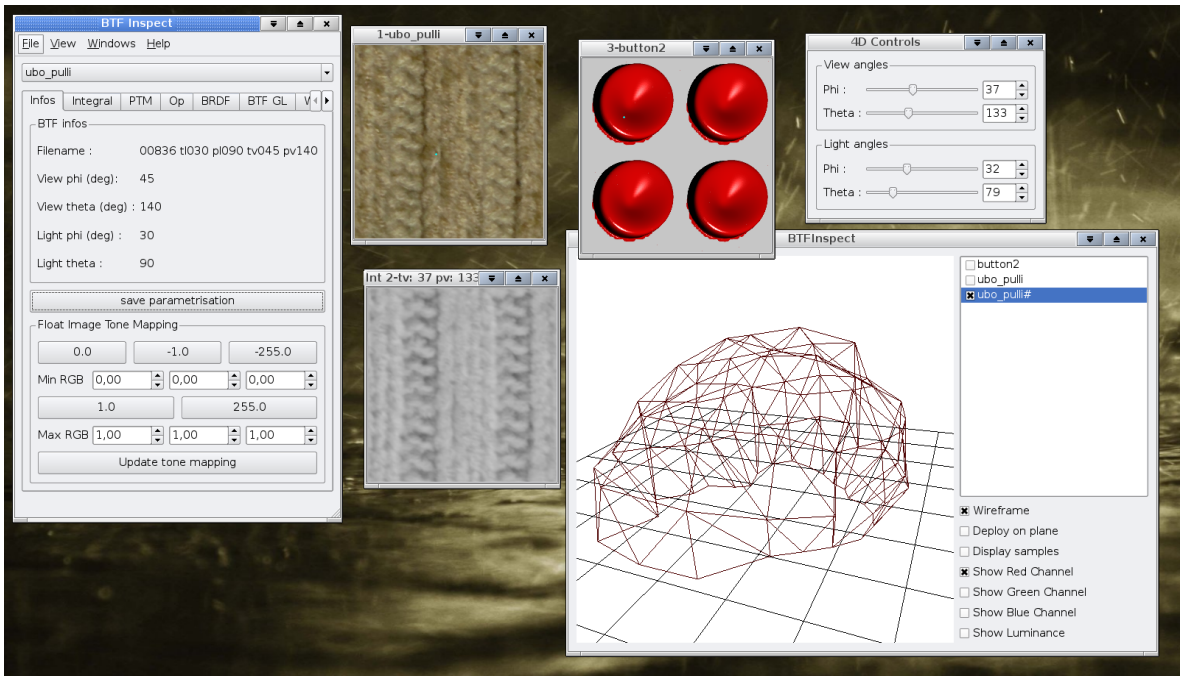


Fig. 4.1: *Un aperçu de l'interface et des fonctionnalités de notre logiciel BTFInspect.*

Les BTFs sont des données complexes à manipuler, du fait de leur taille et de leur grande dimensionnalité. Aucun outil de référence n'existant, nous avons dû penser et développer nos propres outils afin de pouvoir utiliser les différentes BTFs provenant des bases de données en ligne [Bon, MK06]. Notre logiciel *BTFInspect* est un logiciel de recherche et d'étude permettant la manipulation, la visualisation, l'analyse, la compression et le rendu 3D de données issues des BTFs. Son premier objectif est de fournir un moyen de navigation rapide et intuitif dans les données d'une BTF. Les motivations et les détails de cette fonctionnalité sont décrits dans la Section 4.1. Ce logiciel a été la pierre angulaire dans la construction de nos études sur les BTFs ayant pour objectifs de mieux comprendre les différentes phénomènes et de mieux appréhender intuitivement les difficultés auxquelles font face les méthodes de compression par texel. Plus particulièrement, une des fonctionnalités génère un aperçu de la méso-structure et grâce à nos outils de manipulation, nous observons les effets de la parallaxe

dans les images. Enfin, une autre fonctionnalité nous permet d'extraire des ABRDFs depuis les texels d'une BTF et d'observer les variations des différents effets de lumière, d'ombrage et d'occultation à l'aide d'une image ou d'une visualisation en 3D d'une ABRDF.

4.1 Manipulation des BTFs

Généralement, les BTFs sont stockées comme une collection d'images, une par couple point de vue/direction d'illumination (*cf.* les bases de données en ligne [Bon, MK06]). Comme introduit dans la Section 1.5, les directions sont exprimées selon leur coordonnées sphériques (θ, ϕ). Les paramétrisations angulaires des directions de vue et de lumière pour les images diffèrent selon le procédé d'acquisition. Les mesures de l'université de Bonn sont uniformes selon la surface d'un hémisphère et comprennent 81 directions pour les directions de vue et de lumière. Les images d'une BTF sont classées par direction de vue dans des sous-répertoires. Les mesures angulaires de Magda et Kriegman [MK06] ont été faites linéairement avec des pas de 15 degrés pour les directions lumineuses et de 20 degrés pour les directions de vue à la fois pour l'angle θ et l'angle ϕ. Le nombre d'images est supérieur à 10 000 et leur gestion constitue un problème à part entière.

4.1.1 Motivations

La fonctionnalité de base est naturellement la visualisation de la variation de l'apparence de l'échantillon de surface que constitue une BTF, et ce, pour tous les points de vue et les directions d'illumination. La représentation naturelle des BTFs, sous forme d'images dépendantes de la direction de vue et de lumière est donc tout à fait adéquate dans notre cas. Notons que pour leur travaux d'éditions de BTFs, Kautz et al. [KBD07] ont dû réorganiser le stockage des données en petits paquets de 32×32 de résolution spatiale et 3×3 de résolution angulaire pour obtenir un bon compromis entre une édition interactive et des temps de calcul raisonnables pour modifier les données.

La visualisation en images des BTFs implique donc une navigation dans les quatre dimensions restantes : les dimensions angulaires. Un utilisateur doit pouvoir naviguer en continu dans les dimensions angulaires sans être lié aux différentes paramétrisations angulaires des BTFs. Le logiciel doit se charger de trouver interactivement les images les plus proches en fonction des paramètres choisis par l'utilisateur. Enfin, le logiciel doit permettre la navigation pour plusieurs BTFs à la fois dans une même instance. Sachant que la taille des données pour une BTF est de l'ordre du giga octet, la navigation dans les données ne peut se faire que hors-mémoire.

4.1.2 Structure de données

En accord avec nos motivations pour notre logiciel, nous avons développé une structure de données permettant d'indexer des données variant selon quatre dimensions et remplissant les exigences suivantes :

- la gestion de manière générique des différentes paramétrisations angulaires ;
- la dissociation pour chaque vue d'une possible paramétrisation différente pour les directions lumineuses, et inversement ;
- la gestion de manière générique des données indexées ;
- une recherche et un accès rapide de la ou des données les plus proches pour une direction de vue ou/et de lumière donnée.

Les coordonnées sphériques impliquent une gestion de la périodicité de l'angle azimutale ϕ. Afin d'éviter cette gestion et puisque les directions sont exprimées dans un hémisphère, on peut projeter les directions unitaires dans le plan correspondant à l'angle azimutale ϕ (*cf.* Figure 4.2). Les directions s'expriment alors par leurs coordonnées cartésiennes $(x, y) \in [-1, 1]$ et la périodicité est prise en compte implicitement. Pour conserver une indépendance à la paramétrisation, une structure de dictionnaire indexée par les directions nous semble ici bien adaptée. Les deux dimensions de la paramétrisation des directions projetées sont représentées en tant que clef d'un double dictionnaire nommé $HemiDico2D$

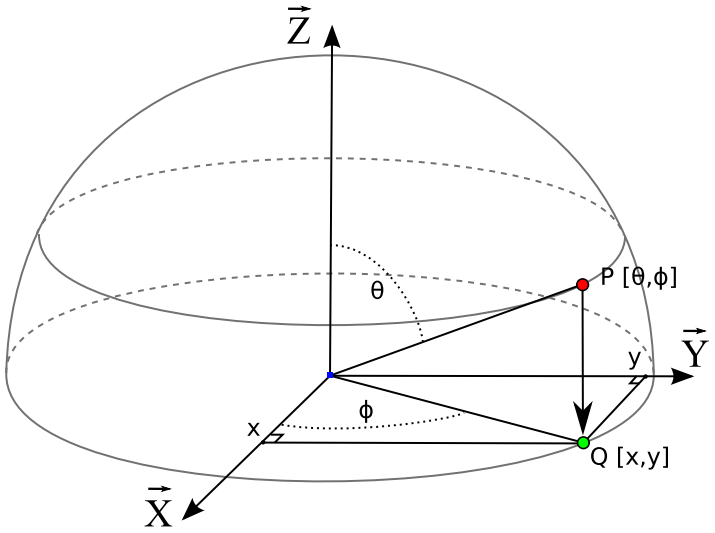


FIG. 4.2: *Les directions sont toujours exprimées selon un hémisphère unitaire et alignées sur l'axe des Z positifs dans un repère orthonormé. Il existe alors une bijection entre la coordonnée sphérique (θ,ϕ) d'un point P et sa projection Q dans le plan XY de la base de l'hémisphère et de coordonnée cartésienne (x,y).*

avec un premier dictionnaire[1] dont la clef est la coordonnée x d'une direction et qui indexe un second dictionnaire ayant pour clef la coordonnée y et qui indexe la donnée correspondante :

$$HemiDico2D(x, y, donnée) = dictionnaire[x](dictionnaire[y](donnée)) \tag{4.1}$$

Cette structure permet un accès direct aux données avec les directions clefs. Pour une direction arbitraire, la structure renvoie la donnée de la direction la plus proche au sens de la distance euclidienne pour les coordonnées (x, y).

En considérant que l'espace angulaire 2D est identique pour les directions de vue et les directions de lumière, notre structure de données 4D nommé $HemiDico4D$ est construite avec une $HemiDico2D$ indexant pour chaque direction de vue, une seconde structure $HemiDico2D$ qui indexe les données pour les directions lumineuses :

$$HemiDico4D(x_r, y_r, x_i, y_i, d) = HemiDico2D(x_r, y_r, HemiDico2D(x_i, y_i, d)) \tag{4.2}$$

avec $(x_r,\ y_r)$ les coordonnées de la direction de vue, $(x_i,\ y_i)$ les coordonnées de la direction d'illumination et d la donnée générique indexée. La structure $HemiDico4D$ propose aussi des itérateurs spécifiques permettant des itérations sur les données des BTFs ayant les mêmes directions de lumière ou les mêmes directions de vue (champ de réflectance).

Cette structure de données nous permet de gérer tout type de paramétrisation et le type de données indexées par la structure est générique. Par conséquent n'importe quelle donnée 4D peut être indexée avec notre structure. Pour des BTFs, nous indexons simplement les noms des fichiers images. De cette manière, la totalité des données est indexée et accessible à tout moment mais les données restent hors-mémoire, seules la ou les images nécessaires pour la visualisation en cours sont chargées. Pour les BRDFs ou les ABRDFs, nous indexons une valeur de luminance ou une couleur RVB.

[1]Nous avons utilisé la structure *map* de la STL en C++ qui implémente un dictionnaire générique

4.2 Étude du phénomène de parallaxe

Le phénomène de parallaxe se définit comme un déplacement de la position apparente d'un corps, dû au changement de position de l'observateur. Pour comprendre la parallaxe dans le cas des BTFs, il faut avoir une idée de la méso-structure sous-jacente des matériaux observés selon les différentes directions de vue.

4.2.1 Mise en valeur de la méso-structure sous-jacente

On sait que la parallaxe est en corrélation avec les directions de vue et chacune d'elles est associée à une certaine visibilité de la méso-structure des matériaux. La connaissance des directions de vue pour chaque image d'une BTF est immédiate mais la méso-structure des matériaux reste inconnue car seuls les effets qu'elle induit participent à la définition d'origine des BTFs. Il existe bien le dispositif de Wang et al. [WTS+05] (*cf.* Section 2.1.1) qui permet à la fois la mesure des BTFs et de la méso-structure mais les données ne sont pas disponibles en ligne. Bien qu'avec quelques images bien choisies, notre cerveau interprète et évalue facilement la méso-structure sous-jacente, l'opération consistant à la reconstruire précisément à partir des images d'une BTF reste très complexe. Nous verrons dans le Chapitre 5 au travers des différentes méthodes existantes que tout un domaine de recherche, notamment dans le domaine de la vision par ordinateur, est consacré à ce problème. Sans recourir à des procédés complexes, on peut obtenir une certaine représentation visuelle implicite de la méso-structure à partir des images et d'un calcul simple. Un opérateur de notre logiciel permet d'estimer d'une certaine manière l'occultation ambiante à partir des champs de réflectance définis pour chaque vue. Comme le montre l'équation 4.3, à une vue donnée la luminance moyenne normalisée est évaluée sur l'ensemble des directions lumineuses :

$$L_{\approx ambient}(x, \vec{v}) = \frac{1}{l_{max}} \sum_{i}^{K} l_i \tag{4.3}$$

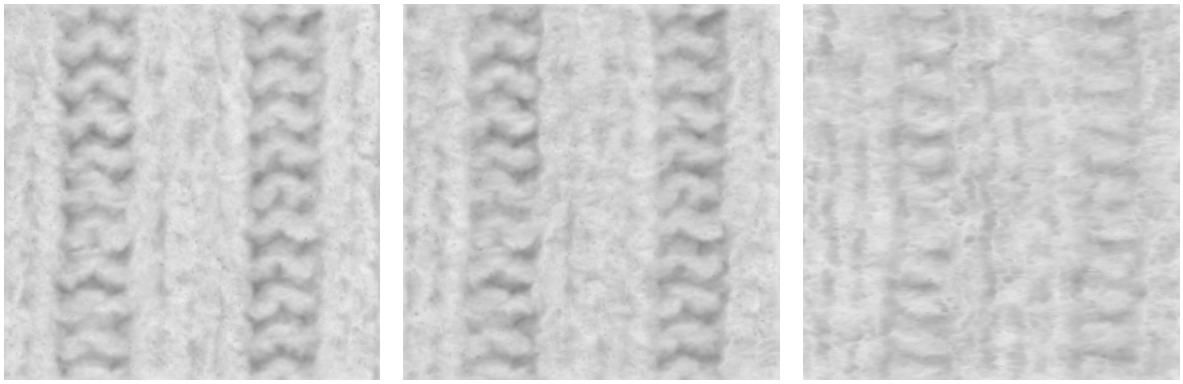
avec x la position dans l'image, $\vec{v}$ la direction de vue fixée, l_{max} la valeur maximale de la luminance pour l'ensemble des directions lumineuses K et l_i la luminance pour la direction i. Dans nos expérimentations, la luminance correspond à la composante L dans l'espace de couleur *Lab*. La formulation de notre opérateur (*cf.* Eq. 4.3) est assez proche de la définition de l'occultation ambiante :

$$A_p = \frac{1}{\pi} \int_{\Omega} V(p, \vec{\omega})(\vec{n} \cdot \vec{\omega})\, d\omega \tag{4.4}$$

avec $V(p, \vec{\omega})$ une fonction binaire de visibilité du point p dans la direction $\vec{\omega}$ et $\vec{n}$ la normale à la surface. La différence est que la visibilité n'est pas déterminée en fonction de la vue mais par une estimation de la visibilité de la source lumineuse. La Figure 4.3 montre les images obtenues avec cet opérateur sur les images de la partie haute de la Figure 4.4. Cet opérateur reste avant tout une estimation de la méso-structure et il n'est utilisé que pour mieux appréhender les phénomènes qui y sont liés dans nos observations.

4.2.2 Observation dans les images

Avec notre logiciel *BTFInspect*, nous avons pu étudier les effets de la parallaxe dans les images des BTFs, en gardant une direction lumineuse fixe et en faisant varier la direction de vue. En partant de l'image de la vue zénithale ($\theta = 0$, $\phi = 0$) comme référence, on observe des **déplacements de la position apparente** pour certaines parties de la surface des matériaux. En plus des déplacements, des parties visibles sur l'image de référence disparaissent tandis que de nouvelles apparaissent au grès de la visibilité de la méso-structure sous-jacente des matériaux. Comme on peut le voir sur la Figure 4.4 pour la BTF d'un échantillon de tricot, plus un point du relief est proche de l'observateur et plus les déplacements de sa position apparente sont importants lorsque la vue devient rasante. Par exemple, le point A pointant sur un creux de la surface du matériau se déplace de sept pixels entre l'image de référence et l'image correspondante à un angle θ de 60 degrés pour l'angle de vue. Pour le point B



$\theta = 0$ $\theta = 30$ $\theta = 60$

FIG. 4.3: *Estimation de l'occultation ambiante pour différents angles d'élévation (θ) de la BTF de tricot.*

pris sur une crête de la méso-structure, la position apparente se déplace de 30 pixels pour le même changement de direction de vue. Le point C fait référence à une partie de la méso-structure visible uniquement pour la vue de référence. La BTF qui représente un échantillon de papier peint, ne possède pratiquement pas de méso-structure. Contrairement à la BTF du tricot où la méso-structure induit d'importants effets de parallaxe, aucun effet n'est discernable pour cette résolution d'échantillonnage spatiale. En d'autres termes, la BTF de papier peint, au vue des faibles variations de sa méso-structure résultant en une absence d'effets de parallaxe, peut être simplement considérée comme une SVBRDF.

4.3 Étude des ABRDFs

Il est assez difficile d'appréhender la complexité des différents phénomènes intervenant dans les ABRDFs des BTFs sans en avoir une approche visuelle. Quelle est l'influence de la parallaxe dans les ABRDFs ? À quoi ressemble une ABRDF issue d'une BTF comparée à un lobe ou un polynôme biquadratique ? Ou encore tout simplement quelle est la complexité visuelle d'une ABRDFs au niveau colorimétrique ou de la luminance ? Pour essayer de répondre à toutes ces questions, nous avons développé plusieurs outils qui permettent de visualiser la totalité des données d'une ABRDF dans une image ou des sous-ensembles en 3D.

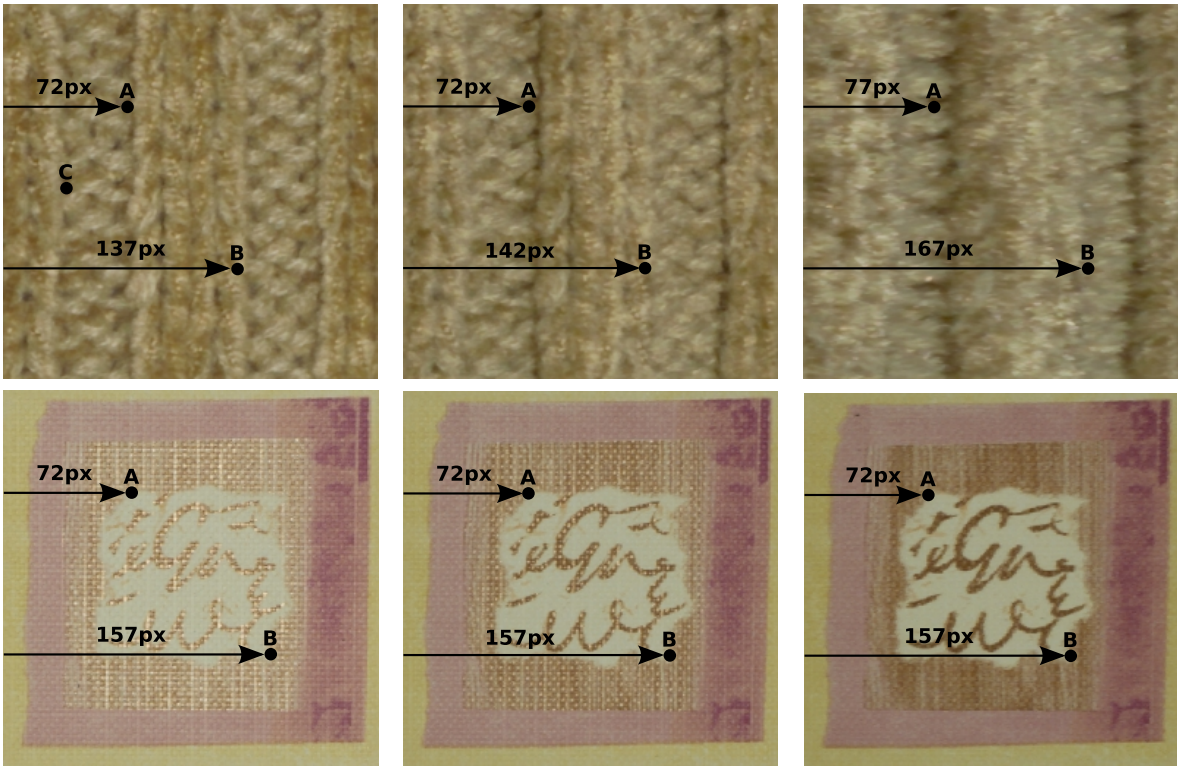
4.3.1 Vue en image

La visualisation d'une ABRDF en totalité dans une image n'est pas triviale car il faut représenter des données variant selon quatre dimensions sur un support variant seulement selon deux dimensions. Cependant l'idée devient possible en reparamétrisant l'espace des directions à l'aide d'une fonction monodimensionnelle comme le parcours en spirale sur la Figure 4.5.

4.3.1.1 Construction

L'ensemble des directions, pour la vue ou la source lumineuse, correspond à différents échantillons sur la surface d'un hémisphère. Une direction est généralement définie par commodité selon deux angles sphériques mais en définissant une fonction monodimensionnelle continue permettant le parcours de tous ces échantillons. Les directions s'expriment alors avec un seul paramètre. Par exemple, on peut prendre une fonction en spirale qui part du pôle de l'hémisphère et qui parcourt les latitudes en devenant de plus en plus rasante (*cf.* Figure 4.5).

En parcourant en spirale les directions, les données d'une ABRDF réduites à deux dimensions peuvent alors être visualisées sous forme d'une image en prenant comme dimensions le parcours des



$\theta = 0$ $\theta = 30$ $\theta = 60$

FIG. 4.4: *Observation de la parallaxe pour différents angles d'élévation (θ) dans les BTFs de Bonn [Bon]. La direction lumineuse est fixe ($\theta = 0, \phi = 0$). En haut, images de la BTF de tricot présentant un relief important et donc une parallaxe importante. En bas, images de la BTF de papier peint présentant une méso-structure quasi-plane, le phénomène de parallaxe n'est pas perceptible dans les images.*

directions de vue et le parcours des directions lumineuses. Les Figures 4.6 et 4.7 montrent différentes ABRDFs extraites de plusieurs BTFs caractéristiques.

4.3.1.2 Interprétation des images

La représentation en image illustrée sur les Figures 4.6 et 4.7 permet de mieux se rendre compte de tous les phénomènes complexes que l'on retrouve dans les ABRDFs issues des BTFs. Il faut prendre en compte dans les observations des phénomènes l'aspect en mosaïque des images qui est produit principalement par la paramétrisation en spirale. Les variations des phénomènes qui dépendent de la direction de vue se lisent selon l'axe vertical des images et se retrouvent logiquement pour toutes les directions lumineuses lorsque les phénomènes ne dépendent que de la direction de vue. De même, les variations des phénomènes qui dépendent de la direction lumineuse se lisent selon l'axe horizontale des images, mais ne se retrouvent pas pour toutes les directions vues à cause de la parallaxe et ce, quelque soient les phénomènes. Les paragraphes qui suivent décrivent succinctement les différentes informations que l'on peut tirer de cette représentation.

Tout d'abord, l'aspect général des images au niveau de la fréquence spatiale permet de juger de la qualité de l'échantillonnage angulaire pour l'apparence des BTFs. On voit clairement que la fréquence d'échantillonnage est insuffisante dans certains cas comme pour les ABRDFs Figures 4.6-(b) et 4.7-(c) qui présentent des images très bruitées. L'aspect général donne aussi un bon aperçu des variations

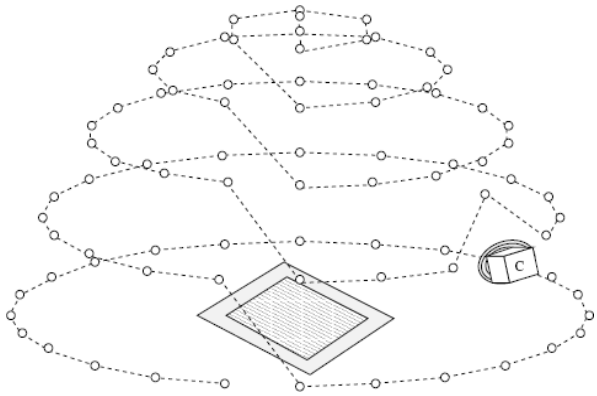


FIG. 4.5: *Parcours en spirale des échantillons de l'espace des directions (schéma de Filip et Haindl [FH04].*

complexes de couleurs et de luminosité dépendant bien à la fois des directions de vue et de lumière. On comprend mieux les différents choix des méthodes de compression existantes et pourquoi les modèles analytiques de BRDF peinent à représenter des ABRDFs.

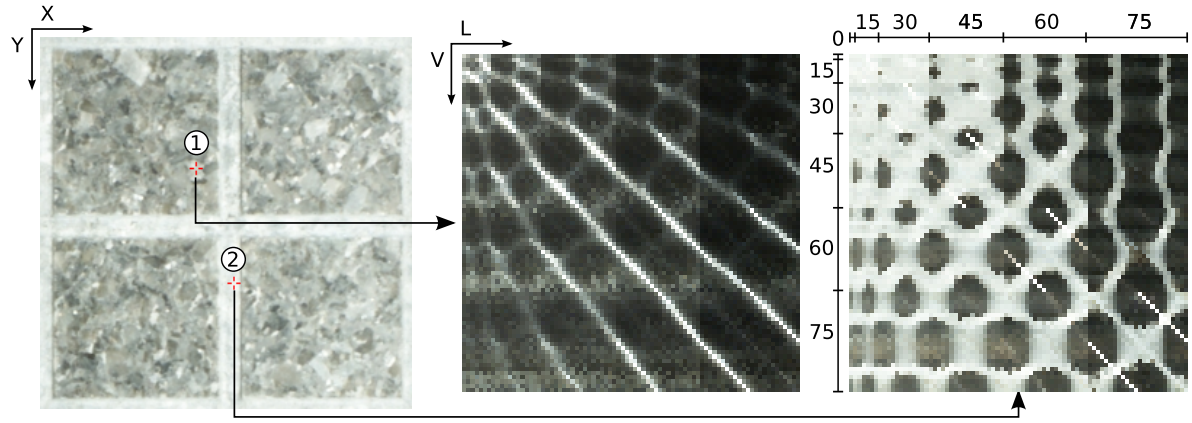
Les images permettent aussi d'interpréter les propriétés des BRDFs comprises dans les ABRDFs. On voit par exemple que l'échantillon de granite en Figure 4.6-(a)-1 présente les spécularités les plus importantes alors que le béton aggloméré en Figure 4.7-(d)-1 est le plus diffus. Les spécularités sont visibles par les grands traits lumineux traversant les images en diagonale et illustrent les configurations de directions de vue et de lumière pour lesquelles la luminosité est maximale. Les propriétés diffuses se discernent par des variations de basses fréquences dans les cellules, une diminution graduelle de la luminosité quand les directions de lumière deviennent rasantes (*cf.* Figure 4.7-(d)-1).

Les phénomènes d'ombrage propre, qui dépendent de la visibilité de la source lumineuse, se lisent sur l'axe horizontal lorsque les directions de lumière deviennent rasantes. Cela correspond dans nos images aux pixels des intervalles correspondant aux angles $\phi = 60$ et $\phi = 75$ sur l'axe L. Les ABRDFs des Figures 4.6-(a)-2, 4.6-(b)-2 et 4.7-(d)-2 illustrent bien les ombres générées par la méso-structure. Notons que si l'on pouvait décorréler les effets d'ombrage propre de la parallaxe, les ombres seraient verticalement constantes dans les images.

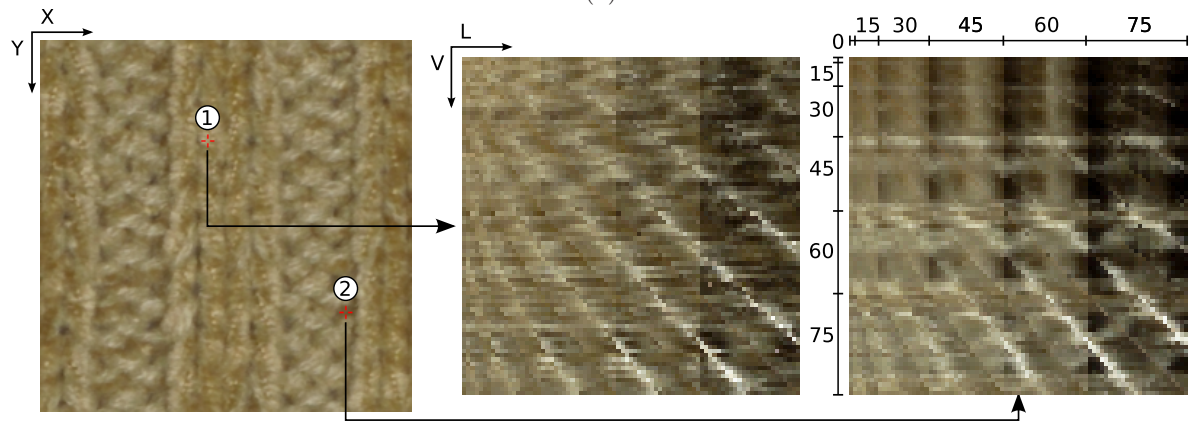
La parallaxe fait partie des phénomènes dépendant des directions de vue qui se lisent sur les variations verticales. Ces dernières peuvent aussi être dues à la dépendance par rapport à la vue des propriétés de réflectance des points de la méso-structure concernés par une ABRDF. En effet, le principal inconvénient de la parallaxe dans les ABRDFs est le fait que plusieurs points de la surface de la méso-structure avec leurs différentes propriétés entrent en jeu pour une même ABRDF pour un texel donné. En d'autres termes, selon la direction de vue, un même texel d'une BTF fait référence à différents points de la méso-structure sous-jacente. La preuve en est que la Figure 4.7-(d)-1 qui représente une ABRDF collectée là où la méso-structure est localement plane, montre que sans parallaxe les variations verticales sont moins complexes et plus constantes.

4.3.2 Vue en 3D

Notre logiciel permet de visualiser en trois dimensions des tranches d'une ABRDF sous forme de lobes pour une direction de vue ou de lumière fixée par l'utilisateur. Un lobe représente donc des données selon deux dimensions. Cette visualisation permet par exemple de comparer la forme des lobes présents dans une ABRDF avec la forme caractéristique des lobes pour les différents modèles analytiques existants. La comparaison des formes permet de constater visuellement les limites des modèles analytiques au niveau de leurs degrés de liberté pour représenter par exemple une chute d'intensité correspondant à une ombre.



(a)



(b)

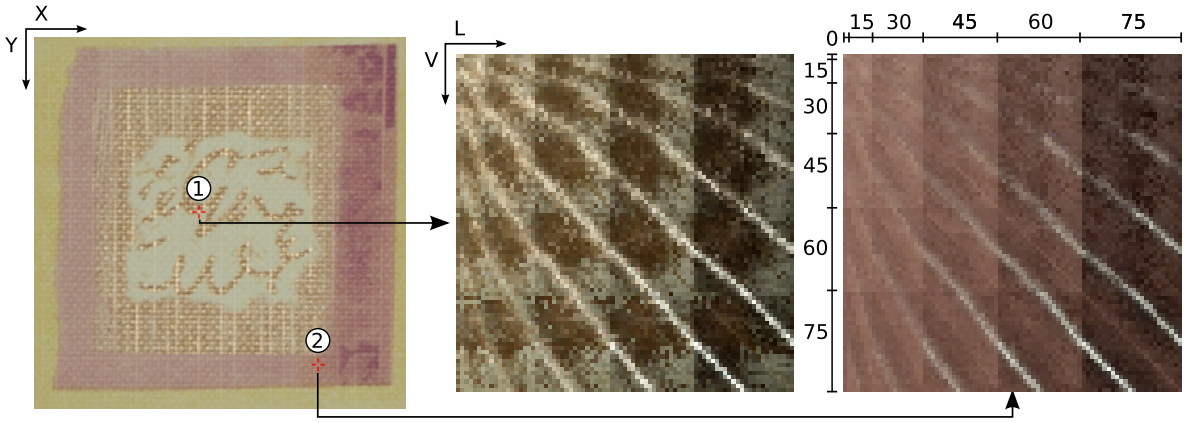
FIG. 4.6: *Sur une ligne, une image d'une BTF et deux images d'ABRDFs pour différents texels. En haut, une BTF issue d'un morceau de granite gravé et en bas, une BTF d'un échantillon de tissage de fibre polyacrylique marron. Les axes, intitulés V et L, dans la deuxième colonne, sont respectivement les directions de vue et les directions lumineuse. Dans la troisième colonne, les échelles indiquent les valeurs fixes pour l'angle ϕ, tandis que l'angle θ varie sur 360 degrés dans chaque intervalle.*

4.3.2.1 Construction

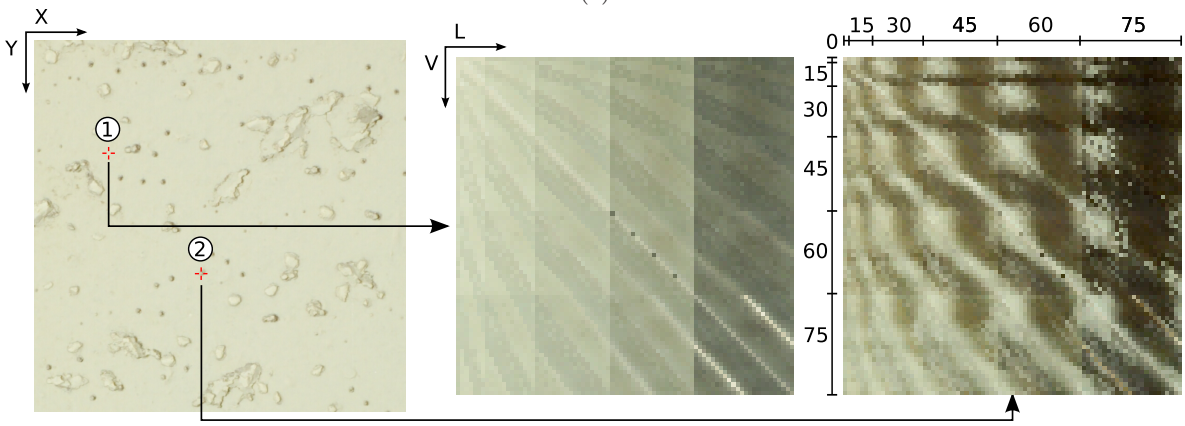
Une tranche d'une ABRDF est représentée par un lobe pour chaque composante RVB. Un lobe est modélisé par un maillage dont les sommets sont déterminés en pondérant les directions unitaires par les valeurs de luminance correspondantes. Pour construire le maillage, les échantillons des directions sur la surface d'un hémisphère sont projetés sur le disque formant la base et une triangulation de Delaunay 2D est appliquée. Si l'échantillonnage des directions est constant, le maillage est aussi constant. Seuls les sommets nécessitent d'être recalculés lorsque les tranches à visualiser changent.

4.3.2.2 Exemple d'utilisation avec les textures polynomiales (PTM)

Comme le montre la Figure 4.8, cette représentation 3D des lobes permet d'observer les pics, les chutes d'intensité pour les ombres propres et de manière générale la variabilité des données d'une ABRDF. Cela permet de mieux se rendre compte des différents degrés de liberté nécessaires pour les méthodes de compression utilisant des modèles analytiques. Par exemple, cette visualisation permet



(c)



(d)

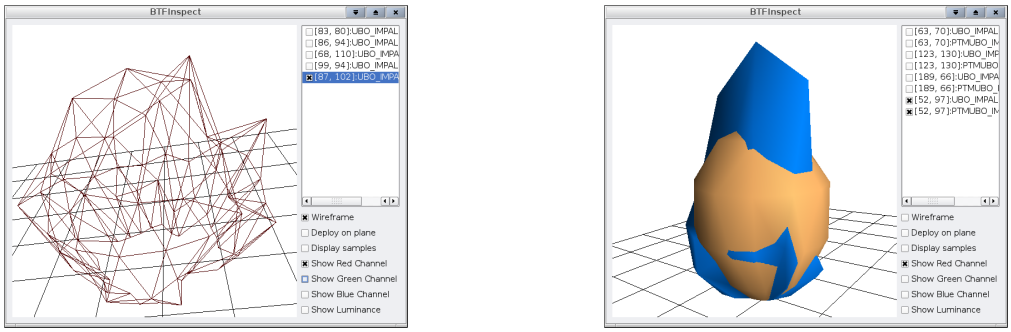
FIG. 4.7: *En haut, une BTF issue d'un échantillon de papier peint et en bas, d'un morceau de béton aggloméré. Les axes, intitulés V et L, dans la deuxième colonne, sont respectivement les directions de vue et les directions lumineuse. Dans la troisième colonne, les échelles indiquent les valeurs fixes pour l'angle ϕ, tandis que l'angle θ varie sur 360 degrés dans chaque intervalle.*

de comparer les lobes des données avec les lobes des modèles analytiques afin de localiser les forces et les faiblesses des différents modèles. La Figure 4.8-(b) montre la superposition d'un lobe par une ABRDF pour une vue fixée et une approximation obtenue avec un modèle polynomiale. Ce modèle reprend le polynôme biquadratique utilisé pour les PTMs [MGW01] :

$$L(u,v;l_u,l_v) = a_0(u,v)l_u^2 + a_1(u,v)l_v^2 + a_2(u,v)l_u l_v + a_3(u,v)l_u + a_4(u,v)l_v + a_5(u,v) \tag{4.5}$$

où (l_u, l_v) sont les projections des vecteurs de lumière normalisés dans le système local de coordonnées de texture (u, v) et l est la luminance résultante pour la surface en cette position. Nous avons utilisé un polynôme par composante RVB et les coefficients du polynôme sont minimisés au sens des moindres carrés avec la contrainte suivante :

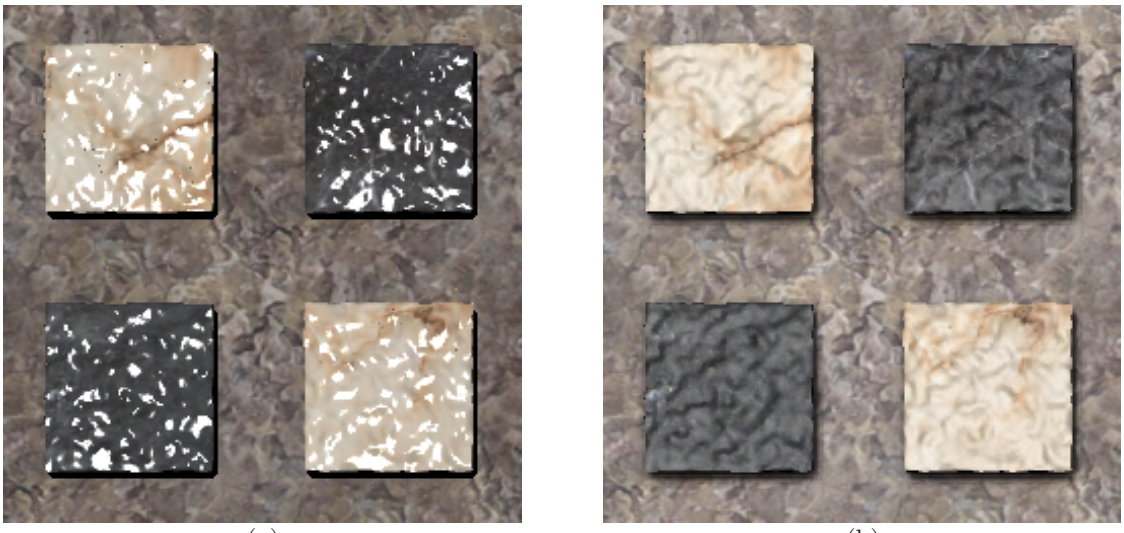
$$\begin{bmatrix} l_{u0}^2 & l_{v0}^2 & l_{u0}l_{v0} & l_{u0} & l_{v0} & 1 \\ l_{u1}^2 & l_{v1}^2 & l_{u1}l_{v1} & l_{u1} & l_{v1} & 1 \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ l_{uN}^2 & l_{vN}^2 & l_{uN}l_{vN} & l_{uN} & l_{vN} & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ . \\ . \\ . \\ a_5 \end{bmatrix} = \begin{bmatrix} L_0 \\ L_1 \\ . \\ . \\ . \\ L_N \end{bmatrix} \tag{4.6}$$



(a) (b)

FIG. 4.8: *(a) La composante rouge pour une tranche d'une ABRDF au pôle (vue : $\phi = 0, \theta = 0$) extraite de la BTF de granite Impalla. On remarque plusieurs pics pour le texel sélectionné. (b) La superposition d'un lobe obtenu pour un texel différent (en bleu) et d'un lobe obtenu représentation l'approximation polynomiale par un PTM (en orange).*

On peut voir clairement dans la superposition des lobes que le polynôme représente mal le phénomène de haute fréquence correspondant à un pic spéculaire et agit comme un filtre basse fréquence sur l'ensemble des données. La Figure 4.9 illustre le fait que les degrés de liberté d'un tel polynôme ne suffisent pas à préserver la qualité de l'apparence dans ce cas.



(a) (b)

FIG. 4.9: *(a) Une image de la BTF synthétique Isba présentant des fortes spécularités pour la vue $[\theta = 30, \phi = 60]$ et la direction lumineuse $[\theta = 0, \phi = 0]$ (b) L'image reconstruite après compression de la BTF par la méthode des PTMs, les spécularités ne sont pas représentées et les couleurs dans l'ensemble ont été atténuées.*

Bilan de notre étude

Dans les chapitres précédents, nous avons vu que les BTFs suscitent un vif intérêt pour la richesse et le réalisme des matériaux qu'elles apportent dans des applications de rendu en temps-réel ou interactif. Leur obtention se fait au prix de dispositifs d'acquisition qui peuvent se révéler coûteux et pour un temps de capture pouvant être assez long. Bien que l'utilisation des BTFs décharge les applications de calculs longs et complexes pour simuler une apparence réaliste, elles demandent dans leur état brut un espace mémoire de stockage conséquent. Un des axes de recherche sur les BTFs est donc consacré à leur compression afin de pouvoir constituer des bibliothèques de matériaux de taille raisonnable et de démocratiser leur utilisation autant que les texture classiques sur cartes graphiques.

Dans le Chapitre 3, nous avons présenté les différentes approches existantes pour compresser les BTFs de manière adaptée aux capacités des cartes graphiques. Dans toutes ces approches, une convergence semble se faire sur la notion d'ABRDF, et donc vers une compression par texel et par point de vue de la réponse lumineuse.

Les ABRDFs devenant les entités à compresser, nous avons donc effectué une étude visuelle des variations complexes qu'elles renferment et des différents phénomènes impliqués. Parmi l'éclairement direct, l'éclairement indirect, les ombres propres et la visibilité de la méso-structure, c'est ce dernier phénomène, que l'on appelle aussi simplement l'effet de parallaxe, qui induit en toute logique les plus importantes variations dans les ABRDFs. L'explication se résume au fait qu'un même texel d'une BTF fait référence à de multiples positions sur la méso-structure et c'est ainsi autant de BRDFs différentes qui influencent une même ABRDF.

Il nous semble alors évident qu'en l'absence du phénomène de parallaxe, qui certes, est inhérent à la définition même des BTFs, les ABRDFs par texel seraient beaucoup plus simples et cohérentes à compresser avec des variations ne dépendant que d'une seule position sur la méso-structure. Dans la suite de ce mémoire, nous proposons une nouvelle piste qui a guidée le développement d'une nouvelle représentation des BTFs, permettant de décorréler les effets de parallaxes des autres effets. Nous l'avons nommées Flat-BTF.

Deuxième partie

Nouvelle représentation des BTFs : décorrélation de la parallaxe

Chapitre 5

Une nouvelle représentation des BTFs : les Flat-BTFs

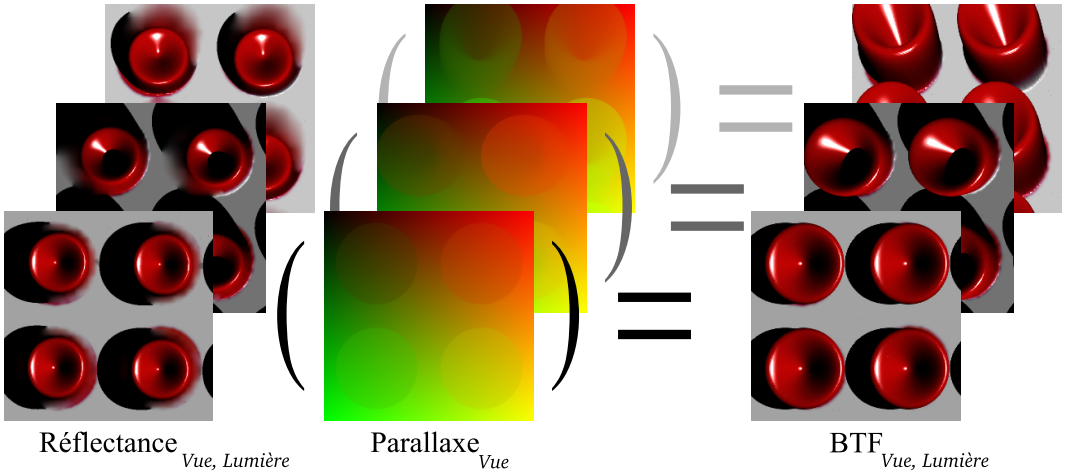


Fig. 5.1: *À gauche notre représentation Flat-BTF. Elle est composée d'une donnée de réflectance 6D et d'une fonction ne stockant que les effets de parallaxe et qui est par conséquent 4D. La composition de ces deux fonctions permet de reconstituer une BTF (à droite).*

Dans ce chapitre, nous introduisons une nouvelle représentation pour les BTFs que nous avons appelée Flat-BTF pour « Flat Bidirectionnal Texture Function » ou BTF aplatie. La principale particularité des Flat-BTFs est de séparer les effets de la parallaxe des autres effets contenus dans les BTFs à l'aide de deux fonctions distinctes (cf. Figure 5.1). Grâce à l'absence de la parallaxe, les effets lumineux en un même texel sont ceux de la même position de la méso-structure et l'ABRDF correspondante ne dépend donc plus que d'une seule BRDF qui va réfléchir l'énergie incidente, directe avec les ombres ou indirecte avec les inter-réflexions. Les données qui constituent les ABRDFs par texel sont alors beaucoup plus cohérentes et sont par conséquent plus facile à compresser.

Cette nouvelle représentation des BTFs a été développée pour essayer de réduire le problème de la parallaxe dans la compression des ABRDFs. Pour résoudre ce problème, nous nous sommes intéressés dans un premier temps aux méthodes permettant d'isoler et d'extraire les effets de parallaxe des BTFs. Le principe de ces méthodes consiste à retrouver la méso-structure des BTFs afin d'en reproduire la visibilité par rapport aux directions de vue. Après avoir isolée l'information de parallaxe, il faut pouvoir la représenter et la stocker dans une structure adaptée. Dans un deuxième temps, nous avons donc recherché une solution pour représenter les effets de la parallaxe une fois ceux-ci extraits des BTFs. Ce problème revient à exprimer et déterminer la visibilité de la méso-structure par rapport aux différentes directions de vue et nous avons donc recherché des solutions dans les méthodes existantes proposant le rendu de méso-structures en temps-réel.

Avant de définir plus en détails la représentation Flat-BTF, nous commençons donc par exposer les différentes approches permettant la reconstruction de la méso-structure des BTFs ainsi que les

méthodes de rendu en temps-réel de la visibilité des méso-structures.

5.1 Reconstruction de la méso-structure des BTFs

En dehors de la méthode de Wang et al. [WTS+05], la méso-structure des matériaux n'est en général pas mesurée explicitement lors des acquisitions de BTFs. Par contre elle est implicitement présente dans les BTFs au travers des différents effets qu'elle induit. Comme notre représentation nécessite la connaissance d'une méso-structure, nous montrons à travers cette section, qu'il est possible d'en extraire une approximation à partir des images d'une BTF.

Il existe deux approches différentes pour reconstruire cette méso-structure. Une première catégorie d'approche consiste à profiter des multiples points de vue des BTFs pour appliquer des méthodes de stéréovision tandis qu'une seconde considère les effets d'éclairement et des hypothèses sur les propriétés de réflectance de la surface pour en estimer la normale puis la géométrie.

5.1.1 Reconstruction par stéréovision

La reconstruction par stéréovision se découpe en 2 étapes. Une première étape consiste à trouver des correspondances entre les texels de deux images dépendant de directions de vue différentes. Les correspondances renvoient à des positions de la méso-structure que l'on retrouve dans les deux images. Une deuxième étape consiste alors à déduire des gradients à partir des déplacements relatifs entre les texels en correspondance. À partir de ces gradients et vecteurs de déplacement, il est possible d'estimer un champ de hauteurs.

C'est exactement avec cette approche que Neubeck *et al.* [NZG05] reconstruisent des fonctions de hauteur pour des BTFs mesurées à l'aide de leur dispositif d'acquisition. Les points de correspondance sont déduits à partir de différentes images de statistiques selon les propriétés des matériaux mesurés. Pour une direction de vue donnée, une image de statistique est calculée avec la totalité des images de l'ensemble des directions lumineuses. Ce procédé est similaire à notre estimation de l'éclairement ambiant décrit dans la section 4.2.1. Les différentes images de statistiques utilisées par Neubeck *et al.* sont par exemple l'image moyenne, l'image médiane ou encore une estimation de la composante spéculaire. En gardant toujours la vue zénithale en référence, Neubeck *et al.* comparent les déplacements relatifs avec chaque direction de vue restante et reconstruisent des fonctions de hauteur pour chaque vue. Le processus de minimisation prend en compte les occultations et une fonction de hauteur finale est déterminée en intégrant les différentes fonctions de hauteur qui ont été calculées pour les différentes paires de vue considérées.

Ces travaux produisent une approximation des méso-structures de bonne qualité pour des BTFs présentant de hautes résolutions angulaires. Les auteurs utilisent leur propre dispositif d'acquisition permettant de mesurer 264 points de vue différents pour 169 directions lumineuses. Les BTFs dont la communauté dispose généralement ne présentent pas de telles résolutions. Par ailleurs, cette méthode sous-entend qu'une fonction de hauteur suffit à représenter la méso-structure sous-jacente d'un matériau.

5.1.2 Reconstruction par photométrie

La reconstruction par photométrie ou « *Shape from shading* » (par exemple [RTG97]) ou encore « *Shape from shadows* » (par exemple [KS92]) consiste à retrouver les attributs géométriques d'une surface à partir de son apparence et d'hypothèses posées sur ses propriétés de réflectance. Par exemple, en utilisant le fait que les modèles analytiques de BRDF comprennent en général des paramètres liés à la géométrie comme la normale à la surface, on peut retrouver la normale en un point d'une surface en minimisant les paramètres d'un modèle analytique par rapport à ses données d'apparence. La principale difficulté consiste alors à trouver le modèle de BRDF en adéquation avec les propriétés de réflectance de la surface. Müller *et al.* [MSK07] ont reconstruit des cartes de normales pour des BTFs à l'aide de la méthode de Rushmeier *et al.* [RTG97] qui suppose que les surfaces à reconstruire sont lambertiennes. La méso-structure représentée par une fonction de hauteur est alors déterminée en intégrant les cartes

de normales selon la méthode de Frankot et Chellappa [FC88]. Avec le même principe, toujours pour des surfaces aux propriétés lambertiennes, Liu *et al.* [LYS01] reconstruisent directement une fonction de hauteur par un processus de minimisation.

Pour des surfaces aux propriétés spéculaires, on cherche en général les directions principales de réflexion de la lumière afin d'en déduire les normales au niveau de la géométrie. Les méthodes existantes commencent par isoler les spécularités dans les images et les normales sont déduites en fonction des configurations de direction de vue et de lumières pour lesquelles les réflexions spéculaires sont maximales. Selon ce principe, grâce à un dispositif d'acquisition présenté dans le paragraphe 2.1.2 et permettant un échantillonnage angulaire très dense, Wang et Dana [WD06] ont reconstruit des cartes de normales pour des matériaux très spéculaires. A l'opposé, Chen *et al.* [CGS06] utilisent un procédé d'acquisition ad-hoc et manuel pour reconstruire les méso-structures pour des matériaux présentant une grande variétés de spécularités, plus ou moins fortes, mais aussi pour des matériaux translucides.

Toutes ces méthodes ne considèrent pas les ombres propres présentes dans les BTFs alors que d'autres travaux en font exclusivement l'usage pour reconstruire la géométrie : en effet, elles représentent la visibilité de la source lumineuse par rapport au relief de la méso-structure. La principale difficulté de ces méthodes réside alors dans la détection des ombres dans les images et plus particulièrement lorsque les matériaux sont sombres. Yu et Chang [YC05] représentent à l'aide d'un graphe toutes les contraintes d'ombres d'une vue donnée et utilisent un processus de minimisation contraint pour rendre consistant les élévations déduites des ombres avec celles d'une reconstruction par éclairement. La méthode est robuste pour les spécularités mais la détection des ombres dans les images de la base CUReT [CUR] nécessite une intervention manuelle. De même, Kautz *et al.* [KBD07] ont utilisé la méthode de Daum et Dudek [DD98] après avoir sélectionné manuellement les ombres dans des images de la base de Bonn [Bon]. Cependant, les fonctions de hauteur reconstruites restent des approximations qui ne représentent pas parfaitement la méso-structure des matériaux.

5.1.3 Mesure de la méso-structure

Wang *et al.* [WTS+05] ont utilisé un laser pour mesurer la géométrie des matériaux sous la forme de plusieurs fonctions de hauteur associées chacune à différentes directions de vue. La méso-structure est utilisée sous cette forme pour compléter un rendu de BTFs par le rendu des silhouettes ainsi que de leurs ombres portées. L'utilisation d'un laser peut poser des problèmes pour l'acquisition de matériaux très spéculaires ou translucides. Récemment, Francken *et al.* [FCM+08] ont proposé un dispositif d'acquisition abordable et efficace pour acquérir des méso-structure sous la forme de cartes de normales à partir des spécularités de la surface. La méthode reste limité aux fonctions de hauteur et ne permet pas la mesure de BTFs mais elle montre qu'une solution d'acquisition facilement réalisable avec une caméra et un moniteur LCD est possible.

5.1.4 Discussions

Malgré certaines limitations, il est possible d'extraire les méso-géométries sous-jacentes des BTFs. À partir des images d'une BTF, les méthodes par stéréovision ou par photométrie permettent d'en reconstruire une approximation. Plus la densité des échantillons angulaires est importante et plus ces méthodes convergent vers un résultat satisfaisant. Malheureusement, les BTFs des bases de données en ligne dont nous disposons ne présentent pas un échantillonnage angulaire suffisant pour obtenir des méso-structures d'une qualité satisfaisante pour permettre une extraction de la parallaxe. Par exemple, nous avons testé le logiciel *BTFShop* de Kautz *et al.* [KBD07] qui requiert une sélection manuelle des zones d'ombres avant l'étape de reconstruction et pour plusieurs de nos BTFs, les fonctions de hauteur reconstruites apparaissent nettement insuffisantes en terme de qualité et de précision.

Un moyen simple de disposer d'une méso-structure précise est de la mesurer directement à l'aide d'un dispositif d'acquisition adapté qui mesure en même temps les images d'une BTF. La mise en œuvre d'un tel dispositif est complexe et surtout trop coûteuse pour le cadre de notre étude. De plus, les mesures effectuées par Wang *et al.* [WTS+05] ne sont malheureusement pas disponibles au public. En conclusion, la seule solution en mesure de garantir une méso-structure exacte est la génération de

BTFs où tous les paramètres, dont la méso-structure, sont connus. Afin de valider notre approche dans un contexte contrôlé, nous avons donc principalement basé nos études sur des BTFs synthétiques.

5.2 Rendu de méso-structures

Il existe de nombreuses méthodes pour le rendu en temps-réel de ces méso-structures avec des effets d'éclairement et des effets de parallaxe. Comme nous l'avons vu précédemment (*cf.* Section 1.4), les premières méthodes telles que le *bump mapping* [Bli77, Coo84, Mam89, HDKS00] supposent des fonctions de hauteur pour la méso-structure et permettent uniquement le rendu des effets d'éclairement sans les effets de parallaxe. Le *parallaxe mapping* [KTI+01] permet d'y ajouter les effets de déformation dus à la parallaxe pour une fonction de hauteur en temps-réel mais ne prend pas en compte les occultations propres ce qui limite la méthode aux méso-structures de faible élévation. De son côté, le *relief mapping* [OBM00] permet le rendu en temps-réel des silhouettes, de l'occultation propre et des ombres propres toujours pour des fonctions de hauteur. Cette méthode utilise un lancer de rayon optimisé pour calculer les intersections avec une fonction de hauteur. Pour accélérer le rendu, Baboud et Décoret [BD06] proposent d'utiliser une recherche binaire combinée avec des pas d'incrémentations pré-calculés pour trouver les intersections plus rapidement. Policarpo *et al.* [PO06] étendent la méthode pour des géométries plus générales, en utilisant plusieurs couches de texture. Récemment, pour améliorer la qualité de la texturation, McGuire *et al.* [MW08] propose une paramétrisation globale de la géométrie représentant la méso-structure.

Dans une approche basée sur un pré-calcul de la visibilité, Wang *et al.* [WWT+03] pré-calculent l'effet de parallaxe et l'encodent dans une texture dépendante du point de vue et de la courbure de la surface de référence pour le placage. Cette méthode implique de très longs pré-calculs et requiert une compression efficace pour réduire la taille de la texture résultante à cinq dimensions (deux pour la direction et une pour la courbure isotrope). Même si les ombres, les silhouettes et l'éclairement locale sont pris en compte, la qualité du rendu est inférieure aux BTFs et la définition de la méso-structure est limitée aux fonctions de hauteur.

Bien que toutes ces méthodes représentent une alternative aux BTFs, aucune ne permet d'atteindre leur réalisme pour le rendu car les effets d'éclairement propre, qui apportent une contribution forte au réalisme du rendu, sont encore trop complexes à évaluer en temps-réel. Wang *et al.* [WTL+04] améliorent leur méthode [WWT+03] avec une approche volumique de la méso-structure afin de modéliser une géométrie plus générale et avec un rendu utilisant des BTFs pour obtenir un éclairement plus réaliste. La méthode rajoute le rendu des silhouettes, manquant aux BTFs, ainsi que leurs ombres portées dans la scène. Cependant, afin d'obtenir un rendu interactif, les BTFs ont été fortement sous-échantillonnées ce qui en réduit la qualité.

Discussion : Les méthodes les plus abouties pour représenter l'ensemble des effets produit par la méso-structure sont les méthodes du type *relief mapping* [OBM00, BD06, PO06, MW08] et les méthodes de Wang *et al.* [WWT+03, WTL+04] qui encodent la visibilité pré-calculée dans une texture dépendante, entre autres, de la direction de vue. Les méthodes de type *relief mapping* misent plus sur la puissance de calcul des cartes graphiques actuelles pour la détermination en temps-réel de la visibilité de la méso-structure par un procédé de lancer de rayon. Une des faiblesse de cette méthode est le nombre fixé d'itérations pour la recherche des intersections car rien ne garantie que la recherche converge dans le nombre imparti. Un autre inconvénient par rapport aux BTFs est que le rendu est beaucoup moins réaliste. Cependant, la proposition de McGuire *et al.* [MW08] est une bonne piste pour combiner *relief mapping* et BTFs.

À l'opposé, Wang *et al.* [WWT+03, WTL+04] misent sur l'espace mémoire en pré-calculant la visibilité de la méso-structure pour différents échantillons de direction de vue. La taille des données est réduite par compression pour être stockée sur carte graphique et la visibilité est reconstruire par décompression et l'interpolation des vues clefs. Cette méthode est très proche des BTFs dans la mesure où la visibilité est pré-calculée pour un certain nombre d'échantillons et reconstruite ensuite par interpolation au moment du rendu. D'ailleurs Wang *et al.* [WTL+04] le prouvent en combinant

facilement leur méthode avec des BTFs.

L'expérience de ces techniques montre que des structures adéquates permettent de rendre efficacement les effets de parallaxe. Une fois la parallaxe extraite des BTFs, ce qui conduirait à une meilleure cohérence pour de la compression, nous développerons une structure permettant de la réintroduire au moment du rendu.

5.3 Définition de la représentation Flat-BTF

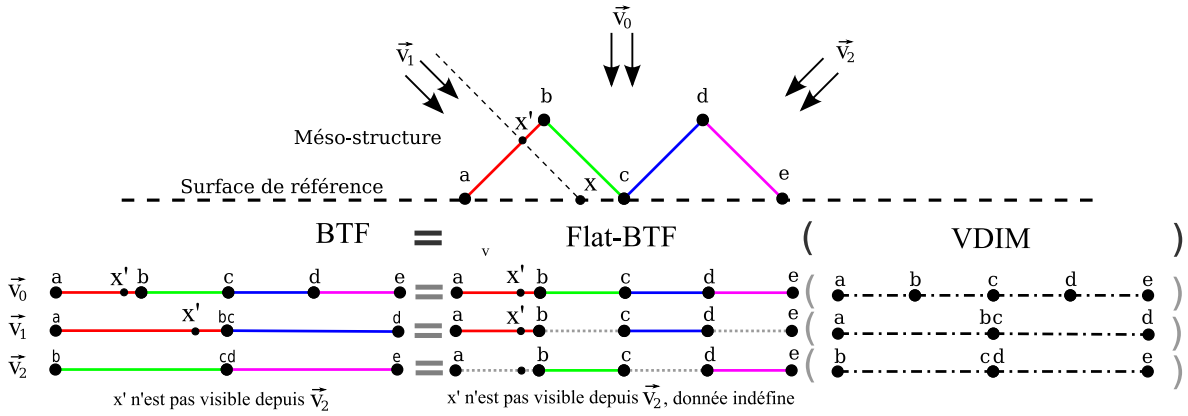


FIG. 5.2: *Dans une BTF standard, l'apparence due à la méso-structure est calculée ou mesurée indépendamment pour une ensemble de direction de vue $\vec{v}_0$, $\vec{v}_1$ et $\vec{v}_2$. La réflectance visible est stockée pour chaque vue mais ces données sont déformées et étirées pour que l'image corresponde à la surface plane de référence (le segment* **ab** *est beaucoup plus long dans la vue $\vec{v}_1$ que dans la vue $\vec{v}_0$). A l'opposé, dans la représentation Flat-BTF, les échantillons sont stockés en fonction de leur localisation sur la méso-structure. Une carte d'indirection dépendante de la direction de vue et renfermant l'information de parallaxe permet alors de convertir une Flat-BTF dans la représentation standard de BTF.*

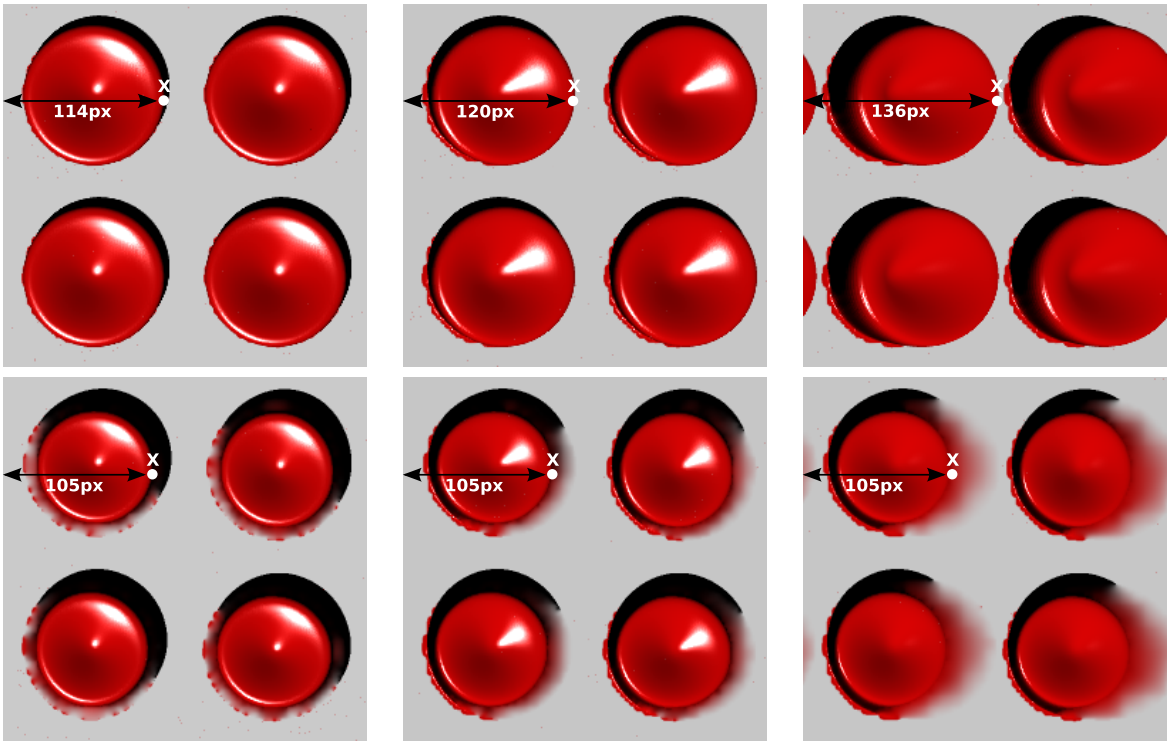
La représentation Flat-BTF sépare les effets d'éclairement contenus dans les BTFs et les effets de parallaxe dépendant de la direction de vue. Une BTF exprimée dans la représentation Flat-BTF est alors composée de deux fonctions distinctes :

- une fonction à 6 dimensions représentée par des textures de réflectance dépendantes d'une direction lumineuse et d'une direction de vue. Dans la suite de ce mémoire, nous appelons ces données, une Flat-BTF ;
- une fonction à 4 dimensions représentées par des cartes d'indirections dépendantes d'une direction de vue. Dans la suite de ce mémoire, nous appelons cette fonction d'indirection, VDIM (pour *View-Dependent Indirection Map*).

Comme on peut le voir sur le Schéma 5.2, le principe de séparation des effets réside dans le fait d'encoder une BTF selon sa méso-structure et non plus selon une surface de référence plane comme il est d'usage dans la représentation standard. Le nom Flat-BTF fait référence à l'effet d'aplatissement[1] de la méso-structure dans l'espace image correspondant pour la Flat-BTF. Cet aplatissement est obtenu par une paramétrisation globale de la méso-structure permettant d'assurer une bijection entre la surface géométrique et l'espace bidimensionnel d'une image. Ce changement de support garantit que chaque texel de la Flat-BTF correspond bien à une unique et même position sur la méso-structure et ce, pour tous les points de vue. En conséquence, tous les effets dus à la parallaxe n'apparaissent plus dans les images et la cohérence des données par texel en fonction des directions de vue est grandement améliorée. Comme le montre la figure 5.3, les variations en fonction des directions de vue ne produisent pas de décalage dans les images. On peut assimiler une Flat-BTF à une SVBRDF à laquelle on aurait rajouté des ombres propres et des phénomènes d'éclairement indirect par texel.

Les effets de parallaxe sont toujours encodés selon une surface de référence plane qui est censée représenter la surface de plaquage des objets. Ces effets qui représentent la visibilité de la méso-structure par rapport à un point de vue sont encodés sous la forme d'indirections par la fonction

[1] Traduction de « flat » : adj, aplati/-e.



$\theta = 0$ $\theta = 30$ $\theta = 60$

FIG. 5.3: *En haut, les images d'une de nos BTFs synthétiques, appelée « bouton », illustrant l'effet de parallaxe de la représentation standard. En bas, les images de la même BTF, dans la représentation Flat-BTF, on constate l'absence de parallaxe. Les zones floues dans les images correspondent aux parties non visibles de la méso-structure qui ont été remplies en accord avec le voisinage.*

VDIM pour accéder aux données de la Flat-BTF. La composée d'une Flat-BTF par la fonction VDIM associée conduit à une BTF dans la représentation standard (*cf.* figure 5.2) :

$$BTF_{\vec{v},\vec{l}}(i,j) = Flat - BTF_{\vec{v},\vec{l}}(VDIM_{\vec{v}}(i,j))\,, \tag{5.1}$$

où $\vec{v}$ est la direction de vue, $\vec{l}$ la direction lumineuse et (i,j) la position d'un texel. Notons que même si une Flat-BTF est exempte de parallaxe, elle reste une fonction lumineuse dépendante de la direction de vue. La fonction VDIM est assez similaire à la méthode « view-displacement mapping » [WWT+03]. La principale différence est qu'à la place de la distance, nous encodons l'indirection sur la méso-structure qui renvoie aux données contenues dans la fonction Flat-BTF.

5.4 Conversion d'une BTF en Flat-BTF

La conversion d'une BTF mesurée dans la représentation Flat-BTF serait en théorie très simple à réaliser si l'on disposait d'une représentation géométrique de sa méso-structure. En effet, cette méso-structure permettrait de générer la fonction VDIM qui renferme les cartes d'indirection correspondantes aux différentes vues de la BTF. Une fois la fonction VDIM construite pour la BTF, les données de la Flat-BTF seraient déterminées simplement par la mise en correspondance avec les texels de la BTF grâce à la fonction VDIM comme le montre l'Algorithme 1 suivant.

```
Pour chaque direction de vue v
   Calculer la fonction VDIM(v) avec la méso-structure
   Pour chaque texel i,j
      Définir i',j' avec VDIM(v,i,j)
      Pour chaque direction de lumière l
         FBTF(v,l,i',j') = BTF(v,l,i,j)
```

Algorithme 1 : Principe de conversion depuis la représentation BTF vers la représentation Flat-BTF.

Dans la pratique, cette conversion supposerait la connaissance d'une méso-structure qui soit assez précise pour reproduire à l'identique, dans la fonction VDIM, les effets de la parallaxe pour la BTF à convertir. Une géométrie trop imprécise donnerait lieu à des erreurs de conversion dans la correspondance des texels avec une erreur maximale au niveau des texels présentant les plus grandes variations de position relative (*cf.* Section 4.2.2). Une attention particulière devrait aussi être portée au moment du remplissage des texels de la Flat-BTF avec ceux de la BTF. L'étude des dispositifs d'acquisition nous montre que plus les angles de vue sont rasants et plus la zone de méso-structure représentée par texel est grande. Cependant, la densité de texels représentatifs est plus faible dans les images avant le détourage et la correction de perspective (*cf.* Section 2.1.3). Une solution consisterait à mettre en place des méthodes de filtrage avec des noyaux de convolution dont la taille varierait en fonction de l'angle de vue. Les texels qui resteraient vides à la fin du processus de conversion sont les texels indéfinis qui représentent les texels non visibles depuis la direction de vue dont ils dépendent.

5.5 Discussion

La représentation Flat-BTF apporte une solution simple aux variations induites par l'effet de la parallaxe dans les ABRDFs d'une BTF. Notre solution consiste à représenter séparément l'information de parallaxe et l'information de réflectance par deux fonctions distinctes : la fonction VDIM et la fonction Flat-BTF. La représentation Flat-BTF devrait apporter un cadre plus favorable à la compression des ABRDFs que la représentation standard et conduit à de nouvelles approches de la compression des données.

La représentation Flat-BTF repose sur l'hypothèse de la connaissance de la méso-structure des BTFs. Il est difficile à l'heure actuelle de reconstruire précisément la méso-structure des BTFs mesurées par les approches existantes. Pourtant, la modélisation de la méso-structure apporte des avantages visuels non négligeables comme par exemple le rendu des silhouettes correspondantes pour les BTFs. D'ailleurs un des derniers dispositifs d'acquisition de BTFs [WTS+05] permet aussi l'acquisition de la méso-structure. On peut donc supposer qu'à l'avenir la définition d'une BTF sera enrichie par la définition exacte de sa méso-structure. Une deuxième hypothèse concerne la possibilité d'une paramétrisation globale de la méso-structure afin de représenter les données selon la surface de la méso-structure dans les images. Cette étape constitue la pierre angulaire de notre représentation Flat-BTF.

Pour valider cette nouvelle représentation des BTFs, nous en proposons une première implémentation sur des données synthétiques dans le Chapitre 6. Les Flat-BTFs et les BTFs de

référence générées sont obtenues à l'aide de méso-structures s'inspirant des BTFs mesurées et dont la paramétrisation globale est aisée. Les résultats obtenus sont ensuite exposés et analysés dans le Chapitre 7.

Chapitre 6

Génération de Flat-BTFs

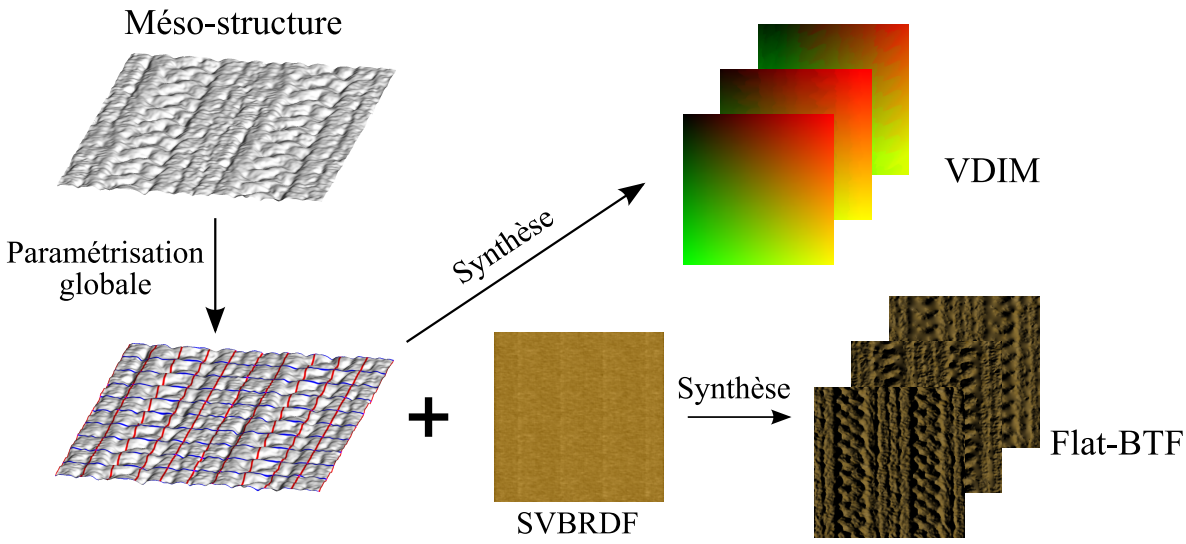
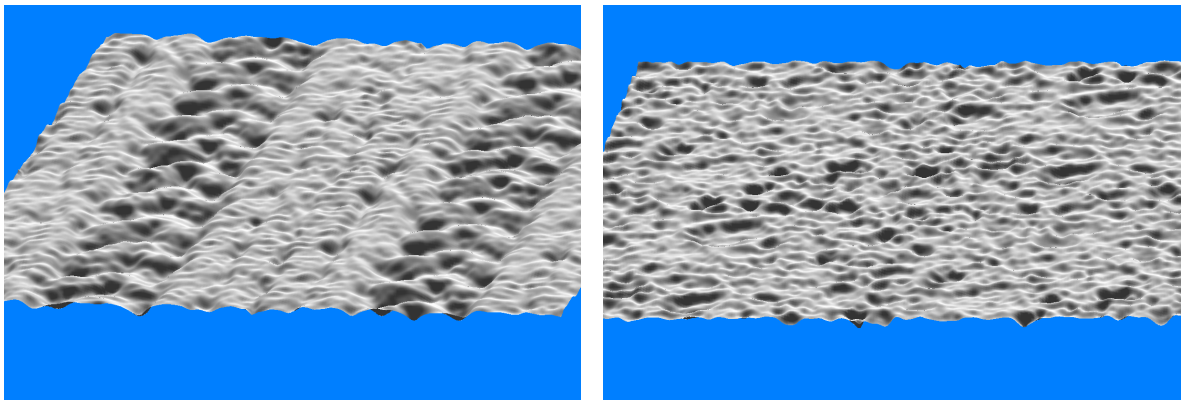


Fig. 6.1: *Les différentes étapes pour la génération de Flat-BTFs de synthèse.*

En plus de la représentation de l'apparence des matériaux telle qu'elle est fournie dans la définition standard des BTFs, la représentation Flat-BTF nécessite une connaissance sur la méso-structure. S'il est possible de retrouver une approximation de cette méso-structure comme montrée précédemment (*cf.* Section 5.1), afin de valider notre approche, nous allons tout d'abord nous placer dans un contexte complètement contrôlé. C'est pourquoi, dans un premier temps, nous avons implémenté la représentation Flat-BTF pour des données synthétiques. Ainsi, toutes les étapes de sa mise en œuvre, illustrées par la Figure 6.1, sont maîtrisées afin d'en contrôler la validité. Elles sont décrites en détail au fil des sections de ce chapitre. La première étape consiste à définir et modéliser la méso-structure. La paramétrisation globale de sa surface, indispensable pour obtenir la représentation Flat-BTFs, est ensuite déterminée. En définissant des BRDFs qui varient sur la méso-structure, les images de la fonction Flat-BTF peuvent alors être synthétisées tandis que les cartes d'indirection de la fonction VDIM sont générées dans un processus à part. Nous avons aussi synthétisé, les BTFs correspondantes dans la représentation standard afin de servir de référence comparative.

6.1 Modélisation de méso-structures

La modélisation d'une méso-structure est la première étape et donc la base de la génération de Flat-BTFs. Elle est utilisée à la fois pour la synthèse des Flat-BTFs et pour le calcul des fonctions d'indirections associées (VDIM). Par analogie aux surfaces mesurées lors de l'acquisition de BTFs, un modèle de méso-structure doit représenter la surface d'un échantillon carré et au niveau des proportions, et par expérience, la hauteur maximale de la surface ne doit pas excéder 10% de la largeur ou de la longueur. Si les variations de hauteur sont trop grandes, l'effet de parallaxe est plus important et le rendu des BTFs doit être complété par le rendu des silhouettes correspondantes pour pouvoir conserver un semblant de réalisme. Un dernier critère est la raccordabilité des textures générées et donc de la méso-structure. Plus particulièrement en rapport avec la représentation Flat-BTF, l'étape suivante consiste en la paramétrisation globale de la méso-structure : cela implique une contrainte forte sur la complexité géométrique des méso-structures. En effet, plus la surface présente une topologie complexe ou une grande variété et plus le processus de paramétrisation peut devenir compliqué. C'est pourquoi, dans un premier temps, nous nous sommes restreint à des méso-structures assimilables à une surface carrée, non fermée et sans trou. Ainsi, ces modèles restent facilement paramétrables.

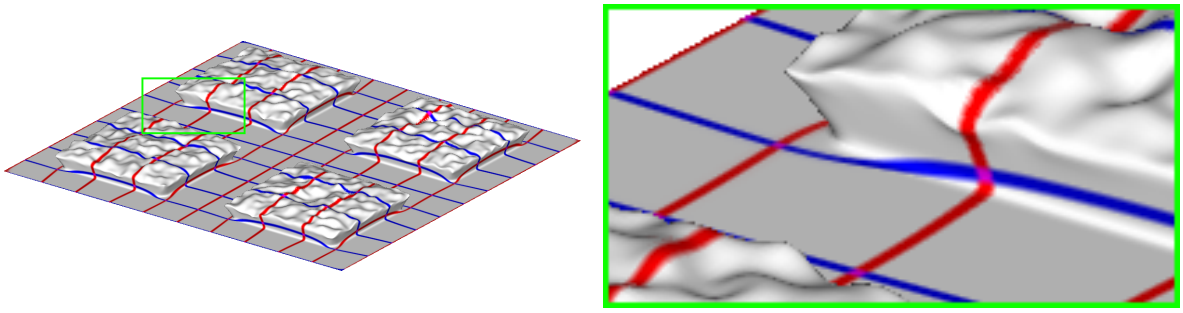


(a) (b)

FIG. 6.2: *Les méso-structures reconstruites à partir de l'occlusion ambiante utilisée comme fonction de hauteur pour une BTF de tissage de fibre polyacrylique en (a) (BTF Pully de l'université de Bonn) et pour une BTF d'une éponge en (b) (BTF Sponge de Madga et Kriegman).*

Nous avons modélisé plusieurs méso-structures aux formes variées à l'aide de maillages d'une résolution assez grande afin de garantir une définition précise de la surface lors du sur-échantillonnage qui peut être effectué lors des étapes de synthèse. Dans une première approche, nous avons exploité les images obtenues grâce à notre estimateur de l'occlusion ambiante pour une vue donnée (*cf.* Section 4.2.1). En retouchant les contrastes de l'image, obtenue en niveaux de gris à partir de l'estimation de l'occlusion ambiante pour la vue zénithale de BTFs mesurées, nous avons pu en déduire une fonction de hauteur afin de déformer une grille 2D définie par un maillage de haute résolution (512×512 sommets). Comme le montre la Figure 6.2, une fois les hauteurs ajustées manuellement, nous avons pu obtenir des méso-structures qui s'approchent de la méso-géométrie sous-jacente des BTFs mesurées. Dans une seconde approche, nous avons modélisé des méso-structures par des fonctions paramétriques qui permettent de créer des formes plus géométriques et régulières tel que par exemple, le modèle *Isba*. Comme le montre la Figure 6.3-(b), cette seconde approche permet de modéliser des méso-structures plus complexes que des celles obtenues par des fonctions de hauteur car la surface peut présenter des repliements qui ne peuvent pas être représentés dans les approches existantes.

6.2 Paramétrisation globale



(a) (b)

FIG. 6.3: *(a) La paramétrisation de la méso-structure du modèle Isba (b) Zoom montrant que la surface se replie sous la dalle. Grâce à la paramétrisation globale, notre représentation ne limite pas le type des méso-structures à des fonctions de hauteur comme la plupart des approches existantes (cf. Section 5.2).*

Après la modélisation de la méso-structure, l'étape suivante qui consiste en la paramétrisation globale de celle-ci, constitue l'élément clef de la représentation Flat-BTF. En effet, c'est grâce à cette paramétrisation que les données d'apparence peuvent être stockées dans les images selon la surface de la méso-structure correspondante (pour plus de détails, voir la section 5.3). En parallèle, la paramétrisation de la méso-structure est aussi déterminante pour générer la fonction VDIM qui comprend les cartes d'indirections dépendantes des directions de vue et encode ainsi les effets de parallaxe. Pour les surfaces définies par maillages, il existe de nombreuses méthodes de paramétrisation aux propriétés diverses : certaines méthodes sont plus robustes que d'autres face à la complexité en genre ou en variété des maillages.

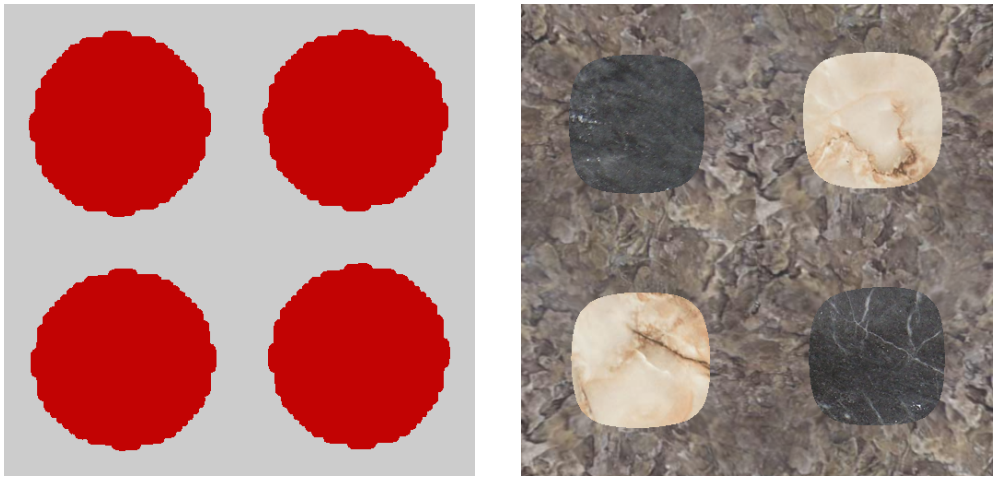
Dans le contexte de la représentation Flat-BTF, la paramétrisation idéale devrait permettre de représenter uniformément et de manière continue la totalité de la surface de la méso-structure dans l'espace image final. En effet, contrairement à la représentation classique des BTFs où les images ne représentent qu'une partie de la méso-structure (certaines parties pouvant être invisibles sous certains points de vue), les images de la fonction Flat-BTF représentent toujours, quelque soit la direction de vue, la totalité de la surface de la méso-structure même si certaines parties ne sont pas visibles.

Parmi les différentes méthodes de paramétrisation existantes, nous avons sélectionné celles [ZWLZ08, SLMBY05, LPRM02, RL03, Flo03] dont les propriétés de paramétrisation semblaient correspondre le plus à nos besoins en terme de d'uniformité sur la surface. Nous les avons toutes testées expérimentalement sur nos méso-structures. Or, seule la méthode intitulée « *Mean Value Coordinate* », introduite par Floater [Flo03], nous a permis d'obtenir des résultats exploitables. Le fait que la surface de nos méso-structures soit non-fermée semble perturber les autres méthodes. Le principe de la méthode de Floater repose sur la dérivée d'une généralisation des coordonnées barycentriques ce qui permet que chaque sommet dans une triangulation planaire puisse être exprimé comme une combinaison convexe des sommets voisins. Cette paramétrisation est robuste pour les surfaces non fermées et elle a la bonne propriété de conserver les aires des facettes pour les maillages. Ainsi la surface d'une méso-structure est représentée uniformément dans les images. En revanche, cette paramétrisation n'est pas régulière au niveau de la préservation des angles. Nous avons utilisé son implémentation dans le logiciel *Graphite* [Gra03] durant nos tests. La Figure 6.3 illustre le résultat de la paramétrisation globale par le placage d'une texture de grille régulière sur le maillage.

6.3 Texturation de la méso-structure et BRDF

Afin de constituer des échantillons virtuels de matériaux à part entière, les méso-structures modélisées doivent être munies de propriétés de réflexion. Nous avons défini ces propriétés par un modèle analytique de BRDF dont les paramètres peuvent facilement varier sur la surface géométrique à l'aide de textures. Parmi les différents modèles existants, nous avons choisi le modèle de Phong [LW94]

qui est le plus couramment utilisé en raison de la simplicité de sa mise en œuvre et de l'évaluation rapide du résultat. Un autre atout de ce modèle est que l'on peut obtenir de manière convaincante des propriétés de réflexion qui varient spatialement en faisant simplement varier, entre autres, la composante diffuse. Cette variation est généralement obtenue à l'aide de textures pour la plupart des applications. En revanche dans notre cas, l'utilisation de textures classiques n'est pas aisée car le domaine de la paramétrisation que nous utilisons n'est pas régulier et par conséquent, le placage direct de textures existantes avec cette paramétrisation entraine des déformations de celles-ci sur la surface. Afin de définir des variations diffuses uniformes sur la surface de la méso-structure, nous avons donc dû texturer nos méso-structures de manière procédurale. À cet effet, nous avons développé notre propre logiciel de texturation interactive et procédurale pour nos méso-structures.



(a) (b)

FIG. 6.4: *Exemple de textures diffuses obtenues de manière procédurale. En (a), pour le modèle Bouton et en (b), pour le modèle Isba qui représente un dallage en marbre sur un fond de granit.*

Lors de la phase de texturation procédurale, l'utilisateur a le choix entres différentes couleurs et différentes textures qui peuvent être attribuées en fonction des paramètres géométriques choisis tels que la position ou l'orientation de la normale d'un sommet. La texture diffuse de la Figure 6.4-(a) comprend deux couleurs qui ont été assignées selon la direction de la normale. La texture diffuse de la Figure 6.4-(b) a nécessité trois textures différentes. La composante z de la position 3D a été utilisée pour distinguer les dalles du sol. Les composantes x et y de la position 3D ont été utilisées à la fois pour les coordonnées de texture et pour la répartition des dalles noires ou blanches.

Une fois la texturation de la surface définie, le résultat est alors sauvegardé sous la forme d'une texture unique et conforme à la paramétrisation de la méso-structure en entrée. Ainsi cette texture peut être directement plaquée avec la paramétrisation actuelle de nos méso-structures et les propriétés de réflectance restent décorrélées de la définition de la géométrie. Lors de la texturation interactive, l'utilisateur visualise directement cette texture. L'espace image conforme à la paramétrisation est obtenu par rendu, ou plus exactement par la rastérisation du maillage modifié des méso-structures. Ce procédé est détaillé dans la section suivante 6.5.1 où le même procédé de génération de textures conformes à la paramétrisation globale de la méso-structure est utilisé.

6.4 Synthèse de BTFs classiques

La génération de plusieurs BTFs dans la représentation classique et leurs équivalents dans la représentation Flat-BTF nous permettra d'effectuer une étude comparative entre les deux

représentations (*cf.* Chapitre 7). Comme nous l'avons vu dans l'état de l'art (*cf.* Section 2.2), les BTFs synthétiques peuvent être obtenues par lancer de rayons. À la différence des BTFs mesurées, les données sont générées de manière à éviter le processus de détourage et de correction de la perspective. Le principe est d'échantillonner la méso-structure depuis la surface de référence qui représente la surface de placage (*cf.* la ligne pointillée sur la Figure 5.2). La surface de référence est alors échantillonnée régulièrement à une résolution identique aux images à générer. Pour chaque échantillon (texel), la valeur d'intensité lumineuse est évaluée pour chaque point de vue et chaque direction de source de lumière. Elle est évaluée à la position qui correspond à l'intersection la plus proche du point de vue, c'est à dire au point de la méso-structure visible pour ce point de vue. Pour chaque direction lumineuse, on teste la visibilité de la source lumineuse depuis la position de l'intersection sur la géométrie. Si le point est visible depuis la source lumineuse, l'apparence du texel est alors calculée avec les propriétés d'apparence associées. Afin de générer des textures répétitives, le modèle raccordable d'une méso-structure est répété plusieurs fois et mis bord à bord dans la scène. Il est aussi possible de sur-échantillonner la surface en lançant plusieurs rayons par texel afin de réduire les effets de crénelage.

6.5 Synthèse de la fonction Flat-BTF

Les données d'apparence des images de la fonction Flat-BTF sont synthétisées à partir de **la géométrie de la méso-structure**, de **sa paramétrisation** et de **ses propriétés d'apparence**. La synthèse de ces données, ne comprenant aucun effet de parallaxe, se déroule en 3 étapes. Une première étape permet de construire une « image géométrique » de la méso-structure. Cette étape qui peut être perçue comme un aplatissement de la méso-structure, permet d'associer à chaque texel de la fonction Flat-BTF une portion de méso-structure selon sa paramétrisation. L'étape suivante concerne la synthèse de l'apparence en chaque texel et enfin une dernière étape s'occupe de traiter les texels indéfinis dans les images générées, c'est à dire les texels correspondant aux parties de la méso-structure qui sont cachées depuis certains points de vue.

6.5.1 Images géométriques d'une méso-structure

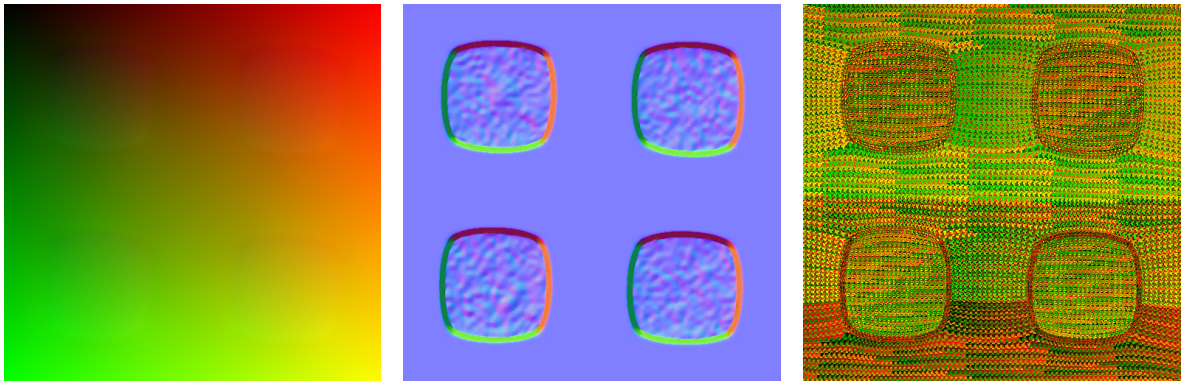
Lors de notre synthèse de BTFs dans la représentation classique, la visibilité de la méso-structure par les images a été déterminée par des rayons lancés depuis la surface de référence. Dans le cas de la fonction Flat-BTF, la problématique est différente car on veut représenter les données de réflectance selon la méso-structure et non plus selon la surface de référence pour le placage. En d'autres termes, on cherche à représenter uniformément la surface de la méso-structure par les texels des textures, chaque texel est alors associé à une partie de la méso-structure découpée uniformément. Par définition, un texel donné fait toujours référence à une même portion de méso-surface quelque soit la direction de vue. Cette représentation de la surface dans l'espace 2D discret d'une image est réalisable grâce à la paramétrisation globale de la méso-structure.

Une première approche consiste à garder la même que pour les BTFs classiques mais en rajoutant la prise en compte des coordonnées (u, v) de la paramétrisation aux points d'intersection des rayons avec la géométrie. Ces coordonnées (u, v) définissent une indirection vers notre texture Flat-BTF, dans laquelle nous cherchons à évaluer les valeur de la BTF. Avec cette approche, chaque texel de la fonction Flat-BTF correspond bien à une même position sur la méso-structure indépendamment de la direction de vue. Cependant, cette méthode ne fournit pas un échantillonnage uniforme de la méso-structure car la portion de surface intégrée par texel varie en fonction du cône de la direction de vue et conduit donc à des solutions bruitées.

La solution que nous avons adoptée consiste, dans une étape de pré-calcul, à construire une image géométrique de la méso-structure. La représentation de la méso-structure que nous construisons est très proche de celle proposée dans les travaux intitulés *Geometry Images* de Gu et al. [GGH02]. L'image géométrique est construite selon une discrétisation linéaire de l'espace de paramétrisation de la méso-structure et des informations sur la géométrie sont alors intégrées pour chaque texel. La surface de la méso-structure est représentée de manière uniforme, à condition que sa paramétrisation globale soit aussi uniforme. Pour les besoins de notre synthèse, nous avons donc créé trois images par méso-

structure où les texels encodent respectivement une position sur la surface, une normale à la surface et un indice de face du maillage.

Techniquement, une représentation en image de la géométrie est obtenue assez facilement par le rendu du maillage modifié de la méso-structure sur GPU. La modification consiste à conserver la topologie du maillage tout en remplaçant la position 3D de chaque sommet par ses coordonnées (u, v) de paramétrisation. Cette modification a pour effet d'aplatir le maillage et d'exprimer la géométrie 3D sous la forme d'une représentation planaire. La rastérisation de ce maillage en vue orthographique et en plein écran permet de construire directement l'image géométrique dans le tampon de rendu du GPU. Les différentes images dont nous avons besoin sont créés grâce à l'utilisation d'un « shader » qui permet de spécifier l'attribut géométrique rendu par texel entre la position, la normale ou l'indice de face. La Figure 6.5 montre les trois différentes images géométriques pour la méso-structure du modèle *Isba*. Les images géométriques sont ensuite utilisées lors du processus de synthèse de l'apparence.



(a) (b) (c)

FIG. 6.5: *Images géométriques de la méso-structure pour le modèle Isba qui représente un dallage en marbre. Respectivement en (a), (b) et (c), l'image des positions 3D (stockage en réel), l'image réels des normales (stockage en réel) et l'image des indices de face (stockage en entier) : on voit apparaître les déformations dues à la paramétrisation, notamment aux niveaux du bord des dalles qui est normalement rectiligne.*

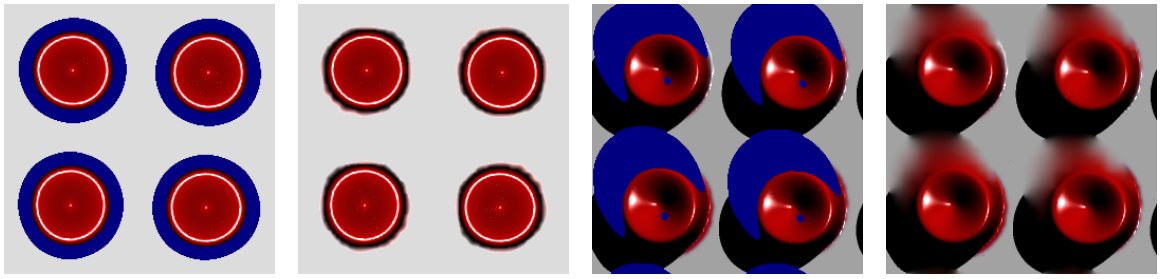
6.5.2 Synthèse de l'apparence

La synthèse de l'apparence des images de la fonction Flat-BTF est effectuée à partir des images géométriques de la méso-structure. La géométrie ainsi définie pour chaque texel sert directement aux calculs de leur apparence. L'image des positions 3D est utilisée avec le maillage original de la méso-structure pour les différents tests de visibilité. Pour chaque direction de vue et pour chaque texel, un rayon est lancé depuis la position définie par le texel courant et un test d'intersection est effectué avec la maillage original. Si une intersection est trouvée, c'est que le texel n'est pas visible et il est marqué indéfini pour la vue concernée. Si le texel est visible, on évalue alors son apparence pour chaque direction de lumière incidente. Un deuxième test de visibilité, cette fois-ci avec la direction de lumière, détermine si le texel est ombragé ou non. Si le texel est visible depuis la source lumineuse, la normale et l'indice de face qui lui corresponde dans les images géométriques sont utilisés pour calculer l'éclairement local. Dans notre implémentation, l'indice de face est utilisé afin de pouvoir retrouver à partir de la face correspondante la fonction de plaquage des textures et l'évaluation du modèle de BRDF correspondant. Pour palier aux problèmes de crénelage, les images géométriques en entrée ont, en général une résolution deux fois supérieure à la résolution des images finales de la fonction Flat-BTF. Tous les calculs d'éclairement sont alors effectués sur les images de résolution supérieure avant d'être réduites par filtrage à la résolution finale.

Pour rajouter du réalisme aux images générées, l'éclairement indirect peut également être évalué pour les texels visibles par une simple simulation de Monte-Carlo [DBB06] avec le maillage de la méso-structure. Son évaluation s'avérant trop longue à l'usage, nous envisageons d'adapter dans une prochaine implémentation la méthode de Daubert et al. [DKS+03] basée sur un pré-calcul de la visibilité de la source lumineuse.

6.5.3 Traitement des texels indéfinis

En sortie de l'étape de synthèse, toutes les images de la fonction Flat-BTF contiennent des trous d'informations correspondant aux texels non-visibles pour certaines directions de vue. Pour une méthode de compression des données par texel, ces trous peuvent simplement être ignorés par l'ajout d'un masque de visibilité indiquant pour chaque vue les texels indéfinis. Cependant pour une utilisation directe des Flat-BTF ou une conversion dans la représentation standard, ces trous doivent être remplis pour ne pas introduire d'erreurs si l'on met en place un filtrage spatial des texels. Une solution simple consiste à utiliser un algorithme « Push-Pull » pour remplir un trou à partir de la moyenne des valeurs de son voisinage. La Figure 6.6 montre les résultats que nous avons obtenus avec cette méthode. Notons, que des méthodes de remplissage plus coûteuses basées sur l'équation de poisson [PGB03] fourniraient un meilleur remplissage.



Avant Après Avant Après

FIG. 6.6: *État avant et après remplissage des zones (en bleu) correspondant aux texels indéfinis dans les images de la Flat-BTF Bouton.*

6.6 Synthèse des cartes d'indirections : la fonction VDIM

Telle que nous l'avons définie (*cf.* Section 5.3), la fonction VDIM encode les effets de parallaxe de la représentation Flat-BTF et sa combinaison avec la fonction Flat-BTF permet au final d'exprimer les données d'apparence de celle-ci selon la surface de référence, comme la représentation classique des BTFs. La fonction VDIM est constituée de cartes d'indirections qui encodent la visibilité de la méso-structure pour les différentes directions de vue. Ces cartes d'indirections sont générées simplement sur le même principe que pour la génération des images d'une BTF pour la représentation classique, c'est à dire par des rayons lancés depuis la surface de référence. Cependant, le processus de synthèse est beaucoup plus simple car seule la visibilité pour les directions de vue est déterminée, il n'est pas nécessaire d'évaluer d'apparence et donc de considérer les différentes directions de lumière incidente. Les indirections sont données par la paramétrisation globale selon laquelle les données d'apparence ont été encodées.

Pour chaque direction de vue $\vec{v}_k$, une carte des indirections $VDIM_{\vec{v}_k}$ est synthétisée. Pour chaque texel (i, j) défini sur la surface de référence, on détermine l'intersection entre la méso-structure et le rayon en partance du texel (i, j) et en direction de la vue $\vec{v}_k$. Les coordonnées (u, v) de paramétrisation au point d'intersection définissent alors les coordonnées (i', j') de l'apparence dans la fonction Flat-BTF pour la position intersectée sur la méso-structure. Le décalage entre les coordonnées (i, j) et (i', j') est stocké dans la carte correspondante $VDIM_{\vec{v}_k}$ aux coordonnées (i, j). Plusieurs rayons sont lancés

par texel afin d'intégrer plus précisément la portion de surface représentée pour un texel. La moyenne des coordonnées (u, v) aux multiples intersections est alors utilisée pour déterminer le décalage. La Figure 6.7 illustre le fait que les décalages sont plus ou moins alignés avec les directions de vue en raison des déformations induites par la méthode de paramétrisation.

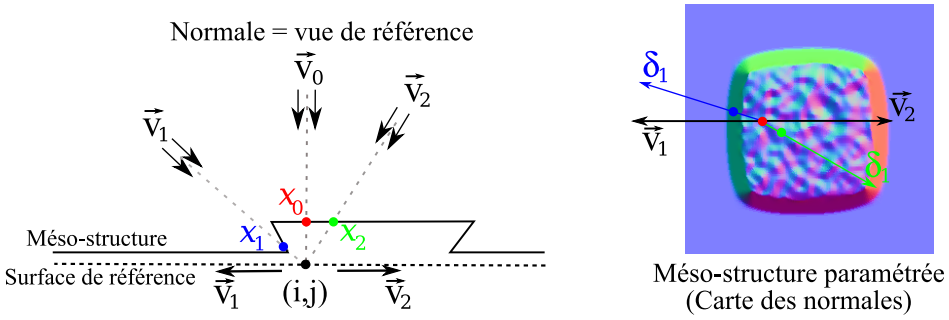


FIG. 6.7: *La corrélation entre les directions de vue et les décalages correspondant pour les indirections de la fonction VDIM. Pour chaque texel (i, j) sur le plan de référence, et chaque vue $\vec{v}_k$, nous calculons l'intersection x_k avec la méso-structure. Les déplacements $\vec{\delta}_1 = x_1 - x_0$ et $\vec{\delta}_2 = x_2 - x_0$ sur l'espace de paramétrisation sont plus ou moins alignés avec les directions de vue projetées $\vec{v}_1$ et $\vec{v}_2$.*

6.7 Discussion

Dans ce chapitre, nous avons montré que la représentation Flat-BTF est facilement implémentable dans le cadre de la synthèse de BTFs. Nous avons introduit les différentes étapes requises afin de générer des Flat-BTFs de synthèse tout en proposant une première solution fonctionnelle pour chacune des étapes. L'implémentation de notre représentation Flat-BTF ne nécessite pas des méthodes complexes pour fournir des résultats visuellement convaincants et exploitables en tant que BTFs.

Les premières étapes de modélisation et de paramétrisation sont directement liées car la modélisation de méso-structures plus complexes comme par exemple un échantillon de vannerie en osier, implique des méthodes de paramétrisation globale plus complexes que celle que nous avons utilisée dans notre implémentation. Les paramétrisations en atlas ne nous paraissent pas incompatibles avec la représentation Flat-BTF et elles constituent potentiellement une bonne solution pour les surfaces complexes à trous bien que le nombre de texels utiles dans les textures résultantes ne sera pas maximal.

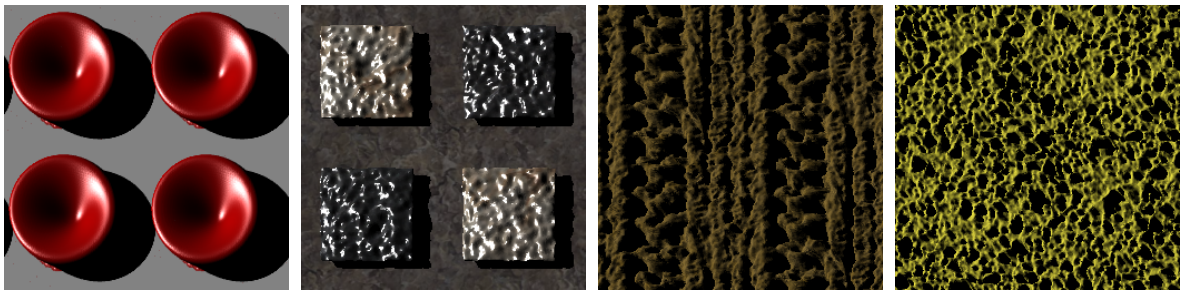
Dans le chapitre suivant, nous présentons différentes Flat-BTFs que nous avons générées avec les BTFs équivalentes en représentation classique. Nous utilisons ensuite ce jeu de données pour réaliser une étude comparative entre les deux représentations. Nous montrons ainsi par une métrique adaptée, que les données par texel deviennent beaucoup plus cohérentes lorsque la parallaxe a été extraite. Nous illustrons ensuite ce constat par une étude en images d'ABRDFs similaires entre une fonction Flat-BTF et une BTF classique. Enfin, nous présentons une étude qualitative entre les deux représentations dans le cas d'une reconstruction vers la représentation classique depuis la représentation Flat-BTF.

Chapitre 7

Résultats expérimentaux et analyse

Ce chapitre présente l'analyse des résultats expérimentaux pour un jeu de données synthétiques de Flat-BTFs (générées comme explicité dans le chapitre précédent) et de leur BTF correspondante. Nous présentons une analyse statistique et visuelle afin d'identifier l'apport de la représentation Flat-BTF comparé à la représentation classique. Nous y étudions aussi l'erreur spatiale qui peut être introduite lors de la reconstruction en BTF classique à partir de la représentation Flat-BTF.

7.1 Jeu de données



Bouton Isba Tricot Éponge

Fig. 7.1: *Un aperçu du jeu de données que nous avons utilisé dans notre analyse. De gauche à droite, la BTF Bouton se présente comme une succession de plots en plastique rouge sur une surface plane grise. La BTF Isba représente un dallage constitué de deux marbres différents sur une surface de granit. La BTF Tricot reprend la méso-structure simplifiée des mailles d'un tricot et la BTF Éponge qui comme son nom l'indique tente de reproduire la surface d'une éponge. Les fonctions de hauteur représentant les méso-structures de ces deux dernières BTFs sont issues de notre opération de reconstruction (cf. Section 4.2.1).*

Pour nos expérimentations, nous avons généré un ensemble de Flat-BTFs ainsi que les BTFs correspondantes pour différentes formes de méso-structures et des propriétés de réflectance variées. Ces modèles sont illustrés en Figure 7.1. Les modèles des BTFs *Bouton* et *Isba* ont été modélisés à l'aide de fonctions analytiques tandis que les modèles des BTFs *Tricot* et *Éponge* proviennent d'une reconstruction de la méso-structure à partir de BTFs acquises. Toutes ces méso-structures correspondent à des fonctions de hauteur excepté celle du modèle *Isba* qui présente des repliements de la surface.

La BTF *Bouton* présente une méso-structure assez régulière avec de fortes variations introduites sur la surface par des plots creux et concentriques. L'effet de parallaxe se traduit principalement par

le déplacement relatif des plots dans les images. Par ailleurs, la surface comprend deux propriétés de réflectance différentes qui sont homogènes spatialement tout en permettant de distinguer les plots et la surface plane. La BTF *Isba* présente une méso-structure assez plane pour le support en granit mais beaucoup plus de détaillée pour les dalles de marbres. Similairement à la BTF *Bouton*, l'effet de parallaxe se traduit par le déplacement relatif des dalles dans les images. Deux textures diffuses de marbre et une autre de granit définissent les propriétés de réflectance qui varient donc spatialement. Les BTFs *Tricot* et *Éponge*, issues de BTFs mesurées, présentent des détails sur toute leur surface avec des fréquences assez hautes. Leurs propriétés sont par contre diffuses et homogènes sur toute la surface de la méso-structure. L'effet de parallaxe dans les images de ces deux BTFs est similaire à celui que l'on peut observer dans les BTFs mesurées (*cf.* Section 4.2.2).

Les Flat-BTFs et BTFs que nous avons générées ont une résolution angulaire de 81×81 directions de vue et de lumière ainsi qu'une résolution spatiale de 256×256. Cette échantillonnage est identique à celui des BTFs de l'université de Bonn qui sont largement utilisées pour les expériences sur les BTFs. L'apparence des données générées prend en compte l'éclairement local et les ombres. En nous plaçant dans un cas extrême, le calcul de l'illumination indirecte n'est pas nécessaire ici car les phénomènes lumineux qui en découlent ont tendance à atténuer les fréquences dans les images (ex : les ombres douces face aux ombres dures), dès lors leur influence ne serait pas significative pour nos analyses.

7.2 Mesure de la cohérence de l'apparence

Comme nous l'avons déjà remarqué, la principale difficulté pour les méthodes de compression par texel provient de la combinaison des effets de parallaxe avec les effets lumineux, ce qui a conduit toutes ces méthodes proposées à compresser distinctement direction de vue par direction de vue. Pour démontrer l'apport des Flat-BTFs dans la variation des données par vue, nous avons analysé la cohérence des variations de l'apparence en fonction des différentes directions de vue.

Pour une vue donnée $\vec{v}_k$, et un texel donné (i, j), nous définissons une fonction de luminance $\vec{L}_k(i, j)$ comme les valeurs d'illuminations résultantes pour chaque direction lumineuse $\vec{l}_m$ telle que :

$$\vec{L}_k\left(i,j\right) = \left\{ BTF_{\vec{v}_k,\vec{l}_m}\left(i,j\right), \forall m \right\}$$

pour une BTF, ou :

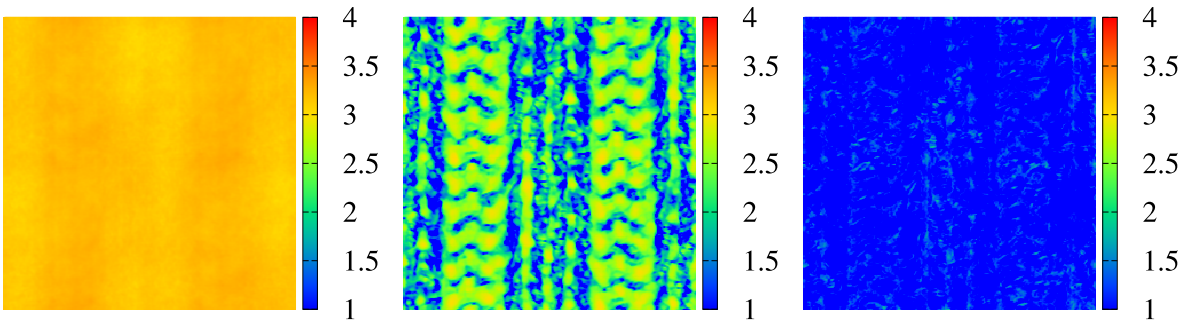
$$\vec{L}_k\left(i,j\right) = \left\{ Flat-BTF_{\vec{v}_k,\vec{l}_m}\left(i,j\right), \forall m \right\}$$

pour la fonction Flat-BTF. Pour chaque texel, la variance de ce vecteur donne une certaine mesure de la cohérence de l'apparence pour l'ensemble des directions de vue. Les variances moyennes et maximales pour l'ensemble des texels indiquent alors globalement la cohérence de l'apparence pour une BTF ou une fonction Flat-BTF. Le Tableau 7.1 présente les différentes mesures de cohérence de l'apparence que nous avons effectuées pour les différentes BTFs et Flat-BTFs synthétisées. Dans le cas des Flat-BTFs, la mesure de la cohérence est réalisée sur la fonction Flat-BTF avec et sans le remplissage des données indéfinies (*cf.* Section 6.5.3). La Figure 7.2 illustre les variances obtenues par texel pour les différentes représentations du modèle *Tricot*.

Analyse : Comme attendu, notre représentation Flat-BTF offre une bien meilleure cohérence de l'apparence en fonction des directions de vue. Pour tous les modèles testés, la variance moyenne est au minimum réduite de 50% et la variance maximale est réduite en moyenne de 25%. Lors d'une compression des données par texel, seules les données qui sont visibles pour une direction de vue donnée sont pertinentes. En d'autres mots, les données indéfinies par texel peuvent être ignorées. La réduction de la variance est alors encore plus importante avec : 66% de réduction au minimum pour la variance moyenne et 50% au minimum pour la variance maximale. Notons que pour les modèles présentant les plus hautes fréquences de détails sur la méso-structure, la variance moyenne est divisée par 8 pour le modèle *Tricot* et par 16 pour le modèle *Éponge* par rapport à la représentation classique. Comme nous l'avons déjà mentionné dans la Section 7.1, l'illumination indirecte aurait ici en théorie assez peu

Nom	*Bouton*	*Isba*	*Tricot*	*Éponge*
	Variance moyenne (Distance Lab)			
BTF	35.38	14.29	23.20	39.82
FBTF$^{(1)}$	16.87	8.75	7.83	18.95
FBTF$^{(2)}$	7.14	5.12	1.38	5.19
	Variance maximale (Distance Lab)			
BTF	41.93	33.14	26.85	41.57
FBTF$^{(1)}$	32.92	29.65	19.46	30.32
FBTF$^{(2)}$	18.89	9.96	10.01	24.8

TAB. 7.1: *Les variances moyennes et maximales de la distance Lab pour le vecteur d'illumination (*81 *directions de vue et de lumière pour une résolution spatiale de* 256×256*). Nous comparons la représentation originale avec deux versions de notre représentation Flat-BTF. Dans la première version* (1)*, les texels non visibles ont été remplis par notre algorithme de remplissage. Dans la version* (2) *les texels non visibles et donc indéfinis ont été ignorés pour l'estimation de la variance.*



(a) (b) (c)

FIG. 7.2: *Images de la variance du vecteur d'illumination en distance Lab selon les différentes directions de vue pour le modèle Tricot. En (a), la variance pour la représentation classique. En (b), la variance pour la fonction Flat-BTF correspondante avec les texels indéfinis remplis. En (c), la variance pour la fonction Flat-BTF sans la prise en compte des texels indéfinis.*

d'influence sur la mesure de la cohérence de l'apparence face aux phénomènes de hautes fréquences tels les ombres ou l'éclairement direct. En utilisant de meilleures techniques d'extrapolation spatiale que la méthode Push-Pull, la variance de la fonction Flat-BTF remplie devrait être encore réduite.

La représentation Flat-BTF semble donc nettement plus adaptée lorsque des méthodes de compression par texel sont utilisées pour réduire la taille des BTFs. L'étude de certaines ABRDFs caractéristiques en image qui suit, nous démontre d'une autre manière l'apport de la représentation Flat-BTF.

7.3 Etude visuelle des ABRDFs

Afin de visualiser l'amélioration de la cohérence de l'apparence des ABRDFs entre les deux représentations, nous avons repris la méthode de visualisation des ABRDFs en image telle qu'elle est décrite dans la Section 4.3.1. Les Figures 7.4 (Page 71) et 7.5 (Page 72) illustrent les différentes ABRDFs significatives que nous avons choisies d'extraire depuis les différentes BTFs et Flat-BTFs

de notre jeu de données. Pour chacun des modèles, nous présentons deux ABRDFs de la BTF en représentation standard suivie juste en dessous des deux ABRDFs équivalentes dans la représentation Flat-BTF. Nous rappelons qu'une ABRDF correspond à toutes les intensités lumineuses prises en un texel pour toutes les directions de vue et de lumière : un texel correspond à une position unique sur la surface pour les BTFs, et à une position unique sur la méso-surface pour les Flat-BTFs.

Comme introduit précédemment à la Section 4.3.1.2, les variations des phénomènes qui dépendent de la direction de vue se lisent selon l'axe vertical des images et se retrouvent logiquement pour toutes les directions lumineuses lorsque les phénomènes ne dépendent que de la direction de vue. De même, les variations des phénomènes qui dépendent de la direction lumineuse se lisent selon l'axe horizontal des images, mais ne se retrouvent pas pour toutes les directions de vue à cause de la parallaxe et ce, quelque soient les phénomènes. Dorénavant, cette dernière affirmation est vraie uniquement pour les ABRDFs extraites des BTFs classiques (*i.e.* les ABRDFs en Figures 7.4-(a), 7.4-(b), 7.5-(c) et 7.5-(d)). Pour les ABRDFs issues des Flat-BTFs (*i.e.* les ABRDFs en Figures 7.4-(a'), 7.4-(b'), 7.5-(c') et 7.5-(d'), les variations des phénomènes qui ne dépendent que des directions lumineuses se retrouvent verticalement pour toutes les vues. L'observation des phénomènes d'ombres dans les ABRDFs des Flat-BTFs nous le confirme si l'on ne tient pas compte des lignes qui correspondent aux directions de vue pour lesquelles la portion de méso-structure n'était pas visible. Ces lignes précédées d'un marqueur bleu sur l'axe des directions de vue correspondent aux données indéfinies qui nous avons complétées après l'étape de synthèse à l'aide de notre méthode de remplissage.

Analyse : D'une manière générale, l'étude comparative des ABRDFs en image entre les deux représentations montre l'impacte et l'importance de la parallaxe dans la cohérence des données par texel. Avec la représentation Flat-BTF qui sépare l'effet de parallaxe du reste des données, les ABRDFs en image issues de la fonction Flat-BTF apparaissent beaucoup plus cohérentes que ce soit pour celles qui correspondent à la même position sur la méso-structure ou pour celles qui correspondent à la même position dans l'image.

La Figure 7.4 présente les ABRDFs extraites pour les modèles *Bouton* et *Isba* dont les propriétés de réflexion varient spatialement. Les ABRDFs en 7.4-(a) et 7.4-(b) issues de la représentation classique démontrent parfaitement le fait que plusieurs BRDFs de la méso-structure contribuent aux données d'un même texel et tout particulièrement pour les ABRDFs 7.4-(a)-1 et 7.4-(b)-2. En revanche, pour les ABRDFs de la fonction Flat-BTF, on observe dans les images une certaine cohérence verticale des données pour les effets lumineux indépendant de la direction de vue tels que les réflexions lambertiennes et les ombres. Hormis les données issues du remplissage (*i.e.* les lignes précédées d'un marqueur bleu), ces ABRDFs sont constituées de l'éclairement local d'une BRDF unique avec des effets d'éclairement indirect.

La Figure 7.5 présente les ABRDFs extraites pour les modèles *Tricot* et *Éponge* dont les propriétés de réflectance sont homogènes et lambertiennes sur la surface. En revanche, ces modèles présentent des méso-structures avec une haute fréquence de détails sur leur surface. Pour les ABRDFs de la représentation classique, l'aspect des images est assez chaotique en raison du fait qu'un texel renvoie à plusieurs positions sur la méso-structure. Le fait que ces modèles présentent des détails de haute fréquence génère de nombreux phénomènes d'ombres sur la surface qui conduisent à cet aspect chaotique des données par texel. Lorsque l'on passe à la représentation Flat-BTF, les données sont identiques pour chaque ligne car aucun des effets lumineux ne dépend de la direction de vue. Ce phénomène démontre que dans le cas de la représentation classique, l'effet de parallaxe est responsable en grande partie de l'incohérence des données par texel.

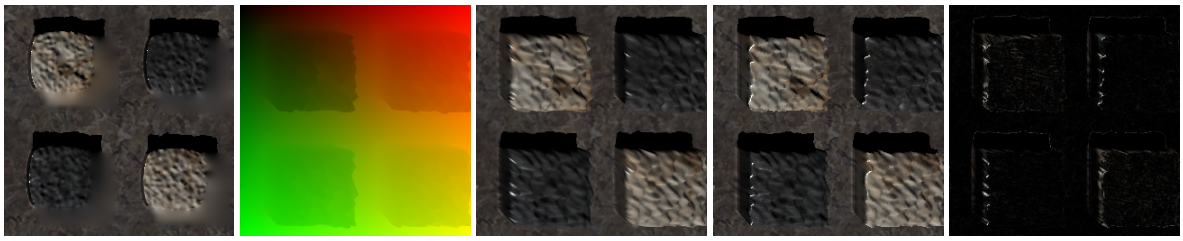
Pour conclure, la comparaison des tailles de ces images après leur compression à l'aide du format *JPEG* ou *PNG* illustre bien le gain de compression apporté par l'amélioration de la cohérence des données grâce à la représentation Flat-BTF (*cf.* Tableau 7.2).

7.4 Erreur de reconstruction

La représentation standard des BTFs peut être obtenue depuis les Flat-BTFs en combinant les

	PNG Taille en ko		JPEG Taille en ko	
ABRDF	Texel 1	Texel 2	Texel 1	Texel 2
Bouton	3.3 < 3.4	3.2 < 5.2	10.6 < 12.6	10.6 < 14.1
Isba	2.1 < 4.8	2.5 < 5.7	5.3 < 6.5	7.9 < 11.2
Tricot	3.8 < 5.6	1.4 < 5.8	8.9 < 12.9	3.7 < 12.8
Éponge	3.5 < 5.6	1.1 < 5.9	9.2 < 14.7	3.4 < 14.8

TAB. 7.2: *Tableau résumant les tailles en kilo-octets des images pour les ABRDFs des Figures 7.4 (Page 71) et 7.5 (Page 72) au format PNG et JPEG pour la qualité maximale. La ligne intitulée « ABRDF » indique le numéro du texel de l'ABRDF concernée sur les figures. Le chiffre à gauche de l'inégalité correspond à la taille pour la Flat-BTF tandis que le chiffre à droite de l'inégalité correspond à la taille pour la représentation classique. Pour certaines ABRDFs des modèles Tricot ou Éponge, la taille dans la représentation Flat-BTF est jusqu'à 4 ou 5 fois moindre.*



(a) (b) (c) (d) (e)

FIG. 7.3: *En (a) Une image issue d'une Flat-BTF. En (b) l'indirection correspondante à la vue issue du VDIM. En (c) l'image reconstruite en utilisant (b) comme indirection dans (a). En (d) l'image de référence correspondante dans la représentation standard. En (e) la différence absolue en RVB entre (c) et (d).*

images de la fonction Flat-BTF avec les indirections de chaque vue de la fonction VDIM. Cependant, l'échantillonnage spatial dans une Flat-BTF est uniforme selon la méso-structure paramétrée alors qu'il est uniforme selon la surface de référence dans la représentation standard. Cette différence de stratégie d'échantillonnage peut conduire à des erreurs d'indirections ou de crénelages pendant la reconstruction. Nous avons donc observé les différences entre les images reconstruites à partir de Flat-BTFs et des images de référence synthétisées dans la représentation classique. Pour réduire les crénelages potentiels, nous avons généré tous les VDIMs avec un sur-échantillonnage spatial et une filtrage linéaire sur les texels voisins est appliqué pour l'indirection. La Figure 7.3 illustre cette erreur de reconstruction pour une image d'une BTF et le Tableau 7.3 présente les moyennes des différences entre les images reconstruites et les images de référence pour différentes BTFs synthétisées. La moyenne est calculée sur l'ensemble des texels des textures et pour toutes les directions de vue et de lumière.

Analyse : Dans le Tableau 7.3, la différence maximale est obtenue pour la BTF *Éponge* principalement dû aux variations importantes de hauteur dans la méso-structure. Cependant, d'après nos observations pour les différents modèles, les lignes de couleurs dans les images de différence représentent les décalages dans les texels reconstruits. Cette erreur spatial est très faible et les décalages dépassent rarement le texel comme en atteste la Figure 7.3.

Nom	*Bouton*	*Isba*	*Tricot*	*Éponge*
Différence Moyenne	3	2.91	5.46	10.23
Différence maximale	8.17	9.56	9.12	15.10
Variance	0.46	2.41	0.30	0.63

TAB. 7.3: *La moyenne, le maximum et la variance des différences (en distance Lab) des texels entre les images de référence de BTF et les images reconstruites depuis la représentation correspondante en Flat-BTF.*

7.5 Analyse globale

Les résultats de notre étude statistique pour les Flat-BTFs et BTFs que nous avons générées démontrent que notre représentation Flat-BTF améliore sensiblement la cohérence des données par texel. L'étude visuelle d'ABRDFs équivalentes dans les deux représentations Flat-BTF et BTF classique illustre bien ce gain de cohérence dans les images. En effet, notre nouvelle représentation réduit les problèmes induits par la parallaxe. Le Tableau 7.3 fourni à la fin de cette étude démontre directement un gain de compression en comparant les tailles des ABRDFs équivalentes compressées à l'aide de l'un des formats de compression d'images standard. Enfin, l'erreur de reconstruction démontre que la recombinaison de la fonction VDIM et de la fonction Flat-BTF n'engendre pas d'erreur visuelle significative par rapport à une représentation Flat-BTF.

Dans un certain sens, on peut se dire qu'il est logique que la cohérence par texel soit améliorée et que la taille des ABRDFs soit réduite lors de la compression, vu que l'on a décorrélé les types d'informations dans les présentes données. En effet, l'information de parallaxe est dorénavant encodée dans la fonction VDIM et pour évaluer précisément le gain de notre représentation, il faudrait aussi prendre en compte la taille de cette information de parallaxe stockée à part. Ce gain se démontre facilement si l'on considère l'une des ABRDFs de la fonction Flat-BTF.

Pour repasser cette ABRDF dans la représentation classique, il faut alors considérer autant d'indirections qu'il y a de directions de vue, soient 81 directions dans nos exemples. La résolution des images de nos BTFs étant de 256×256 texels, on peut donc considérer qu'un octet suffit pour exprimer un décalage selon une dimension, et donc que deux octets suffisent pour encoder une indirection pour une vue donnée. La taille totale des indirections nécessaires pour convertir l'ABRDF est alors de 2 octets $\times$ 81 directions soient 162 octets, ce qui est relativement faible au vue de la taille d'une ABRDF de la fonction Flat-BTF déjà compressée (*cf.* Tableau 7.3). Ainsi, la représentation Flat-BTF constitue une représentation alternative pour les BTFs et permet aux méthodes de compression par texel actuelles ou à venir d'être beaucoup plus efficaces sans aucune perte de données visibles.

Ce chapitre termine la deuxième partie de ce mémoire. Dans la troisième partie, nous nous intéressons aux encodages pour un rendu efficace de l'apparence. Le premier chapitre concerne l'implémentation et l'utilisation des Flat-BTFs sur GPU. Nous y présentons une première solution afin de valider l'utilisation des Flat-BTFs sur GPUs. Le chapitre suivant s'intéresse à l'encodage de l'apparence telle qu'elle est défini dans cette première solution, soit une normale et une couleur diffuse par texel. Pour promouvoir cet encodage efficace de l'apparence, nous nous sommes placés dans un contexte de transmission de données 3D qui est aussi présenté en parallèle.

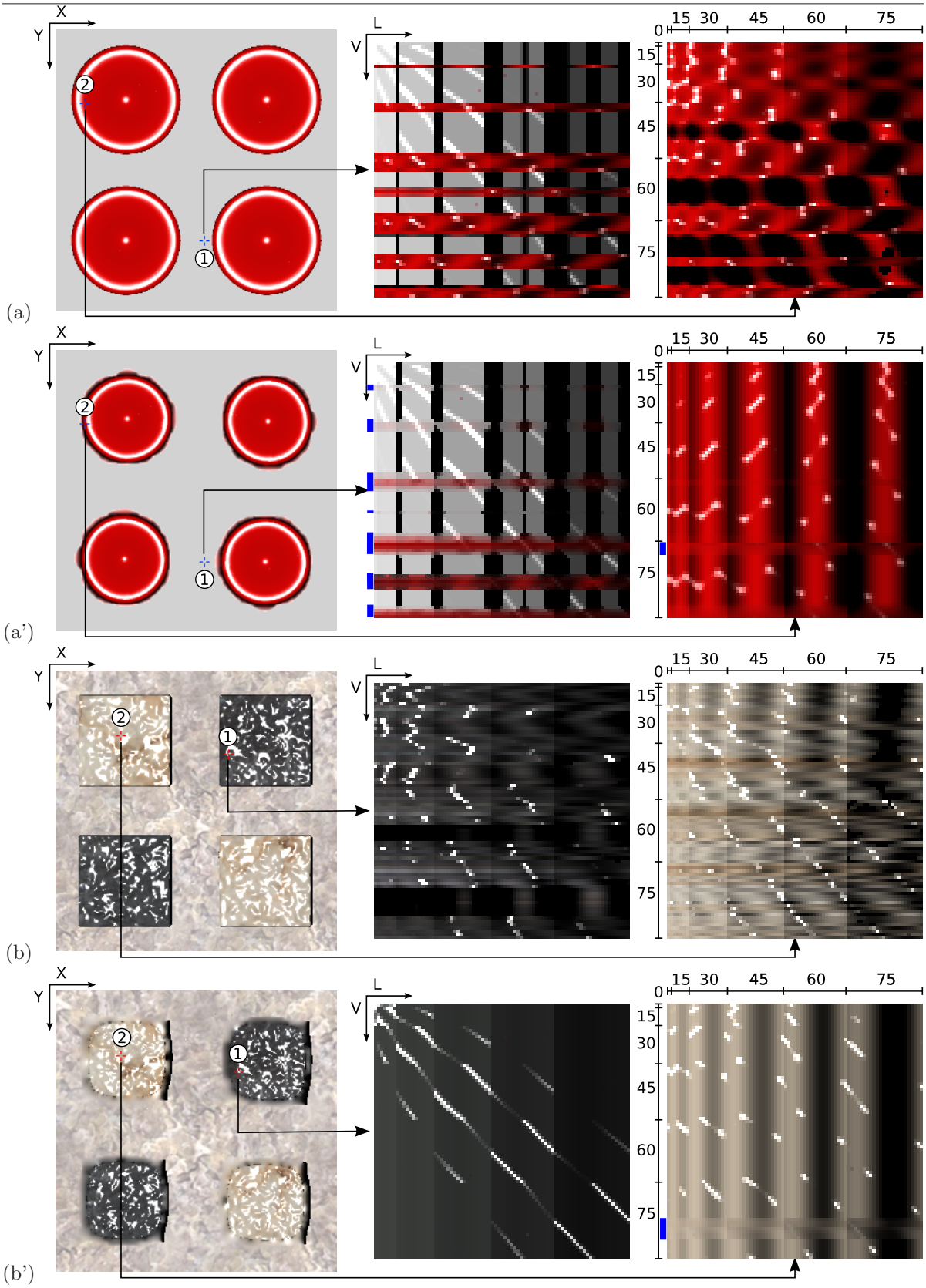


FIG. 7.4: *Une comparaison visuelle d'ABRDFs caractéristiques entre une BTF et une Flat-BTF pour les modèles Bouton (en haut) et Isba (en bas). En (a) et (b), deux ABRDFs de la BTF classique précédées de leur localisation dans le domaine spatial. En (a') et (b'), les ABRDFs équivalentes pour la fonction Flat-BTF. Pour chaque couple d'ABRDFs sur une ligne, l'ABRDF (1) a été sélectionnée pour correspondre à la même position sur la méso-structure sous-jacente, tandis que l'ABRDF (2) correspond exactement au même texel dans l'image.*

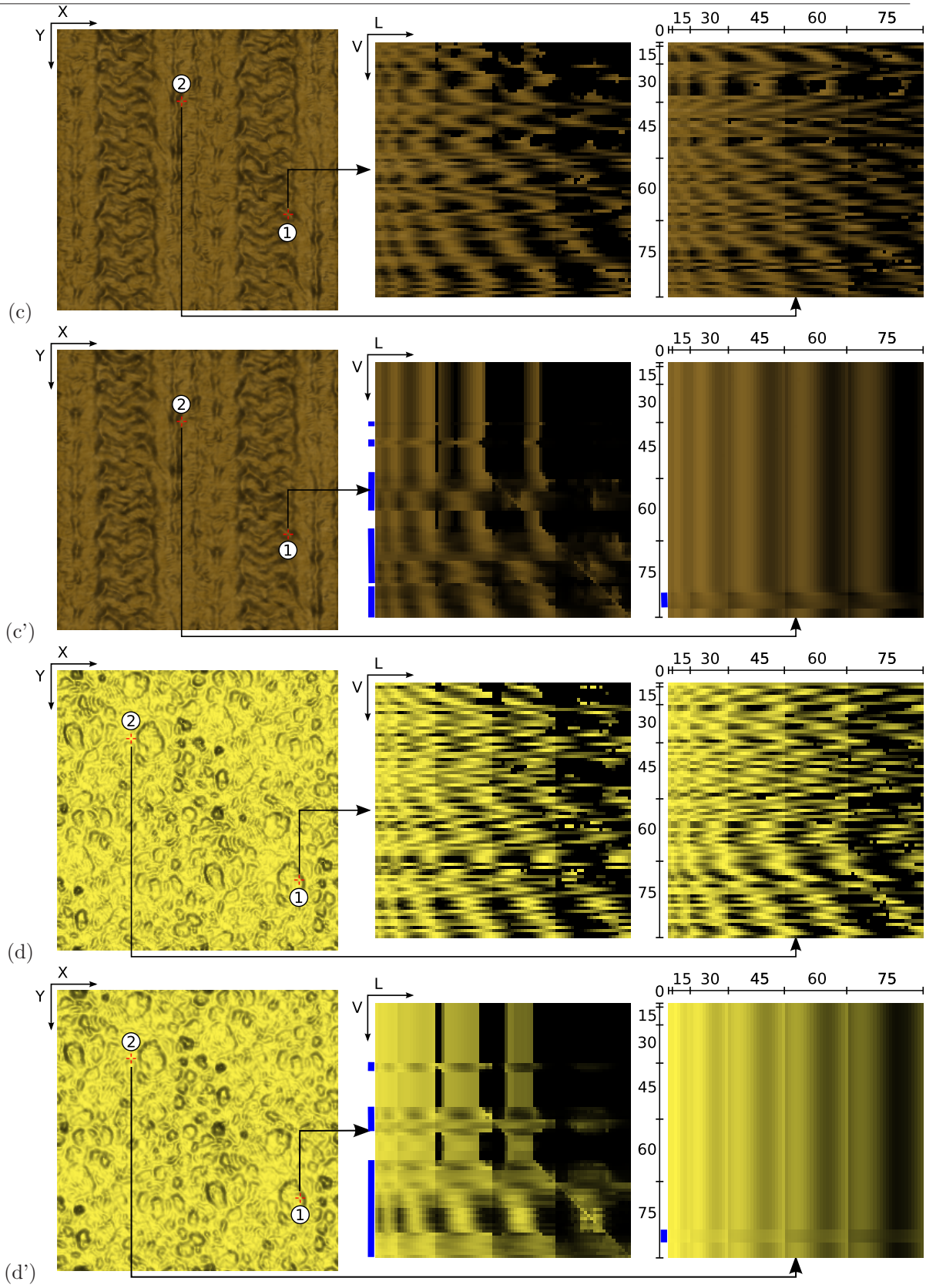


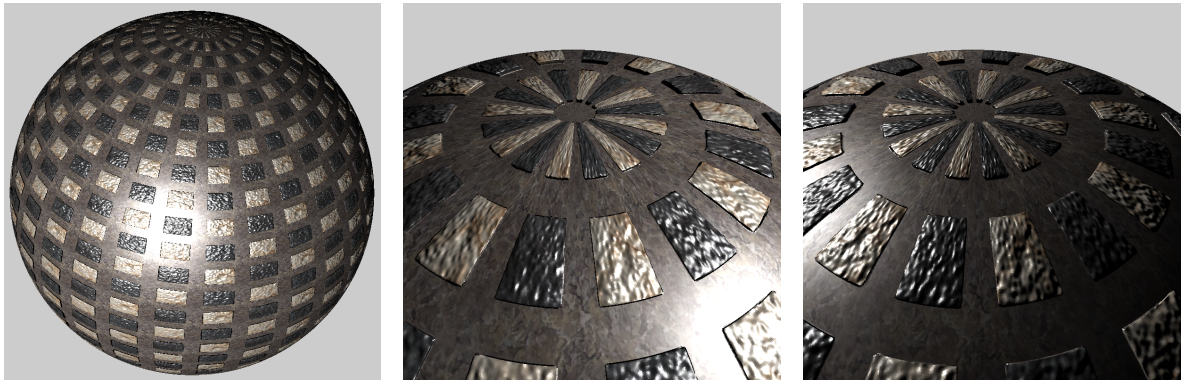
FIG. 7.5: *Une comparaison visuelle d'ABRDFs caractéristiques entre une BTF et une Flat-BTF pour les modèles Tricot (en haut) et Éponge (en bas). En (c) et (d), deux ABRDFs de la BTF classique précédées de leur localisation dans le domaine spatial. En (c') et (d'), les ABRDFs équivalentes pour la fonction Flat-BTF. Pour chaque couple d'ABRDFs sur une ligne, l'ABRDF (1) a été sélectionnée pour correspondre à la même position sur la méso-structure sous-jacente, tandis que l'ABRDF (2) correspond exactement au même texel dans l'image.*

Troisième partie

Mise en œuvre : encodages pour un rendu efficace

Chapitre 8

Implémentation et utilisation des Flat-BTFs sur GPU



(a) (b) (c)

Fig. 8.1: *Une exemple de rendu en 3D avec une implémentation de la fonction VDIM sur GPU (NVIDIA GeForce 8800 GTX) et où la fonction Flat-BTF est définie par une carte de normales et une texture d'albédo diffus. Pour une résolution de* 800×800*, le taux de rafraîchissement est aux alentours de 300 images par seconde. En (a), une vue globale montrant l'impression de relief crée par l'effet de parallaxe. En (b) et en (c), un zoom au pôle avec deux directions lumineuses différentes.*

Dans ce chapitre, nous présentons une première implémentation des Flat-BTFs sur cartes graphiques pour une application de rendu en temps-réel. De cette manière, nous avons pu valider que la représentation Flat-BTF peut être utilisée sur les GPUs actuels. L'utilisation des Flat-BTFs n'est fondamentalement pas très différente de l'utilisation des BTFs classiques. Les Flat-BTFs nécessitent juste une étape supplémentaire de reconstruction où la fonction VDIM et la fonction Flat-BTF sont recombinées afin de retrouver l'apparence dans la représentation classique. Cette étape suggère, tout comme l'utilisation des BTFs classiques, diverses stratégies pour les échantillonnages angulaires et spatiaux. Nous discutons de ces différentes stratégies possibles dans la première section de ce chapitre et nous décrivons le choix retenu pour une première solution de rendu. Dans cette solution, nous calculons d'abord par interpolation l'indirection pour une vue donnée (interpolation pour la fonction VDIM). Cette indirection nous permet ensuite d'accéder à la fonction Flat-BTF. Les sections suivantes du chapitre présentent cette approche en détails avec respectivement par section : l'interpolation spécifique que nous avons développée, l'erreur d'interpolation introduite par l'échantillonnage des directions de

vue clefs, l'implémentation de notre approche sur GPU et enfin les résultats et les performances de cette approche.

8.1 Utilisation des Flat-BTFs sur GPU

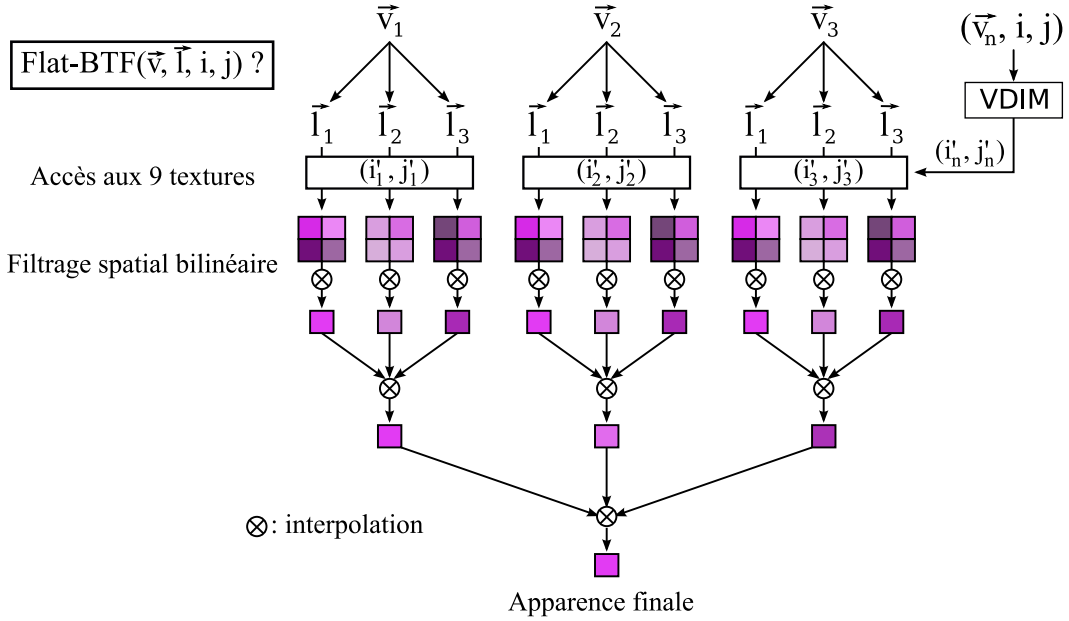


FIG. 8.2: *Détermination de l'apparence finale avec une Flat-BTF pour une direction de vue $\vec{v}$ et une direction de lumière incidente $\vec{l}$ données. On accède d'abord à la fonction VDIM afin de déterminer de nouvelles coordonnées (i'_n, j'_n), qui correspondent à l'indirection, en fonction de (i, j) et de la vue proche $\vec{v}_n$. Dans le cas des BTFs classiques, on utilise directement (i, j) pour toutes les directions de vue proches. Neuf textures sont ensuite utilisées au total pour les directions de vue les plus proches de $\vec{v}$: $\vec{v}_1$, $\vec{v}_2$ et $\vec{v}_3$ et les directions lumineuses les plus proches de $\vec{l}$: $\vec{l}_1$, $\vec{l}_2$ et $\vec{l}_3$. Pour chaque texture, les 4 texels les plus proches pour les coordonnées (i_n, j_n) sont interpolés de manière bilinéaire puis les données résultantes sont interpolées à leur tour de manière barycentrique pour les différentes direction lumineuses puis pour les différentes directions de vues afin d'obtenir l'apparence finale.*

Dans le Chapitre 3, nous avons présenté les différentes méthodes de compression et de rendu de BTFs. Pour discuter des différentes stratégies possibles pour le rendu de nos Flat-BTFs, il convient ici de décrire plus en détails le principe d'utilisation des BTFs en tant que texture. Le placage des BTFs nécessite en premier lieu, en plus des coordonnées de texture habituelles, la définition d'un repère locale à la position de placage. Ce repère locale, définit par la normale à la position et son plan tangent, permet d'exprimer localement la direction d'observation et la ou les directions lumineuses. Ainsi avec les coordonnées (i, j) de texture, nous disposons alors des six dimensions nécessaires afin d'accéder aux données d'une BTF. L'apparence de la position pour une direction d'observation et une direction lumineuse données est déterminée à partir des apparences mesurées par la BTF pour des directions clefs, réparties uniformément sur la surface d'un hémisphère. Pour déterminer les directions les plus proches, on utilise en général une texture d'environnement qui pour une direction donnée indique les directions clefs les plus proches de la BTF. Une fois, les textures requises identifiées (une par direction de vue et de lumière), on peut calculer les différentes interpolations. En générale, on considère les trois échantillons les plus proche pour les dimensions angulaires (*i.e.* les directions de vue et les directions

de lumière incidente) et les quatre échantillons les plus proche pour les dimensions spatiales (*i.e.* les texels de la texture). Ce qui fait donc 9 textures différentes, soient 36 texels différents qui permettront après 13 interpolations d'obtenir l'apparence finale. Si l'on tient compte en plus des deux accès textures nécessaires afin de déterminer les directions les plus proches, on arrive à un total de 11 accès textures différents.

Cette stratégie est directement applicable pour la représentation Flat-BTF et plus exactement pour la fonction Flat-BTF qui stocke aussi une information de réflectance en six dimensions. La différence se situe en amont de ce processus, là où la fonction VDIM intervient en vue de modifier les coordonnées de texture pour les directions de vue les plus proches. Ces modifications, illustrées sur la Figure 8.2, fournissent le décalage dû à la parallaxe nécessaire à chaque vue voisine pour accéder aux données de la fonction Flat-BTF de manière à reconstruire une représentation classique des BTFs. Chacune des directions de vue proches implique un accès aux cartes d'indirections de la fonction VDIM. La représentation Flat-BTF demande donc trois accès textures supplémentaire par rapport à la représentation classique si l'on considère trois vues voisines pour une vue donnée. Une étape de filtrage spatial peut aussi être effectuée lors de l'accès aux texels des cartes d'indirections.

Cette solution est la plus simple et s'applique surtout aux données originelles. Si l'on pose maintenant l'hypothèse que les données de la fonction Flat-BTF sont compressées par texel à l'aide de modèles analytiques de BRDF adaptés. Les directions sont alors représentées de manière continue. De ce fait, les accès textures pour déterminer les directions voisines et les interpolations dépendantes des directions de vue et de lumière deviennent obsolètes pour la fonction Flat-BTF. C'est dans ce contexte que nous avons décidé d'optimiser la détermination des indirections depuis la fonction VDIM. Afin de réduire les 4 accès textures (un accès pour la texture de voisinage des directions et trois accès pour les trois vues voisines) nécessaires, nous avons développé et testé une solution alternative qui ne requiert que deux accès textures au total. Cette solution est similaire à la méthode de rendu des méso-structures de Wang *et al.* [WWT+03] qui est basée sur le pré-calcul de cartes de profondeur pour un ensemble de directions de vue et où la profondeur est ensuite interpolée pour une vue donnée. Nous pouvons appliquer le même principe pour la fonction VDIM en remplaçant les cartes de profondeurs par nos cartes d'indirections. Les sections suivantes décrivent en détails cette solution.

8.2 Encodage GPU de la fonction VDIM

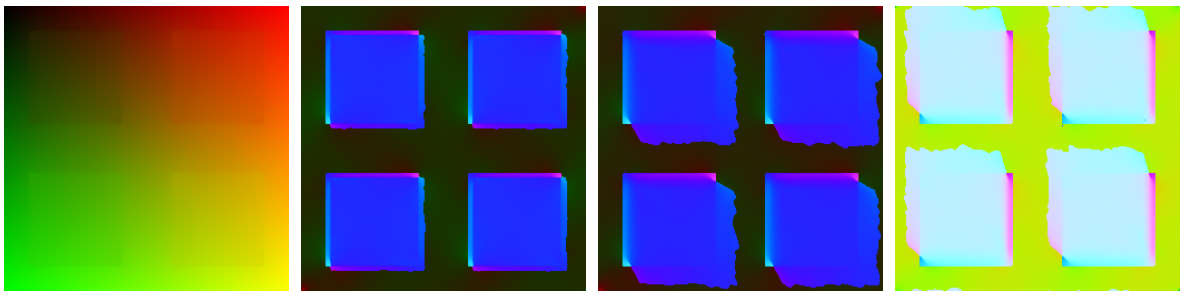
Afin d'extrapoler une indirection unique depuis les indirections calculées pour des directions de vue, nous avons dû encoder ces indirections d'une manière spécifique. Cette encodage est un peu plus coûteux en terme d'espace mémoire mais il est nettement plus précis pour interpoler l'indirection. Lors du processus de synthèse de la fonction VDIM décrit dans la Section 6.6), nous avons construit les cartes d'indirections pour un ensemble de directions de vue. Pour chaque carte d'indirection correspondant à une vue $\boldsymbol{v}$ donnée, chaque texel (i, j) encode le décalage avec les coordonnées (i', j') permettant d'accéder à la fonction Flat-BTF. Avec cet encodage, les indirections intermédiaires se déduisent simplement par interpolation bilinéaire des coordonnées (i', j') entre les vues clefs. En vue d'une interpolation plus précise entre les directions de référence, nous avons encodé les indirections d'une manière différente. Pour un texel (i, j) et une vue $\vec{v}$, les coordonnées d'indirection (i', j') sont exprimées par une position de référence $R(i, j)$ et un décalage $\Delta_{\vec{v}}(i, j)$ sur la paramétrisation et dépendant de la direction de vue (*cf.* Figure 8.3) :

$$(i', j') = VDIM_{\vec{v}}(i, j) = R(i, j) + \Delta_{\vec{v}}(i, j). \tag{8.1}$$

Plus exactement, le déplacement Δ est encodé sous la forme d'une direction 2D $\vec{\delta}$ et d'une distance d : $\Delta = d\,\vec{\delta}$. Les indirections des vues intermédiaires (i.e. entre les directions clefs) sont alors obtenues par une simple interpolation bilinéaire entre des distances clefs et une interpolation suivie d'une normalisation pour des directions clefs. Comme le déplacement est pratiquement aligné avec la projection de la direction de vue sur la surface, nous avons choisi comme indirection de référence l'indirection pour la direction de vue colinéaire à la normale de la surface de référence $\vec{n}$:

$$VDIM_{\vec{v}}(i, j) = VDIM_{\vec{n}}(i, j) + d_{\vec{v}}(i, j)\,\vec{\delta}_v(i, j). \tag{8.2}$$

Une conséquence intéressante de cet encodage est que $d_{\vec{v}} = 0$ quand $\vec{v} = \vec{n}$. De plus, grâce à la paramétrisation conforme, $\vec{\delta}_v$ est quasiment similaire à la projection de $\vec{v}$ sur la surface de référence, et donc $d_{\vec{v}}$ est toujours croissante lorsque $\vec{v}$ est de plus en plus rasante. Ces propriétés permettent une bonne qualité pour la reconstruction car elles réduisent les oscillations lorsque l'on se déplace autour d'une position donnée.



Indirection de référence $\Delta_{45,230}$ $\Delta_{75,230}$ $\Delta_{75,60}$

FIG. 8.3: *La carte d'indirections de référence pour la vue zénithale du modèle Isba suivie de trois exemples de cartes de décalages Δ pour différentes directions de vue. Dans la carte de référence, les coordonnées (i, j) sont représentées respectivement par les composantes rouge et verte. Dans les cartes de décalage, les coordonnées de la direction 2D $\vec{\delta}$ de décalage sont encodées respectivement par les composantes couleurs rouge et verte tandis que la composante bleue encode la distance d.*

8.3 Erreur d'échantillonnage

La précision des indirections intermédiaires ne dépend pas seulement de la méthode d'interpolation mais aussi de la qualité de l'échantillonnage angulaire des directions de vue utilisées pour constituer les indirections de référence. Afin de trouver l'échantillonnage angulaire des directions de vue le mieux adapté mais aussi afin de tester notre méthode d'interpolation, nous avons mesuré les erreurs d'interpolations pour différentes configurations d'échantillonnages uniformes selon les angles ϕ et θ. Pour chaque configuration d'échantillonnage angulaire (θ, ϕ) à évaluer, nous mesurons l'erreur d'indirection pour un grand ensemble de directions prises au hasard dans l'hémisphère Ω. Pour une direction donnée, l'erreur est calculée entre l'indirection déterminée par interpolation et l'indirection calculée directement sur la méso-structure par lancer de rayons. Parmi les différentes configurations possibles, les configurations 8×32 et 16×16 $(\theta \times \phi)$ offrent les meilleurs compromis entre l'erreur d'interpolation et l'espace mémoire occupé par la fonction VDIM. Dans leurs travaux, Wang et al. ont aussi utilisé la configuration 8×32. D'une manière générale et comme on peut le voir sur la Figure 8.4, plus d'échantillons sont nécessaires pour l'angle θ que pour l'angle ϕ. Cela est principalement dû au fait que nous utilisons une interpolation directionnelle sur ϕ (interpolation dans la direction du déplacement) et une simple interpolation linéaire sur θ (interpolation sur la distance). L'erreur peut être réduite en augmentant la résolution pour θ mais elle ne pourra jamais être supprimée. En effet, comme nous avons pu l'observer sur tous nos modèles de BTFs, l'erreur devient significativement plus grande lorsque la vue devient de plus en plus rasante car la définition des BTFs atteint ses limites (absence des silhouettes). Pour le rendu de la fonction VDIM sur GPU, nous utilisons l'une ou l'autre des deux configurations présentées en fonction du modèle de BTF utilisé.

8.4 Implémentation de la fonction VDIM sur GPU

Une fois que les cartes d'indirections pour chaque vue ont été calculées, le fait d'utiliser un échantillonnage uniforme sur les angles ϕ et θ nous permet d'encoder la fonction VDIM en une seule

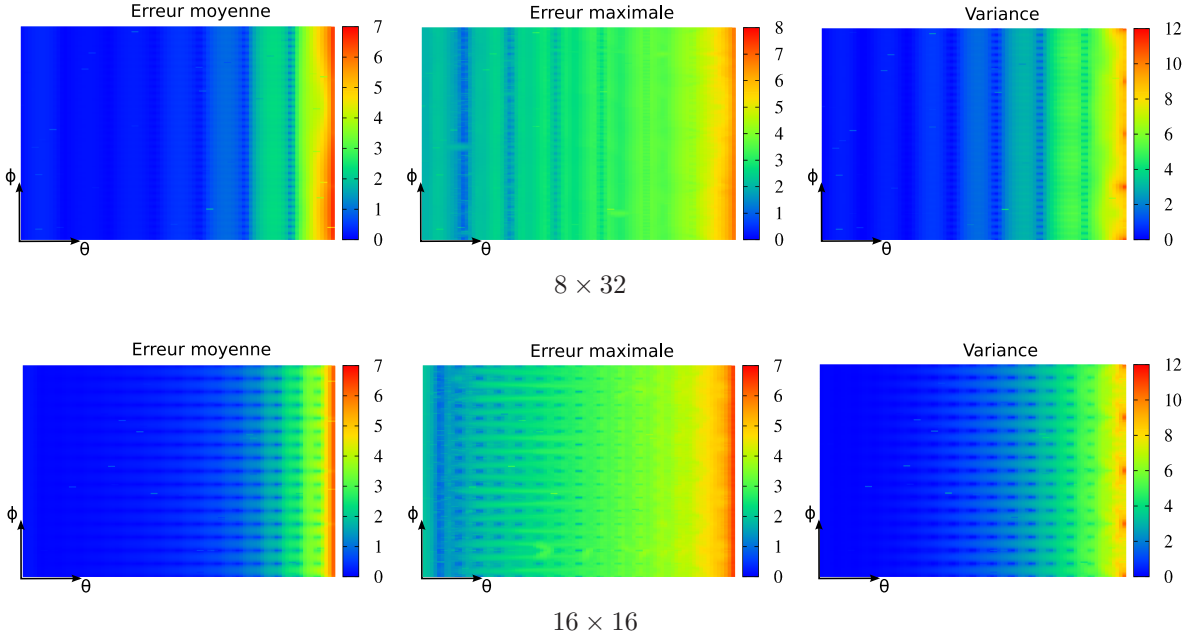


FIG. 8.4: *Erreur d'indirection sur le modèle Bouton pour deux stratégies d'échantillonnage :* 8×32 *(a) et* 16×16 *(b). Pour un grand nombre de directions de vue prises au hasard, l'erreur d'indirection est calculée entre l'indirection déterminée par interpolation des vues les plus proches et l'indirection calculée directement par lancer de rayons sur la méso-structure.*

texture 2D, comme illustré sur la Figure 8.5. Avec cet encodage où toutes les indirections différentes d'un texel sont regroupées spatialement, nous pouvons tirer parti de l'interpolation linéaire matérielle pour réaliser l'interpolation spécifique entre les indirections des directions de vue voisines.

Pour calculer les coordonnées (u, v) adaptées à cette texture VDIM spécifique, nous devons estimer en premier les angles (θ, ϕ) depuis la direction de vue courante $\vec{v}$. Pour cela, il suffit de projeter $\vec{v}$ sur le plan local tangent à la normale. Les angles sont alors ramenés dans l'intervalle $[0, 1]$. Les nouvelles coordonnées (u, v) sont ensuite calculées à partir des coordonnées de textures originales (i, j) de la manière suivante :

$$u = \frac{1}{N}\left(\lfloor i * N \rfloor + \alpha_\theta \frac{2\theta}{\pi} + \beta_\theta\right)$$
$$v = \frac{1}{M}\left(\lfloor j * M \rfloor + \alpha_\phi \frac{\phi}{2\pi} + \beta_\phi\right),$$

où $N \times M$ est la résolution des texels d'origine et $(\alpha, \beta)_\theta$ et $(\alpha, \beta)_\phi$ sont fixés en fonction de l'échantillonnage (θ, ϕ) afin d'assurer l'accès aux indirections regroupées pour un texel tout en garantissant qu'aucune interpolation ne soit faite entre les indirections de deux texels différents. Grâce à cette approche, l'indirection intermédiaire est rapidement déterminée par un seul accès texture. La carte de référence dépendante de la direction de vue zénithale est interrogée en utilisant les coordonnées de texture originales (i, j), sans aucun filtrage. L'activation du filtrage spatial produirait des incohérences avec le filtrage spatial de la texture 2D encodant la fonction VDIM.

8.5 Résultats : performance et qualité visuelle

Pour évaluer notre méthode de calcul des indirections avec la fonction VDIM, nous définissons ici comme données simples de la fonction Flat-BTF : une carte de normales et une texture diffuse qui

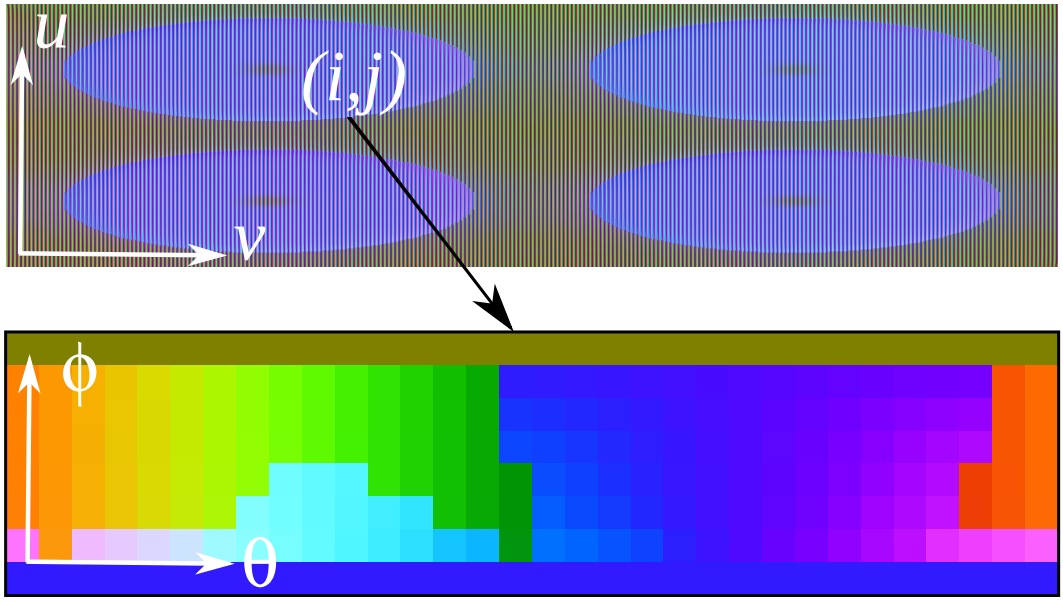


FIG. 8.5: *Organisation des texels de la texture 2D VDIM utilisée pour le rendu : pour chaque texel, nous encodons les valeurs de* $\Delta_{\vec{v}}(i,j)$ *de manière accolées pour les* 8×32 *directions. En haut, la texture 2D représentant l'intégralité de la fonction VDIM. En bas, les différentes valeurs dépendantes de la vue pour un texel donné. Notons que la même valeur est répétée sur toute une ligne pour le pôle* $(\theta = 0, \phi = 0)$*.*

sont exprimées dans l'espace de paramétrisation de la fonction Flat-BTF et qui sont utilisées comme paramètres d'une BRDF de Phong. Ces données de réflectance peuvent être considérées comme le résultat plausible d'une méthode de compression par texel utilisant un modèle analytique de BRDF. En accédant à ces textures à l'aide des nouvelles coordonnées fournies par notre méthode, nous arrivons à reproduire un rendu avec les effets de parallaxe et avec un éclairement produit par les normales et couleurs correspondantes.

Ce type d'implémentation utilise surtout l'unité de traitement des fragments du GPU et de ce fait, le taux de rafraîchissement résultant dépend de la résolution de l'image. Pour les gros plans de la Figure 8.1, nous avons pu afficher des objets 3D avec un taux de rafraîchissement de 300 images par secondes pour une résolution de 800×800 sur notre carte graphique (NVIDIA GeForce 8800 GTX). Sinon, en moyenne pour une telle résolution, le taux de rafraîchissement avoisine les 600 images par secondes (*cf.* Figure 8.6 et Figure 8.7).

En terme de qualité visuelle cette validation de la représentation Flat-BTF, ou plus exactement ce rendu de la fonction VDIM, se juge au niveau du rendu des effets de parallaxe. Même si la méthode que nous utilisons approxime en quelque sorte les indirections par le processus d'interpolation, les effets de parallaxe restent visuellement convaincant et réaliste tout en permettant d'éviter deux accès textures supplémentaires.

8.6 Discussion

Dans ce chapitre, nous avons pu valider que la représentation Flat-BTF est utilisable sur les cartes graphiques modernes. Il n'y pas de différences fondamentales en terme d'accès entre une BTF classique et une Flat-BTF. Un surcoût est engendré par l'utilisation des cartes d'indirections (VDIM), mais grâce à notre implémentation, ce surcoût est réduit à deux accès textures supplémentaires. Les résultats que nous avons obtenus démontrent que l'indirection induite entre les indirections-clefs n'introduit que peu

d'erreurs visibles lorsque leur échantillonnage est choisi avec soin. De plus, cette approche permet de reconstruire les effets de parallaxe de manière efficace pour le GPU.

En se basant sur les conclusions de la Section 3.3 sur les méthodes de compression idéales avec une décompression adaptée, la représentation Flat-BTF offre un nouveau terrain d'expérimentations pour les méthodes de compression par texel qui utilisent des modèles analytiques de BRDF. En effet, la fonction Flat-BTF assure que chaque texel représente toujours la même BRDF pour l'illumination locale quelque soit la direction de vue ou la direction de lumière incidente. Il reste bien évidemment toujours les phénomènes indirects comme les ombres propres ou les occultations propres à représenter mais les données sont déjà beaucoup plus cohérentes par texel dans la représentation Flat-BTF. Actuellement, nous travaillons sur de nouveaux modèles analytiques de BRDFs plus adaptés et plus flexibles afin de représenter par différents types lobes, les différents phénomènes par texel. Ces nouveaux modèles devraient aussi s'appliquer pour une compression par texel de la fonction VDIM. Ainsi une BTF dans la représentation Flat-BTF pourrait être représentée entre une dizaine et une vingtaine de textures de paramètres des différents lobes à la fois pour la fonction Flat-BTF et la fonction VDIM. De plus, la définition continue des dimensions angulaires par les modèles analytiques éviterait le coût de la détermination des directions voisines pour une direction donnée.

Pour poursuivre sur l'encodage de la fonction VDIM dans notre méthode de rendu, nous avons encodé à leur tour la carte de normale et la texture diffuse qui représentent la fonction Flat-BTF dans notre approche. Ces travaux se présentent comme la compression et la quantification de la normale et de la couleur qui définissent l'apparence par texel. Le chapitre suivant présente ces travaux qui ont été réalisés dans le contexte d'une application client/serveur pour de la diffusion de données 3D.

FIG. 8.6: *Visualisation de notre jeu de données de Flat-BTFs plaquées sur différents objets 3D (rendu en moyenne de 500 images par seconde en* 800×800*). Chaque fonction Flat-BTF des modèles est représentée par une carte de normales et une texture diffuse. Sur la première et la deuxième ligne, nos quatre Flat-BTFs sur une bouteille. Sur la dernière ligne, le modèle Isba sur une théière pour deux directions de lumière incidente différentes.*

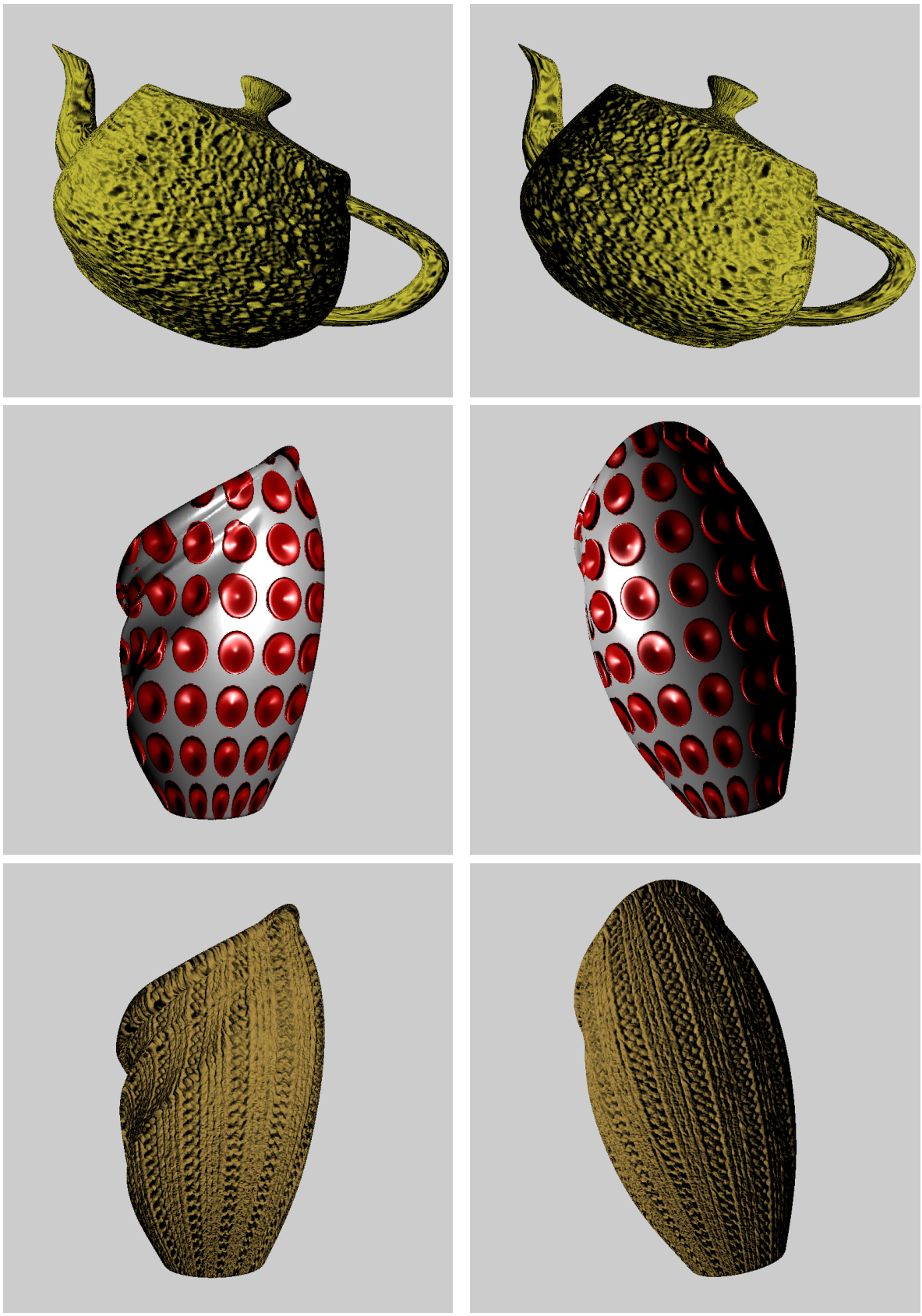


FIG. 8.7: *Visualisation de notre jeu de données de Flat-BTFs plaquées sur différents objets 3D (rendu en moyenne de 500 images par seconde en* 800×800*). Chaque fonction Flat-BTF des modèles est représentée par une carte de normales et une texture diffuse. Sur la première ligne, le modèle Éponge sur une théière pour deux directions de lumière incidente différentes. Sur la deuxième et dernière ligne, les modèles Bouton et Tricot sur un vase pour deux directions de vue différentes.*

Chapitre 9

Quantification de l'apparence

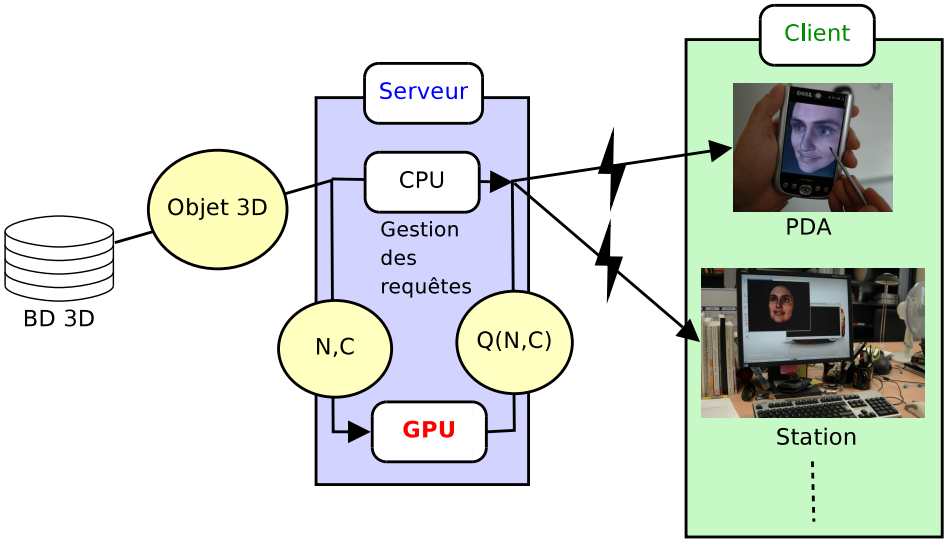


Fig. 9.1: *Le contexte de notre système de quantification à la volée est une application client-serveur de visualisation 3D. La charge de traitement de quantification effectuée à la volée est déléguée au GPU.*

Dans ce chapitre, nous présentons une méthode pour un encodage efficace de l'apparence lorsque celle-ci est définie par une normale et une couleur pour l'albédo diffus. Cet encodage consiste à quantifier, et donc discrétiser, les espaces de définition pour les normales et les couleurs. Afin de valider notre approche, nous la présentons dans le contexte d'un service de distribution de contenu 3D où la quantification des données est en adéquation avec les problèmes de transmission entre un serveur de diffusion et un client.

9.1 Contexte d'application

Dans le cadre d'un service de distribution de contenu 3D de type client/serveur et afin de minimiser les problèmes d'encombrement du réseau ou de latence, une solution classique est de limiter la bande passante requise en réduisant la complexité des modèles 3D. Généralement, cette simplification est combinée avec une méthode d'adaptation aux capacités hétérogènes des clients à l'aide de l'une des deux solutions suivantes :

- stocker plusieurs fois un même objet à diverses résolutions (niveaux de détails), et choisir la version à envoyer au client en fonction de ses capacités ; cette solution est simple et ne nécessite pas de calculs *en ligne*, mais nécessite le stockage/maintient de multiples versions d'un même objet ;
- convertir tous les objets de la base dans une représentation multirésolution, et filtrer un niveau de résolution à diffuser en fonction du client ; cette solution évite le stockage multiple, mais nécessite un prétraitement parfois complexe, et alourdit la charge du serveur lors de la diffusion (extraction du niveau de détail).

Une solution combinant les avantages des deux précédentes serait de ne stocker qu'une version de chaque objet — à pleine résolution — et de construire les niveaux de résolutions à la volée. Malheureusement, le coût dans ce cas rendrait impraticable un nombre conséquent de requêtes simultanées, réduisant son utilisation à un faible nombre de clients. De nos jours, la puissance 3D des terminaux d'affichage atteint pour la plupart des performances raisonnables en affichage (de quelques dizaines de milliers à plusieurs millions de polygones, voir Figure 9.2). Par contre, la charge des réseaux et la latence reste un problème. Il s'agit donc moins d'un problème de complexité géométrique des données que d'un problème de taille des données à transmettre qui se pose. Ce qui nous amène à envisager la compression systématique des données à transmettre.

À nouveau, le problème du stockage multiple se pose. Or, dans ce cas, il est possible d'utiliser un certain type de compression particulièrement simple et effectif : la *quantification*. De toutes les techniques de compression, la quantification a plusieurs avantages :

- le gain est exactement déterminé à l'avance ;
- l'aspect systématique et mécanique de la quantification la rend parallélisable.

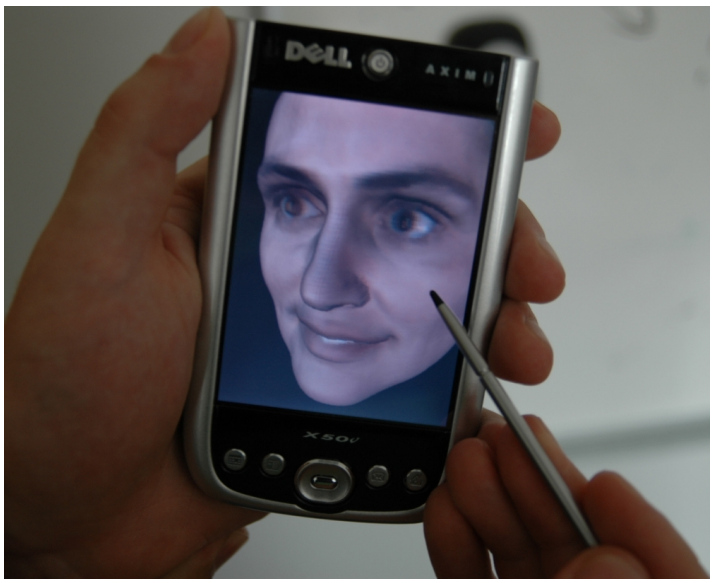


FIG. 9.2: *La puissance des derniers terminaux mobiles en fait des clients potentiels pour des applications de diffusion de contenu 3D. Sur la figure, une déquantification interactive et un rendu sont effectués sur un PDA avec OpenGL ES.*

Notre objectif est de fournir une quantification à la volée d'un flux de données 3D dont l'apparence est définie par un champ de normales et d'albédos diffus colorés, afin d'en réduire les temps de transmission. Malheureusement, même les quantification les plus simples peuvent nécessiter un certain temps de traitement (de l'ordre de quelques dizaines de secondes pour un objet de plusieurs millions d'échantillons), introduisant une forte latence dans l'accès aux données si cette quantification est effectuée *à la volée*.

Afin de réduire cette latence, nous proposons une méthode de quantification spécifique des normales

et des couleurs en tirant parti du GPU (*cf.* Figure 9.1) présent sur le serveur, réduisant la charge CPU au traitement des entrées/sorties. Déplacer cette quantification sur le GPU fait apparaître deux avantages très nets :

- une accélération du traitement d'un minimum de 25% ;
- une diminution de la charge du CPU, augmentant le nombre de requêtes simultanées pouvant être prises en charge.

La combinaison de la réduction de la charge du CPU et de la taille des données augmente significativement la disponibilité des données. Grâce son utilisation à la volée, cette quantification peut facilement être appliquée pendant la transmission au clients finaux et reste compatible avec les autres méthodes de réduction de complexité. Avec cette approche, le stockage de tous les niveaux de détails quantifiés n'est pas nécessaire, et la complexité d'origine des modèles est préservée.

9.2 Système client/serveur

Côté serveur : l'architecture générale de notre système est construite autour d'un serveur de diffusion de données 3D. Classiquement, ce serveur est connecté à une base de données d'objets 3D et en assure la diffusion. Pour notre expérience, notre base de données ne fournit qu'un seul type d'objet(s) 3D dont chaque point est défini par 9 valeurs flottantes (trois pour la position, trois pour la normale et trois pour la couleur). L'originalité de notre système réside dans l'utilisation du GPU du côté du serveur. Le flux provenant depuis la base de données pendant une requête est divisé en *positions* et *propriétés d'apparence* (normales et couleurs). Les positions sont actuellement transmises à l'état brut, alors que les propriétés d'apparence sont transférées sur le GPU, où ces valeurs sont quantifiées par une méthode adaptée au GPU. Le client reçoit alors les positions d'origine et les propriétés quantifiées de l'apparence. Ce principe de séparation de la géométrie et de son apparence rend possible l'utilisation d'une telle approche lorsque l'apparence est définie par des cartes de normales et des textures.

Le flux passant par le GPU est tamponnée, par la segmentation de l'objet d'origine en morceaux de l'ordre de quelques centaines de milliers d'échantillons. Ces morceaux sont traités individuellement et à la suite par l'unité de traitement des fragments du GPU. Le système complet peut fonctionner hors mémoire pour de gros objets en procédant itérativement seulement une sous-partie de l'objet. En conséquence, la mémoire nécessaire pour une requête donnée est fortement réduite ce qui augmente le nombre de requêtes simultanées que le serveur peut supporter. Une fois quantifiée, cette partie du modèle est transmise au client. Notons que, du côté du serveur, un système de cache peut être utilisé pour sauvegarder les données quantifiées afin d'accélérer la transmission d'un objet 3D très demandé, similairement au schéma de cache utilisé pour les pages web avec du contenu dynamique.

Côté client : Quand un client envoie une requête pour un modèle 3D, la taille de la géométrie est transférée en premier pour l'initialisation de la structure de l'objet. Les flux entrants sont ensuite déquantifiés à leur arrivée pour remplir la structure. Avec ce transfert progressif, l'objet peut être rendu en partie ou uniquement après la transmission complète des données.

9.3 Quantification de l'apparence adaptée au GPU

Contexte

Afin de réduire la taille des données pour des objets 3D, une solution classique passe par la simplification des maillages [Lin00, LS01, WK03]. Ces approches simplifient les maillages de grande taille par regroupement de sommets ou la fusion d'arêtes. Malheureusement ces méthodes détériorent la qualité géométrique des maillages. Une autre approche, introduite par Hoppe [Hop96], propose une représentation différente consistant en un maillage grossier avec plusieurs étapes de raffinement pour retrouver la résolution géométrique initiale. Cette méthode est très adaptée pour le téléchargement

progressif (en anglais, *streaming*) de données [Pri00, PR00]. Toutes ces approches sont restreintes aux maillages et requièrent des pré-calculs coûteux. De plus, elles ne s'adressent qu'au problème de réduction de la taille pour la géométrie des objets mais pas de leur apparence.

Des méthodes d'optimisation globales ont été développées [PBCK05] pour fournir un code optimal de quantification pour tous les attributs d'un sommet. Intrinsèquement globale, une telle méthode peut difficilement servir pour le téléchargement progressif de données. L'approche la plus adaptée est celle proposée par Rusinkiewicz et Levoy [RL01]. Cette solution offre une approche multirésolution réduisant la taille des données après une étape de précalcul. Cependant cette solution n'est utilisable que dans le cadre du rendu par points.

Pour adresser le problème de réduction des données de l'apparence, une approche naturelle est la quantification des coordonnées des normales pour un nombre limité de bits [BPZ99]. Facile à mettre en œuvre, cette solution n'est cependant ni uniforme ni isotrope dans sa représentation discrète de l'espace des normales. De plus, elle ne préserve pas la norme unitaire des vecteurs correspondants. Le nombre de bits peut être optimisé par regroupement [KCK04] mais une table d'indexation spécifique à chaque objet est alors nécessaire pour l'étape de décodage des normales.

La quantification basée sur un octaèdre, proposée par Deering [Dee95], a été largement utilisée afin d'obtenir une compression plus uniforme des normales. Dans cette méthode, l'espace des directions est subdivisé en 8 sections, selon un octaèdre. Chaque section est ensuite subdivisée en 6 parties, et sur chaque sextant, les angles des directions sont encodés en utilisant une grille régulière. Deering affirme que, distribuées uniformément et de manière isotrope, 100 000 directions (une densité angulaire de $0,01$) sont suffisantes pour une quantification sans artefacts visibles. Étant donné que la grille régulière ne fournit pas une telle distribution, 200 000 normales sont utilisées en pratique, via un encodage sur 18-bits, afin d'obtenir une quantification de haute qualité.

Taubin *et al.* [THLR98] utilise un « quadtree » régulier sur chaque face de l'octaèdre. L'encodage et le décodage sont alors récursifs, et la normale résultante doit être normalisée. L'avantage majeur de cette approche est une distribution plus uniforme et une meilleure exploitation des bits utilisés. Botsch *et al.* [BWK02] se restreignent à une plage de 13 bits (*i.e.* 8 192 directions) dans le cadre du rendu par points.

QSplat [RL00] introduit une segmentation de l'espace des directions basée sur le cube. Les normales quantifiées sont obtenues en échantillonnant une grille 52×52 sur chacune des six faces, combinée à une fonction de déformation afin d'échantillonner plus uniformément l'espace des directions. Une table d'indexation globale de 16 224 entrées est utilisée pour la décompression, la normale étant encodée sur 14 bits.

Utilisation du cube unitaire

Pour quantifier rapidement les propriétés d'apparence de chaque échantillon géométrique, nous proposons de quantifier le vecteur normal et la couleur en utilisant des méthodes efficaces facilement adaptables pour fonctionner sur GPU. Nous utilisons une approche similaire à la fois pour la normale et la couleur consistant en 2 étapes :

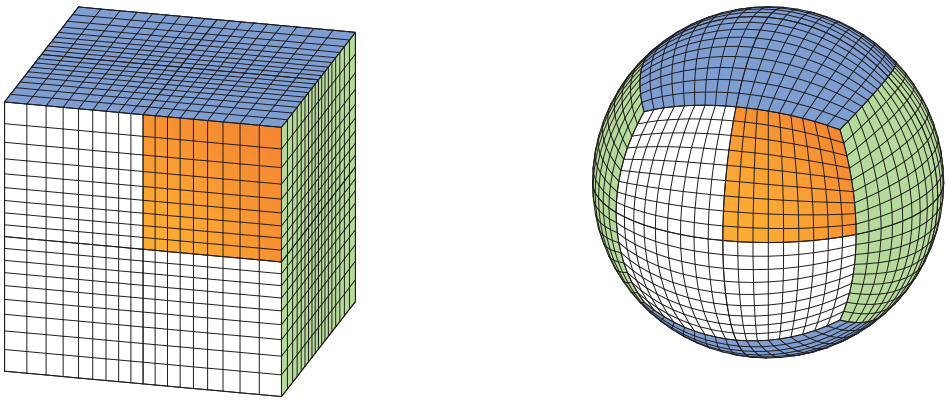
- toutes les valeurs quantifiées sont pré-calculées et stockées dans des structures de données adaptées, respectivement une grille 2D pour les normales et une grille 3D pour les couleurs ;
- les normales et les couleurs à quantifier sont ensuite utilisées comme index des grilles respectives pour accéder aux valeurs quantifiées correspondantes.

Cette méthode permet une quantification des données très rapide car toutes les valeurs quantifiées sont calculées à l'avance et le processus de quantification se résume à un simple accès mémoire. En implémentant les grilles spécifiques comme des textures, le processus est directement utilisable sur le GPU. En outre, l'utilisation d'une plus haute résolution pour les grilles que pour l'espace de quantification permet de générer un échantillonnage non-linéaire au moment de l'accès à la grille.

Finalement, la quantification des normales combinée à la quantification des couleurs permet de quantifier l'apparence simple par sommet pour des objets 3D en un mot de 32 bits. Plus précisément, une normale est quantifiée sur 17 bits et une couleur sur 15 bits de manière à rester compatible avec les formats internes de texture des cartes graphiques pour une précision sur 16 bits

(GL_UNSIGNED_SHORT_5_6_5).

9.3.1 Quantification des normales



(a) (b)

FIG. 9.3: *L'utilisation d'une fonction de déformation sur le cube unitaire (a) permet, une fois projetée sur une sphère unitaire (b), d'obtenir un échantillonnage plus uniforme. La partie orange correspond à la taille de la table d'indexation.*

Puisqu'il n'existe pas de paramétrisation globale uniforme et isotrope sur la surface d'une sphère représentant l'espace des directions, la méthode la plus efficace pour quantifier l'espace des directions est d'échantillonner à l'avance d'une manière aussi uniforme et isotrope que possible, et d'associer à chaque échantillon la représentation flottante correspondante à l'aide d'une table d'indexation. Afin de réduire la taille non négligeable de cette table, le domaine sphérique peut être partitionné en prenant en compte les symétries naturelles qu'il présente. Comme l'a montré Deering [Dee95], seulement $1/24^{ième}$ des échantillons sont nécessaires pour la table d'indexation et le reste des valeurs peut être déduit par symétrie ou par permutation d'index. Notre quantification des normales est une approche basée sur le cube dont chaque face est déformée pour une distribution plus uniforme des normales similairement à [RL00]. Afin de différencier les différentes faces d'un cube aligné selon les axes d'un repère orthonormé, nous avons adopté la convention de nommage et d'indexage utilisé sur les cartes graphiques pour la texture « cubemap » : $\pm X, \pm Y, \pm Z$. En ajustant un cube, aligné selon les axes, de manière à ce que ses sommets soient sur la surface d'une sphère unitaire, nous obtenons six sections sphériques identiques (*cf.* Figure 9.3). De plus, chaque section sphérique comporte deux axes de symétries qui correspondent, dans le repère local de la section, à une symétrie horizontale et une symétrie verticale. Comme le montre la zone orange de la Figure 9.3, la table d'indexation est alors restreinte à un quart de face du cube, ce qui correspond à $1/24^{ième}$ de la sphère dans son intégralité. Chaque élément de la grille 2D contient les coordonnées en représentation flottante des vecteurs unitaires correspondant. Chaque section sphérique associée à une face du cube est échantillonnée par une grille paramétrée de manière quasi-uniforme grâce à une fonction de déformation $f(t) = \tan(4t/\pi)$ sur la paramétrisation (u, v) de la face du cube correspondante :

$$\begin{array}{ll} \forall u \in [-1,1] & u' = f(u) \in [-1,1] \\ \forall v \in [-1,1] & v' = f(v) \in [-1,1] \end{array}$$

Pour chaque échantillon de l'espace des directions quantifiées, nous utilisons un code binaire pour retrouver la représentation flottante à partir de la table d'indexation réduite. L'encodage pour la face du cube et les symétries tient sur 5 bits (*i.e.* les 24 différentes positions de la table d'indexation sur la sphère). Par simplicité la combinaison est faite par des masques :

– la face du cube est encodée sur trois bits ;
– les deux symétries axiales sur un bit chacune.

Une fois que l'en-tête est encodé, tous les bits restants sont utilisés pour définir un index linéaire dans la table d'indexation. Plus précisément, pour une résolution k^2, chaque couple (i, j) sur le quart de face correspond à l'index $i \times k + j$. Les coordonnées i et j sont respectivement encodés sur 6 bits, résultant en une table d'indexation de 4096 entrées et 98 304 normales différentes, ce qui est très proche des 100 000 recommandés par Deering [Dee95].

9.3.2 Déquantification des normales

```
fonction Décodage(index_normale_quantifiée)
    Décoder index, face, les symétries SH et SV
    Retrouver x,y,z à partir de xyzLookUp[index]
    Si (SV) alors z = −z
    Si (SH) alors y = −y
    Selon (face)
        Si '±X' : retourner (±x, y, z)
        Si '±Y' : retourner (z, ±x, y)
        Si '±Z' : retourner (z, y, ±x)
```

Algorithme 2 : Déquantification

Le processus de *déquantification* (*i.e.* reconstruction d'une normale $\vec{n} \in \mathbb{R}^3$) consiste au maximum à un accès à la table d'indexation suivi de trois inversions et deux permutations entre les coordonnées x, y, z de la normale (*cf.* Algorithme 2). Premièrement, l'index est décodé puis utilisé pour retrouver la représentation flottante de la normale dans la table d'indexation. Deuxièmement, la face et les deux symétries sont décodées puis la normale est transformée pour correspondre au quart de face correspondante. Pour atténuer le crénelage que la quantification peut introduire, on utilise une technique de « tremblement local » (en anglais, *i.e. jittering*) pour perturber la normale aléatoirement : notre représentation étant quasi-uniforme, le pas du bruit ajouté doit être maintenu inférieur au pas minimum de la quantification.

9.3.3 Quantification/déquantification des couleurs

Notre approche de quantification pour la couleur utilise une grille 3D où chaque voxel encode une couleur quantifiée. Les composantes couleurs flottantes sont utilisées comme coordonnées dans ce volume pour accéder à la valeur quantifiée correspondante. Chaque voxel peut encoder une couleur sur 15 bits avec respectivement 5 bits pour chaque composante. La quantification de la couleur a été volontairement restreinte à 15 bits pour être combinée facilement avec la quantification des normales sur 17 bits. Ainsi, la quantification de l'apparence tient alors sur 2 mots de 16 bits, comme le montre la Figure 9.4. Comme pour la quantification des normales, si la résolution de la grille de couleur est augmentée, plusieurs voxels correspondraient à une couleur unique. Cette propriété permet la construction d'une grille de couleur avec un espace de couleur alternatif au lieu d'une simple quantification linéaire en RVB.

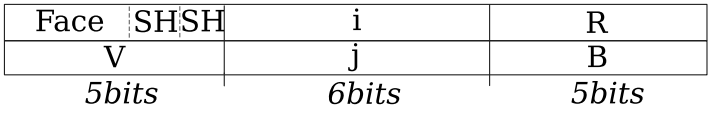


FIG. 9.4: *Les attributs de l'apparence sont stockés sur* 32 *bits. La normale est encodée sur* 17 *bits (5 bits pour la face et les symétries, 6 bits pour i, 6 bits pour j), et la couleur est encodée sur* 15 *bits (5 bits pour chaque composante : rouge, vert, bleu).*

9.4 Implémentation

Nous avons expérimenté notre compression d'apparence sur GPU dans le contexte de la compression d'attributs (normale et couleur) par sommets (resp. point) d'un maillage polygonal (resp. nuage de points non uniforme). L'encodage est conforme au format de texture `RGB_5_6_5` utilisé pour l'instant dans la texture cube pour l'encodage des normales. La conformité à ce format peut permettre aussi, dans le cas de maillage paramétrisé, de spécifier des attributs d'apparence dans des textures. L'algorithme de déquantification nécessite quant à lui des adaptations spécifiques pour fonctionner sur GPU à causes des limitations des unités de calculs de certaines cartes graphiques.

9.4.1 Encodage

La principale contribution de notre méthode est de permettre un encodage en flux tendu (sans stocker à aucun moment la totalité de l'objet en mémoire). La boucle de traitement se scinde ainsi en 3 étapes :

1. Lecture d'un ensemble de taille fixe dans un tampon de sommets sur l'entrée du flux.
2. Rendu sur GPU effectuant la quantification sur l'unité de traitement des fragments.
3. Écriture du résultat sur le flux de sortie.

Le processus de quantification étant ramené à un simple accès d'une texture cube sur la carte graphique et cette opération étant plus efficace dans l'unité de calcul par fragment, les données à encoder doivent donc être transmises via des textures au GPU. De même, le résultat de la quantification du tampon de normales est en pratique contenu dans le tampon de mémoire vidéo (« framebuffer ») courant du GPU.

La quantification consiste alors au rendu d'un unique quadrilatère texturé à la même résolution que la texture utilisée pour les données à encoder. De cette manière, nous nous assurons d'une bijection entre chaque texel de données (normale à quantifier) et chaque pixel en sortie (normale quantifiée). A chaque nouveau tampon de normales lu, on rend une image et le résultat est directement écrit sur le flux de sortie. Le système fonctionne donc à mémoire constante, ce qui permet de traiter plusieurs objets de grande taille en parallèle sur un même serveur : le goulot d'étranglement apparaît désormais sur les entrées/sorties, le CPU étant complètement libéré de toute tâche de quantification.

9.4.2 Optimisation du transfert pour les entrées/sorties

Afin de réduire les temps de transferts des données CPU/GPU, nous avons développé une solution nécessitant une seule passe de rendu. Pour un tampon de taille $B \times B$, les couleurs et les normales à quantifier sont transférées à l'aide d'une texture de résolution $2B \times B$ (*cf.* Figure 9.5). Les valeurs des couleurs et des normales sont stockées de manière entrelacée dans cette texture, et la sélection du processus de quantification est déterminée dans l'unité de traitement des fragments (en utilisant alternativement quantification de couleur et quantification de normale). À la fin du processus, le GPU génère un framebuffer de taille $2B \times B$ qui contient les normales et les couleurs quantifiées et toujours entrelacées. Notons que, avec cette approche purement locale et segmentée, plusieurs objets peuvent être quantifiés en parallèle. Une solution simple étant de combiner les données de différents objets dans un seul tampon. Une autre approche, comme les schémas de traitements séquentiels, serait de mettre en tampon alternativement chaque objet pour chaque tampon de rendu. Avec ces solutions, de multiples requêtes pourraient facilement être supportées.

9.4.3 Décodage

Le décodage de l'apparence s'effectue du côté du client. Les normales sont décodées sur le CPU avec l'algorithme décrit précédemment (*cf.* Algorithme 2). Le décodage des couleurs est encore plus simple puisqu'il s'agit de passer d'un encodage de 5 bits à 8 bits, en effectuant un décalage à gauche de 3 bits, pour chaque composante RVB.

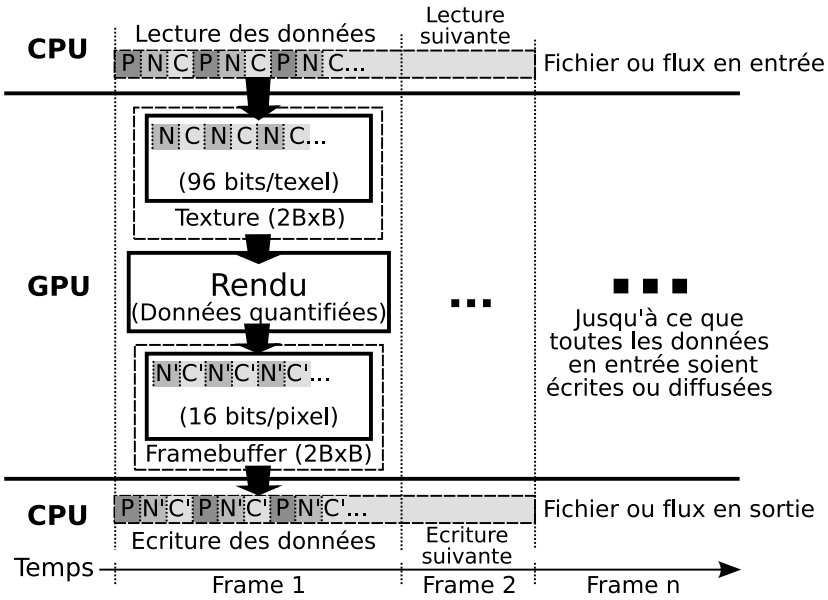


FIG. 9.5: *Pour chaque frame rendue sur le GPU, les couleurs et normales flottantes (resp. N et C, P étant la position) sont lues à partir d'un fichier ou d'un flux en entrée et envoyées entrelacées sur le GPU dans une texture à précision flottante. Une fois le rendu effectué, les normales et les couleurs quantifiées (resp. N′ et C′) sont récupérées depuis le framebuffer avant d'être écrites dans un fichier ou flux de sortie.*

En utilisant des adaptations spécifiques, cet algorithme peut même être implémenté sur l'unité de traitement des sommets du GPU. Sur la carte graphique utilisée durant nos tests, cette unité ne permet pas de stocker la table d'indexation, en tant que tableau statique, dans ses registres en raison de sa taille trop importante. Nous utilisons donc une table d'indexation de 8×8 (64×64 à l'origine pour des normales quantifiées sur 17 bits) et la normale est reconstruite en interpolant linéairement les 4 valeurs les plus proches trouvées dans cette table d'indexation réduite. La normale est ensuite normalisée afin de réduire les artefacts de la quantification. Pour le décodage de l'indice de la face du cube et les symétries, nous utilisons une table contenant les 24 combinaisons encodées dans l'en-tête de 5 bits. Chaque valeur de cette table est un vecteur stockant séparément la face et les deux symétries.

Certains GPUs ne supportent pas les opérateurs binaires ou la fonction modulo nativement. Ces opérateurs peuvent être émulés en utilisant la méthode décrite par Purnomo *et al.* [PBCK05] qui montrent comment utiliser les fonctions d'extraction de parties entières et fractionnelles pour simuler les opérateurs de masquage de bits présents sur les cartes graphiques de dernière génération [NVI06].

Pour réduire un peu plus les artéfacts de quantification, du *jittering* peut être ajouté sur la normale et la couleur. Une manière simple pour obtenir un *jittering* des couleurs est de perturber les coefficients de mélange pendant le calcul de l'éclairement de Gouraud. Pour un éclairement de Phong, ce procédé perturbe aussi les normales. Mettre en place le *jittering* pour la déquantification sur CPU est très facile, mais moins trivial sur le GPU car celui-ci ne dispose pas encore de fonctions génératrices de nombres aléatoires.

En pratique, notre système utilise seulement la déquantification sur CPU car l'implémentation de la déquantification sur GPU n'est pas assez performante pour un rendu interactif à cause des limitations actuelles des GPUs. Du côté du client, le principale avantage de la quantification est le gain de stockage pour les données 3D dans leur forme quantifiée.

9.5 Expérimentations

Dans un premier temps, afin d'évaluer les bénéfices de notre système, nous avons comparé les temps de quantification entre une implémentation CPU et une implémentation GPU. Dans un deuxième temps, nous comparons les temps de transfert client/serveur requis pour des données d'origines et des données quantifiées, en incluant pour les données quantifiées les temps de quantification côté serveur et les temps de reconstruction côté client. Finalement, nous comparons visuellement la différence entre le rendu d'un modèle avec son apparence d'origine et le rendu de ce même modèle mais avec son apparence quantifiée. La station de travail jouant le rôle de serveur est un Pentium 4 à 3Ghz avec 1Go de RAM et une carte graphique Quadro 4400 tournant sous windows XP.

9.5.1 Temps de quantification

Taille (ko)	64	128	256	512	640
Temps 1 (ms)	747	781	770	764	764
Temps 2 (ms)	731	622	564	453	419
Temps 3 (ms)	371	467	592	610	579

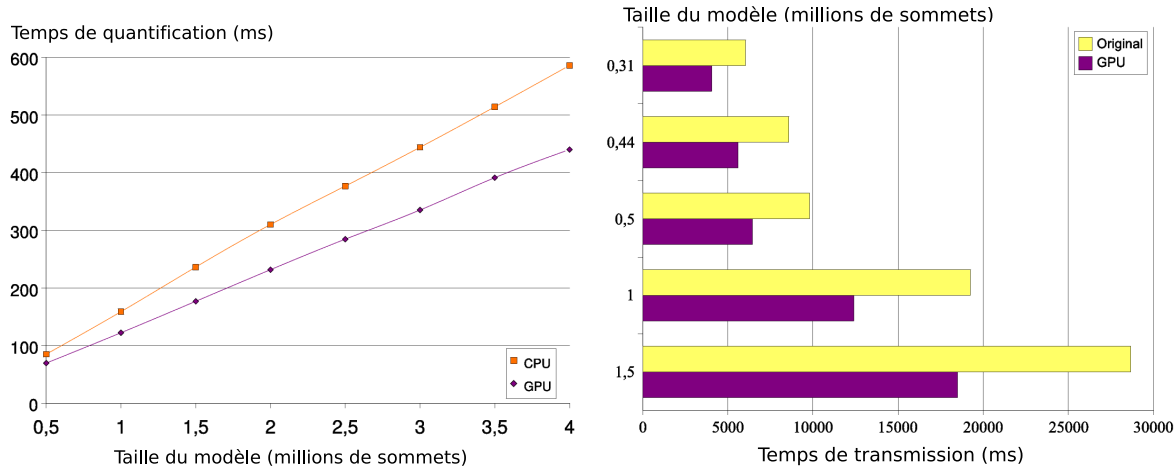
TAB. 9.1: *Impacte de la taille du tampon sur la vitesse de quantification. Les temps de la première ligne sont obtenus pour une implémentation CPU complète avec un Pentium 4 3.4GHz. Les temps de la seconde ligne sont obtenus pour une implémentation GPU sur la même station de travail avec une Quadro FX3400 sur un port PCI Express x16. Pour finir, les temps de la troisième ligne sont obtenus avec une implémentation GPU sur un PC portable muni d'un Pentium-M 2.26 GHz et une Geforce 7800GTX sur un port PCI Express x16.*

Notre système fonctionne hors-mémoire en procédant au traitement par morceau des flux en entrée. En accord avec l'implémentation, chaque tampon est alors directement traité sur le CPU ou transféré sur le GPU. Le Tableau 9.1 illustre l'impact de la taille choisie pour le tampon sur la vitesse de quantification. Visiblement, la taille du tampon a très peu d'influence sur les performances de la quantification purement CPU : quelque soit la taille du tampon, la quantification prend environ 765 millisecondes (ms) sur notre machine de référence. Par contre, pour l'implémentation sur GPU, la taille optimale du tampon dépend grandement de la configuration du serveur et elle doit être choisie judicieusement en fonction de ses capacités. Ce point était prévisible car il est bien connu que, dans le cadre d'une utilisation du GPU pour des calculs d'ordre généraux, les transferts mémoires entre la GRAM et la RAM représentent dans les deux sens le goulot d'étranglement majeur.

Nous avons ensuite analysé la vitesse de quantification pour des modèles présentant une taille de plus en plus grande, avec et sans l'utilisation du GPU. Comme le montre la Figure 9.6-(a), l'utilisation du GPU améliore en générale de 20% à 25% la vitesse de quantification. L'utilisation du GPU ne réduit donc pas seulement la charge du CPU, mais elle accélère significativement le processus de quantification.

9.5.2 Performance de la diffusion des données

Nous avons testé notre système client/serveur dans un contexte de mobilité, en utilisant le réseau WIFI à 54 Mb/s du laboratoire. Ce réseau n'étant pas dédié à nos expérimentations et afin de réduire le biais du à l'encombrement du réseau, nous avons pris la moyenne des temps de transmissions de plusieurs mesures répétées. Pour ces tests, le client était un PC portable avec un Pentium-M 2.26 GHz. Grâce au processus rapide de quantification, le temps de transmission est largement réduit (*cf.* Figure 9.6-(b) - à peu près 35% plus rapide). Avec l'apparence quantifiée, le temps de transmission inclut le temps de quantification sur le serveur et le temps de reconstruction du client.



(a) (b)

FIG. 9.6: *En (a), une comparaison entre le CPU et le GPU des temps de quantification de l'apparence de modèles à base de points. En (b), les temps moyens requis pour la transmission d'un modèle (original et quantifié avec le GPU) en fonction de sa taille. Pour le modèle quantifié, les résultats incluent les temps de quantification et de reconstruction. Les mesures ont été effectuées pour un réseau WIFI avec une bande passante de 54Mb/s.*

Nous avons aussi mesuré l'effet sur le temps de transmission en utilisant tantôt la quantification par GPU, tantôt celle par CPU. Puisque la quantification est plus rapide sur le GPU (*cf.* Figure 9.6-(a)), nous attendions un bénéfice correspondant sur le temps total de transmission. Malheureusement ce n'est pas le cas quand la quantification sur GPU est utilisée. La transmission est plus lente d'environ 2% en moyenne et en décroit avec l'augmentation de la taille des objets. Après plusieurs investigations, nous pensons que, même si la charge du CPU est réduite, il y a plus de transferts de données sur le serveur (transferts du disque dur, entre la RAM et la GRAM et vers l'interface réseau au lieu de simples transferts du disque dur vers l'interface réseau en passant par le CPU). Avec les futurs processeurs multicœurs, combinant GPU et CPU, la surcharge induite par ces transferts internes est susceptible de disparaître.

9.5.3 Erreur de quantification

La Figure 9.7 (page 96) montre un modèle de taille moyenne (8 millions de polygones) rendu avec un matériau spéculaire et plusieurs sources lumineuses. L'erreur RVB donnée est multipliée par 100. On observe seulement une petite perte entre la perception quantifiée et l'originale. En mesurant l'erreur perceptuelle *Delta E* (CIE 1976) sur la même image, on obtient une erreur moyenne de 8.8 10^{-2}% avec une erreur maximale de 9.7% (erreur divisée par le maximum possible quand les composantes couleurs varient entre 0 et 1 dans l'espace RVB).

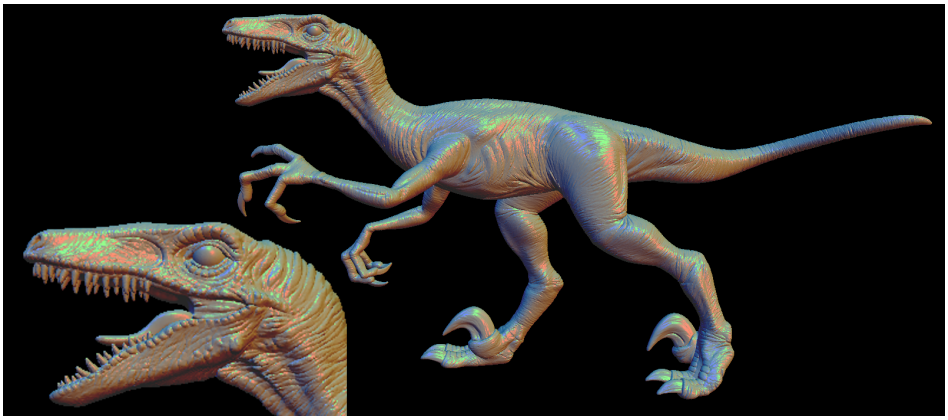
9.6 Conclusion

Nous avons développé un schéma de quantification pour les propriétés d'apparence conçu pour des applications client/serveur de diffusion de contenu 3D. Dans le but de réduire la charge du CPU autant que possible, le processus de quantification fonctionne complètement sur GPU, via un procédé par flux. Notre méthode hors-mémoire est totalement insensible à la taille des modèles 3D (une occupation mémoire fixée est garantie). Nous avons introduit une méthode rapide combinant les méthodes basées sur un octaèdre et basées sur un cube pour représenter l'espace des directions. Notre solution améliore significativement les capacités de quantification pour le serveur et autorise ainsi un plus grand nombre

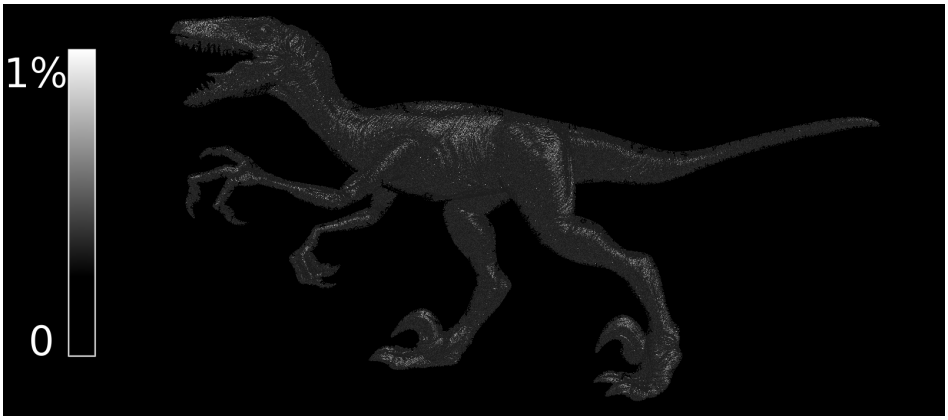
de requêtes simultanées pour la visualisation de grosses bases de données 3D sur des clients hétérogènes.

Pour de futurs améliorations, nous envisageons de déporter au maximum les traitements géométriques sur le GPU du côté du serveur, comme des méthodes de génération de niveaux de détails et de quantification de la géométrie à la volée. Pour améliorer encore l'efficacité de notre système, nous prévoyons une meilleure gestion des transferts de mémoire interne du côté du serveur. Pour limiter le coût de la reconstruction et la taille mémoire requise du côté du client, nous avons commencé des recherches sur des modèles d'illuminations définis dans l'espace quantifié des normales.

Modèle original - 288 bits/sommet
96 bits pour position, normale et couleur



Normales et couleurs quantifiées sur GPU - 128 bits/sommet
96 bits pour la position et 32 bits pour la normale et la couleur



Erreur colorimétrique - Image de différence x100

FIG. 9.7: *Erreur de quantification pour un éclairement de Phong. Trois sources lumineuses ave un matériau spéculaire.*

Conclusion générale

Au cours de cette thèse, nous nous sommes intéressés à la représentation de l'apparence à différents niveaux d'échelles. Dans cette optique, nous avons d'abord étudiés les « Bidirectional Texture Functions » (BTFs) qui permettent d'encoder à une certaine échelle des effets dues à des échelles inférieures. Dans un deuxième temps, nous nous sommes intéressés aux méthodes d'encodage de l'apparence pour des applications de rendu en temps-réel sur cartes graphiques.

Résumé des contributions

Dans la première partie de ce document, nous avons présenté les différentes approches existantes qui permettent de définir une apparence à diverses échelles pour les objets 3D dans la synthèse d'images en temps-réel. Parmi toutes ces méthodes, les « Bidirectional Texture Functions » (BTF) se distinguent comme l'une des meilleures approches à l'heure actuelle de part le réalisme qu'elles apportent tout en demandant un faible coût de calcul pour les applications en temps-réel sur cartes graphiques. Ce réalisme est principalement dû au fait que, comme nous l'avons vu dans le Chapitre 2, les BTFs sont basées sur la mesure en image des variations de l'apparence d'échantillons de matériaux réels selon différents points d'observation et les différentes positions d'une source lumineuse. De plus, elles permettent de représenter à une échelle donnée des phénomènes dus à des échelles inférieures. Bien qu'elles offrent l'avantage d'un bon compromis réalisme/coût de calcul, les données d'une BTF ont l'inconvénient d'occuper beaucoup d'espace en mémoire au point de nécessiter des méthodes de compression adaptées pour une utilisation en temps-réel ou encore la constitution d'une base de matériaux. En analysant les différentes approches de compression au Chapitre 3, nous en avons conclu que les méthodes de compression de texels étaient les mieux adaptées pour le rendu sur GPU, et plus particulièrement les méthodes qui utilisent des modèles analytiques ou paramétriques afin de représenter de manière continue les dimensions angulaires des BTFs. Après avoir soulevé les problèmes induits par les effets de la parallaxe pour ces méthodes, nous avons proposé, au Chapitre 4, une étude approfondie grâce à notre logiciel *BTFInspect* sur l'origine de ce phénomène et ses effets dans la cohérence des données par texel. Au cours de cette étude, nous avons démontré que la parallaxe, induite par la visibilité de la méso-structure, joue un rôle majeur dans l'incohérence des données par texel pour une BTF et qu'en son absence les données deviendraient beaucoup plus faciles à compresser.

Dans la deuxième partie du document, nous avons proposé une nouvelle représentation pour les BTFs, intitulée « Flat Bidirectional Texture Function » (Flat-BTF). En représentant séparément l'information de parallaxe et l'information de réflectance, notre nouvelle représentation améliore fortement la cohérence des données de réflectance par texel et favorise ainsi leur compression. Dans la définition des Flat-BTFs, nous avons montré comment la méso-structure des BTFs pouvait être utilisée pour représenter séparément l'information de parallaxe et l'information de réflectance. Pour valider notre approche, nous avons présenté une première implémentation avec des BTFs de synthèse afin de contrôler toutes les étapes de sa mise en œuvre et obtenir un jeu de données équivalentes dans notre

représentation et la représentation classique. En réalisant une étude statistique puis visuelle sur les Flat-BTFs et BTFs générées, nous avons pu clairement démontré le gain de cohérence apporté par texel avec la représentation Flat-BTF par rapport à la représentation classique. Enfin, nous avons montré que l'erreur visuelle introduite lors de la conversion d'une Flat-BTF en BTF, était suffisamment faible et peu discernable. En résumé, nous avons démontré que notre nouvelle représentation Flat-BTF est plus favorable aux méthodes de compression par texel et que la méso-structure des BTFs qui constitue un élément fondamentale, doit être prise en compte dans la représentation des BTFs.

En troisième et dernière partie, nous avons présenté deux méthodes d'encodage de l'apparence pour un rendu efficace sur GPU. Dans un premier temps, nous avons proposé une première implémentation des Flat-BTFs dans un contexte d'application de rendu d'objets 3D en temps-réel afin de valider son utilisation sur GPU. Après avoir étudié les différentes stratégies d'utilisation des BTFs comme texture, nous avons vu que l'utilisation des Flat-BTFs était relativement similaire à celle des BTFs classiques. En utilisant la fonction d'indirection (ou VDIM) qui encode la parallaxe, avec des données de réflectance simple (une carte de normales et une texture diffuse), nous avons pu réduire le surcoût engendré lors de la reconstruction en représentation classique tout en garantissant une bonne restitution des effets de parallaxe sur les données de réflectance. Dans un deuxième temps, nous sommes allé plus loin dans l'encodage en proposant une quantification des normales et des couleurs qui sont généralement utilisées pour définir une apparence simple des objets 3D. Cette approche a été présentée dans un contexte de diffusion de données 3D où notre encodage adapté au GPU a permis d'accélérer la quantification des données d'apparence à la volée en utilisant la carte graphique du coté serveur.

Travaux futurs et perspectives

Acquisition de BTFs

La représentation Flat-BTF met en évidence l'importance de la méso-structure pour les BTFs et nous pensons que les dispositifs d'acquisition à venir devraient mesurer aussi la méso-structure en plus de l'apparence. Wang *et al.* [WTS+05] ont déjà proposé un premier dispositif qui mesure une fonction de hauteur par direction de vue à l'aide d'un laser. D'une certaine manière, ces travaux illustrent aussi l'importance de la méso-structure en rajoutant le rendu des silhouettes aux BTFs qui leur font défaut dans la représentation traditionnelle. Deux approches sont à envisager pour obtenir la méso-structure des BTFs lors de leur acquisition. On peut d'une part augmenter la densité des échantillons des dimensions angulaires et surtout l'uniformité et la régularité spatiale afin de mieux prendre en compte les effets de la parallaxe et les déformations de perspectives. Ainsi, les méthodes de reconstruction de la méso-structure seront plus efficaces et permettront d'en obtenir une représentation précise. D'autre part, les scanners laser 3D d'aujourd'hui permettent de mesurer la géométrie avec une grande précision et nous envisageons de développer un dispositif d'acquisition qui permettra de coupler la mesure de l'apparence par un dôme de plusieurs capteurs et la mesure de la méso-structure par un scanner 3D. Un tel dispositif permettrait alors de fournir des BTFs directement dans la représentation Flat-BTF.

Nous nous intéressons aussi à la miniaturisation et la portabilité des dispositifs d'acquisition. Ceux-ci restent à l'heure actuelle difficilement exploitables en extérieur ou en dehors d'un laboratoire. Le dispositif récent, miniature et novateur de Moshe *et al.* [BEWW+08] à base de diodes électroluminescentes nous conforte dans cette perspective d'évolution. De notre côté, nous avons imaginé un dispositif à base de fibres optiques où chaque fibre servirait soit à éclairer soit à mesurer l'apparence d'un point d'une surface pour une direction donnée. Ce dispositif léger et portatif ne serait au final pas plus gros qu'une boîte à chaussures et permettrait de mesurer des matériaux ou l'apparence de surfaces disponibles uniquement en extérieur, comme par exemple sur un site archéologique. Au niveau de la réalisation, nous imaginons, placé au dessus de l'échantillon à mesurer, un dôme opaque d'une vingtaine de centimètres de diamètre et qui serait percé sur toute sa surface au niveau des différentes directions à mesurer. De multiples fibres optiques seraient alors connectées par paires au niveau de chaque trou afin que l'une éclaire et que l'autre mesure l'apparence de l'échantillon. À l'autre extrémité des fibres, celles-ci seraient organisées selon une certaine paramétrisation pour former deux matrices

2D différentes : la première pour les fibres éclairantes et la seconde pour les fibres de mesure. On pourrait alors acquérir très rapidement la BRDF des matériaux en faisant correspondre un petit écran LCD pour la matrice éclairante et un capteur CCD pour la matrice de capture. L'écran LCD permettrait de simuler des environnements lumineux complexes ou bien des bases de fonctions telles que des ondelettes sphériques [GAHO07]. Pour mesurer des SVBRDFs, il suffirait de déplacer spatialement le dispositif sur la surface d'un matériau.

Compression de l'apparence

Les modèles analytiques de BRDFs existants représentent de manière continue les dimensions angulaires, ce qui est très avantageux pour le rendu sur GPUs. Seulement, ces modèles ont été conçus spécifiquement pour les phénomènes lumineux inhérents aux BRDFs. Par conséquent, ils sont donc mal adaptés aux phénomènes d'éclairement indirects, tels que les ombres propres. Le fait d'y adjoindre des modifications pour prendre en compte ces phénomènes donne de faibles résultats comme en attestent les travaux déjà proposés [DLHS01, MCC$^+$04, FP05]. Nous travaillons donc actuellement sur le développement d'un nouveau modèle de BRDF plus polyvalent qui peut se composer d'un ou plusieurs lobes de différentes types (*i.e.* diffus, spéculaire, isotrope ou anisotrope). Ce nouveau modèle est aussi développé de manière à ce qu'une approximation dans ce modèle puisse garantir un minimum global sur l'erreur. Ainsi, nous estimons que seulement une dizaine à une vingtaine de textures pourraient suffire à bien représenter les données d'une Flat-BTF (à la fois la fonction Flat-BTF et la fonction VDIM).

En terme de représentation de l'espace des directions pour l'encodage de normales, nous avons testé une alternative au cube que nous avons utilisé dans notre méthode d'encodage de l'apparence. Les quatre faces d'un tétraèdre ont de bonnes propriétés pour représenter l'espace des directions et notamment le nombre de ses faces qui est plus compatible avec un codage binaire que le nombre de faces d'un cube. Avec cette nouvelle représentation, la plage des bits encodant une direction serait pleinement utilisée. Nous travaillons actuellement sur une fonction paramétrique afin d'échantillonner uniformément l'espace des directions sur une face du tétraèdre.

Rendu

Dans le rendu de BTFs, l'interpolation entre les différents échantillons clefs introduit des incohérences spatiales pour les phénomènes qui dépendent de la visibilité de la méso-structure comme les ombres propres ou les portions visibles de la méso-structure. Si l'on considère par exemple une position sur la méso-structure qui devient ombragée entre deux directions lumineuses clefs, l'interpolation linéaire donnera des couleurs intermédiaires entre la couleur éclairée et la couleur ombragée, alors qu'en réalité il n'y a pas de transition entre les deux états. Une solution serait d'utiliser une fonction de visibilité par texel qui représenterait la dimension angulaire de manière continue et qui serait déterminée précisément depuis la méso-structure et les directions de lumière incidente. Le nouveau modèle de BRDF sur lequel nous travaillons pourrait facilement, grâce à sa polyvalence, représenter cette fonction binaire de visibilité.

Lorsque les normales sont encodées, il est nécessaire de les décoder avant toute opération. Comme ce décodage peut être coûteux, nous pensons à développer et définir une algèbre spécifique aux directions quantifiées. Un des grands avantages d'un espace des directions quantifiés est que les normales ont constamment une norme unitaire. Nous réfléchissons actuellement à des méthodes permettant des opérations simples sur des directions encodées et plus particulièrement, la somme de deux vecteurs et le produit scalaire afin de définir un modèle d'éclairement discret.

Table des figures

1.1 À gauche, un rendu réaliste en temps-réel pour une scène représentant l'intérieur d'une maison avec ses meubles. L'apparence réaliste des objets est ici définie par les SVBRDFs de McAllister *et al.* [MLH02] (introduites dans la Section 1.3). À droite, un objet en forme de théière issu des travaux de Wang *et al.* [WTS$^+$05] et dont l'apparence de la surface reproduit un mur de briques à l'aide de BTFs (définies dans la Section 1.5). . . 7

1.2 Les différentes échelles de géométrie entrant en considération dans l'apparence de la surface d'un matériau avec le classement des principales méthodes existantes en fonction des échelles de géométrie considérées. 9

1.3 Configuration géométrique pour la réflexion dans la direction sortante $\vec{\omega_o}$ d'un rayon incident $\vec{\omega_i}$ au point x d'une surface. L'ensemble des directions Ω est représenté par l'hémisphère supérieur au point x et aligné avec la normale au point $\vec{n}$ (schéma de Müller *et al.* [MMS$^+$04]). 10

1.4 Les différents types de réflexion en un point d'une surface. 10

1.5 Les différents paramètres entrant en jeu dans l'interaction de la lumière avec la matière. Soit un photon de longueur d'onde λ_i frappant la surface d'un matériau au temps t_i au point x_i. La direction incidente du photon est définie dans le repère local à la surface au point x_i par le couple d'angles (θ_i, ϕ_i). Le photon traverse la surface et ressort à la position x_r, au temps t_r avec la longueur d'onde λ_r dans la direction (θ_r, ϕ_r). 11

1.6 Images obtenues par Müller *et al.* [MMS$^+$04] utilisant des BTFs pour définir l'apparence des objets. En (a), des BTFs définissent les différentes matériaux constituant l'intérieur d'une Mercedes classe C comme le plastique, le cuir ou le bois. En (b), une BTF est utilisée pour définir l'apparence complexe de la surface d'un tricot. 13

1.7 Une BTF : aperçu d'un échantillon de surface pour quatre différents points de vue (numérotés 1, 2, 3 et 4) et différentes directions de lumière incidente. L'angle θ représente l'angle que fait la direction avec l'axe Z et l'angle ϕ représente l'angle que fait la direction avec l'axe Y dans le plan XY. 14

1.8 Les différents domaines impliqués dans l'utilisation des BTFs. L'**acquisition** et les **BTFs synthétiques** sont les deux moyens d'obtenir des BTFs. La **compression** s'occupe de réduire la taille des données qui peuvent ensuite être exploitée par des méthodes de **rendu**. La **synthèse** de textures peut intervenir avant ou après compression des données pour créer des BTFs sans motifs de répétition à partir des BTFs d'origines. L'**édition** des BTFs est un domaine perpendiculaire à tous ces domaines car elle peut intervenir avant ou après chacun des autres domaines. 15

2.1 (a) Schéma du système d'acquisition inspiré des systèmes de mesure de BRDFs [SSK03]. La caméra se déplace sur un rail alors que la source lumineuse est fixée. Le support du matériau peut pivoter sur deux axes. (b) Photographie du système d'acquisition de l'université de Bonn [SSK03]. On distingue au premier plan le rail circulaire supportant la caméra située en arrière plan. Le bras robotisé recevant les échantillons se trouve au centre et la source lumineuse est au bout du rail en arrière plan. 18
2.2 Le système de Wang *et al.* [WTS+05] avec ses deux supports en arc dont celui supportant les lampes qui est mobile. Sur ce même arc mobile est fixé le scanner laser permettant l'acquisition de la méso-structure sous la forme de plusieurs fonctions de hauteur. Le support de l'échantillon est rotatif pour apporter une dimension supplémentaire aux différentes prises de vue. 19
2.3 Système d'acquisition à composants fixes de l'université de Bonn [MBK05]. Le matériau à mesurer est placé sur le support visible en bas à droite. Quand une lampe s'allume, tous les appareils photos prennent un cliché de l'échantillon. Lorsque toutes les lampes ont été allumées, on a bien une BTF avec N photos prises de différents points de vue pour M sources lumineuses. 20
2.4 À gauche, image acquise pour une vue en perspective ($\theta = 60, \phi = 144$) d'un échantillon de tricot sous lumière directionnelle ($\theta = 60, \phi = 18$). À droite, la même image après détourage et correction de perspective ([MMS+04]). 21
2.5 La génération de BTFs par Suykens *et al.* [SvLD03]. En (a), la méso-structure modélisée pour représenter les mailles d'un tissu, la zone rouge symbolise le morceau d'échantillon sélectionné et raccordable pour générer les BTFs. En (b), la visualisation d'une ABRDF en un point donné et pour un rayon incident (en bleu) qui résulte de la combinaison des propriétés de réflectance et des effets induits par la méso-structure. En (c), un exemple de rendu de la BTF synthétique correspondante plaquée sur un tore. 22

3.1 (a) Vue des BTFs comme des textures dépendantes des directions de vue et des directions de lumière. Une BRDF apparente peut correspondre à chaque texel. (b) Vue des BTFs comme un signal à 6 dimensions. 25
3.2 Comparaison visuelle de la préservation de l'apparence de Müller *et al.* [MMS+04] pour une sélection des différentes méthodes présentées. Sur la ligne du haut, une ABRDF représentée dans une image (*cf.* Section 4.3.1 pour plus de détails) de la BTF d'origine et les ABRDFs reconstruites et sur la ligne du bas, les images inversées de la différence avec l'ABRDF originale. De gauche à droite, une ABRDF originale issue d'une BTF de béton aggloméré, en (a) les lobes pondérés de Lafortune de Daubert *et al.* [DLHS01], en (b) les champs de réflectance par vue de Meseth *et al.* [MMK03a], en (c) la méthode de factorisation chaînée de Suykens *et al.* [SvLD03] avec 4 facteurs, en (d) la compression par ACP par vue de Satler *et al.* [SSK03] avec 8 termes et enfin en (e) la méthode de réduction par partition de Müller *et al.* [MMK03b] avec 32 clusters et 8 termes. 31

4.1 Un aperçu de l'interface et des fonctionnalités de notre logiciel BTFInspect. 33
4.2 Les directions sont toujours exprimées selon un hémisphère unitaire et alignées sur l'axe des Z positifs dans un repère orthonormé. Il existe alors une bijection entre la coordonnée sphérique (θ,ϕ) d'un point P et sa projection Q dans le plan XY de la base de l'hémisphère et de coordonnée cartésienne (x,y). 35
4.3 Estimation de l'occultation ambiante pour différents angles d'élévation (θ) de la BTF de tricot. 37
4.4 Observation de la parallaxe pour différents angles d'élévation (θ) dans les BTFs de Bonn [Bon]. La direction lumineuse est fixe ($\theta = 0, \phi = 0$). En haut, images de la BTF de tricot présentant un relief important et donc une parallaxe importante. En bas, images de la BTF de papier peint présentant une méso-structure quasi-plane, le phénomène de parallaxe n'est pas perceptible dans les images. 38

4.5 Parcours en spirale des échantillons de l'espace des directions (schéma de Filip et Haindl [FH04]. 39
4.6 Sur une ligne, une image d'une BTF et deux images d'ABRDFs pour différents texels. En haut, une BTF issue d'un morceau de granite gravé et en bas, une BTF d'un échantillon de tissage de fibre polyacrylique marron. Les axes, intitulés V et L, dans la deuxième colonne, sont respectivement les directions de vue et les directions lumineuse. Dans la troisième colonne, les échelles indiquent les valeurs fixes pour l'angle ϕ, tandis que l'angle θ varie sur 360 degrés dans chaque intervalle. 40
4.7 En haut, une BTF issue d'un échantillon de papier peint et en bas, d'un morceau de béton aggloméré. Les axes, intitulés V et L, dans la deuxième colonne, sont respectivement les directions de vue et les directions lumineuse. Dans la troisième colonne, les échelles indiquent les valeurs fixes pour l'angle ϕ, tandis que l'angle θ varie sur 360 degrés dans chaque intervalle. 41
4.8 (a) La composante rouge pour une tranche d'une ABRDF au pôle (vue : $\phi = 0$, $\theta = 0$) extraite de la BTF de granite *Impalla*. On remarque plusieurs pics pour le texel sélectionné. (b) La superposition d'un lobe obtenu pour un texel différent (en bleu) et d'un lobe obtenu représentation l'approximation polynomiale par un PTM (en orange). 42
4.9 (a) Une image de la BTF synthétique *Isba* présentant des fortes spécularités pour la vue [$\theta = 30$, $\phi = 60$] et la direction lumineuse [$\theta = 0$, $\phi = 0$] (b) L'image reconstruite après compression de la BTF par la méthode des PTMs, les spécularités ne sont pas représentées et les couleurs dans l'ensemble ont été atténuées. 42

5.1 À gauche notre représentation Flat-BTF. Elle est composée d'une donnée de réflectance 6D et d'une fonction ne stockant que les effets de parallaxe et qui est par conséquent 4D. La composition de ces deux fonctions permet de reconstituer une BTF (à droite). 47
5.2 Dans une BTF standard, l'apparence due à la méso-structure est calculée ou mesurée indépendamment pour une ensemble de direction de vue $\vec{v}_0$, $\vec{v}_1$ et $\vec{v}_2$. La réflectance visible est stockée pour chaque vue mais ces données sont déformées et étirées pour que l'image corresponde à la surface plane de référence (le segment **ab** est beaucoup plus long dans la vue $\vec{v}_1$ que dans la vue $\vec{v}_0$). A l'opposé, dans la représentation Flat-BTF, les échantillons sont stockés en fonction de leur localisation sur la méso-structure. Une carte d'indirection dépendante de la direction de vue et renfermant l'information de parallaxe permet alors de convertir une Flat-BTF dans la représentation standard de BTF. 52
5.3 En haut, les images d'une de nos BTFs synthétiques, appelée « bouton », illustrant l'effet de parallaxe de la représentation standard. En bas, les images de la même BTF, dans la représentation Flat-BTF, on constate l'absence de parallaxe. Les zones floues dans les images correspondent aux parties non visibles de la méso-structure qui ont été remplies en accord avec le voisinage. 53

6.1 Les différentes étapes pour la génération de Flat-BTFs de synthèse. 57
6.2 Les méso-structures reconstruites à partir de l'occlusion ambiante utilisée comme fonction de hauteur pour une BTF de tissage de fibre polyacrylique en (a) (BTF *Pully* de l'université de Bonn) et pour une BTF d'une éponge en (b) (BTF *Sponge* de Madga et Kriegman). 58
6.3 (a) La paramétrisation de la méso-structure du modèle *Isba* (b) Zoom montrant que la surface se replie sous la dalle. Grâce à la paramétrisation globale, notre représentation ne limite pas le type des méso-structures à des fonctions de hauteur comme la plupart des approches existantes (*cf.* Section 5.2). 59
6.4 Exemple de textures diffuses obtenues de manière procédurale. En (a), pour le modèle *Bouton* et en (b), pour le modèle *Isba* qui représente un dallage en marbre sur un fond de granit. 60

6.5 Images géométriques de la méso-structure pour le modèle *Isba* qui représente un dallage en marbre. Respectivement en (a), (b) et (c), l'image des positions 3D (stockage en réel), l'image réels des normales (stockage en réel) et l'image des indices de face (stockage en entier) : on voit apparaître les déformations dues à la paramétrisation, notamment aux niveaux du bord des dalles qui est normalement rectiligne. 62

6.6 État avant et après remplissage des zones (en bleu) correspondant aux texels indéfinis dans les images de la Flat-BTF *Bouton*. 63

6.7 La corrélation entre les directions de vue et les décalages correspondant pour les indirections de la fonction VDIM. Pour chaque texel (i, j) sur le plan de référence, et chaque vue $\vec{v}_k$, nous calculons l'intersection x_k avec la méso-structure. Les déplacements $\vec{\delta}_1 = x_1 - x_0$ et $\vec{\delta}_2 = x_2 - x_0$ sur l'espace de paramétrisation sont plus ou moins alignés avec les directions de vue projetées $\vec{v}_1$ et $\vec{v}_2$. 64

7.1 Un aperçu du jeu de données que nous avons utilisé dans notre analyse. De gauche à droite, la BTF *Bouton* se présente comme une succession de plots en plastique rouge sur une surface plane grise. La BTF *Isba* représente un dallage constitué de deux marbres différents sur une surface de granit. La BTF *Tricot* reprend la méso-structure simplifiée des mailles d'un tricot et la BTF *Éponge* qui comme son nom l'indique tente de reproduire la surface d'une éponge. Les fonctions de hauteur représentant les méso-structures de ces deux dernières BTFs sont issues de notre opération de reconstruction (*cf.* Section 4.2.1). 65

7.2 Images de la variance du vecteur d'illumination en distance Lab selon les différentes directions de vue pour le modèle *Tricot*. En (a), la variance pour la représentation classique. En (b), la variance pour la fonction Flat-BTF correspondante avec les texels indéfinis remplis. En (c), la variance pour la fonction Flat-BTF sans la prise en compte des texels indéfinis. 67

7.3 En (a) Une image issue d'une Flat-BTF. En (b) l'indirection correspondante à la vue issue du VDIM. En (c) l'image reconstruite en utilisant (b) comme indirection dans (a). En (d) l'image de référence correspondante dans la représentation standard. En (e) la différence absolue en RVB entre (c) et (d). 69

7.4 Une comparaison visuelle d'ABRDFs caractéristiques entre une BTF et une Flat-BTF pour les modèles *Bouton* (en haut) et *Isba* (en bas). En (a) et (b), deux ABRDFs de la BTF classique précédées de leur localisation dans le domaine spatial. En (a') et (b'), les ABRDFs équivalentes pour la fonction Flat-BTF. Pour chaque couple d'ABRDFs sur une ligne, l'ABRDF (1) a été sélectionnée pour correspondre à la même position sur la méso-structure sous-jacente, tandis que l'ABRDF (2) correspond exactement au même texel dans l'image. 71

7.5 Une comparaison visuelle d'ABRDFs caractéristiques entre une BTF et une Flat-BTF pour les modèles *Tricot* (en haut) et *Éponge* (en bas). En (c) et (d), deux ABRDFs de la BTF classique précédées de leur localisation dans le domaine spatial. En (c') et (d'), les ABRDFs équivalentes pour la fonction Flat-BTF. Pour chaque couple d'ABRDFs sur une ligne, l'ABRDF (1) a été sélectionnée pour correspondre à la même position sur la méso-structure sous-jacente, tandis que l'ABRDF (2) correspond exactement au même texel dans l'image. 72

8.1 Une exemple de rendu en 3D avec une implémentation de la fonction VDIM sur GPU (NVIDIA GeForce 8800 GTX) et où la fonction Flat-BTF est définie par une carte de normales et une texture d'albédo diffus. Pour une résolution de 800×800, le taux de rafraîchissement est aux alentours de 300 images par seconde. En (a), une vue globale montrant l'impression de relief crée par l'effet de parallaxe. En (b) et en (c), un zoom au pôle avec deux directions lumineuses différentes. 75

8.2 Détermination de l'apparence finale avec une Flat-BTF pour une direction de vue $\vec{v}$ et une direction de lumière incidente $\vec{l}$ données. On accède d'abord à la fonction VDIM afin de déterminer de nouvelles coordonnées $({i'}_n, {j'}_n)$, qui correspondent à l'indirection, en fonction de (i, j) et de la vue proche $\vec{v}_n$. Dans le cas des BTFs classiques, on utilise directement (i, j) pour toutes les directions de vue proches. Neuf textures sont ensuite utilisées au total pour les directions de vue les plus proches de $\vec{v}$: $\vec{v}_1$, $\vec{v}_2$ et $\vec{v}_3$ et les directions lumineuses les plus proches de $\vec{l}$: $\vec{l_1}$, $\vec{l_2}$ et $\vec{l_3}$. Pour chaque texture, les 4 texels les plus proches pour les coordonnées (i_n, j_n) sont interpolés de manière bilinéaire puis les données résultantes sont interpolées à leur tour de manière barycentrique pour les différentes direction lumineuses puis pour les différentes directions de vues afin d'obtenir l'apparence finale. 76

8.3 La carte d'indirections de référence pour la vue zénithale du modèle *Isba* suivie de trois exemples de cartes de décalages Δ pour différentes directions de vue. Dans la carte de référence, les coordonnées (i, j) sont représentées respectivement par les composantes rouge et verte. Dans les cartes de décalage, les coordonnées de la direction 2D $\vec{\delta}$ de décalage sont encodées respectivement par les composantes couleurs rouge et verte tandis que la composante bleue encode la distance d. 78

8.4 Erreur d'indirection sur le modèle *Bouton* pour deux stratégies d'échantillonnage : 8×32 (a) et 16×16 (b). Pour un grand nombre de directions de vue prises au hasard, l'erreur d'indirection est calculée entre l'indirection déterminée par interpolation des vues les plus proches et l'indirection calculée directement par lancer de rayons sur la méso-structure. 79

8.5 Organisation des texels de la texture 2D VDIM utilisée pour le rendu : pour chaque texel, nous encodons les valeurs de $\Delta_{\vec{v}}(i, j)$ de manière accolées pour les 8×32 directions. En haut, la texture 2D représentant l'intégralité de la fonction VDIM. En bas, les différentes valeurs dépendantes de la vue pour un texel donné. Notons que la même valeur est répétée sur toute une ligne pour le pôle $(\theta = 0, \phi = 0)$. 80

8.6 Visualisation de notre jeu de données de Flat-BTFs plaquées sur différents objets 3D (rendu en moyenne de 500 images par seconde en 800×800). Chaque fonction Flat-BTF des modèles est représentée par une carte de normales et une texture diffuse. Sur la première et la deuxième ligne, nos quatre Flat-BTFs sur une bouteille. Sur la dernière ligne, le modèle *Isba* sur une théière pour deux directions de lumière incidente différentes. 82

8.7 Visualisation de notre jeu de données de Flat-BTFs plaquées sur différents objets 3D (rendu en moyenne de 500 images par seconde en 800×800). Chaque fonction Flat-BTF des modèles est représentée par une carte de normales et une texture diffuse. Sur la première ligne, le modèle *Éponge* sur une théière pour deux directions de lumière incidente différentes. Sur la deuxième et dernière ligne, les modèles *Bouton* et *Tricot* sur un vase pour deux directions de vue différentes. 83

9.1 Le contexte de notre système de quantification à la volée est une application client-serveur de visualisation 3D. La charge de traitement de quantification effectuée à la volée est déléguée au GPU. 85

9.2 La puissance des derniers terminaux mobiles en fait des clients potentiels pour des applications de diffusion de contenu 3D. Sur la figure, une déquantification interactive et un rendu sont effectués sur un PDA avec OpenGL ES. 86

9.3 L'utilisation d'une fonction de déformation sur le cube unitaire (a) permet, une fois projetée sur une sphère unitaire (b), d'obtenir un échantillonnage plus uniforme. La partie orange correspond à la taille de la table d'indexation. 89

9.4 Les attributs de l'apparence sont stockés sur 32 bits. La normale est encodée sur 17 bits (5 bits pour la face et les symétries, 6 bits pour i, 6 bits pour j), et la couleur est encodée sur 15 bits (5 bits pour chaque composante : rouge, vert, bleu). 90

9.5 Pour chaque frame rendue sur le GPU, les couleurs et normales flottantes (resp. N et C, P étant la position) sont lues à partir d'un fichier ou d'un flux en entrée et envoyées entrelacées sur le GPU dans une texture à précision flottante. Une fois le rendu effectué, les normales et les couleurs quantifiées (resp. N' et C') sont récupérées depuis le framebuffer avant d'être écrites dans un fichier ou flux de sortie. 92
9.6 En (a), une comparaison entre le CPU et le GPU des temps de quantification de l'apparence de modèles à base de points. En (b), les temps moyens requis pour la transmission d'un modèle (original et quantifié avec le GPU) en fonction de sa taille. Pour le modèle quantifié, les résultats incluent les temps de quantification et de reconstruction. Les mesures ont été effectuées pour un réseau WIFI avec une bande passante de 54Mb/s. 94
9.7 Erreur de quantification pour un éclairement de Phong. Trois sources lumineuses ave un matériau spéculaire. 96

Liste des tableaux

3.1 Résumé des propriétés en termes de représentation compacte des BTFs pour les différentes méthodes présentées. La note maximale pour la décompression temps-réel/interactive correspond au mieux à environ 30 images par seconde. 30

7.1 Les variances moyennes et maximales de la distance Lab pour le vecteur d'illumination (81 directions de vue et de lumière pour une résolution spatiale de 256×256). Nous comparons la représentation originale avec deux versions de notre représentation Flat-BTF. Dans la première version (1), les texels non visibles ont été remplis par notre algorithme de remplissage. Dans la version (2) les texels non visibles et donc indéfinis ont été ignorés pour l'estimation de la variance. 67

7.2 Tableau résumant les tailles en kilo-octets des images pour les ABRDFs des Figures 7.4 (Page 71) et 7.5 (Page 72) au format *PNG* et *JPEG* pour la qualité maximale. La ligne intitulée « ABRDF » indique le numéro du texel de l'ABRDF concernée sur les figures. Le chiffre à gauche de l'inégalité correspond à la taille pour la Flat-BTF tandis que le chiffre à droite de l'inégalité correspond à la taille pour la représentation classique. Pour certaines ABRDFs des modèles *Tricot* ou *Éponge*, la taille dans la représentation Flat-BTF est jusqu'à 4 ou 5 fois moindre. 69

7.3 La moyenne, le maximum et la variance des différences (en distance Lab) des texels entre les images de référence de BTF et les images reconstruites depuis la représentation correspondante en Flat-BTF. 70

9.1 Impacte de la taille du tampon sur la vitesse de quantification. Les temps de la première ligne sont obtenus pour une implémentation CPU complète avec un Pentium 4 3.4GHz. Les temps de la seconde ligne sont obtenus pour une implémentation GPU sur la même station de travail avec une Quadro FX3400 sur un port PCI Express x16. Pour finir, les temps de la troisième ligne sont obtenus avec une implémentation GPU sur un PC portable muni d'un Pentium-M 2.26 GHz et une Geforce 7800GTX sur un port PCI Express x16. 93

Bibliographie

[BD06] Lionel Baboud and Xavier Décoret. Rendering geometry with relief textures. In *Proc. Graphics Interface*, pages 195–201. Canadian Information Processing Society, 2006.

[BEWW+08] Moshe Ben-Ezra, Jiaping Wang, Bennett Wilburn, Xiaoyang Li, and Le Ma. An LED-only BRDF Measurement Device. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[Bli77] Jim Blinn. Models of light reflection for computer synthesized pictures. In *SIGGRAPH '77 : Proceedings of the 4th annual conference on Computer graphics and interactive techniques*, pages 192–198, New York, NY, USA, 1977. ACM.

[Bli78] Jim Blinn. Simulation of Wrinkled Surfaces. In *Proc. SIGGRAPH '78*, volume 12, pages 286–292, 1978.

[Bon] BTF Database Bonn.
http ://btf.cs.uni-bonn.de.

[BPZ99] Chandrajit L Bajaj, Valerio Pascucci, and Guozhong Zhuang. Single Resolution Compression of Arbitrary Triangular Meshes with Properties. In *Proc. IEEE Conference on Data Compression '99*, page 247, 1999.

[BWK02] Mario Botsch, Andreas Wiratanaya, and Leif Kobbelt. Efficient High Quality Rendering of Point Sampled Geometry. In *Proc. EUROGRAPHICS workshop on Rendering 2002*, pages 53–64, 2002.

[Cal98] Patrick Callet. *Couleur-lumière couleur-matière.* Arts et Sciences. Diderot ed., 1998.

[Cat74] Edwin Earl Catmull. *A subdivision algorithm for computer display of curved surfaces.* PhD thesis, 1974.

[CGS06] Tongbo Chen, Michael Goesele, and Hans-Peter Seidel. Mesostructure from Specularity. In *CVPR '06 : Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1825–1832, Washington, DC, USA, 2006. IEEE Computer Society.

[Coo84] Robert L. Cook. Shade trees. 18(3) :223–231, 1984.

[Cor] Cornell Light Measurement Laboratory.
http ://www.graphics.cornell.edu/research/measure/.

[CUR] Columbia-Utrecht Reflectance and Texture database.
http ://www1.cs.columbia.edu/cave/curet/.

[DBB06] Philip Dutré, Kavita Bala, and Philippe Bekaert. *Advanced Global Illumination.* A. K. Peters, Ltd., 2006.

[DD98] Michael Daum and Gregory L. Dudek. On 3-d surface reconstruction using shape from shadows. In *CVPR '98 : Proceedings of the IEEE Computer Society Conference on*

Computer Vision and Pattern Recognition, page 461, Washington, DC, USA, 1998. IEEE Computer Society.

[Dee95] Michael Deering. Geometry Compression. In *Proc. ACM SIGGRAPH '95*, pages 13–20, 1995.

[DHT+00] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In Kurt Akeley, editor, *Proc. ACM SIGGRAPH 2000*, pages 145–156. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[Dis98] Jean-Michel Dischler. Efficiently rendering macro geometric surface structures with bi-directional texture functions. In *Eurographics Rendering Workshop 1998*, pages 169–180, June 1998.

[DKS+03] Katja Daubert, Jan Kautz, Hans-Peter Seidel, Wolfgang Heidrich, and Jean-Michel Dischler. Efficient Light Transport Using Precomputed Visibility. *IEEE Comput. Graph. Appl.*, 23(3) :28–37, 2003.

[DLHS01] Katja Daubert, Hendrik P. A. Lensch, Wolfgang Heidrich, and Hans-Peter Seidel. Efficient Cloth Modeling and Rendering. In *Proc. EUROGRAPHICS Workshop on Rendering Techniques*, pages 63–70. Springer-Verlag, 2001.

[DvNK97] Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. Reflectance and texture of real-world surfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 151–157, 1997.

[DvNK99] Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Trans. Graph.*, 18(1) :1–34, 1999.

[DW04] Kristin J. Dana and Jing Wang. Device for convenient measurement of spatially varying bidirectional reflectance. *Journal of the Optical Society of America A*, 21 :1–12, January 2004.

[FC88] Robert T. Frankot and Rama Chellappa. A Method for Enforcing Integrability in Shape from Shading Algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(4) :439–451, 1988.

[FCM+08] Yannick Francken, Tom Cuypers, Tom Mertens, Jo Gielis, and Philippe Bekaert. High Quality Mesostructure Acquisition Using Specularities. In *CVPR 2008 : Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

[FH04] Jiri Filip and Michal Haindl. Non-linear Reflectance Model for Bidirectional Texture Function Synthesis. In *Proc. International Conference on Pattern Recognition*, pages 80–83. IEEE Computer Society, 2004.

[FKIS02] R. Furukawa, H. Kawasaki, K. Ikeuchi, and M. Sakauchi. Appearance based object modeling using texture database : Acquisition, compression and rendering. In S. Müller and W. Stürzlinger, editors, *Eighth Eurographics Workshop on Virtual Environments*, 2002.

[Flo03] Michael S. Floater. Mean value coordinates. *Computer Aided Geometry Design*, 20(1) :19–27, 2003.

[FP05] Vincent Forest and Mathias Paulin. Rendu temps réel de fonction de textures bidirectionnelles. In *Journées AFIG 2005*, pages 1–12. Université Louis Pasteur - AFIG, novembre 2005.

[GAHO07] Abhijeet Ghosh, Shruthi Achutha, Wolfgang Heidrich, and Matthew O'Toole. BRDF Acquisition with Basis Illumination. In *Proc. of IEEE International Conference on Computer Vision (ICCV) 2007*. IEEE, 2007.

[GGH02] Xianfeng Gu, Steven J. Gortler, and Hugues Hoppe. Geometry images. *ACM Trans. Graph.*, 21(3) :355–361, 2002.

[Gra03] Graphite, 2003. http ://www.loria.fr/ levy/Graphite/index.html.

[GTHD03] Andrew Gardner, Chris Tchou, Tim Hawkins, and Paul Debevec. Linear light source reflectometry. In *SIGGRAPH '03 : ACM SIGGRAPH 2003 Papers*, pages 749–758, New York, NY, USA, 2003. ACM.

[HBR+07] Julien Hadim, Tamy Boubekeur, Mickaël Raynaud, Xavier Granier, and Christophe Schlick. On-the-fly Appearance Quantization on GPU for 3D Broadcasting. In *ACM SIGGRAPH Web3D*. ACM, ACM Press, 2007.

[HDKS00] Wolfgang Heidrich, Katja Daubert, Jan Kautz, and Hans-Peter Seidel. Illuminating micro geometry based on precomputed visibility. In *SIGGRAPH '00 : Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 455–464, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[HF07] Michal Haindl and Jiri Filip. Extreme Compression and Modeling of Bidirectional Texture Function. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(10) :1859–1865, 2007.

[HGS08] Julien Hadim, Xavier Granier, and Christophe Schlick. Flat Bidirectional Texture Funtions. 2008. in submission.

[Hop96] Hugues Hoppe. Progressive Meshes. *Computer Graphics*, 30(Annual Conference Series) :99–108, 1996.

[HP03] Jefferson Y. Han and Ken Perlin. Measuring bidirectional texture reflectance with a kaleidoscope. *ACM Trans. Graph.*, 22(3) :741–748, 2003.

[HS99] Wolfgang Heidrich and Hans-Peter Seidel. Realistic, hardware-accelerated shading and lighting. In *SIGGRAPH '99 : Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 171–178, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[JSX+08] Wang Jiaping, Zhao Shuang, Tong Xin, Lin Stephen, Lin Zhouchen, Dong Yue, Guo Baining, and Shum Heung-Yeung. Modeling and rendering of heterogeneous translucent materials using the diffusion equation. *ACM Trans. Graph.*, 27(1) :1–18, 2008.

[KBD07] Jan Kautz, Solomon Boulos, and Frédo Durand. Interactive editing and modeling of bidirectional texture functions. *ACM Trans. Graph.*, 26(3) :53, 2007.

[KCK04] Deok-Soo Kim, Youngsong Cho, and Hyun Kim. Normal vector compression of 3D mesh model based on clustering and relative indexing. *Future Gener. Comput. Syst.*, 20(8) :1241–1250, 2004.

[KMBK03] Mellissa. L. Koudelka, Sebastian Magda, Peter N. Belhumeur, and David J. Kriegman. Acquisition, Compression, and Synthesis of Bidirectional Texture Functions. In *3rd International Workshop on Texture Analysis and Synthesis*, pages 59–64, 2003.

[KS92] John R. Kender and Earl M. Smith. Shape from darkness : deriving surface information from dynamic shadows. pages 378–385, 1992.

[KTI+01] Tomomichi Kaneko, Toshiyuki Takahei, Masahiko Inami, Naoki Kawakami, Yasuyuki Yanagida, Taro Maeda, and Susumu Tachi. Detailed shape representation with parallax mapping. In *In Proceedings of the ICAT 2001*, pages 205–208, 2001.

[LBAD+06] Jason Lawrence, Aner Ben-Artzi, Christopher DeCoro, Wojciech Matusik, Hanspeter Pfister, Ravi Ramamoorthi, and Szymon Rusinkiewicz. Inverse shade trees for non-parametric material representation and editing. *ACM Trans. Graph.*, 25(3) :735–745, 2006.

[LFTG97] Eric P. F. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-Linear Approximation of Reflectance Functions. *Computer Graphics*, 31(Annual Conference Series) :117–126, 1997.

[LGK+01] Hendrik P. A. Lensch, Michael Goesele, Jan Kautz, Wolfgang Heidrich, and Hans-Peter Seidel. Image-Based Reconstruction of Spatially Varying Materials. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 103–114, London, UK, 2001. Springer-Verlag.

[LHZ+04] Xinguo Liu, Yaohua Hu, Jingdan Zhang, Xin Tong, Baining Guo, and Heung-Yeung Shum. Synthesis and Rendering of Bidirectional Texture Functions on Arbitrary Surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(3) :278–289, 2004.

[Lin00] Peter Lindstrom. Out-of-Core Simplification of Large Polygonal Models. In *Proc. ACM SIGGRAPH 2000*, pages 259–262. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[LPF+07] Man-Kang Leung, Wai-Man Pang, Chi-Wing Fu, Tien-Tsin Wong, and Pheng-Ann Heng. Tileable BTF. *IEEE Transactions on Visualization and Computer Graphics*, 13(5) :953–965, 2007.

[LPRM02] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least squares conformal maps for automatic texture atlas generation. In ACM, editor, *Proc. ACM SIGGRAPH 2002*, Jul 2002.

[LS01] P. Lindstrom and C. Silva. A memory insensitive technique for large model simplification. In *Proceedings of IEEE Visualization 2001*, pages 121–126, October 2001.

[LW94] Eric P. Lafortune and Yves D. Willems. Using the modified phong reflectance model for physically based rendering. Technical report, 1994.

[LYS01] Xinguo Liu, Yizhou Yu, and Heung-Yeung Shum. Synthesizing bidirectional texture functions for real-world surfaces. In *SIGGRAPH '01 : Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 97–106, New York, NY, USA, 2001. ACM.

[MAA01] Michael D. McCool, Jason Ang, and Anis Ahmad. Homomorphic factorization of BRDFs for high-performance rendering. In *SIGGRAPH '01 : Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 171–178, New York, NY, USA, 2001. ACM.

[Mam89] Abraham Mammen. Transparency and antialiasing algorithms implemented with the virtual pixel maps technique. *IEEE Comput. Graph. Appl.*, 9(4) :43–55, 1989.

[Max88] Nelson L. Max. Horizon mapping : shadows for bump-mapped surfaces. *The Visual Computer*, 4(2) :109–117, 1988.

[MBK05] Gero Müller, Gerhard H. Bendels, and Reinhard Klein. Rapid Synchronous Acquisition of Geometry and Appearance of Cultural Heritage Artefacts . In *Proc. International Symposium on Virtual Reality, Archaeology and Cultural Heritage - VAST*, pages 13–20. EUROGRAPHICS, 2005.

[McA02] David K. McAllister. *A generalized surface appearance representation for computer graphics*. PhD thesis, 2002.

[MCC+04] Wan-Chun Ma, Sung-Hsiang Chao, Bing-Yu Chen, Chun-Fa Chang, Ming Ouhyoung, and Tomoyuki Nishita. An efficient representation of complex materials for real-time rendering. In *VRST '04 : Proceedings of the ACM symposium on Virtual reality software and technology*, pages 150–153, New York, NY, USA, 2004. ACM.

[MCT+05] Wan-Chun Ma, Sung-Hsiang Chao, Yu-Ting Tseng, Yung-Yu Chuang, Chun-Fa Chang, Bing-Yu Chen, and Ming Ouhyoung. Level-of-detail representation of bidirectional texture functions for real-time rendering. In *Proc. ACM SIGGRAPH symposium on Interactive 3D graphics and games (I3D '05)*, pages 187–194, 2005.

[MGW01] Tom Malzbender, Dan Gelb, and Hans Wolters. Polynomial texture maps. In *SIGGRAPH '01 : Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 519–528. ACM, 2001.

[MK06] Sebastian Magda and David Kriegman. Reconstruction of Volumetric Surface Textures for Real-Time Rendering. In *Proc. EUROGRAPHICS Symposium on Rendering*, pages 19–29, 2006.

[MKV+03] Tom Mertens, Jan Kautz, Frank Van, Reeth Hans-Peter Seidel, and Limburgs Universitair Centrum. Efficient rendering of local subsurface scattering. In *In Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, page 51, 2003.

[MLH02] David K. McAllister, Anselmo Lastra, and Wolfgang Heidrich. Efficient rendering of spatial bi-directional reflectance distribution functions. In *Proc. ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 79–88, 2002.

[MMK03a] J. Meseth, G. Müller, and R. Klein. Preserving Realism in Real-Time Rendering of Bidirectional Texture Functions. In *OpenSG Symposium 2003*, pages 89–96. Eurographics Association, Switzerland, April 2003.

[MMK03b] G. Müller, J. Meseth, and R. Klein. Compression and Real-Time Rendering of Measured BTFs Using Local PCA. In T. Ertl, B. Girod, G. Greiner, H. Niemann, H.-P. Seidel, E. Steinbach, and R. Westermann, editors, *Vision, Modeling and Visualisation 2003*, pages 271–280. Akademische Verlagsgesellschaft Aka GmbH, Berlin, November 2003.

[MMK04] J. Meseth, G. Müller, and R. Klein. Reflectance Field based real-time, high-quality Rendering of Bidirectional Texture Functions. *Computers and Graphics*, 28(1) :103–112, 2004.

[MMS+04] G. Müller, J. Meseth, M. Sattler, R. Sarlette, and R. Klein. Acquisition, Synthesis and Rendering of Bidirectional Texture Functions. In Christophe Schlick and Werner Purgathofer, editors, *Eurographics 2004, State of the Art Reports*, pages 69–94. INRIA and Eurographics Association, September 2004.

[MMS+05] G. Müller, J. Meseth, M. Sattler, R. Sarlette, and R. Klein. Acquisition, Synthesis and Rendering of Bidirectional Texture Functions. *Computer Graphics Forum*, 24(1) :83–109, March 2005.

[MSK07] Gero Müller, Ralf Sarlette, and Reinhard Klein. Procedural Editing of Bidirectional Texture Functions. In *Proc. EUROGRAPHICS Symposium on Rendering*, 2007.

[MW08] Morgan McGuire and Kyle Whitson. Indirection Mapping for Quasi-Conformal Relief Mapping. In *Proc. ACM SIGGRAPH symposium on Interactive 3D Graphics and games (I3D '08)*, 2008.

[ND06] Addy Ngan and Frédo Durand. Statistical Acquisition of Texture Appearance. In *Proc. EUROGRAPHICS Symposium on Rendering*, pages 31–40, 2006.

[NDM05] Addy Ngan, Frédo Durand, and Wojciech Matusik. Experimental Analysis of BRDF Models. In *Proc. EUROGRAPHICS Symposium on Rendering*, pages 117–226, 2005.

[Nic70] F. Nicodemus. Reflectance Nomenclature and Directional Reflectance and Emissivity. *Appl. Opt.*, 9(6) :1474–1475, 1970.

[Nis] NIST reference reflectometer : STARR facility.
http ://physics.nist.gov/.

[NRH+77] F. Nicodemus, J. Richmond, J. Hsia, I. ginsberg, and Limperis. Geometric Considerations and Nomenclature for Reflectance. In *Monograph 160, National Bureau of Standards (US)*, 1977.

[NVI06] NVIDIA. OpenGL Extensions for Geforce 80, November 2006. http ://developer.nvidia.com/object/nvidia_opengl_specs.html.

[NZG05] A. Neubeck, A. Zalesny, and L. Van Gool. 3D Texture Reconstruction from Extensive BTF Data. In *Texture 2005 Workshop in conjunction with ICCV 2005*, pages 13–19, October 2005.

[OBM00] Manuel M. Oliveira, Gary Bishop, and David K. McAllister. Relief texture mapping. In *SIGGRAPH '00 : Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 359–368, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[PBCK05] Budirijanto Purnomo, Jonathan Bilodeau, Jonathan Cohen, and Subodh Kumar. Hardware-Compatible Vertex Compression Using Quantization and Simplication. In *Proc. ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 53–61, 2005.

[PFTV88] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical recipes in C : the art of scientific computing*. Cambridge University Press, New York, NY, USA, 1988.

[PGB03] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Trans. Graph.*, 22(3) :313–318, 2003.

[PH04] Matt Pharr and Greg Humphreys. *Physically Based Rendering : From Theory to Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[PO06] Fabio Policarpo and Manuel M. Oliveira. Relief mapping of non-height-field surface details. In *Proc. ACM SIGGRAPH symposium on Interactive 3D graphics and games (I3D '06)*, pages 55–62, 2006.

[PR00] R. Pajarola and J. Rossignac. Compressed Progressive Meshes. *IEEE Transactions on Visualization and Computer Graphics*, 6(1) :79–93, 2000.

[Pri00] C. Prince. *Progressive Meshes for Large Models of Arbitrary Topology*. PhD thesis, University of Washington, 2000.

[RL00] Szymon Rusinkiewicz and Marc Levoy. QSplat : a multiresolution point rendering system for large meshes. In *Proc. ACM SIGGRAPH 2000*, pages 343–352, 2000.

[RL01] Szymon Rusinkiewicz and Marc Levoy. Streaming QSplat : a viewer for networked visualization of large, dense models. In *Proc. Symposium on Interactive 3D graphics 2001*, pages 63–68, 2001.

[RL03] Nicolas Ray and Bruno Lévy. Hierarchical least squares conformal maps. In *11th Pacific Conference on Computer Graphics and Applications, Canmore, Canada*, pages 263–270, Octobre 2003.

[RTG97] Holly E. Rushmeier, Gabriel Taubin, and André Guéziec. Appying Shape from Lighting Variation to Bump Map Capture. In *Proceedings of the Eurographics Workshop on Rendering Techniques '97*, pages 35–44, London, UK, 1997. Springer-Verlag.

[SC00] Peter-Pike J. Sloan and Michael F. Cohen. Interactive horizon mapping. In *In Rendering Techniques '00 (Proc. Eurographics Workshop on Rendering)*, pages 281–286. Springer, 2000.

[SLMBY05] Alla Sheffer, Bruno Lévy, Maxim Mogilnitsky, and Alexander Bogom Yakov. ABF++ : Fast and robust angle based flattening. *ACM Trans. Graph.*, Apr 2005.

[SSK03] Mirko Sattler, Ralf Sarlette, and Reinhard Klein. Efficient and realistic visualization of cloth. In *EGRW '03 : Proceedings of the 14th Eurographics workshop on Rendering*, pages 167–177, 2003.

[SvLD03] Frank Suykens, Karl vom Berge, Ares Laga, and Philip Dutré. Interactive rendering with bidirectional texture functions. *Computer Graphics Forum*, 22(3), 2003.

[THLR98] Gabriel Taubin, William Horn, Francis Lazarus, and Jarek Rossignac. Geometry coding and VRML. *Proceedings of the IEEE*, 86(6) :1228–1243, 1998.

[TJP+03] Mertens Tom, Kautz Jan, Bekaert Philippe, Seidel Hans-Peter, and Van Reeth Frank. Interactive rendering of translucent deformable objects. In *EGRW '03 : Proceedings of the 14th Eurographics workshop on Rendering*, pages 130–140, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[TZL+02] Xin Tong, Jingdan Zhang, Ligang Liu, Xi Wang, Baining Guo, and Heung-Yeung Shum. Synthesis of bidirectional texture functions on arbitrary surfaces. In *SIGGRAPH '02 : Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 665–672. ACM, 2002.

[VT04] M. Alex O. Vasilescu and Demetri Terzopoulos. TensorTextures : multilinear image-based rendering. *ACM Trans. Graph.*, 23(3) :336–342, 2004.

[War91] Gregory J. Ward. Real Pixels. In *Graphics Gems II*, pages 80–83. 1991.

[War92] Gregory J. Ward. Measuring and modeling anisotropic reflection. *SIGGRAPH Comput. Graph.*, 26(2) :265–272, 1992.

[WD06] Jing Wang and Kristin J. Dana. Relief texture from specularities. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(3) :446–457, 2006.

[WHON97a] Tien-Tsin Wong, Pheng-Ann Heng, Siu-Hang Or, and Wai-Yin Ng. Image-based Rendering with Controllable Illumination. In *Proc. EUROGRAPHICS Workshop on Rendering Techniques*, pages 13–22. Springer-Verlag, 1997.

[WHON97b] Tien-Tsin Wong, Pheng-Ann Heng, Siu-Hang Or, and Wai-Yin Ng. Image-Based Rendering with Controllable Illumination. In Julie Dorsey and Phillipp Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 13–22, New York, NY, 1997.

[WK03] J. Wu and L. Kobbelt. A stream algorithm for the decimation of massive meshes. In *Graphics Interface'03 Conference Proceedings*, pages 185–192, 2003.

[WTL+04] Xi Wang, Xin Tong, Stephen Lin, Shimin Hu, Baining Guo, and Heung-Yeung Shum. Generalized Displacement Maps. In *Rendering Techniques*, pages 227–234, 2004.

[WTS+05] Jiaping Wang, Xin Tong, John Snyder, Yanyun Chen, Baining Guo, and Heung-Yeung Shum. Capturing and Rendering Geometry Details for BTF-mapped Surfaces. *The Visual Computer*, 21(8-10) :559–568, 2005.

[WWS+05] Hongcheng Wang, Qing Wu, Lin Shi, Yizhou Yu, and Narendra Ahuja. Out-of-Core Tensor Approximation Of Multi-Dimensional Matrices of Visual Data. *ACM Trans. Graph.*, 24(3) :527–535, 2005.

[WWT+03] Lifeng Wang, Xi Wang, Xin Tong, Stephen Lin, Shimin Hu, Baining Guo, and Heung-Yeung Shum. View-dependent displacement mapping. *ACM Trans. Graph.*, 22(3) :334–339, 2003.

[YC05] Yizhou Yu and Johnny T. Chang. Shadow graphs and 3d texture reconstruction. *Int. J. Comput. Vision*, 62(1-2) :35–60, 2005.

[ZWLZ08] Xia Zhou, Yangsheng Wang, Jituo Li, and Daiguo Zhou. ABF based face texturing. In *Edutainment '08 : Proceedings of the 3rd international conference on Technologies for E-Learning and Digital Entertainment*, pages 664–674, Berlin, Heidelberg, 2008. Springer-Verlag.