Université Grenoble I – Joseph Fourier Sciences et Géographie

# Ré–éclairage et Remodélisation Interactifs des Scènes Réelles pour la Réalité Augmentée

Céline LOSCOS

Thèse préparée à *i*MAGIS-GRAVIR/IMAG-INRIA, et présentée pour obtenir le titre de :

**Docteur de l'Université Joseph Fourier**
**discipline informatique**

*Arrêté ministériel du 5 juillet 1984 et du 30 mars 1992*

Soutenue le 20 Décembre 1999 devant la commission d'examen :

| | | |
|---|---|---|
| Jean–Pierre | PEYRIN | Président |
| Pascal | GUITTON | Rapporteur |
| Pierre | POULIN | Rapporteur |
| Bernard | PÉROCHE | Examinateur |
| George | DRETTAKIS | Directeur |
| Claude | PUECH | Directeur |

*Je dédie cette thèse à Jean et Émilienne Noël ainsi qu'à Louisiane Pascal.*

# Remerciements

Au début de cette thèse, modifier l'éclairage d'environnements réels en temps interactif nous paraissait un problème très difficile. L'aboutissement de ce travail en a été d'autant plus gratifiant. Il reste encore beaucoup à faire avant qu'un utilisateur puisse simuler un nouvel éclairage pour des scènes d'intérieur ou d'extérieur. Nous sommes conscients que le travail développé dans cette thèse n'est qu'une goutte d'eau dans un océan. Mais j'espère qu'un jour, je pourrai concevoir virtuellement l'éclairage et l'agencement de mon appartement avec un logiciel automatique, rapide et facile d'utilisation.

J'ai vraiment apprécié ce thème de recherche un peu particulier car il est à l'intersection de nombreux domaines. Dans ce document, se trouvent trois ans de travail qui n'auraient pu aboutir sans l'aide d'un nombre considérable de personnes. George Drettakis a été la personne la plus directement impliquée dans ce travail. Nous avons souvent discuté des problèmes et essayé de trouver ensemble les solutions adéquates. Certaines étapes ont été particulièrement difficiles et George m'a aidée à être persévérante. Durant ces trois ans, il a su trouver le temps nécessaire pour des réunions ou simplement pour m'aider à régler un problème de programmation. Il était d'ailleurs présent les week–ends et les soirs de semaine lorsqu'il le fallait.

Claude Puech a également largement contribué à l'avancement de mon travail de recherche. Malgré son emploi du temps très chargé, j'ai toujours pu discuter avec lui et obtenir son aide sur divers problèmes annexes à ma thèse. Je reste convaincue que la bonne entente qui règne dans l'équipe iMAGIS lui est due, et je l'en remercie car cela permet d'avoir un environnement de travail très agréable.

Je remercie également les rapporteurs de cette thèse, Pierre Poulin et Pascal Guitton, pour leurs remarques pertinentes et leurs encouragements. Je remercie également Bernard Péroche d'avoir accepté d'être membre de mon jury. J'ai eu le plaisir d'avoir, dans mon jury, Jean–Pierre Peyrin, mon professeur de langage et programmation à l'université Joseph Fourier de Grenoble. J'en profite pour le remercier de son soutien lorsque j'ai décidé de partir à Montréal pendant ma maîtrise, et de m'avoir permis d'enseigner à ses côtés durant ma thèse.

Pour mettre en place cette recherche, j'ai travaillé avec différentes personnes. Notamment, cette thèse n'aurait pas pu aboutir si nous n'avions reçu l'aide de Luc Robert et de Pierre Poulin. Ils nous ont donné accès à leur programme de reconstruction à partir d'images que nous décrivons dans ce document. Ils se sont montrés tous les deux disponibles lorsque nous avons rencontré des difficultés d'utilisation.

Ma collaboration pendant trois mois avec Marie–Claude Frasson a été la période la

plus agréable de cette thèse. Nous avons réalisé ensemble un travail considérable, souvent entrecoupé de fous rires. Cette collaboration a été accompagnée de discussions très enrichissantes et productives avec Bruce Walter et Xavier Granier.

Je remercie les relecteurs de cette thèse, Frédéric Elisei, Fabrice Neyret, Xavier Granier, Xavier Decoret, Mathieu Desbrun et Eric Ferley, pour leurs conseils et les corrections de nombreuses fautes d'orthographe.

Je remercie également les « brighters » pour toutes les classes et fonctions de radiosité que je n'aurais jamais eu le temps d'écrire moi–même, pour leur patience lorsque je venais leur demander de l'aide, pour tous les bugs corrigés, pour ce système énorme, dont on ne se lasse jamais de parcourir les lignes de code[†]. Je remercie en particulier le brighter (et non moins très honorable) François Sillion[‡], pour son intérêt et ses conseils. Je remercie également Félix Sillion d'avoir accepté qu'une de ses œuvres d'art apparaisse dans cette thèse.

J'ai particulièrement apprécié de partager le bureau de Laure France, dans lequel nous avons souvent discuté de choses et d'autres, dont les sujets s'égaraient souvent loin de ceux de nos thèses. Laure a été très présente lors de ces dernières années, et son inconditionnel soutien moral a été très efficace dans la réussite de cette thèse.

Je remercie Alexandre Meyer pour m'avoir accompagnée dans le dernier marathon de l'écriture de ce mémoire (lui même en marathon de soumission), et pour avoir réussi à me détendre en me parlant de « threads » et de sémaphores, de poissons multicolors, de mers et d'océans.

Je remercie également Patricio Inostroza et Jean-Michel Trivi pour leurs nombreux encouragements, ainsi que pour les discussions très intéressantes que nous avons pu avoir.

Je remercie Rosine Ah–Tchou pour sa gentillesse et son acceuil toujours souriant, ainsi que Jean–Luc Douville et Jean–Dominique Gascuel qui m'ont aidée à résoudre de nombreux problèmes matériels.

De façon plus générale, je remercie toute l'équipe iMAGIS. Il m'est souvent arrivé de réserver un bureau pendant plusieurs jours pour construire de nouvelles scènes et tout le monde a été très coopératif pour ne pas déplacer les différents meubles et les lampes que j'avais installés. La disponibilité des chercheurs et des thésards d'iMAGIS est exemplaire. Enfin, je ne démentirai pas la réputation des pauses cafés iMAGIS toujours très animées, ainsi que les fameuses soirées toujours très nombreuses et arrosées.

Enseigner pendant cette thèse a été pour moi un bol d'air frais lorsque la recherche devenait difficile. L'interaction avec les étudiants a souvent été positive et enrichissante et les différents enseignements que j'ai réalisés ont toujours été dans des équipes où règnait une bonne entente.

J'ai eu l'occasion durant ces trois ans de rencontrer de nombreuses personnes, liées ou non à mon travail. Certaines m'ont apporté de précieux conseils pour ma recherche, comme Paul Debevec et Steve Marschner. J'ai particulièrement apprécié mes rencontres avec Xavier Pueyo, James Stewart, Fede Perez, Éric Paquette, Éric Plante, François Bérard,

[†]Ne pas croire les mauvaises langues qui disent qu'il y traîne des « awful hacks » et quelques fois des « very awful hacks ».

[‡]François Sillion est co–auteur avec Claude Puech d'un excellent livre sur la radiosité.

Franck Fourel, Frédéric Vernier, Françoise Mocellin, David Thévenin, Eugenia Montiel, ainsi que de nombreuses autres personnes.

Enfin, cette thèse a été parsemée de moments de doutes et de problèmes personnels. J'ai réalisé dans ces moments–là combien certaines personnes m'étaient devenues proches. J'espère conserver longtemps ces nouvelles amitiés et je remercie tous ceux qui m'ont apporté leur soutien.

Je remercie tout particulièrement Philippe, mes parents, mon frère, et tous les membres de ma famille pour leur soutien continuel dans les moments de stress, leur tolérance de mes humeurs inégales, leur compréhension lorsque je travaillais plus que ce que je ne les voyais ; et simplement pour le bonheur de partager avec eux des instants précieux de ma vie.

# Table des matières


**1 Introduction** **17**
- 1.1 Motivations . . . 18
- 1.2 Contributions . . . 20
- 1.3 Structure du document . . . 21

**2 État de l'art** **23**
- 2.1 Radiométrie . . . 23
  - 2.1.1 Propagation de la lumière . . . 23
  - 2.1.2 Fonction de distribution de réflectance bidirectionnelle . . . 25
  - 2.1.3 Rendu . . . 27
- 2.2 Simulation de l'éclairage . . . 29
  - 2.2.1 Radiosité . . . 30
  - 2.2.2 Clustering . . . 35
  - 2.2.3 Contrôle des résultats . . . 36
- 2.3 Environnements dynamiques . . . 36
  - 2.3.1 Radiosité dynamique . . . 36
  - 2.3.2 Lancer de rayons dynamique . . . 38
- 2.4 Éclairage commun pour la réalité augmentée assistée par ordinateur . . . 39
  - 2.4.1 Présentation générale de la réalité augmentée . . . 39
  - 2.4.2 Reconstruction géométrique des scènes réelles . . . 41
  - 2.4.3 Illumination inverse . . . 45
  - 2.4.4 Éclairage commun . . . 50
- 2.5 Discussion . . . 56

**3 Problèmes d'acquisition** **59**
- 3.1 Les caméras utilisées . . . 60
- 3.2 Reconstruction des scènes réelles à partir de photographies . . . 61
  - 3.2.1 Un premier outil de reconstruction : *TotalCalib* . . . 62
  - 3.2.2 Un deuxième outil de reconstruction : *Rekon* . . . 67
- 3.3 Choix de l'éclairage . . . 70
- 3.4 Intégration des modèles 3D dans le système de rendu . . . 71
  - 3.4.1 Les matrices de points de vue . . . 72
  - 3.4.2 Modèle Inventor . . . 74

3.5 Discussion . . . . . 74

**4 Modification des propriétés d'éclairage d'une scène réelle à partir d'un seul éclairage connu 77**
4.1 Présentation générale du système . . . . . 78
4.2 Initialisation du système . . . . . 78
4.2.1 Initialisation de la radiosité hiérarchique . . . . . 79
4.2.2 Limitations du système ainsi initialisé . . . . . 81
4.3 Pré–calcul pour permettre le ré–éclairage . . . . . 82
4.3.1 Raffinement basé sur les textures pour délimiter les ombres . . . . 83
4.3.2 Création de la texture représentant l'éclairage indépendant de la visibilité . . . . . 86
4.3.3 Reprojection des ombres . . . . . 90
4.4 Éclairage commun et ré–éclairage interactif . . . . . 93
4.4.1 Insertion d'objets et de lampes virtuels . . . . . 93
4.4.2 Ré–éclairage interactif . . . . . 94
4.5 Résultats . . . . . 95
4.6 Conclusion . . . . . 98

**5 Modification des propriétés d'une scène réelle connue sous différents éclairages 101**
5.1 Description générale de la méthode . . . . . 102
5.2 Acquisition . . . . . 103
5.3 Estimation de la réflectance . . . . . 107
5.3.1 Approche générale . . . . . 107
5.3.2 Estimation de la réflectance pour chaque image de luminance . . 110
5.3.3 Filtrage des valeurs de réflectance . . . . . 110
5.3.4 Estimation finale de la réflectance . . . . . 114
5.4 Initialisation du système d'éclairage . . . . . 114
5.4.1 Éclairage direct . . . . . 114
5.4.2 Éclairage indirect . . . . . 116
5.4.3 Rendu . . . . . 119
5.5 Ré–éclairage et modification de la géométrie . . . . . 120
5.5.1 Modification de la géométrie . . . . . 120
5.5.2 Génération de textures . . . . . 126
5.5.3 Modification de l'éclairage . . . . . 128
5.6 Résultats . . . . . 129
5.7 Conclusion . . . . . 133

**6 Calibrage photométrique 135**
6.1 Calibrage photométrique . . . . . 136
6.1.1 Propriétés contrôlables et non contrôlables de notre appareil photographique . . . . . 136
6.1.2 Glaire et vignettage . . . . . 136

6.2 Méthode pour créer des images de luminance à partir de photographies . . 136
6.2.1 Méthodes existantes . . . . . 137
6.2.2 Adaptation de la méthode précédente au cas d'un appareil photographique semi–automatique . . . . . 138
6.3 Utilisation d'images de luminance pour les méthodes de ré–éclairage développées . . . . . 141
6.3.1 Application à la méthode basée sur la connaissance d'un seul éclairage . . . . . 141
6.3.2 Application à la méthode basée sur la connaissance de plusieurs éclairages . . . . . 142
6.3.3 Rendu photographique . . . . . 157
6.4 Discussion . . . . . 159

**7 Comparaisons et améliorations 161**
7.1 Discussion sur les méthodes présentées . . . . . 161
7.1.1 Différences algorithmiques . . . . . 161
7.1.2 Utilisation appropriée de chaque méthode . . . . . 164
7.2 Amélioration de la méthode basée sur la connaissance de plusieurs éclairages 164
7.2.1 Déplacement des objets réels . . . . . 165
7.2.2 Parallélisme . . . . . 165
7.3 Combinaison des avantages des deux méthodes . . . . . 166
7.4 Discussion . . . . . 166

**8 Conclusion 167**
8.1 Contributions . . . . . 168
8.1.1 Méthode de ré–éclairage basée sur la connaissance d'un éclairage unique . . . . . 168
8.1.2 Méthode de ré–éclairage basée sur la connaissance de plusieurs éclairages . . . . . 168
8.1.3 Calibrage photométrique . . . . . 169
8.2 Perspectives . . . . . 169
8.2.1 Spécularité . . . . . 170
8.2.2 Changement de point de vue . . . . . 171
8.2.3 Scènes d'extérieur . . . . . 171
8.2.4 Vers le temps réel... . . . . 172

**A Affichage multi–passes 173**

**B Programme de capture automatique de photographies 175**
B.1 Propriétés contrôlables de l'appareil photographique Kodak DC260 . . . . . 175
B.2 Script de capture de photographies sur une plage étendue . . . . . 176

# Table des figures


1.1 Exemple réel de design d'intérieur . . . . . 19
1.2 Exemple de manipulation difficile . . . . . 19

2.1 Angles et paramètres décrivant la luminance . . . . . 24
2.2 Angles et paramètres décrivant une BRDF . . . . . 25
2.3 Différents modèles de réflectance . . . . . 26
2.4 Gonioréflectomètres . . . . . 28
2.5 Description des paramètres de facteur de forme . . . . . 31
2.6 Maillage quadtree . . . . . 33
2.7 Liens en radiosité hiérarchique . . . . . 34
2.8 Hiérarchie de clusters . . . . . 35
2.9 Liens et shafts en radiosité hiérarchique dynamique . . . . . 39
2.10 Exemple d'outils de reconstruction à base d'images . . . . . 44
2.11 Résultats de Sato *et al.* [SWI97] . . . . . 46
2.12 Résultats de Marschner *et al.* [MG97] . . . . . 47
2.13 Résultats de Marschner [Mar98] . . . . . 48
2.14 Résultats de Debevec [Deb98] . . . . . 53
2.15 Résultats de Yu et Malik [YM98] . . . . . 55
2.16 Résultats de Yu *et al.* [YDMH99] . . . . . 55

3.1 Appareils photographiques que nous utilisons . . . . . 61
3.2 Mire de calibrage . . . . . 62
3.3 Les quatre mosaïques servant d'entrée au programme TotalCalib . . . . . 63
3.4 Le système *TotalCalib* . . . . . 63
3.5 Modèle reconstruit avec le système *TotalCalib* . . . . . 64
3.6 Modèle texturé construit avec *TotalCalib* . . . . . 64
3.7 Calibrage de la caméra . . . . . 65
3.8 Exemples de mosaïques . . . . . 66
3.9 Le système *Rekon* . . . . . 68
3.10 Application des contraintes dans *Rekon* . . . . . 69
3.11 Exemple de modèle reconstruit par *Rekon* . . . . . 69
3.12 Lampe utilisée . . . . . 71
3.13 Paramètres pour la perspective . . . . . 73

4.1 Algorithme complet . . . . . 78

4.2 Le modèle reconstruit par un outil de reconstruction . . . . . 79
4.3 Le modèle complet incluant quatre lampes réelles . . . . . 80
4.4 Exemple d'éclairage commun par la radiosité hiérarchique . . . . . 81
4.5 Échec de ré–éclairage . . . . . 82
4.6 Première solution de radiosité . . . . . 84
4.7 Algorithme du raffinement basé sur les textures . . . . . 85
4.8 Raffinement basé sur les textures . . . . . 86
4.9 Correction des textures en ajoutant l'éclairage bloqué . . . . . 87
4.10 Délimitation des régions à modifier . . . . . 89
4.11 Algorithme de sélection d'un élément de référence . . . . . 89
4.12 Texture après une correction basée sur la couleur . . . . . 91
4.13 Radiosité après la correction des textures . . . . . 92
4.14 Reprojection des ombres . . . . . 92
4.15 Reprojection des ombres sur les nouvelles textures par une nouvelle solution de radiosité . . . . . 93
4.16 Modification interactive de l'intensité d'une lampe virtuelle . . . . . 94
4.17 Résultats de ré–éclairage . . . . . 95
4.18 Insertion d'une lampe virtuelle . . . . . 96
4.19 Insertion et déplacement d'un objet virtuel . . . . . 97
4.20 Textures corrigées . . . . . 97
4.21 Résultat de ré–éclairage . . . . . 99

5.1 Photographies servant à la reconstruction 3D de la scène réelle . . . . . 104
5.2 Modèle 3D reconstruit . . . . . 105
5.3 Lampe utilisée pour éclairer les scènes lors de nos tests . . . . . 105
5.4 Images de luminance servant à l'estimation de la réflectance . . . . . 106
5.5 Structure de données pour l'estimation de la réflectance . . . . . 108
5.6 Numérotation des polygones en utilisant un « item buffer » . . . . . 109
5.7 Images de réflectance, et coefficients de validité . . . . . 111
5.8 Algorithme de filtrage . . . . . 112
5.9 Résultats à chaque étape du filtrage . . . . . 113
5.10 Réflectance estimée . . . . . 115
5.11 Structure de données utilisée pour le rendu . . . . . 116
5.12 Éléments de la radiosité . . . . . 117
5.13 Polygones invisibles depuis l'image originale . . . . . 118
5.14 Photographies additionnelles pour l'estimation de la réflectance des polygones invisibles . . . . . 118
5.15 Texture servant à l'estimation plus approximative de la réflectance pour le mur . . . . . 119
5.16 Reprojection des ombres . . . . . 120
5.17 Structure de shaft . . . . . 121
5.18 Localisation des modifications lors de l'insertion d'un objet virtuel . . . . 122
5.19 Algorithme général pour la modification de la géométrie . . . . . 124
5.20 Masques utilisés pour enlever l'ordinateur portable . . . . . 126

5.21 Enlèvement d'un objet réel . . . . . 127
5.22 Modifications de la géométrie pour la première scène testée . . . . . 130
5.23 Modifications de l'éclairage pour la première scène testée . . . . . 131
5.24 Résultats sur une deuxième scène testée . . . . . 132

6.1 Fonction de caméra de Debevec et Malik [DM97] . . . . . 139
6.2 Photographies pour calculer la fonction de la caméra. . . . . 140
6.3 Fonction de la caméra que nous avons retrouvée . . . . . 141
6.4 Nouvelles textures corrigées à partir d'images de luminance . . . . . 143
6.5 Résultats de ré–éclairage en utilisant des images de luminance . . . . . 144
6.6 Zoom sur les résultats de ré–éclairage en utilisant des images de luminance 145
6.7 Photographie de la lampe allumée ayant servi à la capture . . . . . 146
6.8 Mesure de la propriété de diffusion de la lampe servant à la capture . . . . 147
6.9 Algorithme de recalage . . . . . 148
6.10 Images obtenues après la première étape du recalage . . . . . 149
6.11 Réflectance de référence, utilisée pour le recalage des images . . . . . 150
6.12 Images originales et images recalées . . . . . 151
6.13 Réflectance estimée avec des photographies calibrées . . . . . 152
6.14 Réflectance estimée avec des photographies calibrées . . . . . 153
6.15 Nouvel algorithme d'estimation des réflectances . . . . . 153
6.16 Réflectance estimée après dix itérations avec des photographies calibrées 154
6.17 Convergence de l'algorithme d'estimation de la réflectance . . . . . 155
6.18 Reprojection des ombres avec la nouvelle réflectance . . . . . 155
6.19 Nouvelles images de réflectance pour une deuxième scène . . . . . 156
6.20 Ré–éclairage avec la nouvelle réflectance pour la deuxième scène . . . . 157
6.21 Zoom sur le ré–éclairage avec la nouvelle réflectance pour la deuxième scène 158

7.1 Comparaison des deux méthodes de ré–éclairage . . . . . 162

A.1 Algorithme d'affichage en multi–passes . . . . . 174
A.2 Utilisation d'un affichage multi–passes . . . . . 174

# Chapitre 1

# Introduction

La *réalité augmentée assistée par ordinateur* est un domaine émergeant, qui progresse rapidement. Son principe est de mélanger le monde réel et le monde virtuel. De nombreuses applications utilisent déjà la réalité augmentée, notamment dans le domaine du jeu vidéo, du cinéma, de la télévision, ou encore dans des systèmes d'aide et d'apprentissage. Il existe deux façons distinctes pour visualiser des environnements en réalité augmentée. Dans la première, la réalité est perçue au travers de lunettes semi–transparentes sur laquelle se superposent des informations virtuelles. Dans la deuxième, il existe une représentation synthétique de la réalité à laquelle sont mélangées les informations virtuelles. Le mélange est affiché sur ordinateur ou sur des lunettes d'immersion. Cette représentation synthétique de la scène réelle est souvent un modèle géométrique simplifié et texturé. Dans la suite du document, nous appellerons *scènes réelles* une telle représentation de la réalité.

Utiliser des scènes réelles ou des images réelles présente de nombreux avantages par rapport à l'utilisation d'environnements purement virtuels, modélisés à la main. En effet, la modélisation demande des efforts importants pour obtenir une géométrie complexe des scènes virtuelles nécessaire au réalisme. L'utilisation de scènes réelles permet de réduire considérablement ces efforts, car même représentées par un modèle texturé très simple, les « scènes » réelles sont très réalistes. En effet, les scènes reconstruites comportent moins de polygones que les scènes virtuelles du même type, et sont ainsi plus faciles à traiter. L'utilisateur est alors confronté à un environnement familier, dans lequel il s'adapte facilement, ce qui permet une immersion plus rapide et plus naturelle. Leur utilisation présente donc un avantage à la fois algorithmique et visuel.

L'objectif des recherches menées jusqu'à présent en réalité augmentée a principalement porté sur le calibrage en *temps réel* des scènes réelles et le placement en temps réel des objets virtuels dans la scène réelle, comme le décrit l'étude d'Azuma [Azu97]. En réalité augmentée, on entend par *temps réel* un taux de rafraichissement de moins de trente images par seconde pour des algorithmes de calibrage et d'alignement, afin d'éviter un délai qui provoquerait des incohérences visuelles.

Notre intérêt concerne plutôt l'*éclairage commun* entre les objets réels et les objets virtuels. L'éclairage commun consiste à prendre en compte les interactions lumineuses (ombres, réflexions) entre les lampes réelles et virtuelles et les objets réels et virtuels. La prise en compte de ces effets aide à obtenir des images plus réalistes, ainsi qu'une cohérence

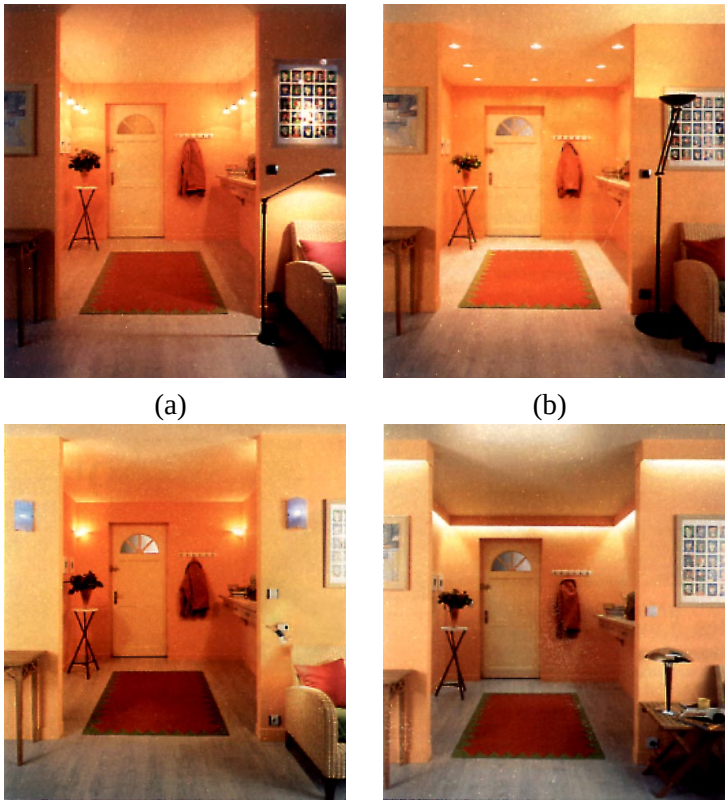
visuelle de l'éclairage. Lorsqu'un objet virtuel ne projette pas d'ombre dans la scène où il est inséré, il est visuellement très difficile à localiser. L'information de l'ombre permet alors de connaître les distances entre les objets, et leurs positions relatives. Pour simuler ces effets d'ombre, il est courant d'utiliser des méthodes de simulation d'éclairage. Cependant, la simulation d'éclairage en synthèse d'images requiert des calculs complexes, non réalisables en temps réel. La plupart des travaux précédents [JNP+95, NHIN86, YM98, SSI99, Deb98, FGR93, DRB97] traitant l'éclairage commun se sont focalisés sur la qualité des images produites, en ignorant l'aspect temps de calcul.

## 1.1 Motivations

Contrairement aux approches existantes, nous désirons dans cette thèse nous rapprocher du temps réel pour permettre l'interaction de l'utilisateur avec la scène réelle. Nous aimerions fournir des outils pour produire des montages photographiques réalistes dans des temps interactifs (une à trois images par seconde). Nous nous intéressons plus particulièrement au ré–éclairage d'intérieur pour fournir des outils utilisables par des professionnels du design d'intérieur. Actuellement, dans le design d'intérieur, la plupart des décorateurs réalisent leur conception sur dessin ou maquette réelle. Pour de gros travaux, comme dans le cas des musées par exemple, l'intérieur est entièrement modélisé sur ordinateur, et l'environnement est traité de façon complètement virtuelle. Une autre solution est la construction de maquettes, de taille réelle ou réduite, ce qui implique un coût financier très élevé et un investissement de temps très important.

Un exemple typique est montré sur les photographies présentées dans la figure 1.1. Ces images ont été extraites d'une publicité illustrant différents types d'éclairage possibles. Le concepteur d'intérieur a présenté différents types d'éclairage (ampoules, halogènes, néons) sur différents supports. Pour mettre en évidence ces différents éclairages et le style associé, le concepteur d'intérieur a pris le soin de changer les tableaux fixés au mur. De plus, pour la photographie (d), il a déplacé le canapé pour ajouter une table, sur laquelle il a posé une nouvelle lampe. Pour obtenir ces quatre photographies, le concepteur d'intérieur a dû déplacer et ajouter des objets, enlever des lampes et en ajouter des nouvelles, et finalement choisir une bonne intensité lumineuse pour chacune des lampes. Le concepteur d'intérieur a réalisé un travail manuel important, tant au niveau du placement des meubles, qu'au niveau de l'installation électrique. On peut également imaginer que de nombreux essais lui ont été nécessaires avant d'obtenir les résultats présentés sur ces photographies.

Dans cette thèse, nous souhaitons apporter de nouveaux outils pour assister les tâches de la décoration et du design. Plutôt que de travailler soit entièrement dans la réalité, soit sur dessin, soit sur une scène exclusivement virtuelle, le travail est réalisé sur ordinateur, en utilisant une représentation de la scène réelle obtenue à partir de quelques photographies de l'environnement existant. Le système d'interaction avec la scène réelle doit permettre de tester différents éclairages et de changer (jusqu'à un certain degré) la géométrie de la scène. Il est important que cette nouvelle méthode de travail soit le moins fastidieuse possible, et plus rapide que celle que le concepteur avait l'habitude d'utiliser ; il nous a donc paru indispensable que la phase de capture de l'environnement réel soit rapide et simple.



(a) (b)

(c) (d)

FIG. 1.1: Photographies montrant la même pièce, prise depuis le même point de vue, mais sous un éclairage différent. Pour obtenir ces photographies, le concepteur d'intérieur a changé les lampes, et le style de la pièce, en changeant les tableaux accrochés aux murs et en déplaçant quelques meubles, comme le canapé dans l'image (d). Ceci lui a permis d'ajouter une table sur laquelle il a posé une nouvelle lampe.



(a) (b)

FIG. 1.2: Ces images ont été simulées à partir d'un des systèmes développés dans cette thèse (voir le chapitre 5). L'image (a) représente la scène réelle originale. Dans l'image (b), nous avons virtuellement enlevé la porte (en 1.2 secondes), de façon convaincante. Cette manipulation nécessiterait dans la réalité des travaux importants et fastidieux.

De plus, les systèmes utilisés sur ordinateur doivent produire des résultats réalistes dans des temps interactifs, et permettre certaines modifications que nous n'aurions pas pu faire aisément dans la réalité. Par exemple, avec une des méthodes que nous avons développées, nous pouvons interactivement enlever des objets, tels qu'une porte comme le montre la figure 1.2, ce qui serait fastidieux dans la réalité.

## 1.2 Contributions

Dans cette thèse, nous présentons les différentes méthodes que nous avons développées pour atteindre l'objectif du ré–éclairage et de la remodélisation dans des temps interactifs. Pour chacune de ces méthodes, nous nous sommes efforcés de n'utiliser que des données d'entrée simples (non exhaustives), et un matériel non spécifique, facilement accessible. Pour les deux méthodes de ré–éclairage, nous considérons que les scènes réelles ont un éclairage diffus, et qu'elles sont visualisables depuis un point de vue fixe.

Nous avons développé une première méthode [LDR98, LDR99] qui permet le ré–éclairage interactif des scènes réelles, en modifiant l'intensité lumineuse des lampes réelles, et en insérant de nouvelles sources de lumière dans la scène. Elle permet également d'insérer et de déplacer des objets virtuels en temps interactif. Dans cette méthode, notre connaissance de la scène réelle est limitée à une photographie, depuis laquelle nous extrayons des textures à appliquer au modèle géométrique simple de la scène réelle. L'éclairage original est déjà inclus dans les textures, et resterait donc perceptible si on le modifiait d'une façon naïve. Pour permettre le ré–éclairage, nous transformons les textures d'origine pour qu'elles représentent un éclairage sans ombre. Depuis ces nouvelles textures, des propriétés radiométriques sont estimées et un nouvel éclairage est simulé. L'affichage est accéléré par l'utilisation du matériel graphique. Après cette phase de pré–traitement, l'utilisateur peut modifier l'éclairage réel et virtuel, et déplacer les objets d'une façon interactive grâce à des méthodes incrémentales.

Cette méthode comporte cependant quelques limitations. Premièrement, la qualité des textures transformées n'est pas garantie. À la fin du traitement, les textures peuvent contenir des artefacts. Deuxièmement, les propriétés radiométriques ne sont pas toujours correctement estimées, car nous ne connaissons la scène réelle que sous un seul éclairage. De plus, l'utilisation du rendu par polygone, assisté par le matériel d'affichage graphique, nécessite l'utilisation de textures, ce qui limite notre contrôle sur les valeurs exactes de chaque pixel affecté. Nous avons développé une deuxième méthode [LFD+99b, LFD99a] basée sur une approche différente, qui utilise plusieurs photographies de la scène réelle sous des éclairages différents. Elle autorise des modifications interactives comme le ré–éclairage, l'insertion d'objets virtuels, ainsi que l'enlèvement d'objets réels (non permis par la première méthode), ce qui était l'un de nos objectifs (voir figure 1.2). Dans cette approche, nous travaillons pixel par pixel pour estimer les propriétés radiométriques, plutôt que sur des polygones texturés comme c'est le cas dans la première méthode. Ceci permet un contrôle plus fin de l'estimation des valeurs. Les propriétés radiométriques sont estimées à partir de plusieurs photographies de la même scène, prises sous différents éclairages contrôlés. Ceci nous permet d'avoir une connaissance indépendante de l'éclairage

original de la scène. Les modifications sont localisées et mises à jour dans des temps interactifs, comme par exemple en 1.5 secondes pour supprimer la porte sur la figure 1.2.

Nous avons utilisé pour ces deux méthodes des images générées par tout appareil photographique numérique. Malgré l'utilisation de ces données brutes, nous montrons plusieurs exemples pour lesquels les résultats de ré–éclairage correspondent à nos attentes. Nous avons cependant développé une nouvelle méthode de calibrage de données simples, pour lever les limitations possibles. Nous perdons notamment de l'information lorsqu'il y a une saturation soit vers des couleurs trop sombres, soit vers des couleurs trop claires. De plus, la caméra modifie les couleurs en appliquant sur la photographie une fonction de transfert liée à ses caractéristiques. Le traitement pour estimer l'information radiométrique peut donc aboutir à un résultat erroné. Pour éviter ces problèmes, nous avons utilisé une méthode de calibrage des données photographiques, avec des photographies prises à différents temps d'exposition pour avoir une information plus complète. Cette méthode a été adaptée à l'appareil photographique que nous avons utilisé, pour lequel le contrôle manuel est limité, notamment pour le temps d'exposition. Nous avons réduit les artefacts de la première méthode de ré–éclairage en utilisant les images ainsi calibrées. Pour la deuxième méthode, nous avons appliqué un deuxième algorithme de calibrage, pour assurer une cohérence des images pour toutes les différentes conditions d'éclairage. De plus, nous avons amélioré le processus d'estimation des propriétés de l'éclairage en utilisant un algorithme basé sur une mise à jour itérative de l'éclairage indirect et des valeurs de réflectance. La qualité de la réflectance estimée est sensiblement meilleure après cette méthode itérative.

Nous avons enfin comparé les deux méthodes présentées, en tenant compte de l'amélioration obtenue après traitement des données photographiques. Nous avons fait ressortir les avantages et les inconvénients de chacune. Notamment, la deuxième méthode est plus propice à des extensions. Nous montrons que déplacer des objets réels est possible grâce à l'utilisation d'une texture dépendante du point de vue extraite de l'image de réflectance (ne contenant plus d'effets lumineux). L'interactivité nous apparaissant comme très importante, nous montrons également que l'algorithme peut être modifié pour permettre le calcul sur une machine parallèle.

## 1.3 Structure du document

Le ré–éclairage et la remodélisation de scènes réelles est un problème très difficile, qui recouvre plusieurs domaines, et différents types de recherches telles que :

- la reconstruction de scènes réelles à partir d'images pour transformer des informations dans le repère image en informations géométriques dans l'espace,
- la simulation d'éclairage réaliste en synthèse d'images,
- l'*illumination inverse* qui permet de retrouver les propriétés radiométriques des scènes réelles,
- et enfin l'*éclairage commun* pour la réalité augmentée, qui prend en compte les interactions lumineuses entre les objets réels et virtuels.

Nous nous appliquons dans le chapitre 2 à donner les éléments nécessaires pour la compréhension du document. Nous détaillons les travaux précédents dans les domaines de l'illumination inverse et de l'éclairage commun sans pour autant rentrer dans les détails de chaque domaine de recherche.

Travailler avec les scènes réelles n'est pas quelque chose de simple, car les inconnues sont beaucoup plus nombreuses que dans la synthèse d'images purement virtuelles. Nous présentons dans le chapitre 3 un résumé du travail de préparation et d'acquisition nécessaire avant de travailler sur les méthodes de ré–éclairage des scènes réelles. Nous expliquons les difficultés rencontrées et les solutions choisies, ce qui fournit ainsi une information pour les personnes désirant travailler dans ce domaine. Dans ce travail de mise en place préliminaire, il faut choisir le matériel servant à l'acquisition des scènes réelles, tel que les caméras, les lampes, et les outils de reconstruction. Ensuite, il faut transformer les données pour qu'elles soient utilisables dans nos systèmes.

Nous décrivons ensuite les différentes méthodes de recherche que nous avons développées.

La première méthode de ré–éclairage s'appuie sur la connaissance d'un seul éclairage pré–existant. Nous présentons dans le chapitre 4, les différents algorithmes que nous avons développés pour permettre le ré–éclairage d'une scène réelle représentée par un modèle texturé à partir de cette unique photographie. Avec des algorithmes incrémentaux, l'éclairage original peut être modifié, et des lampes et des objets virtuels peuvent être ajoutés à la scène réelle dans des temps interactifs.

Nous présentons ensuite dans le chapitre 5 la deuxième méthode de ré–éclairage, basée sur une connaissance plus étendue de la scène réelle, éclairée différemment sur plusieurs photographies. Les propriétés radiométriques peuvent être estimées de façon plus précise. La simulation du ré–éclairage et de l'éclairage commun est réalisée dans des temps interactifs, en combinant une localisation des informations à modifier dans l'espace 3D et dans l'espace image. Nous pouvons interactivement manipuler les intensités des lampes, ajouter des objets virtuels et des lampes virtuelles, ainsi que supprimer des objets réels de la scène.

Ces deux méthodes de ré–éclairage ont été développées pour donner de bons résultats avec des données simples, nécessitant une capture facile et rapide. Elles peuvent néanmoins être améliorées avec des données mieux calibrées. Nous présentons dans le chapitre 6, une nouvelle méthode de calibrage de données photographiques, adaptée à un appareil photographique dont le contrôle manuel est limité. Grâce à un meilleur calibrage des données, nous pouvons ainsi améliorer la qualité des résultats.

Enfin, dans le chapitre 7, nous comparons les algorithmes, les résultats et les applications possibles des deux méthodes de ré–éclairage. Nous proposons ensuite quelques améliorations, comme le déplacement d'objets réels et la parallélisation de l'algorithme pour accélérer les calculs.

La conclusion et les perspectives sont présentées dans le chapitre 8.

# Chapitre 2

# État de l'art

Dans cette thèse, notre objectif est de mélanger des images réelles avec des images virtuelles, en conservant et en simulant des effets d'éclairage réalistes. Notre souci est d'obtenir cette composition dans un temps interactif. À ces fins, nous sommes amenés à utiliser des travaux de plusieurs domaines.

La première composante nécessaire est la simulation de l'éclairage. Nous présentons tout d'abord les différents éléments permettant de décrire la propagation de la lumière dans un environnement (section 2.1). La simulation d'éclairage en synthèse d'images est brièvement présentée dans la section 2.2. Il existe plusieurs méthodes permettant de simuler un rendu phot–réaliste. Dans la section 2.2.1, nous présentons la radiosité qui est la méthode que nous utilisons dans cette thèse. La simulation de l'éclairage est réalisée par des algorithmes complexes. La manipulation des objets dans une scène n'est pas interactive si on utilise les algorithmes initiaux. Différentes méthodes ont été développées pour permettre l'interactivité dans les scènes de synthèse d'images tout en gardant un rendu de qualité (section 2.3).

Ces travaux précédents concernent la simulation de l'éclairage pour des environnements entièrement virtuels. Dans nos travaux, nous utilisons également des objets réels. Le mélange entre objets virtuels et objets réels est appelé *réalité augmentée*. De nombreuses recherches sont réalisées dans ce domaine. Nous nous intéressons plus particulièrement aux interactions lumineuses entre objets réels et objets virtuels ; la prise en compte de ces interactions s'appelle *éclairage commun*. Dans la section 2.4, nous décrivons quels sont les besoins pour atteindre un tel objectif. Il est d'abord nécessaire d'obtenir un modèle géométrique (section 2.4.2) et radiométrique (section 2.4.3) de la scène réelle. Ensuite, nous obtenons la composition de la réalité avec le synthétique en utilisant des algorithmes spécifiques (section 2.4.4).

## 2.1 Radiométrie

### 2.1.1 Propagation de la lumière

La radiométrie étudie les transferts d'énergie lumineuse, et permet une description de la propagation de la lumière. La *luminance* $L(x, \theta, \phi)$ représente la quantité d'énergie lu-

mineuse se propageant à partir d'un point $x$, dans une direction donnée $(\theta, \phi)$, par unités de temps, d'angle solide et de surface. Les angles $\theta$ et $\phi$ sont décrits sur la figure 2.1. Les unités de la luminance sont en Watts par mètre carré et par stéradian.

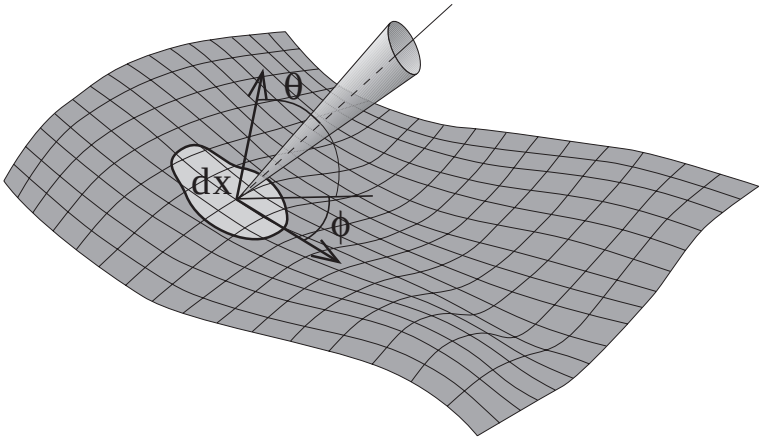



FIG. 2.1: Angles et paramètres décrivant la luminance (Figure extraite de [SP94], page 9).

L'énergie lumineuse pour un angle solide $d\omega$, pour une surface d'aire $dx$, et pour un intervalle de temps $dt$, est égale à :

$$L(x,\theta,\phi)dxcos\theta d\omega dt. \tag{2.1}$$

Cette quantité dépend également de la longueur d'onde. Pour alléger les équations, nous ne ferons pas apparaître ce paramètre, bien qu'il soit présent dans toutes les équations que nous allons présenter.

La puissance $dP$ associée par unité de temps est :

$$d^2P = L(x,\theta,\phi)dxcos\theta d\omega. \tag{2.2}$$

La *radiosité* est définie par la puissance totale quittant une surface, par unité de surface. La radiosité $B$ en un point $x$ est définie à partir de la puissance lumineuse :

$$B(x) = \frac{dP}{dx} = \int_{\Omega} L(x,\theta,\phi)cos\theta d\omega, \tag{2.3}$$

$\Omega$ étant l'hémisphère supérieur représentant l'ensemble des directions $(\theta, \phi)$ couvrant la surface autour du point $x$. Les unités de la radiosité sont en Watts par mètre carré. De la même façon, on peut définir la quantité d'énergie arrivant sur une surface en un point, appelée *irradiance*. Les unités de l'irradiance sont les mêmes que celles de la radiosité.

En plus de réflechir de l'énergie, certaines surfaces peuvent également en émettre (source de lumière). Cette quantité émise est appelée *exitance*. Cette quantité est très importante car c'est d'elle que dépend l'éclairage d'un environnement. Les surfaces comportant une exitance sont les sources de lumière. Comme précédemment, nous pouvons définir une luminance émise $L_e(x,\theta,\phi)$, en un point $x$ et pour une direction $(\theta, \phi)$. Le cas le plus simple

est celui des sources non directionnelles diffuses, pour lesquelles l'exitance ne dépend pas de $\theta$ et $\phi$. Cependant, dans la réalité, la plupart des sources de lumière sont directionnelles. De façon similaire à la radiosité, l'*exitance* est définie par :

$$E(x) = \int_{\Omega} L_e(x,\theta,\phi) cos\theta d\omega. \tag{2.4}$$

### 2.1.2 Fonction de distribution de réflectance bidirectionnelle

**Définition**

La façon dont la lumière est réfléchie par les surfaces du monde réel dépend de la composition et de la structure de ces surfaces. Pour chaque surface, on peut définir une *réflectance* qui décrira comment la lumière arrivant en un point est absorbée puis réfléchie. La fonction de distribution de réflectance bidirectionnelle (*BRDF*, en anglais « Bidirectional Reflectance Distribution Function ») décrit la propriété de réflexion d'une surface. En particulier, elle représente le rapport entre la luminance $L$ (énergie sortante), et l'irradiance $L_i$ (énergie incidente). Elle dépend donc de la direction d'incidence $(\theta,\phi)$ et de la direction sortante $(\theta_0,\phi_0)$ (figure 2.2).

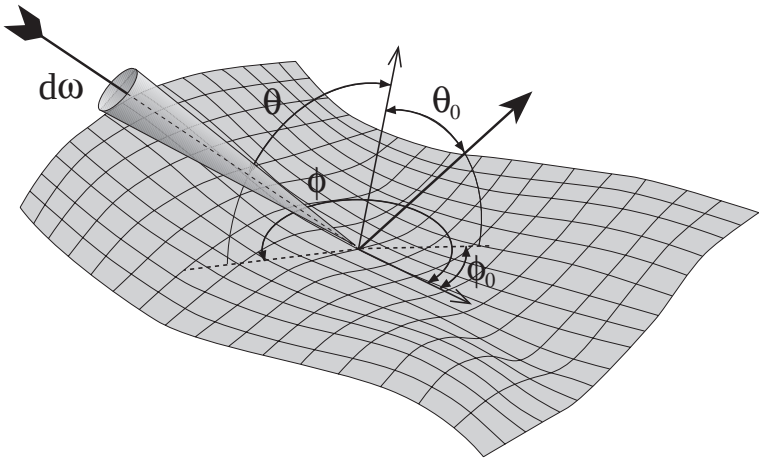



FIG. 2.2: Angles et paramètres décrivant une BRDF (Figure extraite de [SP94], page 15).

La réflectance bi–directionnelle $\rho_{bd}$ est donc égale à :

$$\rho_{bd}(\theta_0,\phi_0,\theta,\phi) = \frac{L(x,\theta_0,\phi_0)}{L_i(x,\theta,\phi) cos\theta d\omega}. \tag{2.5}$$

Les chercheurs en synthèse d'images ont approximé cette fonction par différentes compositions de termes simples, à partir d'un modèle d'optique physique générale [HTSG91]. Le modèle diffus directionnel dépend de façon complexe de la direction, en fonction de la rugosité des surfaces, et d'autres caractéristiques physiques. Le modèle de réflectance peut ensuite être simplifié, et les propriétés des surfaces classifiées.

En synthèse d'image, une surface dite complètement *diffuse* ou *lambertienne*, diffusera la lumière reçue en intensité égale dans toutes les directions. La propriété de *spécularité* correspond au contraire à une diffusion unidirectionnelle, symétrique à la direction de la lumière incidente par rapport à la normale de la surface. Une surface purement spéculaire est un miroir parfait. La combinaison des propriétés de diffusion et de spécularité est appelée propriété diffuse directionnelle. Ces différents modèles de réflectance sont illustrés sur la figure 2.3. Dans la nature, les surfaces se comportent d'une façon bien plus complexe, qui peut être vu comme à la fois diffus et spéculaire.

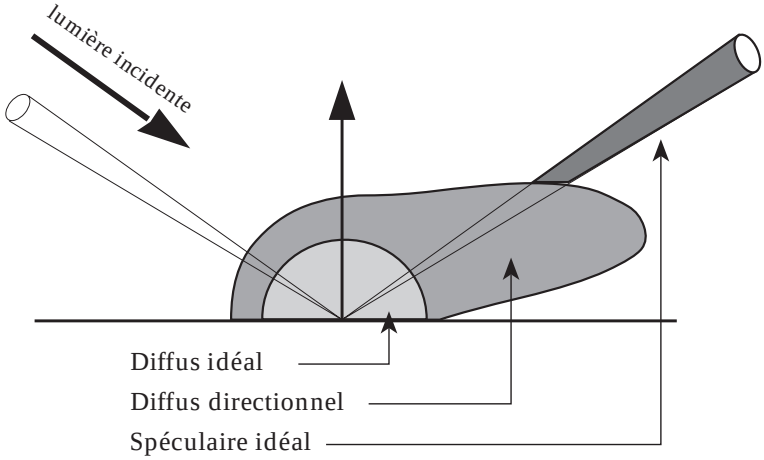



FIG. 2.3: Différents modèles de réflectance (Figure extraite du livre de [SP94], page 19).

La définition de la BRDF donnée par l'équation (2.5) peut être utilisée pour des mesures physiques comme nous le décrivons ci–dessous. Les données résultantes peuvent être utilisées comme échantillons mais ceci nécessite un stockage bien trop coûteux. Pour permettre un simulation en synthèse d'images, des modèles analytiques simplifiés ont été développés. Un des plus utilisés dans le domaine du graphisme est le modèle de Phong [Pho75], dont la représentation décompose simplement la BRDF en deux composantes : une composante diffuse, et une composante spéculaire (non idéale).

Le modèle de Torrance–Sparrow [TS67], simplifié par Blinn [Bli77] et Cook et Torrance [CT82], permet d'avoir un modèle plus physique, en introduisant l'hypothèse que les surfaces soient formées de microfacettes. Cependant, dans ce modèle, les facettes sont supposées planes et orientées uniformément, ce qui correspond à un modèle *isotrope*. L'apparence visuelle d'un tel effet spéculaire est une tache de lumière souvent arrondie pour certains points de vue.

Pour simuler un éclairage virtuel, il faut donc définir une BRDF pour chacune des surfaces. Il est souvent difficile de déterminer les paramètres correspondant à certains matériaux réels. De plus, pour obtenir un rendu efficace, il est préférable de trouver une bonne appriximation des modèles de réflexion.

#### Mesure de la BRDF

Il existe des appareils pour mesurer la réflectance des surfaces réelles. Le plus connu est le *gonioréflectomètre*, comme par exemple celui montré sur la figure 2.4 (a) construit par Murray–Coleman et Smith [MCS90]. Cet appareil est composé d'un système mécanique à quatre degrés de liberté et d'un photomètre. Les mesures réalisées avec cet appareil n'ont fourni qu'une BRDF isotrope. Cet appareil ne permet cependant que de mesurer une BRDF isotrope. Une rotation de l'échantillion permettrait des mesures anisotropiques.

Un instrument de mesure de BRDF plus complet a été défini par Ward [War92]. Cet appareil mesure aussi le cas de spécularité *anisotrope*, ce qui correspond à des microfacettes orientées non uniformément. L'effet visuel correspond à une tache de lumière plus allongée (pour certains points de vue). Pour mesurer cette composante, Ward a utilisé un appareil appelé *Imaging Gonioreflectometer* montré sur la figure 2.4 en (b) et (c), développé par le groupe de recherche sur les systèmes d'éclairage au laboratoire Lawrence Berkeley. Cet appareil est conçu de façon à capturer non pas l'image du matériau lui–même, mais sa réflexion sur un miroir en hémisphère. La capture est réalisée par une caméra CCD avec une lentille *fish–eye*. Un mécanisme permet de contrôler la position d'une lampe halogène, et la position de l'échantillon $A$ du matériau.

Ces appareils de mesure sur échantillons ne permettent pas d'évaluer la réflectance sur tous les objets de notre environnement. Deux surfaces d'un même matériau peuvent avoir une réflectance différente car elles auront vieilli différemment, ou encore parce que les motifs et couleurs qui les composent seront différents. De plus, il n'est pas toujours possible d'avoir accès à un échantillon d'un objet pour pouvoir évaluer ses propriétés, et la taille des échantillions nécessaire pour une mesure fiable n'est pas la même pour tous les objets. Certaines recherches ont été menées pour évaluer différemment les propriétés des matériaux. Ces travaux sont décrits dans la section 2.4.3.

### 2.1.3 Rendu

#### Équation du rendu

Afin de reproduire virtuellement les conditions d'éclairage réaliste d'un environnement, il faut combiner la façon dont la lumière se propage et la façon dont les surfaces réfléchissent cette lumière. Dans un environnement fermé, on parle d'*éclairage global*. En effet, la lumière produite par des surfaces émétrices va fournir un éclairage *direct*. Les surfaces réflechissent ensuite la lumière et fournissent un éclairage *indirect*.

L'*équation du rendu* [Kaj86] permet de décrire de façon globale tous les échanges d'énergie entre les surfaces d'un environnement :

$$L(x,\theta_0,\phi_0) = L_e(x,\theta_0,\phi_0) + \int_\Omega \rho_{bd}(x,\theta_0,\phi_0,\theta,\phi) L_i(x,\theta,\phi) cos\theta d\omega. \qquad (2.6)$$

Cette équation montre que la luminance totale en un point $x$ est égale à la luminance émise, plus la luminance réfléchie. Cette équation comporte une intégrale, et il est impossible de la résoudre dans le cas général de façon analytique. Des méthodes telles que les

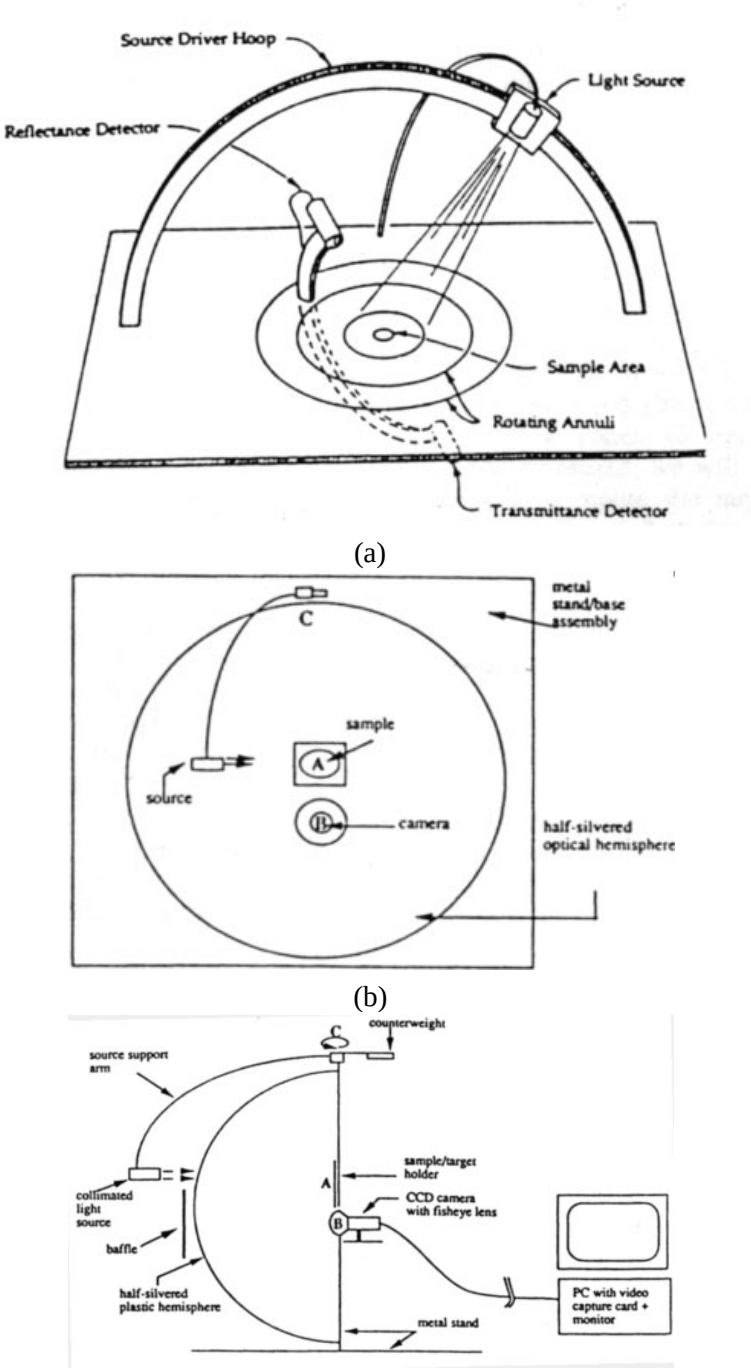



FIG. 2.4: (a) Gonioréflectomètre construit par Murray–Coleman et Smith. (b) et (c) Gonioréflectomètre à images permettant de mesurer une réflectance spéculaire anisotrope. (b) Vue de dessus. (c) Vue de coté. (Figures extraites de l'article de [War92]).

méthodes de radiosité ou les méthodes par lancer de rayons, permettent de simuler l'éclairage en résolvant cette équation. La méthode par lancer de rayons va simuler l'éclairage à partir des pixels, alors que la méthode de radiosité discrétise explicitement les surfaces de l'environnement pour simuler l'équilibre énergétique. Ces méthodes sont expliquées dans la section suivante 2.2.

## 2.2 Simulation de l'éclairage

Les équations établies dans la section précédente 2.1 doivent être transformées en algorithmes pour pouvoir simuler l'éclairage global. Il existe de nombreuses méthodes que l'on peut séparer en deux classes : les méthodes par *lancer de rayons* et les méthodes de *radiosité*.

Les méthodes par lancer de rayons peuvent permettre de simuler un éclairage complexe de haute qualité. La première méthode proposée [Whi80] permet de simuler des effets spéculaires. Depuis, d'autres méthodes ont été développées, qui prennent en compte les effets plus complexes. Le principe de la méthode par lancer de rayons est de calculer localement une valeur d'éclairage, pour un point d'une surface intersectée par un rayon depuis le point de vue au travers d'un des pixels de l'image. Pour cela, de nombreux rayons sont envoyés vers les surfaces émétrices ainsi que les autres surfaces de la scène. Dans une méthode de lancer de rayons classique, un nombre considérable de rayons sont envoyés et réflechis afin de simuler un éclairage avec un minimum d'artefacts. Ce type de simulation est très coûteux et dépendant du point de vue. Si l'utilisateur veut voir la scène simulée sous un autre point de vue, la solution doit être entièrement recalculée.

Une méthode générale de lancer de rayons appliquant l'équation du rendu est le tracer de chemins [Kaj86]. Un autre système appelé *RADIANCE* [War94b] a été développé, essayant de résoudre l'équation du rendu pour de nombreuses conditions. Ces conditions incluent les réflections spéculaires, diffuses et directionnelles–diffuses, pour des surfaces de type varié, y compris des surfaces courbes. La simulation de l'éclairage global est réalisée par une combinaison de techniques de lancer de rayons, déterministes et stochastiques.

Les méthodes de radiosité sont basées sur les éléments finis. L'environnement est subdivisé en facettes, et une valeur de radiosité est calculée pour chacune de ces facettes. Le résultat est indépendant du point de vue, ce qui permet une visualisation interactive sur des stations de travail graphiques. De plus, son approche par discrétisation permet de choisir entre qualité et temps de calcul. Une simulation approximative et rapide de l'éclairage est possible, si on choisit une subdivision grossière.

Ces deux différentes classes de méthodes présentent des avantages et des inconvénients différents. Dans notre travail, nous utilisons principalement les algorithmes de radiosité, et notamment la radiosité hiérarchique. Les différentes méthodes de radiosité sont présentées dans cette section.

### 2.2.1 Radiosité

#### Équations

Dans la méthode de radiosité, pour simplifier le problème de la simulation de l'éclairage, on se place en général dans le cas de scènes complètement diffuses. La réflectance devient alors une fonction indépendante des angles d'incidence et d'exitance. De cette façon, la radiosité (émise) est indépendante des angles d'exitance. Depuis l'équation (2.6), la radiosité $B(x)$ en un point $x$ peut être écrite comme étant :

$$B(x) = E'(x) + \rho_d(x)E(x), \tag{2.7}$$

où $E'(x)$ est l'exitance, $\rho_d(x)$ la réflectance, et $E(x)$ l'irradiance en ce point. L'indépendance envers les angles d'incidence et d'exitance enlève une des difficultés du problème de la simulation. Cependant, la radiosité d'une surface dépend de la radiosité de toutes les autres surfaces de la scène.

Pour résoudre cette dépendance, la méthode de radiosité utilise un modèle discret. Les surfaces sont découpées en facettes, sur lesquelles on considère que la radiosité est constante, de même que la réflectance. Grâce à cette discrétisation, on peut définir une nouvelle équation, appelée *équation de radiosité* [GTGB84], où pour chaque élément $i$ :

$$B_i = E_i + \rho_i \sum_{j=1}^{N} F_{ij} B_j. \tag{2.8}$$

La décomposition des surfaces de l'environnement en éléments finis, pour résoudre l'équation discrète (2.8), a été tout d'abord utilisée pour les échanges thermiques [Spa63]. Dans l'équation (2.8), pour une facette $P_i$, la radiosité $B_i$, son exitance $E_i$ et sa réflectance $\rho_i$ sont supposées constantes sur toute la facette. L'irradiance est exprimée comme étant la somme sur les $N$ autres facettes, de la radiosité $B_j$ modulée par le facteur de forme $F_{ij}$ entre la facette $P_i$ et la facette $P_j$. Le facteur de forme détermine la portion d'énergie transférée d'une surface à une autre.

#### Facteur de forme

Le facteur de forme est la valeur la plus difficile à calculer, et représente la fraction d'énergie émise par $P_i$ et reçue par $P_j$. Sa formule analytique est :

$$F_{ij} = \frac{1}{A_i} \int_{x \in i} \int_{y \in j} \frac{cos\theta cos\theta'}{\pi r^2} V(x,y) dy dx, \tag{2.9}$$

où $A_i$ est l'aire de la facette $P_i$, $V(x,y)$ est la valeur de la visibilité entre le point $x$ de la facette $P_i$ et le point $y$ de la facette $P_j$, et $r$ est la distance entre $x$ et $y$. Si un objet est placé entre le point $x$ et le $y$, les deux points sont considérés invisibles l'un de l'autre. Les angles $\theta$ et $\theta'$ correspondent chacun à l'angle entre la normale en chaque point et le segment entre ces deux points (figure 2.5).

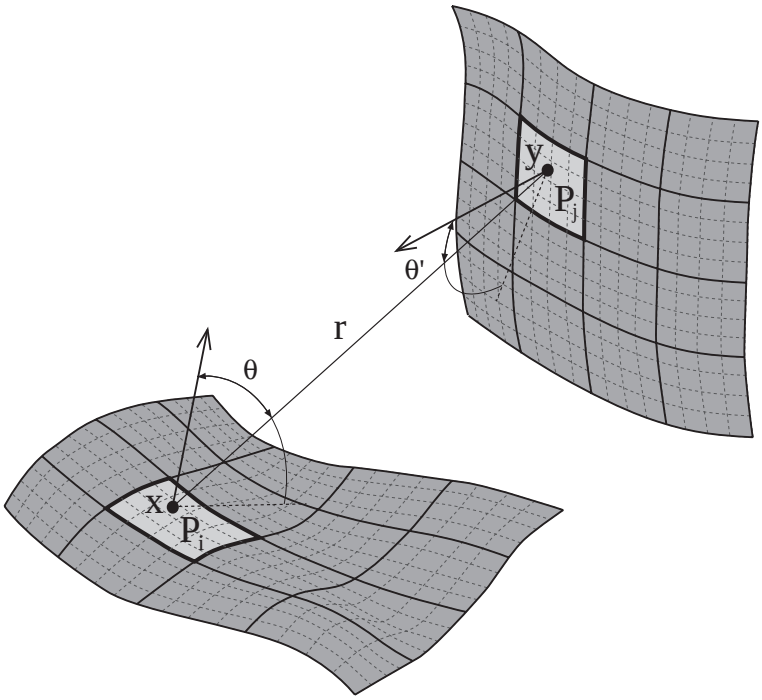



FIG. 2.5: Description des paramètres du facteur de forme entre deux facettes (Figure extraite du livre de [SP94], page 29).

Sa formule analytique étant difficile et coûteuse à calculer, de nombreuses méthodes de simplification des calculs ont été développées. Un première approximation est de calculer la visibilité $V$ de façon indépendante, en la sortant de l'intégrale ; la valeur du facteur de forme résultante est une approximation de la valeur analytique. Le calcul restant à faire est celui d'un facteur de forme indépendant de la visibilité :

$$F_{ij} = \frac{1}{A_i} \int_{x \in i} \int_{y \in j} \frac{cos\theta cos\theta'}{\pi r^2} dydx. \tag{2.10}$$

Une première façon de calculer les valeurs du facteur de forme est d'intégrer la formule sur les contours $C_i$ et $C_j$ des facettes $P_i$ et $P_j$.

$$F_{ij} = \frac{1}{2\pi A_i} \oint_{C_i} \oint_{C_j} \ln r \, \vec{dx} \cdot \vec{dy}. \tag{2.11}$$

$\vec{dx}$ et $\vec{dy}$ sont les vecteurs différentiels sur ces contours. Il est à noter que cette résolution peut poser des problèmes lorsque les surfaces sont proches, et en particulier en contact. Cette équation peut néanmoins être résolue de façon quasi–analytique [Sch93]. Il est aussi possible de résoudre l'équation du facteur de forme entre un point et un polygone par une formule analytique simple [BRW89].

Un calcul plus grossier mais plus rapide peut se faire en ne considérant que le centre des facettes. Ce calcul peut être une approximation justifiée si les surfaces sont relativement

éloignées l'une de l'autre ou si leur aires sont particulièrement petites. L'équation alors utilisée pour le calcul est :

$$F_{ij} = A_j \frac{cos\theta cos\theta'}{\pi r^2}. \tag{2.12}$$

Une autre façon de réaliser les calculs est de procéder par projection sur une sphère de rayon unitaire par exemple (analogie de Nusselt [Nus28]), ou encore sur un hémi–cube [CG85].

Le coût dominant du calcul des facteurs de forme reste cependant le coût des requêtes de visibilité.

#### Méthodes de résolution

Dans l'équation de radiosité, chaque facette est dépendante de la résolution de la radiosité des autres facettes. Nous avons donc un système d'équations à résoudre :

$$\begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{pmatrix} + \begin{pmatrix} \rho_1 F_{11} & \rho_1 F_{12} & \cdots & \rho_1 F_{1n} \\ \rho_2 F_{21} & \rho_2 F_{22} & \cdots & \rho_2 F_{2n} \\ \vdots & & & \\ \rho_n F_{n1} & \rho_n F_{n2} & \cdots & \rho_n F_{nn} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{pmatrix} \tag{2.13}$$

soit en forme matricielle :

$$\mathbf{B} = \mathbf{E} + \mathbf{MB}. \tag{2.14}$$

Pour résoudre ce système, une solution est de procéder par itérations jusqu'à convergence. Il existe deux approches différentes. La première consiste à amasser la lumière sur une facette après l'autre (*gathering*), c'est–à–dire de calculer la radiosité d'une facette après l'autre. Sur l'équation (2.14), ça veut dire qu'un élément de la matrice **B** de gauche sera calculé à partir de toutes les valeurs de radiosités contenues dans la matrice **B** de droite. La solution est calculée par des itérations *Gauss–Seidel*, et nécessite la matrice complète des facteurs de forme. Le principe de la deuxième approche est basé sur la méthode de relaxation de *Southwell*. Elle consiste au contraire à envoyer de la radiosité d'une facette vers toutes les autres facettes (*shooting*). Toujours sur l'équation (2.14), tous les éléments de la matrice **B** de gauche sont initialisés avec un élément de la matrice **B** de droite. Ceci ne nécessite que le calcul d'une colonne de la matrice des facteurs de forme à la fois, ce qui conduit à un raffinement progressif. Les facettes desquelles l'énergie est propagée sont triées et sélectionnées dans l'ordre d'importance de leur contribution à la solution finale. Avec cette approche, les résultats sont en général visuellement plus proches de la solution finale dès les premières itérations, et la place mémoire utilisée est moins grande. Cette méthode est aussi appelée *radiosité progressive* [CCWG88, WEH89].

Il est cependant à noter que les deux méthodes convergent dans un temps équivalent vers la même solution.

### Subdivision

Nous avons jusque là ignoré le problème de la subdivision des surfaces en facettes. Les surfaces pourraient être subdivisées de façon régulière. Cependant, la solution demande une finesse dans la précision à certains endroits comme dans les frontières d'ombre ou les zones d'échanges importants de lumière, et moins à d'autres. Il faut donc établir une subdivision adaptative permettant de ne pas avoir un coût trop élevé dans les zones ne nécessitant pas de calculs très précis. Pour décider de la subdivision, il existe de nombreux critères.

Dans le cas d'une subdivision d'objets à quatre sommets, on utilise en général des « quadtrees » [Sam90]. Cette subdivision consiste à diviser chaque élément en quatre sous–éléments de taille similaire. Une représentation de la subdivision est illustrée sur la figure 2.6 (a). Cet exemple montre aussi une subdivision additionnelle, nécessaire pour éviter une différence de niveau trop grande entre deux éléments. Il existe une approche de subdivision équivalente pour les triangles où chaque élément est divisé en quatre sous–éléments de taille égale (voir figure 2.6 (b)).

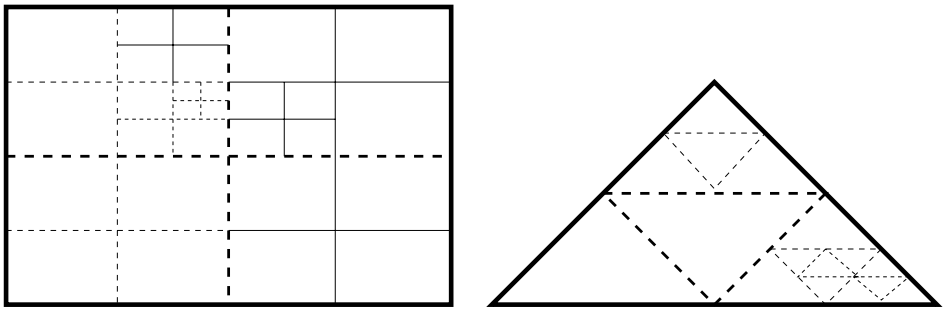



FIG. 2.6: (a) Maillage de type quadtree. Les lignes en continu représentent la maillage additionnel, évitant d'avoir une différence de niveau supérieure à un entre deux éléments voisins (maillage contraint). (b) Maillage de type quadtree appliqué sur les triangles.

D'autres méthodes de maillage peuvent être appliquées comme le maillage de discontinuité [LTG92, Hec92], le maillage construit à partir d'arbres BSP [CF92], etc. Ces types de maillage réduisent souvent des artefacts, comme une représentation en escalier des ombres. Ils sont cependant plus complexes à utiliser et à implanter. De plus le maillage de discontinuité peut se montrer très sensible aux problèmes de stabilité numérique.

### Radiosité hiérarchique

Les deux méthodes de radiosité présentées précédemment n'utilisent dans leur calcul que les éléments « feuilles » de la subdivision. Pourtant, les échanges d'énergie ne nécessitent pas toujours d'être représentés au niveau le plus bas de la subdivision. Les temps de calcul sont passés en majorité dans le calcul des facteurs de forme. Calculer tout au niveau le plus bas peut se révéler très coûteux. Si deux murs sont très éloignés l'un de

l'autre par exemple, il n'y a pas beaucoup de différence dans la solution, que l'on évalue leur échange d'énergie à un niveau bas de la subdivision ou plus haut. On préférerait donc économiser du temps de calcul en estimant cet échange d'énergie à un niveau de subdivision supérieur. Par contre, pour deux éléments plus proches, on préférerait avoir un niveau de représentation plus bas pour les échanges d'énergie. Ce principe de *hiérarchisation* des échanges lumineux est la base de la *radiosité hiérarchique* [HSA91]. Les échanges lumineux se font à différents niveaux de subdivision qui ont conservés ; on parle alors de *hiérarchie des éléments*. Comme pour les méthodes de radiosité de type *gathering* ou progressive, l'algorithme est itératif. Les différents pas de l'itération sont :

1. raffinement de la solution (subdivision et choix du niveau de calcul),
2. calcul des valeurs d'irradiance pour chaque élément de la subdivision (*gather*),
3. mise à jour cohérente des valeurs à chaque niveau de la hiérarchie par descente des valeurs d'irradiance jusqu'aux feuilles, puis remontée des valeurs de radiosité jusqu'au plus haut niveau (*push–pull*). Ceci permet d'avoir une multi–résolution cohérente.

À chaque étape de la solution, la représentation des échanges lumineux se fait par des *liens*. Un lien est une structure de données où sont stockés le facteur de forme et le facteur de visibilité (aussi appelé facteur d'atténuation) associés à cet échange. Les liens sont habituellement stockés sur le récepteur et ils représentent l'énergie incidente. Un exemple de liens est illustré sur la figure 2.7, entre une source de lumière (en haut) et un récepteur subdivisé en quatre facettes.

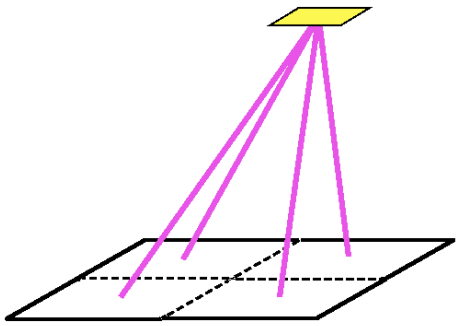


FIG. 2.7: Liens représentant l'irradiance arrivant sur chaque facette, depuis la source de lumière.

Dans la première partie d'une itération, les échanges lumineux entre surfaces sont raffinés selon le critère de subdivision choisi. Ensuite, la hiérarchie est parcourue et pour chaque élément, une valeur d'irradiance est calculée en additionnant les valeurs d'irradiance associée à chaque lien. La hiérarchie est à nouveau parcourue, d'abord de haut en bas pour additionner les valeurs d'irradiance aux feuilles. Puis une valeur de radiosité est calculée pour chaque feuille de la hiérarchie en multipliant les valeurs d'irradiance amassées par la réflectance de l'élément. Les valeurs de radiosité sont ensuite remontées en les moyennant jusqu'au niveau le plus haut de la hiérarchie.

La radiosité hiérarchique présente deux avantages. Le premier est que le nombre de calculs est réduit par rapport à une solution de radiosité progressive ou dite de « matrice

complète » par exemple. Le deuxième est que nous possédons une structure de données permettant de représenter les échanges d'énergie tout en les stockant. Nous avons connaissance à tout moment d'une représentation complète de tous les échanges lumineux.

### 2.2.2 Clustering

Le calcul de la solution de radiosité peut cependant se faire de façon plus efficace. Considérons par exemple un groupement de petites surfaces. La méthode que nous venons de présenter établirait un lien entre chaque paire de ces surfaces. Pourtant dans certains cas, seule une approximation de l'énergie arrivant ou quittant un sous–ensemble de ces surfaces serait nécessaire. Des algorithmes de regroupement de surfaces ont été intégrés à la méthode de radiosité hiérarchique [SAG94, SDS95, CLSS97] pour optimiser la représentation des échanges. Ainsi la hiérarchie des éléments est composée en une hiérarchie de « clusters », une hiérarchie des surfaces, et une hiérarchie mixte où les éléments fils d'un cluster sont composés à la fois de clusters et de surfaces. Un exemple de clusters construits pour une scène est montré sur la figure 2.8. Le calcul de la solution de radiosité hiérarchique suit le même algorithme, mais les échanges sont représentés entre clusters, entre surfaces et clusters, ou entre surfaces. Une méthode spéciale de calcul de facteur de forme entre clusters ou entre un cluster et une surface est alors appliquée. Il est à noter qu'un cluster n'est jamais une feuille de la hiérarchie ; nous ne reflétons jamais l'irradiance sur un cluster à la fin du « push ». Ils n'ont donc pas de réflectance associée. L'algorithme est initialisé par un cluster englobant toute la scène. L'algorithme choisi pour la formation des clusters est très important pour le résultat final. Il en existe plusieurs qui ont été comparés par Hasenfratz *et al.* [HDSD99].

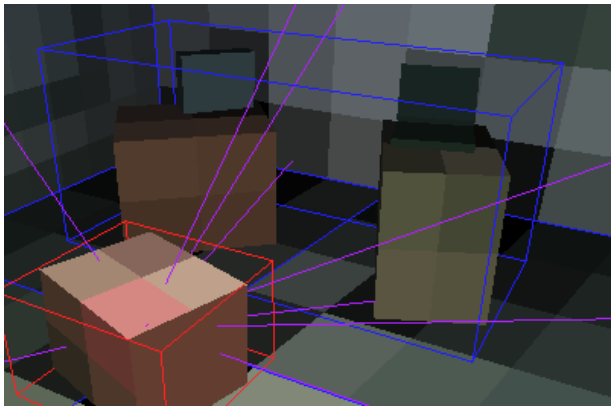


FIG. 2.8: Hiérarchie de clusters. Les liens vers le cluster englobant la boîte au premier plan sont établis depuis d'autres clusters comme celui englobant les autres objets, ou depuis des surfaces comme les murs et le sol.

### 2.2.3 Contrôle des résultats

La radiosité est une méthode fournissant un résultat indépendant du point de vue. Cependant, il est quelque fois inutile de calculer un éclairage détaillé dans des régions que l'utilisateur ne regarde pas. Un terme d'importance peut être ajouté dans la solution de radiosité, influençant les critères de subdivision. Ce terme d'importance a été introduit dans un premier temps par Smits *et al.* [SAS92].

Un des principaux problèmes dans les résultats de simulation d'éclairage par une méthode de radiosité est que l'éclairage indirect est souvent mal contrôlé. Dans certaines scènes où l'éclairage est principalement indirect, il est important de s'assurer que le résultat sera correct. Une des solutions est d'utiliser des bornes d'erreur [LSG94, GH96b]. Dans l'algorithme de radiosité hiérarchique, on associe à chaque lien deux nouvelles quantités : une estimation d'une borne inférieure de la radiosité, et une estimation d'une borne supérieure. Dans le critère de raffinement, on tiendra compte de l'erreur sur l'échange d'énergie pour chaque lien. On décidera de raffiner si l'erreur est supérieure à un seuil fixé par l'utilisateur.

## 2.3 Environnements dynamiques

On appelle *environnement dynamique* un environnement où les objets ou les sources de lumière peuvent être ajoutés, déplacés ou enlevés. Ce type d'objet est appelé *objet dynamique*. Dans ces environnements, les propriétés des surfaces peuvent aussi être modifiées, de même que les exitances des sources de lumière. De plus, il est important de pouvoir changer le point de vue.

Si nous voulons conserver un éclairage de qualité tout en permettant une mise à jour rapide lors des modifications, des algorithmes adaptés doivent être appliqués. Les algorithmes de radiosité présentent l'avantage de proposer des solutions indépendantes du point de vue (à l'exeption de méthodes comme le lancer de particules). Des solutions pour une mise à jour rapide lors de modifications géométriques sont présentées dans la section 2.3.1.

Le rendu effectué par des algorithmes de lancer de rayons a permis d'avoir des solutions de haute qualité pour des environnements ayant des propriétés de surface très diverses. Par contre, chaque solution est dépendante du point de vue. Des algorithmes permettant de se déplacer rapidement tout en gardant la qualité de simulation ont été développés et sont en partie présentés dans la section 2.3.2. Dans cette section, sont aussi présentés des algorithmes permettant des modifications géométriques.

### 2.3.1 Radiosité dynamique

Nous avons vu que les méthodes de radiosité permettent de simuler, pour les scènes diffuses, un éclairage réaliste et indépendant du point de vue. Après le calcul d'une solution, l'utilisateur peut se déplacer dans la scène interactivement. Par contre, il ne peut réaliser de modification géométrique, comme le déplacement d'un objet, sans qu'une solution complète soit réalisée. Pourtant, lorsqu'un objet se déplace, il n'y a en général que peu de changements visuels, surtout si l'objet est petit par rapport aux objets aux alentours.

Un utilisateur percevra plus les changements dus à l'éclairage direct comme les positions d'ombre de l'objet dynamique, et les projections des ombres des objets statiques sur cet objet. Quelques fois seulement, les modifications de l'éclairage indirect sont importantes visuellement.

Afin d'éviter de recalculer une solution complète de radiosité lors du déplacement d'un objet, il existe plusieurs méthodes qui localisent les modifications, et permettent une mise à jour locale.

Une première approche consiste à pré–calculer une solution de radiosité sans l'objet dynamique, puis à insérer l'objet dans cette solution. À cet effet, Baum *et al.* [BWCG86] utilisent un volume englobant l'objet dynamique, dans sa position initiale et finale. Ce volume est projeté sur l'écran selon le point de vue de l'utilisateur. Ensuite, il y a deux mises à jour complémentaires. La première met à jour les surfaces visibles ou cachées pour un point de vue. Cette solution est locale dans la région délimitée par la projection du volume englobant. Ensuite, les facteurs de forme associés à l'objet dynamique sont calculés et insérés dans la solution globale de radiosité. Cette approche est très intéressante car elle utilise la cohérence visuelle. Par contre, l'indépendance du point de vue de la solution est perdue lors de la phase dynamique.

Une autre solution consiste à mettre à jour de façon incrémentale la solution de radiosité pour un objet dynamique. Cette solution a été proposée par Chen [Che90] et George *et al.* [GSG90]. Ces méthodes utilisent la radiosité progressive comme méthode de base. Lorsque la géométrie de la scène change (objet ajouté, déplacé ou enlevé), ou lorsque l'exitance d'une lampe est modifiée, une énergie de correction est propagée dans la scène. La localisation de modification de l'éclairage est cependant différente pour les deux méthodes. Chen [Che90] utilise la boîte englobante de l'objet dynamique pour localiser les changements. Seules les surfaces statiques visibles directement depuis cette boîte englobante sont considérées dans la phase de modification. George *et al.* [GSG90] localisent les modifications des ombres dues à l'objet dynamique par un volume d'ombre. De plus, les surfaces ayant le plus d'énergie à redistribuer sont sélectionnées en priorité lors de la phase de mise à jour. Muller et Schöffel [MS94] ont amélioré ces méthodes incrémentales, en gérant la cohérence de la scène par une structure de données où sont stockées les informations sur la visibilité et les facteurs de forme. Un algorithme de parcours sur cette structure de données permet une mise à jour efficace et locale lors de modifications de la géométrie, mais aussi des réflectances des surfaces.

Ces méthodes ont mis en évidence le problème de mise à jour des facteurs de forme et de la localisation des modifications de l'éclairage dues à un objet dynamique. La localisation peut être plus facile si on utilise les algorithmes de radiosité hiérarchique. En effet, avec la structure de données des liens (voir la section 2.2.1), nous connaissons comment se réalisent les échanges lumineux. Si nous localisons les liens affectés par une modification géométrique, il ne reste plus qu'à mettre à jour l'information associée à ces liens (visibilité et facteur de forme), et de modifier avec ces liens la solution de radiosité. Forsyth *et al.* [FYT94] ont exploité en premier cette propriété en modifiant la position des liens dans la hiérarchie selon la position de l'objet dynamique. Les liens de la scène statique sont gardés, et seule l'information associée aux liens est modifiée. Par contre les liens attachés

à l'objet dynamique sont placés à différents niveaux de la hiérarchie selon un critère basé sur les facteurs de forme.

Shaw [Sha97] a amélioré la structure de la radiosité hiérarchique en introduisant la notion de « liens fantômes » ou *liens passifs*. Ce type de lien correspond à un échange d'énergie à un niveau de subdivision supérieur à celui considéré dans le calcul de la radiosité finale. Ils sont mis en opposition aux *liens actifs* qui servent au calcul de la solution de radiosité. Les liens passifs servent à avoir un historique de chaque étape du raffinement, et l'information associée, tels que les facteurs de forme et la visibilité, y est stockée. Les modifications de l'éclairage sont localisées par un volume de mouvement englobant la position initiale et finale de l'objet dynamique. Seuls les liens intersectant ce volume sont considérés dans la phase de modification. Durant cette phase, le niveau de subdivision associé à un échange varie. Si la subdivision est raffinée, des nouveaux liens sont établis et les liens du niveau précédent sont conservés comme étant passifs. À l'inverse, si on considère que la subdivision est trop fine, les liens passifs sont ré–activés.

Cette idée de liens actifs et passifs a été reprise par Drettakis et Sillion [DS97] qui ont ajouté une structure de données pour gérer les modifications, et ont utilisé un algorithme de clustering. Cette structure de données est appelée « hiérarchie d'espace de droites ». Elle est composée de la hiérarchie des éléments avec laquelle est stocké tout d'abord un intervalle de numérotation. Chaque feuille comporte un numéro, et l'élément supérieur dans la hiérarchie comportera l'intervalle de numérotation des feuilles qui lui sont inférieures. Cette indexation permet de parcourir la hiérarchie de façon rapide pour trouver un élément. Ensuite, une structure appelée *shaft*[*] [HW94] est associée à chaque lien. Un shaft permet de délimiter une région dans l'espace entre deux éléments. Un exemple de shaft est montré sur la figure 2.9 (c). Les liens affectés par le déplacement d'un objet sont détectés par le calcul de l'intersection entre un shaft et la boîte englobante de l'objet dynamique. Cette recherche des liens modifiés est d'abord effectuée en partant du haut de la hiérarchie. Si un shaft père n'intersecte pas la boîte englobante de l'objet dynamique, le parcours de cette branche est arrêté. Par contre, tant qu'il y a intersection, on descend dans la hiérarchie. Lorsqu'on arrive au plus bas niveau des éléments associés au shaft, les éléments sont marqués comme touchés. Cette phase de localisation des liens affectés permet de faire une mise à jour efficace de la solution de radiosité. Comme pour Shaw, selon les modifications à faire, les liens deviennent actifs ou passifs (voir figure 2.9). De plus, la radiosité est mise à jour de façon incrémentale en utilisant la *hiérarchie d'espace de droites* pour localiser les calculs, lors du *gather* et du *push–pull*.

Ces différentes méthodes ont su exploiter la cohérence des échanges lumineux pour accélérer les calculs lors d'une modification des scènes virtuelles, tout en conservant un éclairage de qualité.

### 2.3.2 Lancer de rayons dynamique

Le problème principal d'une image calculée par lancer de rayons est que son rendu dépend du point de vue. La production d'une image de qualité en lancer de rayons est loin

[*] Traduction littérale en français : puit ou cage

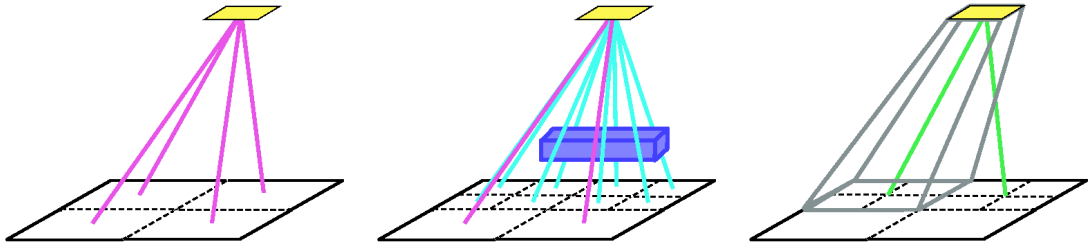



FIG. 2.9: (a) Maillage initial et liens correspondant à une première solution. (b) Insertion d'un objet dynamique dans la scène, et subdivision conséquente. Huit liens ont été créés. (c) Les anciens liens sont devenus des liens passifs. Shaft correspondant à un lien passif.

d'être interactive. De ce fait, tout déplacement du point de vue n'est pas interactif, et la solution complète doit être re–calculée. De plus, on aimerait aussi pouvoir manipuler les objets et les lampes d'une scène virtuelle. Nous ne présentons que deux exemples des nombreux travaux dans ce domaine.

Brière et Poulin [BP96] ont proposé une solution pour manipuler les objets ainsi que leurs propriétés de réflectance, pour une scène rendue par lancer de rayons. La solution est d'utiliser des arbres de couleurs et de rayons pour accélérer la mise à jour. Le rendu est accéléré mais n'est pas toujours interactif.

Walter *et al.* [WDP99] ont utilisé un « tampon de rendu » pour permettre le déplacement interactif dans une scène rendue par un algorithme de lancer de rayons. Cette méthode permet également les déplacements interactifs d'objets et de lampes. Le tampon de rendu génère des images plus rapidement que le système de rendu est capable de générer des échantillons (pixels) pour toute l'image. Pour cela, l'image est générée en projetant les échantillons précédents et en détectant les pixels à mettre à jour. Ces pixels sont classés par ordre de priorité. Seuls ces derniers pixels sont recalculés par le système de rendu. Les autres pixels seront interpolés à partir des autres échantillons. Cette méthode permet de déplacer le point de vue et de manipuler des objets en temps interactif, dans une scène dont le rendu est de haute qualité et contenant des effets spéculaires, malgré certains artefacts visuels.

## 2.4 Éclairage commun pour la réalité augmentée assistée par ordinateur

### 2.4.1 Présentation générale de la réalité augmentée

La réalité augmentée consiste à utiliser un environnement réel et à y ajouter des éléments virtuels tels que des objets 2D ou 3D ou du texte. Ces applications sont diverses, comme le cinéma, l'apprentissage, l'aide à l'utilisation d'outils, etc.
Azuma [Azu97] a présenté un document contenant une vue générale de la réalité augmentée, comportant une bibliographie étendue.

Dans le cadre de manipulation directe avec l'environnement réel, on utilise des lunettes semi–transparentes sur lesquelles les éléments virtuels sont superposés à la réalité. Il est également possible d'utiliser une caméra qui capture la scène réelle, ensuite le mélange réel et virtuel est réalisé par ordinateur, et le tout est reprojeté dans un casque d'immersion virtuelle (≪ head mounted display ≫).Les données réelles sont traitées et mélangées avec le virtuel directement sur ordinateur. C'est ce qu'on appelle la *réalité augmentée assistée par ordinateur*. Les problèmes à résoudre sont le calibrage, le placement des objets dans l'espace, et le délai d'affichage. Il est évident que ce délai doit être minimal pour éviter les incohérences. Le problème de calibrage et de placement des objets dans la scène en temps réel est un problème difficile. Cependant les applications comme le cinéma ne nécessitent pas toujours le temps réel et l'immersion, et le traitement peut être réalisé

La plupart des travaux réalisés en réalité augmentée se sont focalisés sur les problèmes de calibrage et de placement des objets dans l'espace en temps réel. Les problèmes de cohérence d'éclairage ont été longtemps laissés de côté, car ce sont des calculs souvent trop coûteux pour assurer le temps réel nécessaire à la réalité augmentée. Pourtant, il est plus facile pour un utilisateur de déterminer la position des objets dans l'espace s'il y a des ombres projetées et une illumination commune. De plus, pour les applications telles que le cinéma, le réalisme dans le mélange de la synthèse et du réel est nécessaire. Il est souvent très important de tenir compte de l'*éclairage commun*, c'est–à–dire la restitution des effets lumineux (ombres, réflexions, etc.) entre les objets réels et les objets virtuels. Pour l'instant, l'éclairage commun réaliste est assuré par des artistes qui réalisent des montages photographiques avec des retouches. Dans cette thèse, notre objectif est d'automatiser ce processus.

L'éclairage commun en réalité augmentée est la façon de mélanger le réel et le virtuel en tenant compte des modifications d'éclairage. Ceci inclut le ré–éclairage virtuel, c'est–à–dire la modification de l'éclairage original de la scène réelle, comme par exemple l'extinction virtuelle des lampes réelles. Il est à noter que le ré–éclairage est un problème plus difficile que l'insertion d'objets virtuels dans une scène réelle. Lorsqu'on ajoute des objets virtuels à une scène réelle, on utilise des algorithmes de rendu pour projeter les ombres de l'objet virtuel sur les objets réels, et des objets réels sur l'objet virtuel. Pour plus de réalisme, on peut aussi simuler l'éclairage indirect, en utilisant une estimation de la réflectance des objets réels. Un éclairage indirect approximatif peut être suffisant pour obtenir un effet convaincant. Dans le cas d'insertion d'objets virtuels, une estimation très approximative de la réflectance des objets réels est souvent suffisante. Par contre, dans le cadre de ré–éclairage virtuel d'une scène, de nombreux artefacts peuvent apparaître si l'estimation de la réflectance est trop approximative.

Des travaux récents ont été réalisés pour retrouver les propriétés radiométriques des scènes réelles. Ces propriétés sont ensuite exploitées par des algorithmes de rendu. Le domaine de recherche de ces travaux est appelé *illumination inverse*, car le processus est inverse au processus de rendu virtuel classique. Ces travaux sont présentés dans la section 2.4.3. L'estimation de ces paramètres radiométriques permet ensuite de réaliser un ré–éclairage convaincant.

Pour pouvoir retrouver correctement les propriétés radiométriques d'une scène réelle,

il est souvent nécessaire de connaître la géométrie en trois dimensions de cette scène. De nombreux travaux ont été menés dans le domaine de la vision par ordinateur pour résoudre le plus automatiquement possible ce problème, notamment en utilisant la géométrie épipolaire. La reconstruction automatique ne produisant pour l'instant pas de modèles fiables et utilisables en infographie, certains travaux se sont concentrés au contraire sur des méthodes nécessitant l'intervention de l'utilisateur. Certaines approches (notamment celles que nous avons utilisées dans cette thèse) sont présentées dans la section suivante 2.4.2.

À partir de la connaissance des propriétés géométriques et radiométriques des scènes réelles, on peut construire un modèle de la scène réelle, et l'utiliser pour simuler l'éclairage commun entre les objets réels et des objets virtuels ajoutés. Les algorithmes développés pour traiter ce problème font l'objet de la section 2.4.4.

### 2.4.2 Reconstruction géométrique des scènes réelles

Il existe de nombreux travaux ayant trait à la reconstruction 3D à base d'images. En effet, les applications sont nombreuses, et ne se limitent pas à celles de l'infographie. Les roboticiens ont par exemple besoin d'outils de reconstruction automatiques, pour aider un robot à reconnaître un environnement et à s'y déplacer sans collision. La reconstruction à partir d'images est un thème de recherche important et difficile dans le domaine de la vision 3D.

Les modèles générés sont souvent insuffisants si on veut les utiliser pour des algorithmes de rendu en synthèse d'images. Dans le problème qui nous intéresse, nous avons besoin d'un modèle polygonal 3D qui soit le plus juste possible dans ses mesures et dans ses propriétés. Certains appareils comme les scanners 3D permettent d'acquérir une bonne connaissance de la forme d'un objet [MNP+99, Nyl]. Cependant, passer au scanner tout un environnement peut parfois s'avérer impossible, surtout pour les environnements en extérieur. De plus, il peut y avoir des erreurs dues aux problèmes d'échelle (comme pour une grille) ou à des effets spéculaires (miroir). Cette nouvelle technologie est également à confirmer quant à sa fiabilité.

#### Vision 3D et géométrie épipolaire

Pour reconstruire un modèle 3D, il faut d'abord *calibrer* la caméra. Le résultat d'un calibrage doit nous fournir une matrice de projection, décrivant comment les points 3D se projettent dans une image donnée par la caméra (appareil photographique ou caméra vidéo). Cette matrice de projection est décomposable en deux matrices, une décrivant les *paramètres intrinsèques* de la caméra, l'autre les *paramètres extrinsèques*. Les paramètres intrinsèques décrivent la distance focale, le centre de projection et les coefficients d'échantillonnage des pixels. Les paramètres extrinsèques décrivent la position et l'orientation de la caméra dans l'espace, par rapport à un référentiel fixe. Le calibrage d'une caméra peut se faire de plusieurs manières. Une première est d'utiliser une mire [FT86, Rob95, Fau93]. Les points 3D positionnés sur cette mire sont connus, et donnés en paramètre au système.

Lorsque nous ne connaissons pas la caméra qui a servi à prendre des images, ce type de calibrage n'est pas possible. Il faut alors utiliser des méthodes de calibrage automatiques

ou semi–automatiques. Ces méthodes de calibrage utilisent des correspondances entre les images établies manuellement ou automatiquement. Pour cela, on peut établir des correspondances pour six points et huit lignes [LHO90], pour retrouver les paramètres d'une caméra. En utilisant seulement des points, il est possible de retrouver une paramétrisation mais elle sera à un facteur d'échelle près [Fau92, Fau93]. En utilisant la *géométrie épipolaire* [Fau93], on peut réaliser une reconstruction utilisable sans avoir à calibrer parfaitement la caméra.

C'est en partant de ce principe que s'est développé l'outil *TotalCalib* [BR97, FLR+97, Rob95, ZFD97]. Cet outil a été ensuite amélioré et est en cours de commercialisation par la société *Realviz* [LdlFK+96, Rea]. Nous avons utilisé en partie pour nos travaux de recherche l'outil TotalCalib. Sa description est détaillée dans le chapitre 3.

#### Systèmes de reconstruction à base de contraintes

Le développement des outils a évolué vers des systèmes assistés par l'utilisateur, mais construisant des modèles plus cohérents et utilisables par les infographistes. Ces systèmes prennent en entrée non plus des points mais des primitives telles que des lignes, des polygones, ou des polyèdres. Ensuite, on peut ajouter des contraintes permettant d'affiner la connaissance globale de l'environnement en ajoutant des contraintes de distance, de parallélisme, de perpendicularité. L'utilisateur est entièrement impliqué dans la reconstruction. Les erreurs résultantes sont plus facilement contrôlables et rectifiables.

Le système *Façade*, développé par Debevec *et al.* [DTM96], permet de reconstruire des scènes architecturales. Le principe est que les bâtiments sont en général très symétriques avec des propriétés de perpendicularité et de parallélisme. Les objets de base de reconstruction sont des *blocs*, pouvant être ajustés par des paramètres tels que la taille, la forme et la position. Différents blocs sont proposés à l'utilisateur, tels que des polyèdres comme des pyramides ou des cubes, ou des surfaces de révolution comme les sphères. Lútilisateur modélise une scène avec ces blocs, et indique les correspondances des blocs sur plusieurs images et les contraintes appliquées à ces blocs, comme par exemple une position relative de deux blocs. Le système résoud les équations pour retrouver les meilleurs valeurs des paramètres. Le positionnement des blocs, les contraintes appliquées et les correspondances, conduisent ainsi à le calibrage du système et donc à la génération d'un modèle 3D. Ce type de reconstruction s'adapte à de nombreux environnements réels et permet un contrôle précis sur le système. Cependant la reconstruction d'objets complexes ne correspondant pas aux blocs proposés ne peut être qu'approximative ou impossible.

Une autre approche similaire a été développée par Meas–Yedid *et al.* [MYTG94]. Dans cette approche, le calibrage se fait à partir de rectangles et de segments. Puis des primitives telles que des polyèdres, des sphères ou des cônes sont utilisées pour reconstruire les objets. La représentation finale de la scène est hiérarchique, et le niveau de représentation est choisi par l'utilisateur.

Dans un objectif similaire, le système *Rekon* développé par Poulin *et al.* [POF98] utilise des primitives simples telles que des points, des lignes et des polygones. Les contraintes sont basées sur les propriétés de coplanarité, de perpendicularité, ou de parallélisme. Ce système de reconstruction est le deuxième que nous avons utilisé dans nos travaux. Il est

décrit en détail dans le chapitre 3.

Les modèles construits par ce type de système sont plus facilement utilisables, et plusieurs modeleurs commercialisés utilisent ce principe d'interaction avec l'utilisateur, tels que Photomodeler [Pho], Renoir [Ren], et Canoma [Can].

Le système de reconstruction AIDA [WG96] est basé sur la connaissance. Le système comporte une base de contraintes correspondant aux objets que l'on s'attend à trouver dans les scènes, et d'autres sur des propriétés 3D et 2D. Après avoir calibré et segmenté les images par des algorithmes classiques, le système sélectionne automatiquement ces contraintes après avoir réalisé une phase d'interprétation. Ce système est un mélange des algorithmes de vision et des systèmes à base de contraintes. Les résultats dépendent de l'applicabilité de la base de contraintes sur les images capturées. De plus, ce fonctionnement automatique peut produire des erreurs difficiles à corriger.

#### Modèles texturés

Pour rendre le modèle reconstruit plus réaliste, on peut appliquer des textures. La méthode la plus simple est de produire une texture rectangulaire pour un objet. Cette méthode simple comporte souvent des artefacts. D'une part, si l'angle de vue pour lequel la texture est extraite est rasant, la qualité de la texture sera très mauvaise. D'autre part, si l'objet est caché depuis le point de vue choisi, la texture comportera l'image d'un autre objet.

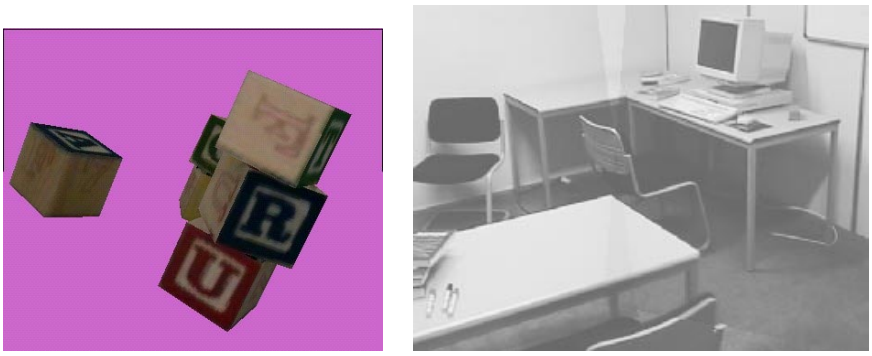
Dans le système *Rekon* [POF98], la connaissance de la géométrie 3D et de la position de la caméra permet de traiter le cas d'occlusion. Les textures sont extraites depuis toutes les images où le polygone pour lequel on veut extraire la texture est visible. Les textures sont ensuite combinées en tenant compte des occlusions, et par pondération, les pixels sont validés en éliminant les « défauts » tels que des taches de spécularité ou ceux dus aux angles rasants. Les textures ainsi récupérées sont indépendantes du point de vue. Cependant, l'information de perspective peut être rendue floue par la combinaison des images, ce qui oblige à avoir une scène assez détaillée pour qu'elle ne soit pas trop incohérente.

On peut également extraire des textures dépendantes du point de vue, en les déformant pour qu'une fois reprojetées, elles soient réalistes [FLR+97]. L'objectif est de ne pas faire la différence entre une image perçue et la scène 3D visualisée selon le même point de vue. Cette méthode permet d'extraire des textures de polygones non rectangulaires, ce qui est souvent le cas dans les modèles reconstruits.

Cependant, ces textures ne sont visibles que depuis un même point de vue. Dès que l'on change de point de vue, les informations deviennent erronées et des incohérences peuvent apparaître. De plus, les modèles extraits sont souvent des simplifications de la géométrie réelle. Dans certains cas, de petites variations de surfaces sont perdues dans la géométrie. Pour éviter cette perte visuelle d'information pour les textures dépendantes du point de vue, Debevec *et al.* [DTM96] appliquent une méthode pour faire varier ces effets de profondeur, sans avoir la géométrie. Pour cela, des morceaux de textures extraits depuis plusieurs images sont composés. Chacun est choisi comme correspondant le mieux au point de vue de la caméra. Les transitions entre chaque morceau de textures sont assurés par un lissage pondéré. Ceci permet de faire un choix dans les textures, mais pas de visualiser les effets de profondeur. Pour cela, une carte de profondeur est construite pour

calculer par la déviation entre un point du modèle reconstruit et la projection de ce point par géométrie épipolaire.

Trois exemples de modèles texturés sont présentés sur la figure 2.10 : en (a), un modèle reconstruit par *Rekon*, en (b), un modèle reconstruit par *TotalCalib*, et en (c), un modèle reconstruit par *Façade*.



(a) (b)



(c)

FIG. 2.10: (a) Modèle texturé reconstruit par le système *Rekon*, avec des textures indépendantes du point de vue. (b) Modèle texturé reconstruit par *TotalCalib* avec des textures dépendantes du point de vue. (c) Modèle texturé reconstruit par *Façade* avec des textures dépendantes du point de vue. (Figures extraites des pages web correspondant aux différents systèmes : `http ://www.iro.umontreal.ca/labs/infographie/projects/rekon/`, `http ://www-sop.inria.fr/robotvis/personnel/sbougnou/Results/`, `http ://www.cs.berkeley.edu/~debevec/Research/` )

Ces méthodes d'extraction et d'application de texture permettent de visualiser le modèle reconstruit et d'avoir un effet réaliste. Cependant, dans ces textures, l'effet lumineux au moment de la capture des photographies est inclus. Ces modèles texturés sont difficilement utilisables dans des systèmes de simulation d'éclairage, dans lesquels on voudra simuler un autre éclairage. Certaines méthodes ont été développées pour transformer l'information de ces textures et la rendre indépendante de l'éclairage de capture. Ces méthodes sont expliquées ci–dessous.

### 2.4.3 Illumination inverse

Le principe de l'illumination inverse est très proche des objectifs de l'éclairage commun pour la réalité augmentée assistée par ordinateur. Dans des algorithmes classiques de rendu, les données sont la géométrie d'une scène virtuelle, les propriétés de réflectance pour chacun des objets, et les sources lumineuses avec les intensités associées. Les résultats sont un calcul des échanges lumineux dans la scène virtuelle ainsi définie, et un affichage selon un point de vue choisi. Au contraire, les données de l'illumination inverse sont une ou plusieurs images d'une scène réelle, et les paramètres de la caméra qui a pris ces images. L'objectif de l'illumination inverse est de retrouver les paramètres de l'illumination perceptible sur ces images. La simulation de l'éclairage avec ces paramètres estimés peut ensuite être directement comparée avec les images initiales. Les paramètres à retrouver sont la géométrie de la scène, la réflectance des objets, et l'intensité des sources lumineuses. En général, les chercheurs focalisent leur attention sur le problème de l'estimation de la réflectance. Ces paramètres estimés permettent ensuite de faire du ré–éclairage en utilisant des algorithmes de rendu.

Les chercheurs ont abordé le problème d'estimation de la réflectance de plusieurs façons selon leurs objectifs. Certains s'intéressent à retrouver la réflectance pour un objet unique. Cette approche permet d'avoir un dispositif localisé sur cet objet. D'autres s'intéressent à retrouver les réflectances d'un environnement. Cette approche est plus complexe dans le sens où l'acquisition des données est souvent coûteuse, en main d'œuvre et en mémoire. Il est également plus difficile d'avoir un modèle géométrique précis pour un environnement complexe.

Ces deux approches ne nécessitent pas les mêmes dispositifs, et sont décrites dans deux parties différentes de ce chapitre.

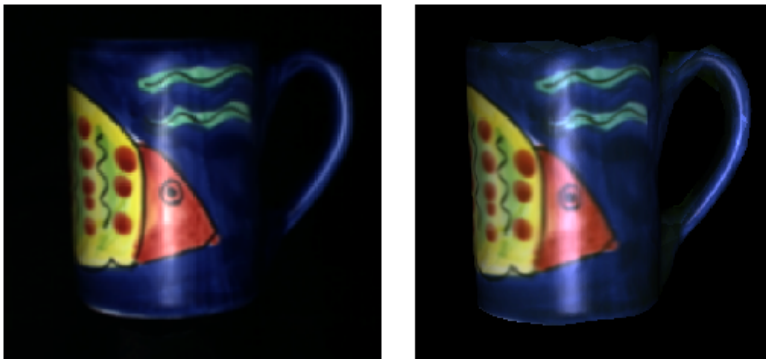
#### Estimation de la réflectance pour un objet

Les approches pour estimer la réflectance d'un objet utilisent en général un dispositif muni d'une caméra, et d'une source presque ponctuelle. En faisant varier la position soit de l'objet, soit de la caméra, soit de la source de lumière, on obtient des images contenant de l'information sur les variations lumineuses en fonction de chacune de ces positions.

Sato *et al.* [SWI97] ont mis au point un système pour estimer la géométrie et la réflectance d'un objet. Ils ont pour cela un dispositif calibré composé d'un bras articulé sur lequel est fixé l'objet à modéliser, et d'une source ponctuelle fixe. La géométrie de l'objet est retrouvée en utilisant la méthode développée par Wheeler *et al.* [WSI97]. La réflectance des objets est retrouvée en plusieurs étapes. Les composantes de réflexion sont estimées à partir d'une décomposition du modèle de Torrance–Sparrow [TS67] en deux équations matricielles isolant chacune les paramètres diffus des paramètres spéculaires. La première étape est donc de calculer en tout point une composante diffuse ; la deuxième est d'estimer pour quelques régions seulement la composante spéculaire qui sera appliquée à tout l'objet. Les régions pour calculer la composante spéculaire sont choisies en fonction de plusieurs critères. Le premier est que les composantes diffuse et spéculaire dans ces régions doivent être très différentes. Le deuxième est que la valeur représentative de la réflexion

spéculaire doit être grande, et le troisième que la valeur de la composante diffuse doit être grande.

Comme le montre la figure 2.11, les résultats sont assez bons. Les auteurs comparent les images initiales avec un rendu simulé avec les paramètres estimés. Les images simulées sont très similaires aux photographies originales. En revanche, cette estimation comporte quelques limitations. Premièrement, l'estimation de la réflectance se fait directement sur la couleur des pixels, sans pré–traitement sur les variations d'éclairage. La combinaison des images permet d'atténuer ces éventuelles variations, mais n'offre pas de garantie sur le résultat final. Une deuxième limitation est que d'une part la composante spéculaire est supposée isotrope, d'autre part elle est supposée constante sur tout l'objet. Cette restriction ne permet donc pas de traiter n'importe quel objet. Enfin, cette méthode permet de ne traiter que des scènes très simples sans inter–réflexions importantes.



(a) (b)

FIG. 2.11: Résultats de Sato *et al.* [SWI97]. (a) Photographie originale. (b) Image simulée avec l'éclairage original. (Figures extraites de l'article).

Marschner et Greenberg [MG97] ont développé une méthode pour permettre des modifications virtuelles d'éclairage pour un objet vu sur plusieurs photographies sous des éclairages différents. Les variations d'éclairage ont été réalisées avec une source ponctuelle placée assez loin de l'objet. Le modèle 3D de l'objet est retrouvé à partir d'un *scanner 3D à laser*. En utilisant les différentes images auxquelles est associée une position de la source de lumière, une réflectance diffuse est estimée. La méthode met en place une matrice reliant les valeurs des pixels représentant la luminance réfléchie par l'objet, et la distribution lumineuse. Cette matrice représente un système à résoudre aux moindres carrés pour des images virtuelles, ou par un système à décomposition en valeurs singulières (SVD) pour des images provenant d'un appareil photographique, et donc bruitées. Une fois que la réflectance est estimée, un rendu est effectué avec une nouvelle position de lampe. Cette approche de ré–éclairage est intéressante car elle ne nécessite pas une estimation exacte de la réflectance. Le rendu est basé sur la connaissance d'un éclairage de base à partir des photographies originales. Le résultat affiché est égal à la photographie originale, modulée par un rendu synthétique correspondant au nouvel éclairage, divisé par un rendu synthétique réalisé pour l'éclairage correspondant à la photographie. Les résultats obtenus sont intéres-

sants, surtout si on considère la simplicité de l'algorithme. Cet algorithme est appliqué au ré–éclairage de visage qui est tout à fait convaincant (voir figure 2.12). Ce travail pourrait être amélioré avec une estimation des paramètres spéculaires.

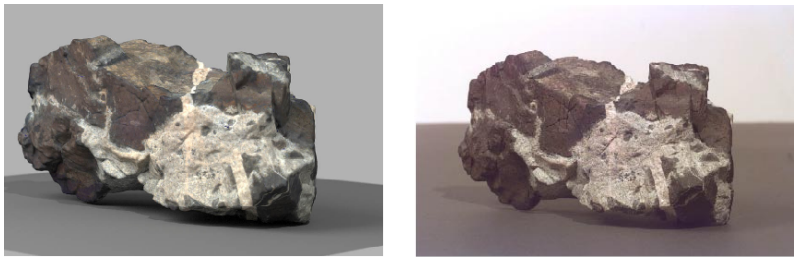


(a) (b)

FIG. 2.12: Résultats de Marschner *et al.* [MG97]. (a) Photographie originale. (b) Image simulée avec un éclairage différent. (Figures extraites de l'article).

Dans le même esprit, Marschner [Mar98] a amélioré l'estimation de la réflectance pour obtenir des textures indépendantes des effets lumineux, contenant des informations sur la BRDF. Il utilise pour cela des photographies d'un objet tournant sur une plate-forme, prises avec un appareil photographique restant à position fixe et avec une source ponctuelle restant fixe. Comme pour Sato *et al.* [SWI97], la composante diffuse est calculée indépendamment de la composante spéculaire, et la composante spéculaire est approximée comme étant unique pour un même objet. La réflectance diffuse est calculée en divisant la luminance observée en chaque pixel par l'irradiance directe. Dans cette approche, l'éclairage indirect est ignoré. Cette estimation est faite pour chaque pixel et pour chacune des images d'entrée. Le résultat final est une moyenne pondérée par un coefficient de validité associé à chacun des pixels. Ce coefficient est calculé en fonction de trois critères : l'angle d'incidence par rapport à la normale doit être faible, l'angle de vue par rapport à la normale doit être faible, la réflexion spéculaire doit être faible. Ce dernier critère est évalué en fonction de l'angle de vue et de l'angle de la réflexion spéculaire possible. Une fois la réflectance diffuse calculée, on peut comparer après simulation de l'éclairage le résultat avec les images d'entrée (voir figure 2.13). Les pixels montrant une variation sont utilisés pour estimer une composante spéculaire. Les résultats utilisant cette composante spéculaire doivent être similaires aux photographies d'entrée.

Ces différentes approches permettent d'établir des valeurs de BRDF pour des objets quelconques. Cependant, ces mesures ne tiennent pas compte de la complexité d'éclairage dans un environnement, telles que les inter–réflexions. Ces méthodes ont également en commun de vouloir séparer l'estimation du caractère diffus du caractère spéculaire de la BRDF. Les résultats montrés sont convaincants lors du ré–éclairage à partir des données estimées. Cependant, ces approches seraient fastidieuses si nous voulions retrouver la BRDF pour tous les objets d'un environnement complexe. Certains objets seraient non déplaçables à cause de leur taille ou de leur poids. De plus les dispositifs utilisés dans les différentes



(a) (b)

FIG. 2.13: Résultats de Marschner [Mar98] (a) Photographie originale. (b) Image simulée avec l'éclairage original. (Figures extraites de la thèse).

méthodes ne pourraient pas toujours être appliqués.

**Estimation de la réflectance pour un environnement**

Des méthodes plus générales ont été développées afin de retrouver une BRDF pour un ensemble d'objets contenus dans un environnement.

Yu *et al.* [YDMH99] ont développé un algorithme d'illumination inverse, pour pouvoir ré–éclairer les scènes réelles. Les données d'entrée sont un modèle géométrique d'une pièce d'intérieur fermée, de nombreuses photographies prises sous des angles différents, et sous le même éclairage pour lequel la position des lampes est connue. Toutefois, la capture n'est pas exhaustive. Le modèle géométrique est reconstruit en utilisant *Façade* [DTM96]. Le fait que la pièce soit fermée garantit que l'éclairage estimé sera proche de l'éclairage réel. Les photographies sont transformées en images de luminance (voir le chapitre 6, section 6.2) qui servent à estimer la réflectance et l'intensité des sources de lumière. La réflectance est estimée par un algorithme itératif. L'idée principale pour estimer la BRDF est d'utiliser la luminance connue car elle est contenue dans les pixels des images. L'algorithme est fait en deux passes. Tout d'abord, les composantes spéculaires et diffuses sont estimées pour chaque facette résultant d'une discrétisation des surfaces de la scène. Puis, une carte d'albédo est calculée en tout point pour chacune des surfaces.

Pour estimer les composantes spéculaires et diffuses, des liens vers les autres surfaces sont créés pour chacune des facettes. Ces liens sont hiérarchiques, afin d'éviter une explosion du coût de la mémoire. Sur chacun des liens, on associe une valeur de luminance calculée en faisant la moyenne des pixels correspondant à la surface pour une image choisie. Cette image est choisie telle qu'un effet spéculaire y soit visible. Le choix de cette image provoque une erreur sur la luminance réelle à associer à ce lien. Cette erreur est due à l'effet spéculaire. Une autre entité $\delta S$ est associée à ce lien pour tenir compte de l'erreur. Cette entité est initialisée à zéro au départ. Ensuite, l'algorithme itératif est appliqué. À partir des valeurs ainsi associées au lien, les composantes diffuses et spéculaires sont estimées en utilisant un approche de résolution aux moindres carrés. Enfin, le terme $\delta S$ est mis à jour en utilisant la composante spéculaire nouvellement estimée.

À la fin de cette itération, on a pour chacune des facettes les composantes diffuses

et spéculaires de BRDF. Les composantes spéculaires sont supposées uniformes sur les facettes. Par contre, les composantes diffuses doivent être estimées plus finement pour obtenir un effet texturé des surfaces. Une nouvelle composante diffuse est donc estimée pour chacune des surfaces, pour créer une carte de réflectance diffuse. Pour cela, on sélectionne parmi les images celles pour lesquelles la composante spéculaire est la moins perceptible, de façon à ne pas générer trop d'erreur sur l'estimation de la composante diffuse. Ensuite, une valeur de réflectance diffuse est calculée pour chacune des images, en utilisant la luminance diffuse (égale à la luminance du pixel moins la luminance spéculaire) et l'irradiance arrivant en ce point. La valeur finale de la composante diffuse est une moyenne pondérée de ces valeurs initiales. Les poids sont inversement égaux à la luminance spéculaire. Ces poids tiennent aussi compte de l'angle de vue pour écarter de la solution les angles trop rasants.

Cet algorithme donne une estimation comparable aux BRDFs mesurées malgré les approximations réalisées lors de la discrétisation et la création des liens hiérarchiques. Il nécessite malgré tout la capture de nombreuses images pour avoir plusieurs photographies sous des angles de vue différents pour chacune des surfaces.

Yu et Malik [YM98] se sont intéressés au ré–éclairage des scènes d'extérieur par un re–positionnement du soleil à différentes heures de la journée. La scène est décomposée en quatre ensembles : le soleil, le ciel, le panorama et les bâtiments. Comme précédemment, pour estimer la BRDF, il faut tout d'abord modéliser la luminance. Dans un premier temps la luminance due au soleil et au ciel, puis la luminance due au panorama sont évaluées. Le ciel, le panorama et le soleil sont considérés comme émettant de la lumière. À partir de ces luminances, deux pseudo–BRDFs sont estimées par surface. Une correspondant au soleil, une autre correspondant à l'environnement. Ils évaluent d'abord une réflectance diffuse locale pour les bâtiments reconstruits. Pour cela, ils utilisent au moins deux photographies par surface à estimer. Sur une des photographies, la surface du bâtiment est éclairée directement par le soleil. Sur la deuxième, elle ne l'est pas. La surface est triangulée de façon à estimer une luminance par image, qui est la moyenne des pixels. Grâce à ces images, on peut estimer deux réflectances diffuses, une associée à l'irradiance de l'environnement (ciel plus panorama), l'autre associée au soleil. Le ciel et le panorama sont subdivisés pour faciliter le calcul de l'irradiance.

Cette réflectance est ensuite ajustée par une résolution aux moindres carrés pour retrouver un terme spéculaire global, constant pour tout un bâtiment donné. Encore une fois, deux termes spéculaires sont estimés, un associé au soleil, un autre associé à l'environnement.

Cette approche montre combien il est difficile de déterminer l'illumination inverse sur des environnements extérieurs. Il est en effet très difficile d'avoir une bonne représentation de l'éclairage en extérieur car la lumière y est distribuée de façon très complexe notamment à cause des réflexions avec le ciel et de la complexité géométrique des environnements. Les conditions d'éclairage varient également selon la météo, les saisons et l'heure du jour. La séparation des composantes de la BRDF en deux composantes distinctes pour le soleil et pour l'environnement simplifie l'approche.

### 2.4.4 Éclairage commun

Nous avons présenté dans les deux sections précédentes des méthodes pour modéliser la géométrie de scènes réelles à partir d'images, et des méthodes pour estimer la BRDF des surfaces de ces scènes. Une fois cette connaissance établie, il est possible de modifier l'éclairage de ces scènes et d'y ajouter des objets virtuels, tout en gardant un éclairage réaliste.

Cependant, les algorithmes d'illumination inverse recherchent une certaine précision qui n'est pas utile si on veut simplement ajouter des objets virtuels, sans réaliser de modification de l'éclairage réel. Plusieurs méthodes ont été développées afin d'estimer moins finement la réflectance et la luminance par des algorithmes plus simples.

#### Insertion d'objets virtuels sans estimation de la radiométrie de la scène réelle

L'approche la plus simple a été proposée par State *et al.* [SHC+96]. Leur travail se concentrait plus sur le calibrage et le placement des objets les uns par rapport aux autres. Il a montré la justesse de son calibrage en insérant un objet dynamique et en projetant son ombre sur les objets réels. L'objet est éclairé par une lampe ponctuelle réelle déplacée par un bras mécanique. La géométrie des objets sur lesquels l'ombre se projette a été numérisée. L'ombre de l'objet virtuel est simulée en temps réel par une carte d'ombre [Wil78, SKv+92]. La simulation d'éclairage direct par une source ponctuelle permet d'avoir la sensation des ombres, nécessaire au placement visuel des objets par rapport aux autres, mais ne permet pas un mixage réaliste.

Une autre approche ne nécessite pas non plus de retrouver la réflectance des objets réels. Jancène *et al.* [JNP+95] ont basé leur méthode sur le principe de la composition d'images. Leur objectif est d'ajouter le plus automatiquement possible des objets virtuels animés dans une vidéo, dont les images sont calibrées. La vidéo finale sera la composition de la vidéo initiale avec une vidéo virtuelle. La vidéo virtuelle contient les objets réels reconstruits et modélisés en noir et des objets virtuels animés. Le rendu dans la séquence virtuelle est réalisé par un algorithme de lancer de rayons. Si la réflectance des objets réels est modifiée dans la séquence virtuelle, la réflectance finale est modifiée en composant la nouvelle réflectance avec la réflectance d'origine. Les ombres des objets virtuels sur les objets réels sont modélisées par un facteur d'atténuation à prendre en compte lors de la composition. Un masque d'occlusion est aussi créé pour tenir compte des masquages des objets les uns par rapport aux autres lors de la composition.

L'utilisation de ces deux méthodes est limitée à l'insertion d'objets virtuels. L'éclairage simulé est un éclairage direct. Ne connaissant pas les propriétés des surfaces réelles, il est impossible de simuler un éclairage réaliste mettant en avant des inter–réflexions.

#### Insertion d'objets virtuels avec une estimation simple de la radiométrie de la scène réelle

Nakamae *et al.* [NHIN86] ont proposé les premiers une méthode pour mélanger des photographies avec des objets virtuels, en estimant un éclairage initial. Les images sont calibrées, et une géométrie simple est extraite de la scène. Le point de vue d'une pho-

tographie est retrouvé et le point de vue virtuel est fixé de telle façon à s'accorder avec la photographie. Les photographies sont capturées en extérieur. La position du soleil est calculée en fonction de l'heure, de la date et de la position de la caméra (longitude et latitude). L'intensité du soleil et un terme ambiant sont calculés à partir de deux polygones dans l'image. Cette estimation est très approximative. L'illumination approximative est appliquée virtuellement sur le modèle géométrique, puis elle est ré–ajustée pour satisfaire l'illumination originale de la photographie. L'ajout d'objets virtuels et de brouillard est ensuite réalisé en modifiant un par un les pixels de l'image. Les résultats obtenus par cette méthode sont limités, car les paramètres qu'ils utilisent sont très approximatifs. Cependant, ils étaient les premiers à mettre en évidence la nécessité de trouver un modèle radiométrique des environnements réels pour améliorer la qualité des montages photographiques.

Sato *et al.* [SSI99] proposent eux aussi de n'utiliser que la luminance de l'environnement réel sans en déterminer la réflectance. L'avantage de leur méthode est la simplicité de la capture. Elle nécessite en tout quatre photographies dont la photographie originale qui sera composée. Une autre photographie est prise sous le même point de vue pour permettre de calibrer l'image. Une mire de calibrage plate est placée sur un plan et permet de déterminer une matrice de transformation entre les points 3D et les points 2D de l'image. Le plan sur lequel est posée la mire est un plan de référence ayant pour coordonnées $z = 0$. Ensuite, deux autres séries de photographies sont prises sous des points de vue différents avec une lentille *fish–eye*. Chaque série est composée de photographies prises avec des temps d'exposition différents pour en extraire des images de luminance (voir le chapitre 6, section 6.2). À partir de ces deux dernières séries, des points significatifs sont détectés. Ils correspondent aux points les plus lumineux. Par la matrice de transformation précédemment calculée, une position 3D de ces points est calculée. Ces points sont liés par un maillage triangulaire afin de représenter l'environnement réel. La géométrie des objets n'est donc pas retrouvée. Seul un modèle général est estimé. À partir des images de luminance, la luminance de chaque élément triangulaire est retrouvée. Lors de la composition de l'image réelle avec des objets virtuels, un algorithme simple de rendu est appliqué et chaque pixel de l'image est parcouru pour être modifié. Un rayon est lancé depuis le point de vue au travers du pixel. Si le rayon intersecte un objet virtuel, l'équation de Torrance–Sparrow [TS67] simplifiée est utilisée pour effectuer le rendu. L'irradiance est calculée en utilisant un dôme centré sur l'objet maillé par une géodésique. Les différentes luminances y sont projetées. L'irradiance est égale à la somme des luminances accumulées sur le dôme. Les occlusions par les objets réels ne sont pas traitées. Ceci est évidemment dû à la simplicité du modèle géométrique représentant la scène réelle. Les ombres des objets virtuels sont projetées sur les surfaces réelles en corrigeant la couleur initiale du pixel par un facteur, établi en divisant l'irradiance calculée avec l'objet virtuel, et l'irradiance sans objet virtuel. Si une partie de l'irradiance est bloquée, le facteur assombrira le pixel, ce qui donnera la sensation d'ombre.

Debevec [Deb98] a estimé les propriétés radiométriques de scènes réelles de façon à pouvoir y ajouter des objets virtuels. Contrairement aux méthodes présentées précédemment, la composition est réalisée en tenant compte de l'éclairage global. Dans cette mé-

thode, la scène réelle est décomposée en deux, une scène distante et une scène locale. La scène distante n'est pas visualisable, mais sert à calculer l'éclairage. Des valeurs de luminance sont établies sur une géométrie très simple. Cette luminance est déterminée en photographiant par des images de luminance un miroir sphérique reflétant tout l'environnement, placée à l'endroit de la scène locale. Les valeurs de luminance de la scène distante servent à calculer l'éclairage pour la scène locale. Par contre, il n'y a pas de valeurs de réflectance estimées pour la scène distante, car elle ne réfléchit pas la lumière lors de la simulation.

La scène locale est l'endroit où les objets virtuels sont insérés. Ce modèle local est en interaction directe avec les objets virtuels insérés. Il est important d'avoir un modèle de leur réflectance lors du calcul des échanges lumineux. L'estimation de la réflectance se fait par un algorithme itératif :

1. Choisir un modèle de réflectance (diffus, directionnel, diffus directionnel, ou BRDF arbitraire).
2. Choisir des valeurs initiales approximatives pour les paramètres de la réflectance.
3. Calculer une solution d'éclairage global, avec les paramètres courants.
4. Comparer le résultat simulé avec une ou plusieurs photographies originales.
5. Ajuster les paramètres de réflectance si le résultat n'est pas convaincant, et ré–itérer l'algorithme à partir de l'étape 3.

En pratique, les paramètres sont d'abord fixés pour être idéalement diffus, puis ajustés par itérations. Comme il n'y a pas de modification de l'éclairage, la composante spéculaire devient importante si la réflexion des objets virtuels sur les surfaces réelles est importante pour un résultat convaincant, ou encore si le point de vue change. L'étape x5 de l'algorithme n'est pour l'instant pas automatique, en ce qui concerne le cas non diffus. De plus, les auteurs n'indiquent pas de méthode particulière pour résoudre l'étape 1. Il est à supposer que l'utilisateur fournit les paramètres en entrée.

La résolution de la réflectance dans le cas diffus se fait en utilisant les valeurs de luminance locales, et les valeurs de luminance du reste de la scène. Ces valeurs de luminance sont obtenues en utilisant des photographies sur une plage étendue (voir chapitre 6, section 6.2). Cette méthode d'estimation permet d'obtenir une approximation suffisante pour permettre d'insérer localement des objets virtuels. Cependant, elle n'est pas suffisante pour permettre de réaliser des modifications de l'éclairage initial.

La composition avec les objets virtuels est faite par addition de la scène locale avec la scène virtuelle. Cependant, la simulation de la scène locale est différente de la photographie originale. Un terme d'erreur est calculé entre la scène locale simulée et la scène locale sur la photographie. Le rendu final est la luminance de la scène locale photographiée moins celle de la scène locale simulée plus celle de la scène virtuelle. Les ombres projetées sont affichées lorsque la luminance de la scène virtuelle est plus sombre que celle de la scène locale simulée. Les effets spéculaires sont affichés lorsque la scène virtuelle a une luminance supérieure à la scène locale simulée.

Cette méthode décomposant la scène réelle en deux parties permet d'éviter un modèle trop complexe. Le rendu est effectué avec le système RADIANCE [War94b] qui réalise

la simulation d'éclairage par lancer de rayons. Ceci permet de prendre en compte les effets spéculaires, et produit des résultats très réalistes (voir figure 2.14). Cependant, à cause de l'utilisation du système RADIANCE, la simulation de chaque image prend un temps conséquent (en général 20 minutes par image par RADIANCE).



(a) (b)

FIG. 2.14: Résultats de Debevec [Deb98]. (a) Photographie originale. (b) Ajout d'objets virtuels dans la scène avec prise en compte de l'éclairage commun (ombres et effets spéculaires). (Figures extraites de l'article).

Fournier *et al.* [FGR93] sont les premiers à avoir considéré l'éclairage global pour la simulation d'éclairage commun. La scène réelle est approximée par des boîtes, dont une incluant le reste de la scène. Quelques points de cette reconstruction « manuelle » servent à calibrer l'image originale et à retrouver le point de vue. Des textures sont extraites pour chacun des objets à partir de la photographie originale. Le modèle est simple et doit servir à calculer une approximation de l'éclairage global initial. Afin de calculer un éclairage global, il faut estimer des paramètres de réflectance pour les objets réels, et l'intensité des sources de lumière. Seules des réflectances diffuses sont estimées. La réflectance des objets est estimée directement à partir des pixels des textures, par une moyenne des pixels, pondérée par une radiosité ambiante et une réflectance ambiante. Une intensité pour chacune des sources de lumière est ensuite estimée à partir des radiosités des objets. Nous détaillons l'estimation de ces paramètres dans la section 4.2.1 car nous l'utilisons dans une de nos méthodes présentées au chapitre 4. Grâce à ces paramètres, une simulation de l'éclairage réel est calculée par une solution de radiosité progressive. Cette solution de radiosité sert de référence. Lors de l'insertion d'objets virtuels, une nouvelle solution de radiosité est calculée. La solution affichée tiendra compte d'un facteur étant égal à la radiosité modifiée par l'insertion d'un objet, divisée par la solution de référence. Le rendu final est affiché pixel par pixel. Si le pixel contient un objet virtuel, le rendu de l'objet virtuel est affiché, sinon, une correction du pixel original est calculée en se basant sur le facteur de correction présenté ci–dessus. Dans les régions d'ombre de l'objet virtuel, la texture est donc assombrie. La solution présentée par Fournier *et al.* est non–interactive en ce qui concerne les modifications de la scène réelle.

Drettakis *et al.* [DRB97] ont étendu le travail de Fournier *et al.*, et l'ont adapté à des algorithmes de radiosité hiérarchique sur un modèle texturé. Le modèle texturé est construit par des méthodes de vision réunies, telles que la construction de mosaïques à partir de trois images consécutives [ZFD97], la reconstruction semi–automatique possible par

TotalCalib [BR97], et l'extraction de textures dépendantes du point de vue [FLR+97]. Les techniques d'évaluation des paramètres sont similaires à celles de Fournier *et al.* [FGR93]. Nous les décrivons en détail dans le chapitre 4. L'affichage est cependant différent. Il permet de modifier directement les textures par un facteur de correction entre la radiosité modifiée et la radiosité originale, sans passer par un lancer de rayons. Dans leur implantation, Drettakis *et al.* [DRB97] utilisent les algorithmes de radiosité dynamique [DS97] pour insérer les objets virtuels et les déplacer dans la scène *interactivement*. C'est la seule méthode précédant cette thèse qui permettait le traitement interactif de l'éclairage commun global.

**Estimation valide de la radiométrie permettant le ré–éclairage de scènes réelles**

Les méthodes présentées ci–dessus traitent simplement, et pour certaines interactivement, l'insertion d'objets virtuels dans une scène réelle en tenant compte de l'éclairage commun. Quelques–unes [Deb98, FGR93, DRB97] simulent l'éclairage global. Les autres [SHC+96, JNP+95, NHIN86, SSI99] se contentent d'un éclairage direct approximatif. Toutes ces méthodes ne permettent pas de réaliser un ré–éclairage des scènes réelles, en modifiant l'éclairage original, présent lors de la capture.

Les travaux que nous avons présentés dans la section 2.4.3, retrouvent les paramètres de l'illumination inverse et permettent un ré–éclairage de scènes réelles.

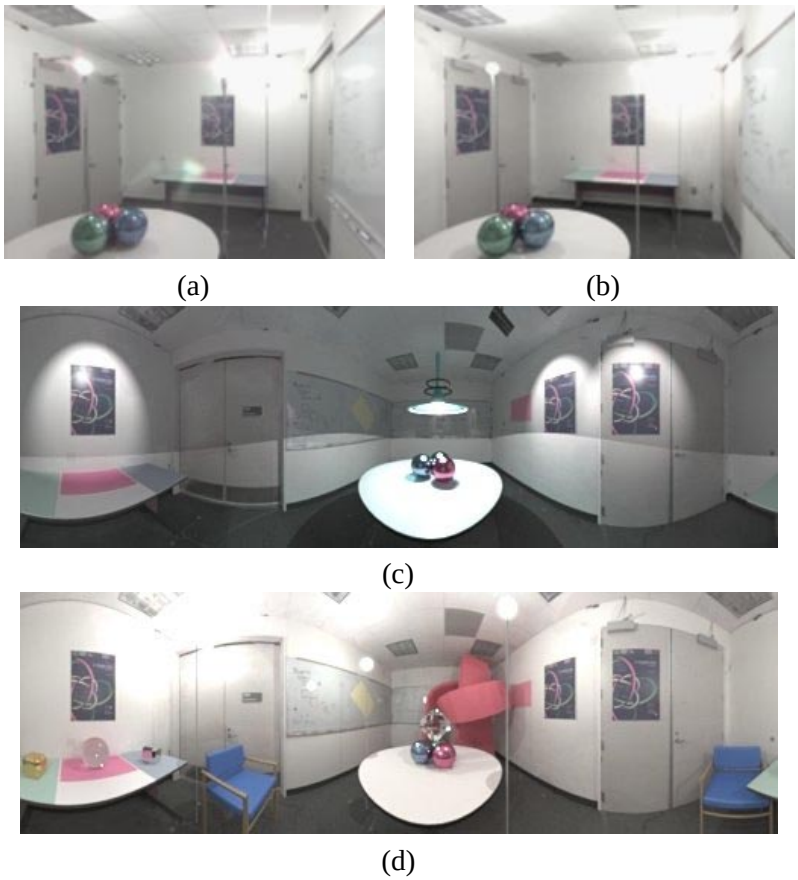
Yu et Malik [YM98] ont retrouvé les paramètres d'illumination inverse pour des scènes d'extérieur. Ils proposent un rendu permettant de simuler une position du soleil différente tout au long de la journée. Le premier problème est de déterminer une position cohérente du soleil. Ceci est fait de la même façon que Nakamae *et al.* [NHIN86]. La couleur du ciel dépend ensuite de la position du soleil. Le ciel est modélisé par un dôme maillé. À chaque maille sont associés trois paramètres. Un premier paramètre est un facteur d'échelle de luminosité. Ce facteur reste le même, sauf au lever et au coucher du soleil. Un deuxième paramètre définit la couleur générale du ciel en fonction de la position sur le dôme. Un dernier paramètre reflète l'auréole du soleil et dépend de la position du soleil. Avant de faire le rendu final, une luminance approximative est estimée pour le panorama, après avoir déterminé quelques normales par un algorithme de vision. Le rendu final n'utilise plus les images d'entrée. Seuls les bâtiments modélisés et texturés apparaissent dans la simulation (voir figure 2.15). On peut regretter que les auteurs ne composent pas mieux le rendu synthétique avec les images réelles, en utilisant la géométrie approximative qu'ils ont pour le ciel (dôme) et le panorama.

Yu *et al.* [YDMH99] ont proposé la seule méthode qui réalise un ré–éclairage de scènes réelles en conservant tous les détails présents dans les photographies. Nous avons détaillé dans la section 2.4.3 comment les auteurs retrouvaient les paramètres de réflectance des objets réels. Les objets sont modélisés de façon assez précise. Ils ont tous une texture de réflectance diffuse, et une composante spéculaire par polygone. Comme pour une scène purement virtuelle, la simulation d'éclairage est réalisée par un algorithme de lancer de rayons, RADIANCE [War94b]. Des objets virtuels et des lampes virtuelles peuvent être ajoutés, les lampes réelles éteintes. En utilisant RADIANCE, la simulation des modifications ne peut être réalisée en temps interactif.



(a) (b)

FIG. 2.15: Résultats de Yu *et al.* [YM98]. (a) Photographie originale. (b) Image simulée avec l'éclairage original. (Figures extraites de la page web : `http ://www.CS.Berkeley.EDU/~yyz/research/arch.html` ).



(a) (b)

(c)

(d)

FIG. 2.16: Résultats de Yu *et al.* [YDMH99]. (a) Photographie originale. (b) Image simulée avec l'éclairage original. (c) Image ré–éclairée par un éclairage virtuel. (d) Objets virtuels insérés dans la scène. (Figures extraites de la page web : `http ://www.CS.Berkeley.EDU/~yyz/research/igi.html` ).

Évidemment, une telle méthode ne pourrait pas être appliquée pour un environnement extérieur. De plus, le processus de capture et d'estimation des paramètres nécessite de nombreuses prises de photographies. Comme le montre la figure 2.16, les résultats sont de qualité, mais les modifications de l'environnement réel ne sont pas traitées de façon interactive.

## 2.5 Discussion

Dans ce chapitre, nous avons présenté plusieurs domaines de la synthèse d'images, et en partie de la vision par ordinateur, qui concernent notre travail, comme la simulation interactive de l'éclairage, la réalité augmentée et les méthodes d'estimation de la réflectance pour le ré–éclairage des scènes réelles.

Les problèmes liés à la simulation de l'éclairage sont divers. Il faut d'abord définir les paramètres de la simulation, comme la BRDF des objets. Ensuite, il faut choisir un algorithme adapté au résultat que l'on souhaite avoir. Les algorithmes par lancer de rayons permettent d'obtenir des résultats de haute qualité. Cependant, sauf après adaptation de techniques spéciales, le point de vue ne peut être changé, et le coût de calcul pour chaque image est très élevé. Les algorithmes de radiosité permettent de simuler un éclairage global de qualité, indépendant du point de vue. Ils ne permettent cependant pas facilement d'intégrer des effets spéculaires, et se limitent souvent à des environnements diffus. Des méthodes incrémentales ont été développées pour permettre des modifications géométriques interactives.

Nous avons présenté également les différentes méthodes permettant d'extraire les paramètres d'une scène réelle afin de les modifier virtuellement. Certains algorithmes se contentent d'insérer des objets virtuels en maintenant un éclairage global cohérent. D'autres ont recherché des paramètres plus justes pour pouvoir ré–éclairer des scènes réelles. De nombreux travaux ont mis en avant la nécessité d'une capture non exhaustive des scènes réelles tout en permettant ces simulations.

La réalité augmentée implique un aspect temps réel. Peu des travaux précédents se sont préoccupés de cette propriété. Le temps réel ne semble pas accessible pour l'instant si on veut tenir compte de l'éclairage global. Cependant, nous souhaitons nous rapprocher du temps réel en diminuant les temps de calcul et en permettant l'interactivité.

Le but du ré–éclairage non–interactif est atteint en partie pour la méthode de Yu *et al.* [YM98] et Yu *et al.* [YDMH99]. Il est à noté que ces travaux ont été développés en même temps que nos recherches. De plus, ces méthodes sont très différentes des notres, et ne visent pas du tout des modifications interactives, qu'elles ne sont pas capables de traiter. Malgré leur qualité, elles souffrent de restrictions que nos travaux présentés dans ce document n'ont pas.

Dans les chapitres 4 et 5, nous décrivons deux approches, qui se donnent comme objectif commun de s'appuyer sur un processus simple et accessible en coût matériel à un professionnel du design et d'architecture. Cependant, la mise en place de notre travail de recherche comporte des difficultés matérielles. Nous avons dû choisir et utiliser des techniques de capture du monde réel. Bien plus que pour des méthodes purement virtuelles,

ce travail nécessite une période longue et souvent pénible pour obtenir la configuration *physique* désirée (lampes, pièces spéciales, appareil photographique, etc.). Nous discutons les moyens mis en place pour la capture dans le chapitre suivant 3.

Dans nos méthodes, nous recherchons une estimation suffisamment cohérente des paramètres pour permettre le ré–éclairage et la modification géométrique des scènes réelles. Cette estimation doit se faire dans un temps de calcul raisonnable. Enfin, nous avons développé des méthodes permettant de gérer ces modifications virtuelles dans un temps interactif tout en conservant un éclairage de qualité.

Enfin, nous présentons dans le chapitre 6 un algorithme de calibrage des photographies prises avec un appareil semi–automatique, dont le temps d'exposition n'est pas contrôlable. Ceci nous permet d'obtenir, avec du matériel peu coûteux, des données équivalentes à des images de luminance [DM97] obtenues par des appareils photographiques de haute qualité et manuellement contrôlables, notamment au niveau du temps d'exposition. Ceci nous permet d'estimer de façon plus fine la réflectance pour chacune des deux méthodes et d'en améliorer la qualité des résultats. Nous proposons un dernier chapitre 7 dans lequel nous comparons les deux méthodes, et proposons de nouvelles extensions comme le déplacement des objets réels et la parallélisation d'un de nos algorithmes.

# Chapitre 3

# Problèmes d'acquisition

Dans cette thèse, nous nous intéressons au problème de modifications virtuelles des scènes réelles. Comme nous l'avons expliqué dans le chapitre précédent, ceci nécessite l'acquisition d'un modèle 3D texturé des scènes réelles. Dans ce chapitre, nous décrivons l'étendue du travail préliminaire et les différents choix que nous avons faits. Cette thèse n'est pas une thèse classique de synthèse d'image. Travailler avec la réalité présente de nombreuses difficultés auxquelles il faut faire face avant de s'attaquer à la résolution des problèmes de recherche qui nous intéressent. La réalité est l'environnement le plus complexe que l'on peut trouver ; le fait de le capturer oblige déjà à se restreindre et à appliquer des approximations. À chaque approximation, il y a une perte de l'information, ce qui peut amener à de nombreuses erreurs. Nous avons dans ce chapitre décrit les différents choix que nous avons faits. Nous espérons que cette description servira de bon point de départ à d'autres chercheurs désirant travailler sur le mélange du virtuel et de la réalité.

Pour avoir une connaissance de la géométrie de la scène réelle, nous avons décidé d'utiliser des outils de reconstruction à base d'images. Nous avons donc besoin de *capturer* l'environnement réel, pour en obtenir des images. Ceci implique de nombreux aspects techniques que nous décrivons dans ce chapitre. Parmi ces aspects, il y a celui du choix du matériel, tel que la caméra servant à la capture, ou les lampes éclairant la scène réelle. Ce choix de matériel a été guidé à la fois par des aspects de coût financier et par la simplicité d'utilisation.

Ensuite, nous avons eu à choisir et à utiliser des systèmes de reconstruction à base d'images. Nous en avons choisi deux. Le premier est un système développé par l'équipe Robotvis à Sophia–Antipolis, appelé *TotalCalib*, qui est en cours de commercialisation par la société *Realviz* [Rea]. Le deuxième système choisi est *Rekon*, un outil de reconstruction à base de contraintes développé dans le laboratoire d'informatique graphique (LIGUM) de l'Université de Montréal. Enfin, il a fallu traiter toutes ces données pour les rendre compatibles à nos systèmes de simulation.

Dans ce chapitre, nous décrivons tous les choix réalisés, ainsi que les difficultés générales auxquelles il faut faire face lorsque l'on travaille avec la réalité.

## 3.1 Les caméras utilisées

Avant de reconstruire un modèle géométrique de l'environnement réel, nous avons besoin de le *capturer* au moyen d'une caméra produisant des images de l'environnement réel, et de ramener ces données images sur un ordinateur. Nous avions plusieurs choix de capture possibles.

Nous pouvions capturer ces images avec un appareil photographique traditionnel, et numériser ces photographies. Dans ce contexte, la résolution et la qualité des photographies sont bien supérieures à celles d'appareils numériques. De plus, les prix des appareils traditionnels de bonne qualité sont abordables. En revanche, cette solution comporte de nombreux inconvénients. En effet, elle ne facilite pas la tâche des captures. En utilisant un appareil traditionnel, il faudrait prendre toutes les photographies, puis développer la pellicule pour savoir si les photographies correspondent vraiment à nos besoins, et enfin les essayer dans nos systèmes. Si les résultats se révélaient incomplets, il nous faudrait recommencer la série de captures. De plus, les variations des réactions chimiques lors du développement, ainsi que les compensations lors du développement des films font que les photgraphies ne sont pas véritablement représentatives de la réalité. Enfin, le coût engendré par les multiples développements nous a paru compensé par l'achat d'un appareil numérique.

À cause de ces inconvénients, nous avons choisi des caméras numériques. Dans notre objectif de départ, nous désirions développer un système qui fonctionnerait pour des environnements intérieurs. Nous avons tout d'abord choisi une caméra Sony EVI-D31, branchée directement sur une station de travail graphique SGI-O2 munie d'une carte de numérisation vidéo, pouvant être télécommandée en zoom ou en position autour de son axe. Cette caméra est en photographie sur la figure 3.1 (a). Elle est très adaptée à l'outil de reconstruction *TotalCalib* car elle est pratique pour créer des mosaïques, qui sont des données d'entrée au système de reconstruction. Le *déplacement télécommandé* et la possibilité de capture directement sur ordinateur nous ont paru être des avantages significatifs. Cependant, un des inconvénients est que nous utilisions cette caméra branchée sur ordinateur. De plus, la résolution des images ($640 \times 480$) ne nous suffisait plus.

À cause des inconvénients cités ci-dessus, nous nous sommes également intéressés à d'autres appareils photgraphiques numériques, dont la technologie évolue rapidement. Afin de mieux diriger notre choix, nous avons essayé de définir clairement nos besoins. Parmi ceux-ci se trouvait notre souhait de produire des images de luminance (voir le chapitre 6), à partir d'images prises à différents temps d'exposition. De plus, nous voulions avoir une résolution de bonne qualité. Les appareils répondant à ces besoins sont des appareils manuels traditionnels, montés sur un système de capture numérique. Le coût de tels appareils était au–dessus de notre budget*. Nous avons donc restreint nos choix sur des appareils numériques semi–automatiques, ayant une bonne résolution d'image. L'appareil choisi est le Kodak DC260, montré en photographie sur la figure 3.1 (b). Cet appareil est programmable par des scripts. Les fonctions manuelles sont pour certaines accessibles uniquement au travers de ses scripts. Les scripts doivent être chargés sur l'appareil pho-

*en–dessous de 10,000 FF



(a) (b)

FIG. 3.1: Les deux appareils que nous utilisons pour prendre en photographie des scènes réelles. (a) Caméra Sony EVI-D31. (b) Appareil photographique Kodak DC260.

tographique sur la carte servant aussi à stocker les photographies. Ils sont écrits dans un langage pseudo–code spécifique. Nous avons utilisé ces scripts, et nous décrivons un exemple d'utilisation dans le chapitre 6. Cet appareil s'est révélé très pratique, bien que limité dans ses fonctions manuelles.

## 3.2 Reconstruction des scènes réelles à partir de photographies

Comme nous l'avons expliqué précédemment, nous avons besoin d'obtenir un modèle géométrique de la scène réelle afin de l'utiliser dans nos systèmes. Dans l'état de l'art (section 2.4.2), nous avons présenté différentes méthodes pour reconstruire des scènes réelles à partir d'images. Ces méthodes pouvaient être classées en deux grands groupes : les méthodes automatiques, et les méthodes semi-automatiques. Nous avons sélectionné deux méthodes semi-automatiques afin d'avoir un meilleur contrôle sur le résultat. De plus, les systèmes choisis sont des systèmes de recherche. Nous désirions ainsi bénéficier d'une collaboration entre les équipes pour adapter les systèmes selon nos besoins. Le premier système utilisé est l'outil de reconstruction *TotalCalib* [BR97] de l'équipe ROBOTVIS de Sophia–Antipolis. Cet outil est basé sur des concepts de vision 3D. Nous le présentons dans la section suivante. Le deuxième outil choisi est le système *Rekon* [POF98] développé dans le laboratoire LIGUM à Montréal. Ce système est basé sur l'intervention de l'utilisateur, qui doit fixer des contraintes géométriques dans le système afin de l'aider dans la résolution. Alors que *TotalCalib* utilise des points et des contraintes épipolaires, l'outil *Rekon* utilise des points, des lignes et des polygones. Le calibrage dans *Rekon* est aidé par des contraintes géométriques telles que le parallélisme, la perpendicularité, et la co–planarité. Nous présentons le système *Rekon* dans la section 3.2.2. Le système *Rekon* a une approche très similaire de l'outil de reconstruction *Façade* [DTM96].

### 3.2.1 Un premier outil de reconstruction : *TotalCalib*

Le premier outil de reconstruction que nous avons utilisé s'appelle *TotalCalib*. Ce système de reconstruction a été développé par l'équipe ROBOTVIS à Sophia–Antipolis. Son implantation [FLR+97] est basée sur les principes de la vision 3D, et utilise différentes techniques : le calibrage automatique de la caméra [Rob95], la construction de mosaïques [ZFD97], la géométrie épipolaire qui permet d'obtenir un modèle géométrique de la scène [BR97], et enfin l'extraction de textures dépendantes du point de vue.

Le calibrage de la caméra est réalisé en dehors de *TotalCalib*, par une technique développée par Robert [Rob95]. Ce calibrage est semi–automatique. L'utilisateur donne pour six points de référence, leurs coordonnées dans un espace 3D, et leurs coordonnées sur l'image. Ces points doivent être distribués sur une mire de calibrage, comme par exemple sur l'image 3.2. Ces six coordonnées sont ensuite utilisées par le système pour déterminer les paramètres intrinsèques et extrinsèques de la caméra.

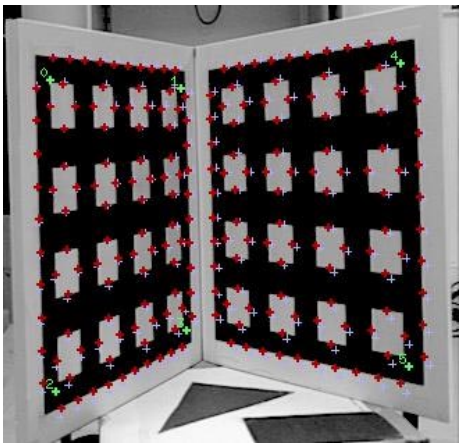


FIG. 3.2: Mire de calibrage, et les six points d'entrée pour retrouver les paramètres de la caméra. (Figure extraite de la page web : `http ://www-sop.inria.fr/robotvis/personnel/lucr/detecproj.html`)

Ensuite, des mosaïques sont reconstruites pour une séquence de trois images [ZFD97]. Un exemple de quatre mosaïques servant à reconstruire un modèle est montré sur la figure 3.3. Pour reconstruire ces mosaïques, un ensemble de 12 images est nécessaire (3 images par mosaïque). *TotalCalib* n'impose pas des mosaïques en entrée. Cependant, l'utilisation de mosaïques permet une reconstruction plus large, permettant d'avoir des polygones couvrant plus qu'une seule image (notamment l'image centrale).

Ces mosaïques sont passées en entrée dans le système *TotalCalib*, qui va permettre de reconstruire le modèle 3D des objets visibles sur ces mosaïques. Ce système montré sur la figure 3.4 combine différentes techniques de vision par ordinateur. L'utilisateur entre les points en cliquant sur une image avec la souris. Pour les premiers points, la correspondance de points est réalisée par l'utilisateur qui associe les points correspondant au même point 3D entre eux. Ensuite, le système est capable de retrouver automatiquement les points correspondants sur les trois autres mosaïques, en utilisant la matrice de projection et la recon-

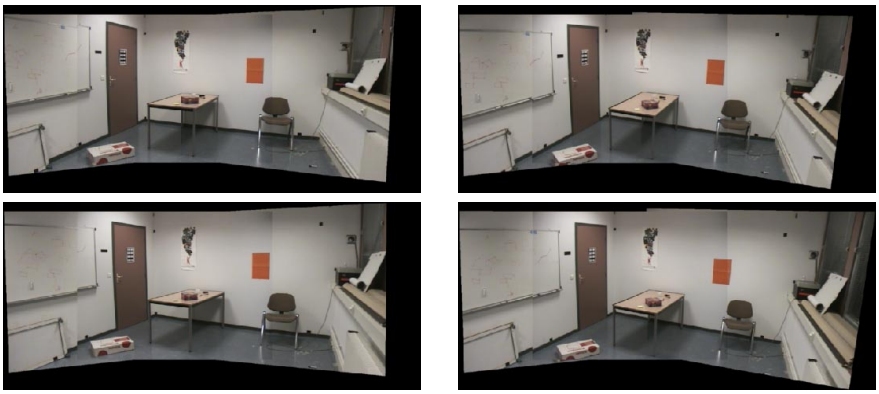


FIG. 3.3: Les quatre mosaïques servant d'entrée au programme TotalCalib.

naissance de discontinuités locales dans l'image. Avec à peu près trente points entrés et mis en correspondance sur les images, le système est capable de retrouver les matrices fondamentales (matrices de projection et les matrices de paramètres intrinsèques), en utilisant une méthode non–linéaire [ZDFL95]. Les polygones sont ensuite reconstruits manuellement, en reliant les points entre eux. Les coordonnées 3D de ces polygones sont retrouvés par le système en utilisant la matrice de projection. Sur la figure 3.4, le système *TotalCalib* est illustré, avec les quatre mosaïques servant à la reconstruction. Sur la fenêtre en haut à gauche, les polygones ont été entrés par l'utilisateur.

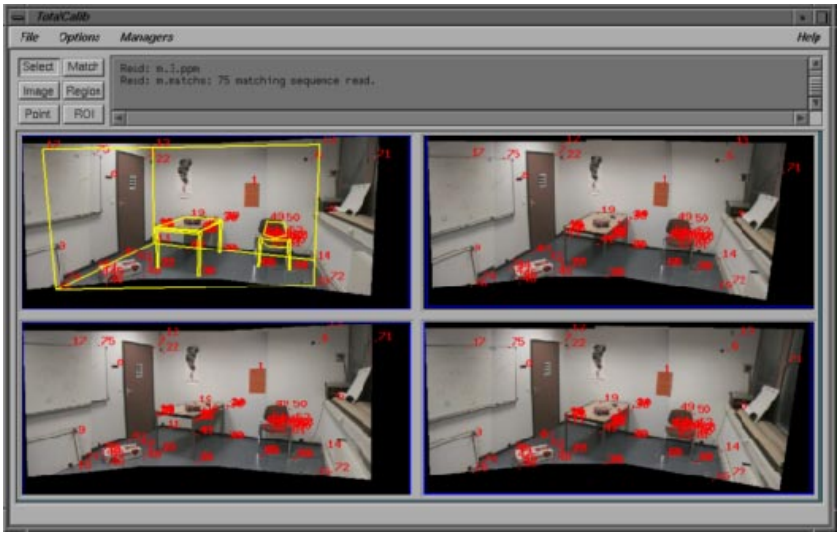



FIG. 3.4: Le système *TotalCalib*. Dans la fenêtre en haut à gauche, des polygones ont été créés à la main par l'utilisateur, en reliant des points précédemment introduits. Ces points ont servi au calibrage des images et pour retrouver les matrices de projection pour chacune des mosaïques.

Les paramètres intrinsèques ne sont pas toujours correctement retrouvés par le système. Dans ce cas, nous utilisons les paramètres intrinsèques préalablement retrouvés en utilisant

le calibrage automatique de la caméra à l'aide d'une mire. Un exemple de modèle reconstruit par *TotalCalib* est présenté sur la figure 3.5.

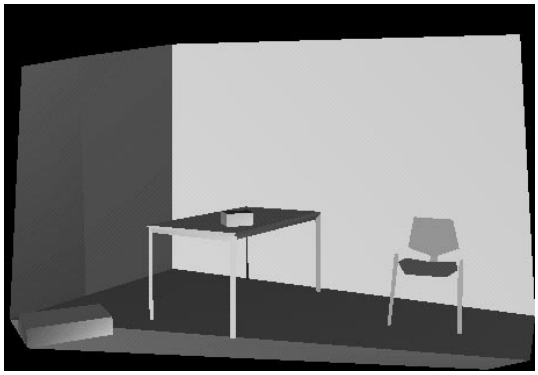


FIG. 3.5: Modèle reconstruit avec le système *TotalCalib*, correspondant aux images présentées sur la figure 3.3.

Des textures dépendantes du point de vue sont extraites [FLR+97] à partir du modèle reconstruit, d'une des mosaïques, et des matrices de projection. Pour chaque polygone reconstruit, l'image originale (mosaïque) est détordue et ré–appliquée sur le polygone. Le modèle texturé est montré sur la figure 3.6 (a). En (b), le modèle texturé est montré depuis un point de vue différent de celui de l'image d'origine. Les textures ainsi extraites contiennent les images des objets qui sont entre le polygone et le point de vue.



(a) (b)

FIG. 3.6: (a) Modèle texturé construit avec *TotalCalib*. (b) Même modèle mais sous un autre point de vue. Cette image met en évidence que les textures sont dépendantes du point de vue.

Comme nous l'avons décrit, le système est semi–automatique. L'intervention de l'utilisateur est réduite à l'introduction de points dans *TotalCalib*, ainsi qu'à la création des polygones sur l'image. Le modèle géométrique ainsi reconstruit représente correctement la réalité. Cependant, le système ne garantit pas les propriétés géométriques, telles que la planarité d'un polygone, la perpendicularité entre polygones ou le parallélisme.

**Mise en place**

Nous avons exposé ci-dessus la description technique de la méthode de reconstruction. Nous décrivons plus en détail ici, comment nous avons utilisé ces techniques.

Dans un premier temps, nous avons calibré notre caméra, pour une distance focale donnée. Pour toutes les captures, nous avons utilisé la même distance focale (fixée à l'infini). Les mires de calibrage couramment utilisées en vision sont assez chères. Nous avons donc construit notre propre mire, avec un support carton, et un schéma régulier créé par ordinateur et imprimé sur du papier. Cette mire est présentée sur la figure 3.7.



FIG. 3.7: Notre mire de calibrage, construite avec un socle en carton, sur lequel sont fixés deux schémas de grille.

Nous avons ensuite mesuré les six points requis par le programme de calibrage, qui a déterminé les paramètres de notre caméra.

La capture des scènes se faisait ensuite dans les locaux du laboratoire. Comme le montreront les chapitres suivants, nous avions besoin d'avoir une scène close, pour être sûrs d'avoir un éclairage global cohérent. N'ayant ni rideaux opaques, ni stores masquant vraiment la lumière extérieure, les captures se faisaient lorsqu'il faisait nuit. La capture pour le système *TotalCalib* a été effectuée par la caméra Sony présentée dans la section 3.1. L'ordinateur sur lequel était branchée la caméra devait être déplacé pour se trouver dans la salle où nous réalisions les captures.

Pour capturer les mosaïques, nous prenons trois images successives, en télécommandant la caméra pour une rotation horizontale. La capture des mosaïques s'est révélée longue. L'ajustement des images entre elles ne fonctionnait pas à chaque fois, et nécessitait souvent plusieurs essais. Dans la figure 3.8, nous montrons un exemple d'échec d'ajustement des images pour créer une mosaïque. Les trois premières images utilisées, (a), (b), et (c) n'ont pas permis de reconstruire une mosaïque correcte en (d) (voir la région encadrée). Les images ne se recouvraient pas assez. Nous avons repris une nouvelle photographie pour l'image centrale en (f), qui nous a permis d'obtenir une meilleure mosaïque en (h).

Une fois les mosaïques reconstruites, nous utilisons le système *TotalCalib* pour reconstruire la scène. *TotalCalib* s'est révélé être un outil très pratique. Les reconstructions étaient relativement rapides (quelques heures). Il nous a cependant fallu du temps pour

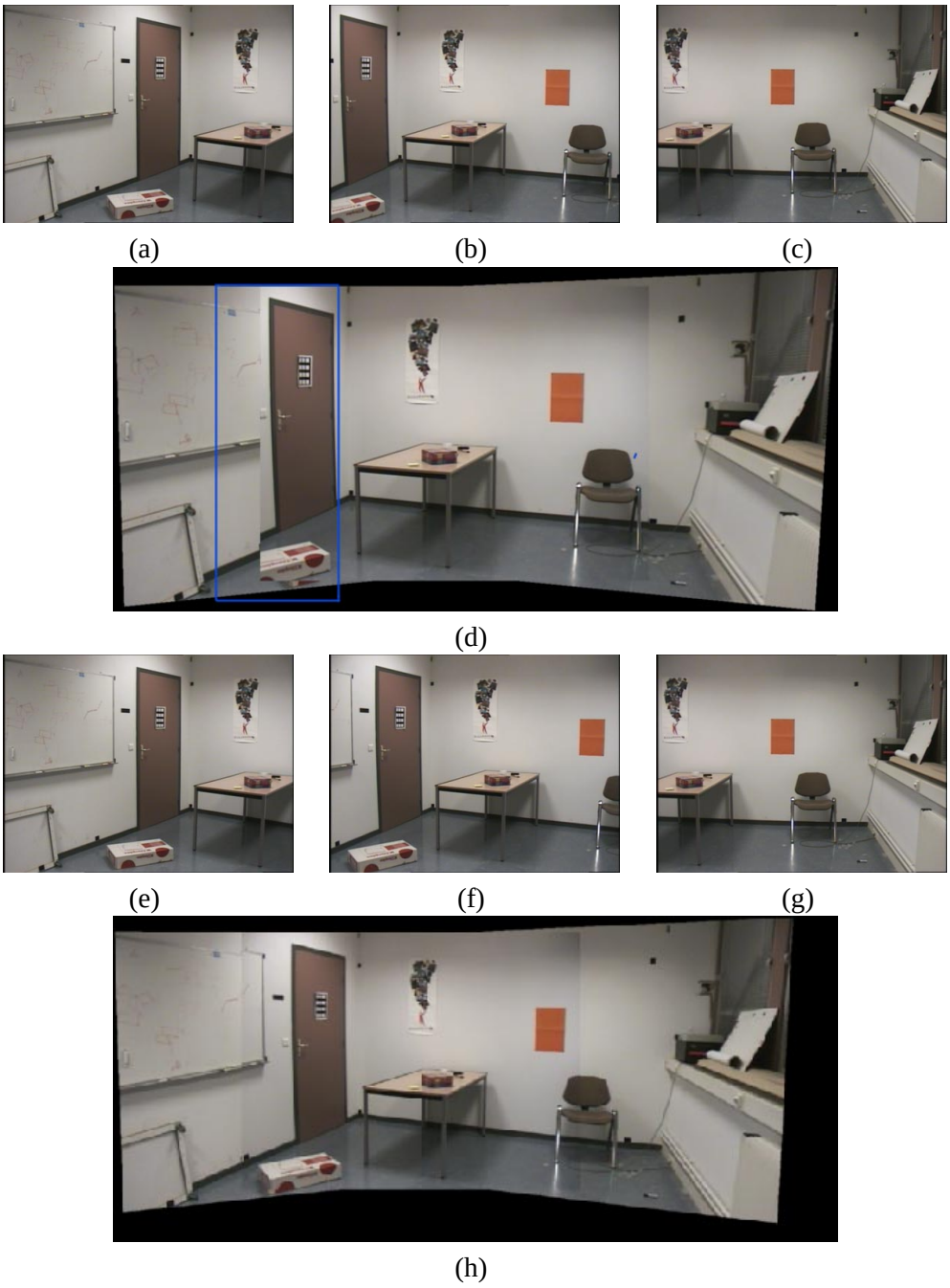



FIG. 3.8: (a), (b) et (c) ont servi à crée la mosaïque (d). Cette mosaïque n'a pas été ajustée correctement sur la gauche (voir la région encadrée), car les images (a) et (b) ne se recouvrent pas suffisamment. (e), (f), et (g) ont servi à la création de la mosaïque (h), qui cette fois est très bien ajustée. Pour réussir l'ajustement de cette mosaïque, l'image du centre (b) a été remplacée par l'image (f). Les autres images (a) et (e), et (c) et (g) sont identiques.

acquérir les compétences nécessaires, et prendre l'habitude de bien distribuer les points dans les images, ou prendre les mosaïques d'entrée sous des points de vue suffisamment différents. Durant cette phase d'apprentissage, la reconstruction s'étendait alors à plusieurs jours. Pour améliorer le modèle, il fallait parfois reprendre de nouvelles photographies, avec de meilleurs repères et plus d'objets pour varier le placement des points. Le calibrage de la caméra avec la mire s'est avéré très utile. Nous avons pu aider *TotalCalib* à calibrer notre scène.

Un mauvais calibrage avait souvent pour conséquence une mauvaise correspondance entre les textures extraites et les polygones. Étant donné que le modèle est regardé sous un seul point de vue, cela pouvait être acceptable dans une certaine mesure, car l'image générale restait cohérente.

Ce système nous a permis de reconstruire des scènes réelles avec des défauts de reconstruction minimes. La reconstruction générale se fait en plusieurs étapes. Nous devions recommencer du début si par exemple nous n'arrivions pas à calibrer correctement le système en retravaillant la scène, ou en prenant de nouvelles photographies sous un autre point de vue. En général, la capture complète de la phase d'arrangement de la scène, puis la prise de photographies, jusqu'à l'obtention du modèle, nous prenait une semaine. *TotalCalib* est en cours de commercialisation par la société *Realviz* [Rea]. Cet outil n'est plus un outil de recherche, et sa configuration, notamment son interface, est plus adaptée à un utilisateur inexpérimenté.

### 3.2.2 Un deuxième outil de reconstruction : *Rekon*

Nous avons également travaillé avec un autre outil de reconstruction, qui est le système *Rekon* développé à l'Université de Montréal, dans le laboratoire d'informatique graphique (LIGUM). Il est différent du système *TotalCalib* sur plusieurs plans. *TotalCalib* se veut le plus automatique possible. L'intervention de l'utilisateur est restreinte. *Rekon* au contraire demande souvent à l'utilisateur d'intervenir. Cette intervention permet de réduire le nombre de points nécessaires à une première reconstruction. Alors que *TotalCalib* nécessite une trentaine de points, *Rekon* en demande six. Ces six points doivent cependant être connus parfaitement de l'utilisateur, et si possible non planaires. Ce peut être par exemple des points placés sur les coins d'une boîte. L'utilisateur entre les coordonnées de ces six points à la main. Le système utilise cette correspondance entre le 2D et le 3D pour calibrer la caméra et construire une première matrice de projection. Ce premier calibrage peut être comparé au calibrage sur une mire, comme nous l'avons fait pour *TotalCalib*. Pour réaliser une reconstruction, le système a besoin que pour chaque image, six points soient calibrés. Ce peut être soit des points dont les coordonnées 3D ont été entrées par l'utilisateur, soit des points déjà calibrés dans les autres images. Ce qui veut dire que nous devons avoir deux images au moins contenant les six points de départ entrés par l'utilisateur. On peut ensuite ajouter des nouveaux points 2D sur des images calibrées, et leurs coordonnées 3D seront calculées par le système. Dans la phase de calibrage, l'utilisateur peut entrer des points, des polygones ou des lignes. La mise en correspondance des polygones permet de les étiqueter, et de savoir que le même polygone se trouve sur plusieurs images. Pour obtenir un meilleur calibrage et une reconstruction de meilleure qualité, *Rekon* a besoin d'images d'entrée dont le point de vue est très différent.

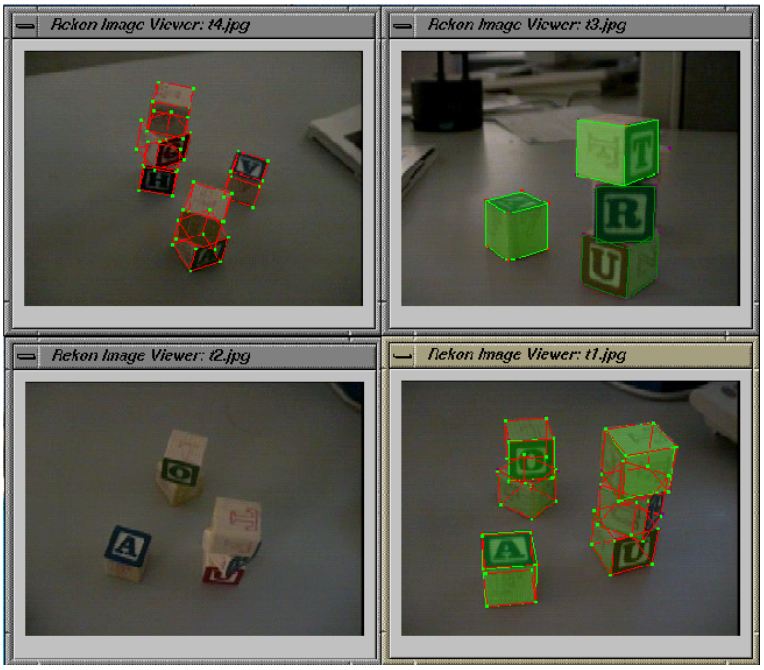



FIG. 3.9: Le système *Rekon*, avec quatre images sur lesquelles l'utilisateur construit un modèle. Le modèle 2D dessiné par l'utilisateur est montré par exemple sur l'image en haut à droite. Le modèle 3D reprojeté sur l'image est montré par exemple en haut à gauche et en bas à droite. (Figure extraite du mémoire de Frasson [Fra99])

Pour améliorer le calibrage, l'utilisateur *aide* le système en lui indiquant les contraintes géométriques qui existent dans la scène. Ce sont des contraintes de planarité, de co–planarité, de perpendicularité et de parallélisme. La contrainte de co–planarité peut s'appliquer entre deux polygones, ou entre un point et un polygone. Cette contrainte est très pratique lorsqu'on veut indiquer au système que deux polygones se touchent. L'utilisateur peut ajouter autant d'images dans le système, qu'il lui en faut pour reconstruire les polygones de la scène. Le système itére autant de fois que c'est nécessaire pour atteindre une convergence.

L'extraction des textures est intégrée directement dans le système. Les textures construites sont indépendantes du point de vue. L'utilisateur choisit un polygone sur une image, et le système sélectionne automatiquement toutes les images où le polygone a été reconstruit. Le système extrait des textures sur chacune des images, et les combine en les pondérant par une valeur de validité. Cette validité est calculée en tenant compte de la visibilité depuis l'œil, de l'angle de vue, et d'autres paramètres.

Le système *Rekon* est comme *TotalCalib* un système complet, qui fournit un modèle géométrique texturé. L'intervention de l'utilisateur est plus importante cependant que pour le système *TotalCalib*.

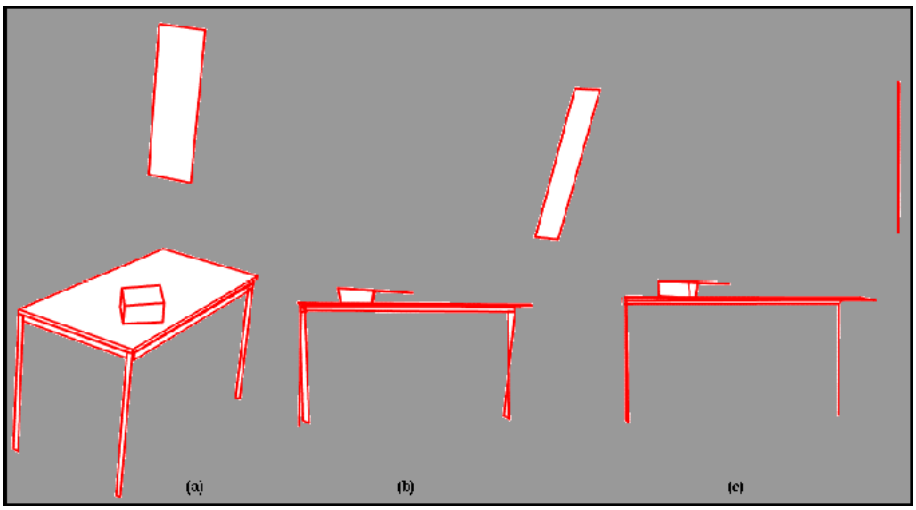



FIG. 3.10: À gauche, on voit la scène reconstruite sous un point de vue. Au milieu, on voit la scène reconstruite sous un point de vue différent mettant en évidence les erreurs de reconstruction. À droite, le système de reconstruction a pris en compte des contraintes géométriques qui ont permis d'améliorer la reconstruction. (Figure extraite de la thèse de Marie–Claude Frasson [Fra99])

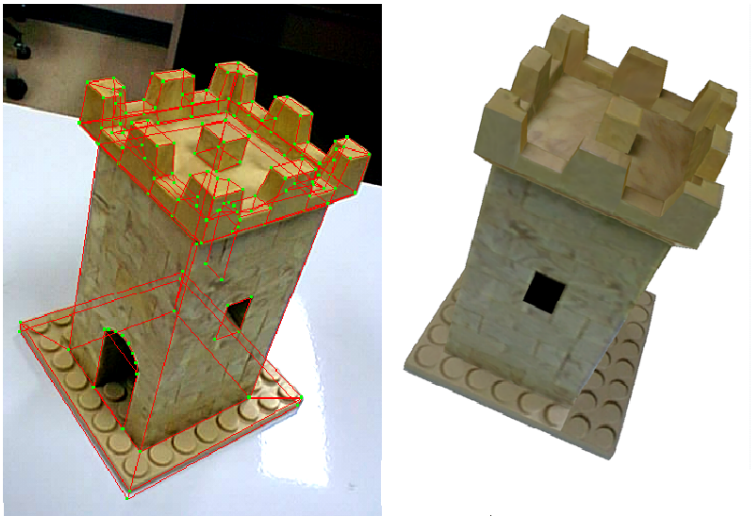


FIG. 3.11: Exemple de modèle reconstruit par *Rekon*. À gauche, le modèle est reprojeté sur l'image lors de la reconstruction. À droite, le modèle est placé sous un nouveau point de vue. Les textures ont été extraites à partir de plusieurs images. Elles sont indépendantes du point de vue. (Figure extraite du mémoire de Frasson [Fra99])

**Mise en place**

Comme nous l'avons fait pour *TotalCalib*, nous décrivons maintenant notre utilisation de *Rekon*.

La reconstruction d'une scène en utilisant *Rekon* est assez différente de *TotalCalib*. Elle nécessite une connaissance préliminaire de certaines mesures dans la scène. Comme pour *TotalCalib*, nous avons eu à reprendre des photographies pendant la phase de reconstruction, et à conserver la scène quelques jours en état. Les problèmes que nous avons rencontrés étaient divers. Il est assez difficile pour un utilisateur de visualiser les points communs entre les images lors de la prise de photographies. Il est arrivé d'avoir suffisamment de photographies pour visualiser les objets, mais pour lesquels les points communs n'étaient pas assez nombreux. La prise de nouvelles photographies peut cependant être évitée, si on connaît les mesures de certains points. La reconstruction ne prenait que quelques heures. Cependant, si des problèmes apparaissaient, nous pouvions atteindre quelques jours, le temps de reprendre les photographies, ou de nouvelles mesures.

Le système est assez robuste. Les modèles reconstruits sont assez justes, même si l'utilisateur place un point 2D à quelques pixels de sa position exacte. L'algorithme permet une correction des erreurs de l'utilisateur dans le système. Par contre, certaines erreurs de coordonnées 3D, de mauvaises correspondances, ou de mauvaises contraintes géométriques ont une influence assez grande dans le système. Comme l'utilisateur doit intervenir assez souvent, il lui est facile d'introduire des erreurs d'inattention qu'il est ensuite difficile de localiser.

Enfin, nous avons ajouté certaines fonctionnalités dans le système *Rekon*, pour nous permettre d'obtenir le modèle texturé tel que nous en avions besoin. Tout d'abord, l'extraction de textures dépendantes du point de vue a été implantée par Frasson dans le système. Ensuite, les matrices de *Rekon* ont été transformées en des matrices de paramètres de la caméra, que nous savions transformer pour obtenir un point de vue dans nos systèmes (voir section 3.4.1).

## 3.3 Choix de l'éclairage

Notre objectif est d'estimer les propriétés radiométriques des scènes réelles, afin de modifier leur éclairage. L'estimation des paramètres radiométriques des scènes réelles est un problème très difficile. Cette complexité est due à la variété des types d'éclairage, ainsi qu'à celle des propriétés des surfaces (BRDF complexes). De plus, l'éclairage dépend de la configuration de l'environnement, qui implique plus ou moins d'éclairage indirect. L'état de l'art dans le domaine de l'illumination inverse est très jeune. Il paraît plus judicieux de s'attaquer à des environnements moins complexes afin d'élaborer des méthodes de base et de comprendre les problèmes, avant d'essayer de travailler sur des environnements très complexes.

Nous avons donc fait certains choix sur l'arrangement de nos scènes. Nous avons privilégié d'abord les scènes d'intérieur fermées. Nous prenions soin de n'avoir qu'un éclairage connu, et non un éclairage provenant de l'extérieur comme par des fenêtres, ou par une

porte. Nous avons choisi de préférence des scènes éclairées directement par les lampes, plutôt que par un éclairage indirect. En effet, il est en général plus facile d'estimer des valeurs pour l'éclairage direct que pour l'éclairage indirect. De plus, nous avons choisi des lampes ayant une intensité homogène, pour simplifier notre estimation de l'intensité des lampes. Dans la première méthode de ré–éclairage, présentée au chapitre 4, nous avons utilisé les lampes néons déjà en place. Le support de ces lampes est rectangulaire et il est fixé au plafond. Pour notre deuxième méthode, expliquée au chapitre 5, nous avons utilisé une lampe de jardin, facile à modéliser, car son support est rectangulaire. Cette lampe est une lampe halogène. Pour que l'éclairage de cette lampe s'approche plus à une émission diffuse, nous avons fixé un papier de soie sur le support (voir figure 3.12).



(a) (b)

FIG. 3.12: (a) Lampe utilisée pour éclairer les scènes lors de nos tests. (b) La lampe est recouverte d'un papier de soie, permettant de se rapprocher du caractère diffus.

Enfin, nous avons essayé d'avoir des objets les plus diffus possibles sans pour autant nous limiter à de tels objets. Dans la plupart de nos scènes, les propriétés des surfaces sont très variées.

## 3.4 Intégration des modèles 3D dans le système de rendu

Nous avons pu construire un modèle 3D texturé d'une scène réelle grâce aux deux outils présentés ci–dessus. Ces deux outils nous ont permis d'obtenir un modèle au format Inventor®, avec des textures RGB, que nous pouvons lire et utiliser dans nos programmes

graphiques. Nous avons cependant réalisé quelques adaptations pour rendre compatibles les données des outils de reconstruction avec celles de nos programmes.

### 3.4.1 Les matrices de points de vue

Les premières données à adapter sont les matrices de projection fournies par les outils de reconstruction. Les deux matrices fournies sont la matrice de paramètres intrinsèques, et la matrice de paramètres extrinsèques [Fau93]. La matrice de paramètres intrinsèques fournit des indications sur la distance focale, et la projection du centre de la caméra sur l'image. Ces paramètres dépendent uniquement des réglages de la caméra. La forme de la matrice de paramètres intrinsèques est la suivante :

$$\begin{pmatrix} A_u & \gamma & u_0 \\ 0 & A_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.1}$$

où $A_u$, $A_v$, et $\gamma$ sont des paramètres indiquant la distorsion d'un pixel, et la distance focale, et $u_0$ et $v_0$ sont la projection du centre optique sur l'image. La matrice des paramètres extrinsèques fournit la position relative de la caméra, en termes de rotations (matrice $R$ $3 \times 3$) et de translation (vecteur $T$ $3 \times 1$). Cette matrice est évidemment différente pour chacune des images utilisées pour la reconstruction. La forme de la matrice est la suivante :

$$\begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix}. \tag{3.2}$$

Les deux matrices fondamentales qui nous intéressent doivent être celles associées à l'image utilisée pour extraire les textures.

Dans nos programmes, nous utilisons la librairie graphique OpenGL®pour réaliser le rendu. Dans OpenGL®, les paramètres utilisés pour définir le point de vue ne sont pas les matrices de vision. Il faut donc convertir les données des matrices typiquement utilisées en vision par ordinateur, décrivant les paramètres intrinsèques et extrinsèques, en paramètres acceptables par OpenGL®.

Les paramètres correspondant aux paramètres intrinsèques sont ceux de *gluPerspective* et *glViewport*.

*glViewport* a quatre paramètres. Les deux premiers $(x, y)$ définissent le coin en bas à gauche de la fenêtre de vue. Les deux autres $(width, height)$ définissent la taille de la fenêtre de vue. La fenêtre de vue peut se déduire des paramètres intrinsèques. Idéalement, on a :

$$x = \frac{-u_0 \times width}{2} \text{ et } y = \frac{-v_0 \times height}{2}. \tag{3.3}$$

Cette équation provient d'une transformation du repère où sont calculées les coordonnées $(u_0, v_0)$ du centre de la caméra, dans le repère de la fenêtre de vue. Le repère du centre de la caméra varie en abscisses et en ordonnées entre $-1$ et 1. Le repère associé à la fenêtre varie en abscisses entre 0 et *width*, et en ordonnées entre 0 et *height*. Une fois le changement de repère effectué, il faut appliquer le fait que $(u_0, v_0)$ correspond au centre de la fenêtre de visualisation, et $(x, y)$ au coin en bas à gauche. Les signes négatifs proviennent du fait que

les axes des deux repères ne sont pas orientés de la même façon. Pour avoir une bonne impression lors de la visualisation de la scène, il est préférable d'avoir le rapport $\frac{width}{height}$ égal à celui de la taille de l'image originale.

En réalité, nous n'avons jamais exactement utilisé $(x, y)$ calculés par l'équation (3.3). Cependant, ceci nous donnait un point de départ, et nous ajustions ensuite les paramètres $(x, y)$ directement et manuellement dans le système, jusqu'à que la scène 3D reprojetée soit alignée avec l'image 2D. Nous pensons que ces erreurs sont dues soit aux inexactitudes du calibrage, soit à l'idéalisation de la caméra par OpenGl®.

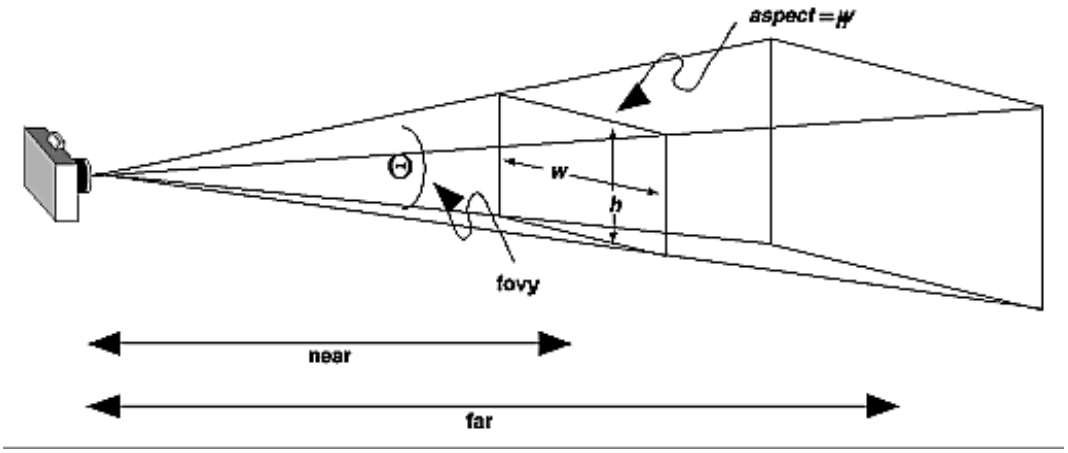



FIG. 3.13: Schéma illustrant les paramètres nécessaires pour simuler la perspective dans nos systèmes graphiques. (Figure extraite du manuel de la librairie graphique OpenGL®) [WNDS99]

Souvent en vision, on ne s'intéresse pas à la vraie profondeur 3D, alors qu'elle est nécessaire dans un système de synthèse d'image. Une façon de la prendre en compte est de donner des valeurs de profondeur de champ. *gluPerspective* permet de fixer les plans de profondeur devant et derrière la scène virtuelle (*near* et *far* sur la figure 3.13). Il faut aussi déterminer l'aspect de la fenêtre, ce qui est simplement donné par $\frac{width}{height}$. Enfin, il faut déterminer le champ de vue ($fovy$, pour *field of view in y* sur la figure 3.13). Ceci se fait en fonction de vecteurs placés sur le rectangle de la fenêtre dans la position la plus proche avant coupure (*near*). Ces vecteurs sont calculés en fonction de la distance focale de la caméra accessible dans la matrice de paramètres intrinsèques, dans la sous–matrice $(2 \times 2)$ de la matrice 3.1 :

$$\begin{matrix} A_u & \gamma \\ 0 & A_v \end{matrix} \tag{3.4}$$

Les coordonnées au milieu de chaque arrête de la fenêtre sont :

$$\begin{aligned} left &= \tfrac{-near}{A_u} \\ right &= \tfrac{-near}{A_u} \\ bottom &= \tfrac{(near \times -\gamma)}{A_v} \\ top &= \tfrac{(near \times \gamma)}{A_v}. \end{aligned} \tag{3.5}$$

Les deux vecteurs utilisés pour calculer l'angle du champ de vue sont :

$$\begin{array}{l} V_1(\frac{right+left}{2}, top, -near) \\ V_2(\frac{right+left}{2}, bot, -near). \end{array} \tag{3.6}$$

L'angle du champ de vue est déterminé par son cosinus donné par le produit scalaire des deux vecteurs normalisés $V_1$ et $V_2$.

Les paramètres extrinsèques permettent de positionner la caméra dans l'espace. À partir de la matrice de paramètres extrinsèques, nous avons extrait trois angles de rotations, et un vecteur de translation. Pour extraire les angles, nous utilisons les neuf premiers paramètres de la matrice de paramètres extrinsèques, c'est–à–dire la sous–matrice $R$ $(3 \times 3)$ en haut à gauche de la matrice générale $(4 \times 4)$ de l'équation (3.2). Les trois colonnes de la matrice $(3 \times 3)$ nous indiquent les axes du repère obtenus après rotation. Pour chacun de ces axes, nous avons extrait trois vecteurs normalisés. Ensuite, nous avons obtenu le cosinus des angles en calculant le produit scalaire entre ces vecteurs et les vecteurs originaux $((1,0,0),(0,1,0),(0,0,1))$. Les angles ainsi retrouvés nous indiquent la direction dans laquelle la caméra regarde la scène. À partir de la dernière colonne $T$ de la matrice $(4 \times 4)$, nous obtenons la translation déterminant la position de l'œil dans l'espace. Afin d'obtenir la translation dans le repère original, il faut appliquer la transformation de rotation au vecteur de translation.

En utilisant les paramètres intrinsèques et extrinsèques, nous avons déterminé les paramètres de vue, positionnant l'œil dans l'espace virtuel, et appliquant la perspective et la fenêtre de vue correspondant à notre caméra.

### 3.4.2 Modèle Inventor

Les modèles 3D texturés, obtenus par *TotalCalib* ou *Rekon*, sont décrit dans le format Inventor®. Ce format est très pratique car il est extensible et permet d'ajouter de l'information dans les fichiers.

Nous avons notamment ajouté des numéros à chaque polygone construit. Nous avons également ajouté une étiquette nous permettant de reconnaître si le polygone était visible sur l'image d'origine. Ceci nous permet de faire une distinction sur la validité des polygones dans notre système. Ceux reconstruits depuis l'image d'origine sont considérés plus importants que ceux non visibles sur cette image, comme par exemple pendant le raffinement de la radiosité hiérarchique.

## 3.5 Discussion

Dans ce chapitre, nous avons décrit l'aspect matériel de nos recherches. Cet aspect n'est pas à négliger car il représente une quantité de travail considérable. Tous les choix réalisés ont une influence sur les méthodes de recherche présentées dans les chapitres suivants. Alors que certains chercheurs sur le même thème ont privilégié la qualité des équipements (appareils photographiques hauts de gamme, lampes diffuses ou ponctuelles, scan-

ner 3D), nous avons préféré nous placer dans le cas d'un utilisateur ayant un matériel restreint par un certain budget.

Les outils de reconstruction choisis sont des outils de recherche. Il nous a fallu une période d'adaptation pour les manipuler correctement. Comme nous l'avons décrit dans ce chapitre, nous sommes très contents des deux outils de reconstruction utilisés. Ils nous ont permis d'obtenir des modèles 3D texturés de qualité, et facilement intégrables dans nos systèmes.

Dans la suite du document, nous décrivons les méthodes de recherche que nous avons développées pour permettre le ré–éclairage interactif des scènes réelles, capturées et reconstruites par les méthodes décrites dans ce chapitre.

# Chapitre 4

# Modification des propriétés d'éclairage d'une scène réelle à partir d'un seul éclairage connu

Comme nous l'avons souligné dans les chapitres précédents, l'objectif général de cette thèse est d'apporter des solutions au problème du ré–éclairage interactif des scènes réelles. Un exemple d'application de telles méthodes est la compositions d'éclairage pour des scènes d'intérieur. Nous souhaitons fournir des solutions pour permettre la modification interactive des propriétés d'éclairage d'une scène réelle.

Dans ce chapitre, nous décrivons une méthode permettant d'atteindre cet objectif, avec une méthode de capture simple et l'utilisation des cartes d'accélération graphique. Nous utilisons en tout une douzaine d'images d'entrée (quatre mosaïques) pour créer un modèle géométrique de la scène réelle. Des textures sont ensuite extraites depuis une seule mosaïque, avec l'éclairage d'origine. Nous utilisons ensuite ce modèle texturé pour simuler l'éclairage et le modifier, ce qui permet la manipulation virtuelle et interactive de l'intensité des lampes réelles présentes dans notre modèle, et l'insertion d'objets et de lampes virtuels. La modification de l'intensité des lampes réelles est un problème difficile, car les textures contiennent déjà l'éclairage réel initial, et en particulier des ombres.

Parallèlement à notre travail, d'autres solutions ont été publiées [YDMH99, YM98]. Nous les avons décrites précédemment dans la section 2.4. Toutes nécessitent une connaissance de la scène sous plusieurs éclairages, et la résolution se fait pixel par pixel. Nous proposons une approche moins complexe, en nous contentant d'une seule image en entrée du système d'éclairage. Pour permettre le ré–éclairage, nous modifions les textures du modèle pour qu'elles ne prennent plus en compte les occlusions dans l'éclairage qu'elles représentent. Pour cela, nous utilisons une méthode de radiosité hiérarchique*, et nous nous appuyons sur les travaux de Fournier *et al.* [FGR93] et de Drettakis *et al.* [DRB97]. En utilisant un système de radiosité, nous nous restreignons à traiter le cas des scènes diffuses (émissions et réflectances diffuses). Les textures originales comprenant les effets d'éclairage tels que les ombres, sont transformées en textures correspondant à un éclairage

*Pour le lecteur qui ne connaît pas la radiosité, nous suggérons la lecture préalable de la section 2.2 du chapitre 2

calculé sans prise en compte de la visibilité. Pour modifier les textures, nous utilisons le matériel d'accélération graphique en affichantles polygones texturés aux feuilles de la subdivision de la radiosité hiérarchique. La phase interactive permet de modifier l'éclairage de la scène, d'insérer et de déplacer des objets virtuels.

## 4.1 Présentation générale du système

Notre objectif est de permettre la modification interactive de l'intensité des lampes réelles (et notamment l'extinction totale de l'éclairage réel), l'insertion des lampes virtuelles et la modification de leurs intensités, ainsi que l'insertion et le déplacement interactif des objets virtuels. Toutes les mises à jour doivent permettre d'obtenir un éclairage commun cohérent et précis.

L'algorithme que nous présentons dans ce chapitre comporte trois parties : une partie pour reconstruire et à initialiser le système, une partie de pré–calcul, qui permet en particulier de modifier les textures originales, et une partie d'interaction avec le système. La figure 4.1 présente l'algorithme complet.

**Reconstruction 3D et Initialisation**
- Construction d'un modèle 3D simple de la scène réelle
- Mise en place du système de radiosité

**Pré–calcul pour le ré–éclairage**
- Raffinement spécial pour délimiter les ombres
- Création des textures indépendantes de la visibilité
- Ré–initialisation du système et reprojection des ombres
- *Pré–calcul supplémentaire pour l'insertion de lampes et d'objets virtuels*

**Modification interactive**
- Modification des lampes réelles et virtuelles
- Mise à jour de l'éclairage lors du mouvement d'un objet virtuel

FIG. 4.1: Algorithme complet.

Nous reconstruisons tout d'abord un modèle géométrique texturé représentant la scène réelle. Ce modèle est utilisé pour initialiser le système de radiosité. Cette initialisation est présentée dans la section 4.2. La deuxième partie de l'algorithme est une phase de pré–calcul qui permet de corriger les textures, et de créer des textures d'éclairage sans occlusion, puis qui ré–actualise le système à partir de ces nouvelles textures (voir section 4.3). Enfin, nous utilisons des algorithmes adaptés (voir section 4.4), afin de modifier interactivement la scène réelle. Les résultats sont présentés dans la section 4.5.

## 4.2 Initialisation du système

Dans un premier temps, nous retrouvons un modèle géométrique texturé de la scène réelle, à partir de photographies de celle–ci. Nous utilisons pour cela un des outils présentés

dans le chapitre précédent en section 3.2. Les textures utilisées sont dépendantes du point de vue. Ceci présente un grand avantage, qui est de préserver les détails de perspective tout en ayant un modèle simple. En revanche, le modèle ne peut être visualisé dans notre système que d'un seul point de vue. Le modèle est décomposé en deux ensembles. Le premier ensemble contient les objets visibles sur la mosaïque servant à la reconstruction. Cette mosaïque est appelée *photographie originale*. Pour une de nos scènes de test, c'est par exemple la mosaïque en haut à gauche sur la figure 4.2. Les objets non visibles sur cette photographie, sont placés dans un deuxième ensemble et sont traités différemment.

Le modèle visible sur la photographie originale est reconstruit en utilisant les outils semi–automatiques de reconstruction à partir d'images. Sa reconstruction par l'outil *TotalCalib* [BR97] est illustrée par la figure 4.2. Le reste de la scène est reconstruit de façon grossière à la main, et texturé par des textures représentatives, de basse résolution. Les sources de lumière sont modélisées à la main. Les objets du reste de la scène ainsi que les lampes auraient pu être modélisés en utilisant un outil de reconstruction. Cependant, pour nos besoins, il s'avère que le modèle non visible sur l'image n'a pas besoin d'être précis. Seule une géométrie grossière suffit pour simuler les échanges lumineux. De plus, ceci nous évite de multiplier les données photographiques. Un exemple de modèle reconstruit est présenté sur la figure 4.3. Dans ce modèle, on distingue le modèle texturé dans le fond, les murs et un ensemble de bureaux reconstruits grossièrement sur le devant, et quatre lampes à néons (A, B, C, D) représentées par des rectangles.

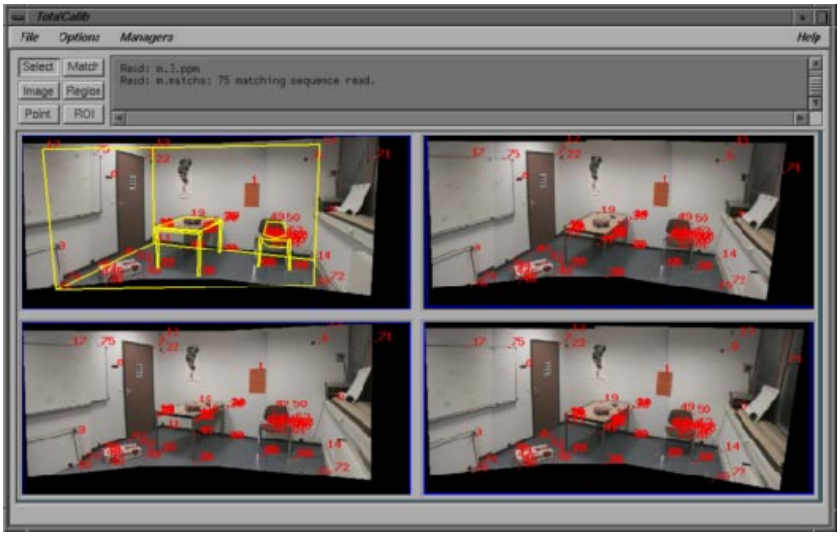



FIG. 4.2: Le modèle reconstruit par un outil de reconstruction à base d'images, ici *TotalCalib*.

### 4.2.1 Initialisation de la radiosité hiérarchique

Le modèle 3D ainsi reconstruit est utilisé dans notre système de radiosité hiérarchique. Avant de calculer une solution de radiosité pour le modèle de la scène réelle, nous devons estimer plusieurs paramètres : la réflectance des objets et l'intensité des sources. Ces

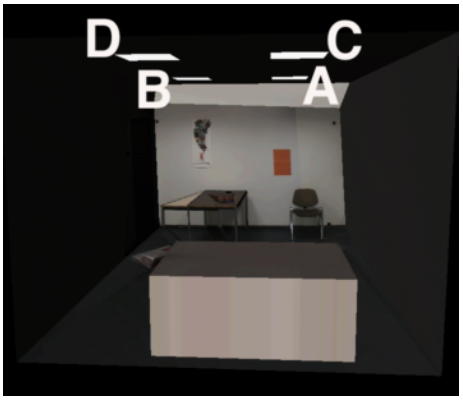



FIG. 4.3: Le modèle complet incluant quatre lampes réelles (A, B, C, D).

paramètres sont fixés en utilisant les équations de Fournier *et al.* [FGR93] et de Drettakis *et al.* [DRB97], qui estiment les paramètres directement à partir des textures (c'est–à–dire des images d'entrées). Pour chaque polygone, une valeur de réflectance diffuse est déterminée en utilisant la moyenne des intensités des pixels. L'équation est basée sur la définition de la réflectance, égale à l'énergie émise divisée par l'énergie reçue. Pour chaque polygone $i$, la réflectance $\rho_i$ est :

$$\rho_i = \frac{\hat{B}_i}{\hat{B}_A} \times \hat{\rho} \tag{4.1}$$

où $\hat{B}_i$ est la moyenne de l'intensité des pixels identifiés par la projection de cet objet sur l'image d'origine (photographie). à partir de la texture de $i$, $\hat{B}_A$ est l'intensité moyenne de toute la scène, calculée à partir de la photographie originale, et $\hat{\rho}$ est la réflectance moyenne de la scène. Dans un premier temps, $\hat{\rho}$ est fixée par l'utilisateur. D'après l'équation d'estimation (4.1), les valeurs de réflectance dépendent de la couleur des textures, et sont donc plus sombres dans les régions d'ombre. Dans un deuxième temps, la réflectance moyenne $\hat{\rho}$ est ré–estimée en faisant la moyenne des valeurs de réflectance calculées. Puis, les réflectances sont mises à jour en utilisant l'équation (4.1) pour cette nouvelle réflectance moyenne.

Ensuite, l'exitance $E_i$ des sources de lumière réelles est estimée en utilisant l'équation :

$$\sum_i E_i A_i = (1 - \hat{\rho})\hat{B}_A \sum_i A_i \tag{4.2}$$

où $A_i$ est l'aire du polygone $i$. Cette approximation est basée sur l'estimation d'un terme ambiant dans l'algorithme de radiosité progressive [CCWG88]. Pour simplifier la résolution, et étant donné que c'est le cas dans nos scènes, les sources de lumière sont considérées comme ayant la même exitance. Un système d'équations pourrait être résolu pour le cas de lampes ayant des exitances non–homogènes.

Une fois que la réflectance et l'intensité des sources ont été estimées, le système de radiosité hiérarchique est initialisé. La scène est subdivisée et les liens sont créés. Lors de la subdivision, les réflectances sont mises à jour pour chaque nouvel élément de subdivision

en utilisant l'équation (4.1), à partir des textures correspondantes. De cette initialisation est issue une radiosité $B_i$ pour chaque élément. Cette radiosité ainsi calculée sert à initialiser un facteur d'affichage $D_i$ ($D_i = B_i$). Si la radiosité de la scène est modifiée, ce facteur permet de moduler les textures et ainsi de prendre en compte les modifications d'éclairage. Ceci se fait en utilisant l'équation suivante pour chaque élément $i$ du maillage de radiosité[†] :

$$\frac{B_i}{D_i} \times T_i \tag{4.3}$$

où $T_i$ est la texture de l'élément $i$. Un exemple de maillage est montré plus loin sur la figure 4.6. Par exemple, si un objet virtuel est inséré dans la scène, la valeur de radiosité dans ses ombres sera inférieure au facteur d'affichage $T_i$. À cet endroit, la texture sera donc assombrie. Ceci est montré par la figure 4.4. En (a), la radiosité n'est pas modifiée. Le rapport d'affichage permet de visualiser le modèle texturé sans modification d'éclairage dans notre système. En (b), un objet virtuel a été inséré en utilisant la méthode de radiosité dynamique de Drettakis *et al.* [DRB97]. Les ombres de l'objet sont projetées sur la table en utilisant le rapport d'affichage décrit par l'équation (4.3) (voir le zoom sur l'image en (c)).



(a) (b) (c)

FIG. 4.4: (a) La scène vue dans notre système de radiosité hiérarchique. (b) Un objet virtuel est inséré dans la scène. Les ombres de l'objet virtuel sont projetées sur la table. (c) Zoom sur la projection des ombres virtuelles.

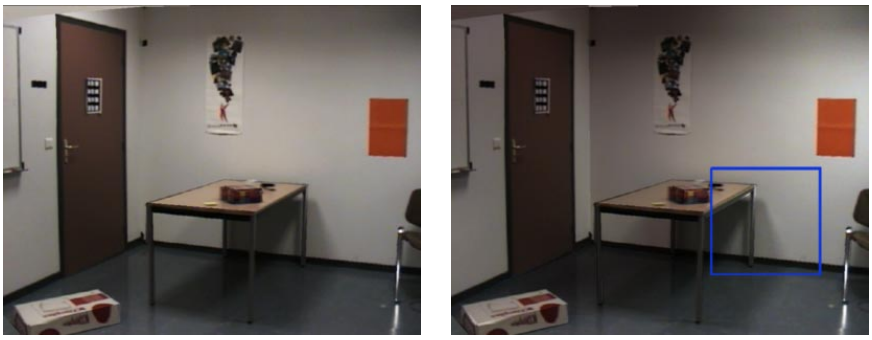
Il est important de noter que les valeurs de radiosité n'ont pas besoin d'être très précises. Comme l'affichage se fait grâce à une modulation par un rapport qui est de un s'il n'y a pas de modifications, ce qui importe est que la modification de la valeur de radiosité $B_i$ soit cohérente avec les valeurs précédentes. De plus, l'œil humain perçoit surtout les différences d'intensités, plutôt que les intensités elles–mêmes.

### 4.2.2 Limitations du système ainsi initialisé

En utilisant le système initialisé avec les équations de Fournier *et al.* [FGR93] et de Drettakis *et al.* [DRB97], nous ne pouvons pas modifier les lumières réelles de façon satisfaisante. L'effet est illustré par la figure 4.5.

Dans la figure (b), la scène n'est éclairée que par une seule lampe, celle de droite. Pourtant, les ombres dues à la lampe de gauche sont toujours perceptibles. Ce résultat est

[†]En pratique, les valeurs sont interpolées aux sommets de chaque élément pour donner un aspect lisse



(a) (b)

FIG. 4.5: (a) Éclairage original de la scène. Deux lampes projettent les ombres de la table sur le mur qui nous fait face. (b) En utilisant la méthode de Drettakis *et al.* [DRB97], nous éteignons la lampe de gauche. Les ombres dues à cette lampe sont toujours visibles sur le mur, alors qu'elles ne le devraient pas.

prévisible si on utilise la méthode de Drettakis *et al.* [DRB97]. En effet, l'affichage est réalisé directement en utilisant les textures de la photographie originale. Les textures contiennent les effets d'éclairage originaux, tels que les ombres des objets réels. Lors de l'affichage, elles sont modulées par un facteur qui est la radiosité potentiellement modifiée, divisée par la radiosité originale (c'est–à–dire le facteur d'affichage).

Dans la figure 4.5 (b), la région encadrée contient l'ombre de la table, due à la lampe de gauche, et une région qui ne contient pas d'ombre. En éteignant la lampe de gauche, la radiosité courante dans cette région varie peu, en fonction des valeurs de facteurs de forme. La texture dans les deux régions est modulée par la radiosité modifiée divisée par la radiosité originale. Comme la région d'ombre sur la texture est plus sombre que la région qui ne l'est pas, elle reste plus sombre après la modification de l'éclairage, car le rapport radiosité modifiée sur radiosité originale est similaire dans les deux régions. Pour avoir un résultat correct, il faudrait que la texture affichée ait une couleur similaire dans les deux régions. Les deux méthodes précédentes ne peuvent donc pas traiter de façon satisfaisante la modification de l'éclairage due aux lampes réelles.

Nous levons dans ce chapitre la limitation des deux méthodes précédentes, en transformant les textures pour qu'elles représentent un éclairage calculé sans prise en compte de la visibilité.

## 4.3 Pré–calcul pour permettre le ré–éclairage

Dans notre méthode, nous utilisons le rapport d'affichage décrit dans l'équation (4.3), permettant d'exploiter les capacités d'accélération graphique du matériel. Nous initialisons les paramètres du système à partir des textures, comme nous l'avons présenté dans la section précédente. Afin de permettre le ré–éclairage de la scène en modifiant l'intensité des lampes réelles, nous ajoutons un algorithme de pré–calcul (la méthode est très différente des méthodes précédentes [FGR93, DRB97]). Grâce à cet algorithme, nous construisons des textures correspondant à un éclairage indépendant de la visibilité, c'est-à-dire

que la lumière ne rencontre pas d'obstacle avant d'arriver sur l'objet.

La création de ces textures se fait en deux étapes. Dans une première étape, nous ajoutons la lumière bloquée à celle déjà présente dans les textures, en utilisant l'information contenue dans la structure de la radiosité hiérarchique. À la fin de cette première étape, la texture comporte des imperfections, principalement dues aux nombreuses approximations faites dans notre système. Pour résoudre ces problèmes, nous appliquons, dans une deuxième étape, une correction basée sur la couleur d'un élément de texture de référence, donnant une indication sur la couleur de texture à obtenir. À la fin de cette étape, les textures obtenues représentent effectivement un éclairage calculé sans tenir compte de la visibilité ou ≪sans occlusion ≫.

Pour chaque étape, il est important d'avoir un maillage adapté, délimitant correctement les régions d'ombre. Nous avons pour cela développé un raffinement basé sur la couleur des textures que nous décrivons dans la section suivante. Ensuite nous décrivons les deux étapes pour la modification des textures. Nous terminons par l'initialisation du système de radiosité basé sur les nouvelles textures.

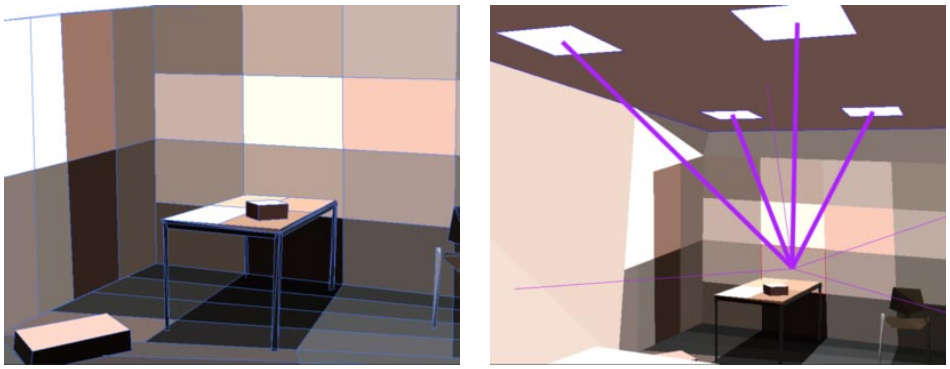
### 4.3.1 Raffinement basé sur les textures pour délimiter les ombres

Nous désirons appliquer des modifications pour obtenir des textures représentant l'éclairage sans ombres. Les régions où il y a le plus de modifications à faire sont les régions où l'on perçoit les ombres. Pour mieux délimiter les zones d'ombre, nous avons besoin d'un maillage adapté, localement plus fin dans les régions d'ombre, surtout dans les zones de pénombres. Si on utilise les méthodes standard de raffinement, telles que le raffinement à critère BF [HSA91] ou dirigé par le contrôle de l'erreur [GH96a], nous n'obtenons pas un maillage adapté à nos besoins. Le problème principal de ces approches est qu'elles ne garantissent pas une bonne délimitation des ombres, même en utilisant un facteur de visibilité [HSA91]. De plus, les limites de ces raffinements se perçoivent bien dans notre cas, où la géométrie reconstruite et la visibilité calculée par lancer de rayons sont approximatives. Le maillage de discontinuité [LTG93] se révèle tout aussi inadapté, et serait incorrect étant donné que la géométrie de nos scènes n'est pas précise. En conséquence, nous gardons la subdivision par maillage de type ≪ quadtree ≫ (voir section 2.2.1), mais nous utilisons un nouveau raffinement basé sur la texture.

L'idée générale est d'utiliser l'information de couleur contenue dans les textures, qui sont extraites directement des photographies, ainsi que l'information de visibilité accessible par le système de radiosité hiérarchique. Les ombres des objets réels sont inclues dans les textures, et sont distinguées par des régions de couleur plus sombre. En utilisant le type de la visibilité (VISIBLE, PARTIEL ou INVISIBLE) associé aux liens, et la différence de couleur entre deux éléments voisins, nous pouvons forcer la subdivision dans les régions d'ombre.

Ce raffinement a lieu après la phase d'initialisation décrite dans la section précédente (section 4.2.1), qui a permis d'initialiser les paramètres de la scène, tels que la réflectance, et l'intensité des sources de lumière. La subdivision initiale, obtenue avec un critère de raffinement BF, est grossière, comme le montre la figure 4.6 (a). À la fin de cette solution, tous les liens depuis les sources de lumière sont attachés aux feuilles de la hiérarchie

des éléments, afin de procurer une meilleure précision sur l'information de visibilité (voir figure 4.6 (b)).



(a) (b)

FIG. 4.6: (a) Subdivision grossière après la première solution de radiosité, initialisant les paramètres. (b) Tous les liens depuis les sources de lumières sont attachés aux feuilles de la hiérarchie pour assurer une meilleure précision dans les valeurs de visibilité.

L'algorithme de raffinement basé sur la texture compare la visibilité et la couleur de deux éléments feuilles voisins. La visibilité doit être cohérente avec les différences de couleur. Nous considérons deux cas, pour un élément et ces voisins :

1. Si les deux éléments ont des couleurs similaires, ils doivent avoir une visibilité similaire. Si ce n'est pas le cas, on subdivise.
2. Si les deux éléments ont des couleurs différentes, ils doivent avoir une visibilité différente. Si ce n'est pas le cas, on subdivise.

Nous considérons que la visibilité est similaire, si la visibilité est de même type (VISIBLE, PARTIEL, INVISIBLE) et si la différence entre les facteurs d'atténuation est inférieure à un certain seuil fixé par l'utilisateur. De la même façon, nous considérons que deux éléments ont une couleur similaire si leur différence de couleur est inférieure à un certain seuil fixé par l'utilisateur. Pour calculer la différence de couleur, nous convertissons les valeurs RGB en valeurs CIE LAB [Fai98]. En effet, les couleurs RGB ne sont pas facilement comparables. Le modèle CIE LAB est plus représentatif des variations de couleurs telles qu'elles sont perçues par le système visuel humain. La distance de couleur calculée dans ce modèle est également plus similaire à l'interprétation de distances de l'œil humain. La transformation des couleurs RGB en couleurs CIE LAB est exécutée en deux temps. Les valeurs sont d'abord converties dans l'espace de couleurs CIE XYZ, puis dans l'espace CIE LAB [Fai98]. La distance est ensuite calculée simplement par une équation de distance euclidienne :

$$Distance_{couleur}(i,n) = \sqrt{(L_i - L_n)^2 + (A_i - A_n)^2 + (B_i - B_n)^2}. \qquad (4.4)$$

L'algorithme complet de raffinement est montré sur la figure 4.7. Cet algorithme traverse la hiérarchie des éléments de façon récursive et compare chaque paire d'éléments voisins. Un élément est subdivisé s'il ne respecte pas les conditions du cas 1 et 2 ou si

dans le cas 2, le type de visibilité est PARTIEL. Cette dernière condition nous permet de mieux raffiner dans les régions de pénombre. L'algorithme examine ensuite les éléments fils créés. Si aucun changement de visibilité n'est détecté, la subdivision est annulée et l'élément devient à nouveau une feuille de la hiérarchie.

**Raffinement pour délimiter les ombres**
**Pour** chaque feuille *i*, comparaison avec les éléments feuilles voisins *n*
**si** *i* a une couleur similaire par rapport à *n*
**et** une visibilité depuis les sources de lumière différente
**alors** subdiviser *i*
**sinon si** *i* a une couleur différente de celle de *n*
**et** une visibilité depuis les sources de lumière similaire
**alors** subdiviser *i*
**sinon si** type = PARTIEL **alors** subdiviser *i*
**si** *i* a été subdivisé **alors**
**si** *i* n'a pas de différence de visibilité depuis
les sources de lumière avec ses éléments fils
**alors** annuler la subdivision de *i* (*i* est à nouveau une feuille)
**sinon pour** chaque fils de *i*
recommencer l'algorithme

FIG. 4.7: Algorithme du raffinement basé sur les textures pour délimiter les régions d'ombre.

Le cas numéro 1 apparaît dans les limites d'ombre. Cette subdivision aide à avoir un maillage plus fin dans les limites entre les régions d'ombre, et les régions éclairées directement par les sources de lumière. La subdivision d'un élément est arrêtée lorsque le niveau maximum de subdivision (basé sur la taille minimum de l'élément) est atteint, ou si les conditions des cas 1 et 2 sont respectées.

Le cas numéro 2 apparaît quand le lancer de rayons a échoué pour identifier correctement la visibilité. L'élément peut être non–subdivisé si la différence de couleur n'est pas due à une région d'ombre, mais à un changement de couleur dans la texture, comme c'est le cas pour le poster orange sur le mur sur la figure 4.8 (a).

À la fin du raffinement, la subdivision est fine dans les régions d'ombre (surtout de pénombre), et moins fine dans les régions visibles. La subdivision est montrée sur la figure 4.8 (b). Durant le raffinement, les réflectances ont été mises à jour.

Pour préparer la phase de modification de texture, les liens depuis toutes les sources (sources de lumière, et autres éléments qui sont des sources secondaires) sont attachés aux feuilles de la hiérarchie des éléments. Des itérations de *gather* et *push–pull* sont réalisées avec cette nouvelle subdivision, ce qui permet de mettre à jour les valeurs de radiosités $B_i^{orig}$ et les valeurs de facteurs d'affichage $D_i^{orig}$ ($D_i^{orig} = B_i^{orig}$), en fonction de la nouvelle subdivision.



(a) (b)

FIG. 4.8: (a) L'élément encadré a une couleur différente par rapport à ses voisins, à cause de la couleur de l'affiche. En revanche, il n'y pas de différence de visibilité entre eux. L'élément n'est pas subdivisé. (b) Maillage résultant du raffinement basé sur la texture. Les régions d'ombre sont délimitées correctement.

### 4.3.2 Création de la texture représentant l'éclairage indépendant de la visibilité

Comme nous l'avons vu précédemment, nous désirons créer des nouvelles textures à partir des textures d'origine. Ces nouvelles textures doivent représenter un éclairage calculé sans occlusion. La création de nouvelles textures se fait en deux étapes successives.

#### Première étape pour la création des nouvelles textures

La première étape consiste à amasser l'énergie bloquée par des objets réels, et de l'ajouter aux textures d'origine $T^{orig}$ qui correspondent à la photographie originale. Pour cela, nous utilisons le nouveau raffinement présenté ci–dessus, et nous définissons une nouvelle quantité d'énergie $\bar{E_i}$. Cette nouvelle quantité est l'irradiance d'un élément $i$ depuis les sources $s$, bloquée par des objets réels. Une source $s$ est soit une source de lumière, soit un autre élément de la hiérarchie (source d'éclairage secondaire). Cette irradiance additionnelle est la somme des radiosités $E_{is}$ de chaque source, multipliée par les facteurs de forme $F_{is}$ correspondants, ainsi que le complément du facteur d'atténuation $(1 - V_{is})$.

En considérant chaque source $s$, l'irradiance additionnelle $\bar{E_i}$ pour chaque élément $i$ est égale à :

$$\bar{E_i} = \sum_s F_{is}(1 - V_{is})E_{is}. \tag{4.5}$$

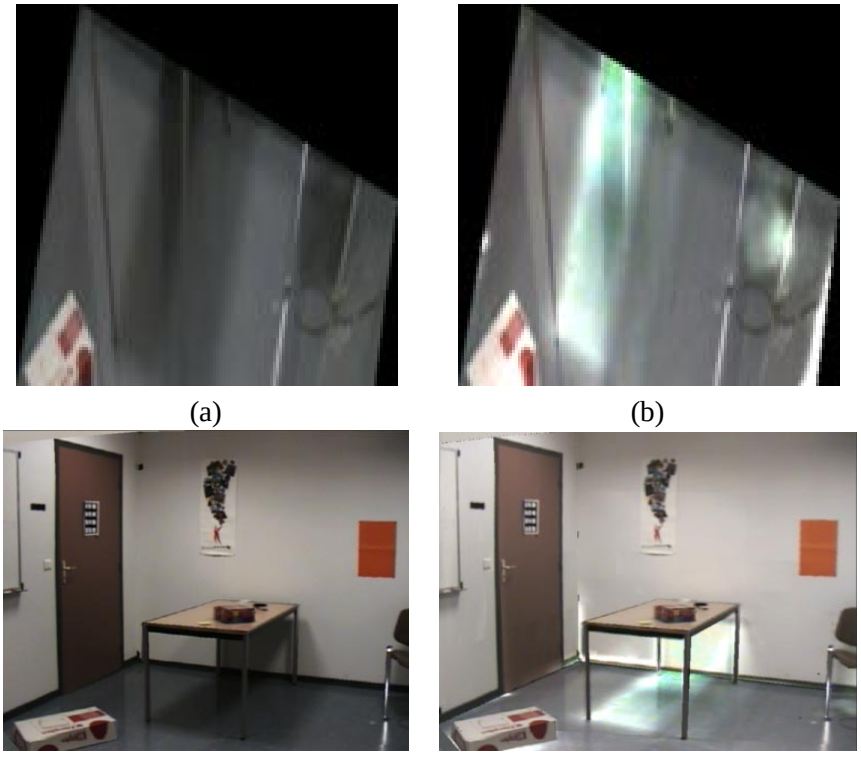
Le fait que tous les liens soient attachés aux feuilles assure un calcul plus juste de l'énergie additionnelle. Pour plus de précision, nous avons tenu compte de l'irradiance indirecte bloquée. Cependant, étant donné que la géométrie de la scène est simplifiée et non précise, nous ne considérons pas cette irradiance indirecte bloquée comme très fiable.

Pour créer les nouvelles textures en considérant la lumière bloquée, la texture originale est modulée par un facteur de correction calculé à chaque sommet des éléments feuille de la hiérarchie. La modulation de la texture aux sommets des éléments permet une modification lissée. Le facteur de correction utilise l'irradiance additionnelle décrite dans l'équation (4.5).

Pour inclure cette énergie bloquée dans les textures et créer les nouvelles textures $T^{inter}$, nous modulons la texture $T_i^{orig}$ associée à chaque élément $i$ par le facteur de correction suivant :

$$T_i^{inter} = \frac{\rho_i E_i + \rho_i \bar{E_i}}{D_i^{orig}} \times T^{orig}. \tag{4.6}$$

Dans cette équation, $\bar{E_i}$ est l'irradiance bloquée par des objets réels, et la radiosité $B_i^{orig} = D_i^{orig} = \rho_i E_i$. La texture intermédiaire $T_i^{orig}$ est générée en rendant chaque feuille de la hiérarchie avec une modulation appropriée comme la décrit l'équation (4.6). Si le facteur de modulation est supérieur à un, nous utilisons un affichage en passes multiples, décrit dans l'annexe A. Un exemple de modification de texture est montré sur la figure 4.9 (b), pour la texture du sol d'une de nos scènes. La reprojection des textures modifiées sur les polygones de la scène est montrée sur la figure 4.9 (d). Comme le montre cet exemple, la modification de texture est inexacte, ce qui produit des zones trop claires. Ceci est dû à l'estimation de l'énergie bloquée qui est inexacte, étant donné les approximations réalisées sur les paramètres de la radiosité, comme l'intensité des sources de lumière, les réflectances, les facteurs de forme et les facteurs d'atténuation, représentant les valeurs de visibilité.



(a) (b)

(c) (d)

FIG. 4.9: (a) Texture d'origine pour le sol d'une de nos scènes. (b) Texture $T^{inter}$ résultant de l'addition de l'énergie lumineuse bloquée par des objets réels. (c) Modèle 3D avec les textures d'origine reprojetées. (d) Modèle 3D avec les textures modifiées en ajoutant l'énergie bloquée.

La texture obtenue après cette étape de modification est utilisée pour mettre à jour des

valeurs de réflectance $\rho_i^{inter}$, et des valeurs de radiosité $B_i^{inter}$. Le facteur d'affichage est mis à jour en fonction de ce que représente la nouvelle texture, c'est–à–dire l'énergie totale, comprenant l'énergie bloquée par les objets réels ($D_i^{inter} = \rho_i^{inter}(E_i^{inter} + \bar{E}_i^{inter})$).

Comme la figure 4.9 (d) le montre, nous ne pouvons pas utiliser les textures ainsi modifiées. Pour compenser les erreurs de modification, nous appliquons une correction, basée sur une heuristique prenant en compte la couleur des textures.

#### Deuxième étape pour la création des nouvelles textures

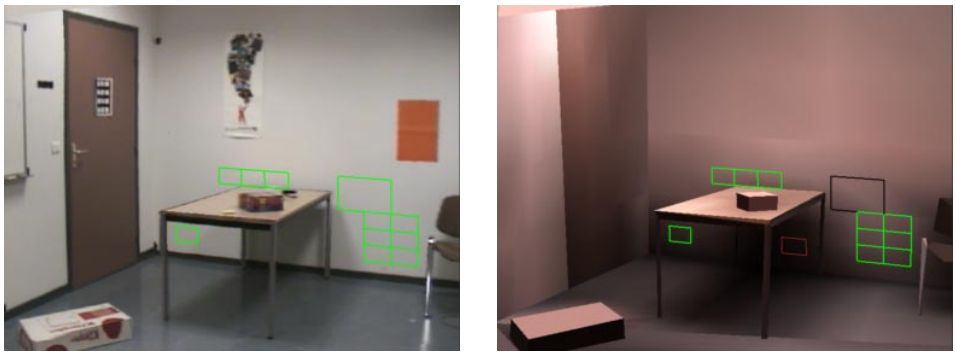
Nous voulons maintenant corriger les textures nouvellement générées en ajoutant la lumière bloquée. L'idée générale de la correction est que l'on peut estimer la couleur désirée d'un élément dans l'ombre à partir de la couleur d'un élément complètement visible depuis les sources de lumière, appartenant au même objet. Nous aimerions donc que la couleur de texture d'un élément dans l'ombre soit similaire à celle d'un élément directement éclairé. La similarité correspond à une égalité de couleur, au rapport de facteurs de forme près. En effet, la texture que nous voulons obtenir doit représenter un éclairage direct complet (non bloqué), et non une réflectance indépendante de l'éclairage. La texture voulue $T_i$ d'un élément *i* dans l'ombre doit être égale à la texture d'un élément voisin *r* directement éclairé, au rapport de facteurs de forme près :

$$T_i = T_r \times \frac{\sum_s F_{is}}{\sum_s F_{rs}}. \tag{4.7}$$

Comme nous l'avons fait précédemment, nous modifions la texture élément par élément, en la modulant. Pour corriger les textures, nous associons à chaque élément des régions d'ombre, un *élément de référence* qui servira à corriger la texture. Pour chaque élément du maillage dans l'ombre, nous cherchons donc un élément appartenant à une région éclairée. Cet élément doit être à la fois proche dans le sens de distance euclidienne, et d'une couleur similaire.

Pour cela, nous sélectionnons tout d'abord une liste d'éléments appartenant à une région éclairée, et voisins d'éléments appartenant aux régions d'ombre. Ces éléments forment une liste appelée *frontière*. Comme les liens depuis les sources de lumière sont attachés aux feuilles de la hiérarchie, la classification *VISIBLE*, *PARTIEL*, et *INVISIBLE* est relativement fiable. Grâce à cette classification, il est facile d'établir automatiquement la liste des éléments *frontière*, composée des éléments ayant tous les types des liens associés aux sources de lumière *VISIBLE*, et ayant un voisin au moins dont le type de lien associé à une source soit *PARTIEL* ou *INVISIBLE*. Cette frontière délimite la région où se trouvent les éléments à modifier lors de la correction des textures. Notre algorithme n'a cependant pas besoin que la région ainsi délimitée soit fermée. Nous appliquons une deuxième sélection sur ces éléments appartenant à la frontière, en ne conservant que ceux qui sont visibles depuis le point de vue. Cette restriction est justifiée par l'algorithme d'extraction de texture lors de la construction du modèle : il n'y a pas de calcul de visibilité réalisé, et la texture extraite pour un polygone contient les textures des éléments qui sont devant lui par rapport au point de vue. Nous avons affiché les éléments de la frontière ainsi sélectionnés

pour une subdivision grossière, pour le mur du fond, masqué par la table (voir figure 4.10 (a)).



(a) (b)

FIG. 4.10: (a) Frontière composée d'éléments appartenant à une région complètement visible depuis les sources de lumière, et voisins d'éléments de régions d'ombre. (b) L'élément de subdivision en noir de la frontière a été sélectionné comme élément de référence pour l'élément dans l'ombre de la table.

Pour chaque élément $i$ dans l'ombre, un élément de référence $r$ est choisi parmi les éléments de la frontière. L'élément choisi est l'élément le plus proche ayant une couleur similaire. Par exemple, pour l'élément dans l'ombre de la table sur la figure 4.10 (b), l'élément en noir a été choisi comme élément de référence parmi la liste d'éléments composant la frontière. Le choix de l'élément de référence se fait selon l'algorithme décrit à la figure 4.11.

**Pour** chaque feuille $i$
  $min_{distance} = \infty$
  $min_{couleur} = \varepsilon$
  $Reference = i$
  **Pour** chaque élément $n$ de la frontière
    **si** $Distance(i, n) < min_{distance}$
      **et** $Distance_{couleur}(n, i) < min_{couleur}$
    **alors**
      $Référence = i$
      $min_{distance} = Distance(i, n)$
      $min_{couleur} = Distance_{couleur}(i, n)$

FIG. 4.11: Algorithme de sélection d'un élément de référence.

Pour comparer les couleurs, nous utilisons les valeurs de réflectance, calculées directement depuis les textures et accessibles dans le système de radiosité. Comme nous l'avons fait précédemment lors du raffinement, nous comparons les couleurs dans l'espace CIE-LAB. Les couleurs entre deux éléments sont considérées comme similaires, si la distance euclidienne (équation (4.4)) est inférieure à un certain seuil $\varepsilon$ fixé par l'utilisateur. Si aucun élément de référence n'est trouvé par l'algorithme, l'élément de référence affecté est

un élément par défaut choisi manuellement auparavant par l'utilisateur.

Une fois qu'un élément de référence est choisi pour chacun des éléments dans les régions d'ombre, nous utilisons l'équation (4.7) pour déterminer le facteur de correction à appliquer sur la texture. Au lieu d'utiliser les textures comme le suggère l'équation (4.7), nous utilisons les réflectances qui représentent à un facteur près la couleur des textures dans le système de radiosité. Comme l'élément de référence est à une certaine distance de l'élément dans l'ombre, la réflectance de l'élément de référence est modulée par le rapport des facteurs de forme des sources de lumière pour ces deux éléments, $\sum_s F_{is}$ pour l'élément $i$ dans l'ombre, $\sum_s F_{rs}$ pour l'élément de référence $r$. En tenant compte de cette distance, la réflectance corrigée (que nous désirons obtenir) pour l'élément $i$ est :

$$\rho_i^{corr} = \rho_r \frac{\sum_s F_{is}}{\sum_s F_{rs}}. \tag{4.8}$$

En utilisant la réflectance corrigée $\rho_i^{corr}$, les textures finales correspondant à un éclairage indépendant de la visibilité sont créées. Les feuilles de la hiérarchie sont affichées avec un facteur de correction appliqué aux sommets pour moduler la texture, comme pour la génération de la texture intermédiaire. Pour les éléments dans l'ombre seulement, la texture $T_i^{inter}$ est modulée par l'équation suivante pour générer la texture finale $T_i^{final}$ :

$$T_i^{final} = \frac{\rho_i^{corr}}{\rho_i^{inter}} \times T_i^{inter} = \frac{\rho_r}{\rho_i^{inter}} \frac{\sum_s F_{is}}{\sum_s F_{rs}} \times T_i^{inter}. \tag{4.9}$$

Le facteur $\frac{1}{\rho_i^{inter}}$ est nécessaire car il permet d'atténuer l'effet contenu dans la texture $T_i^{inter}$ et dû à la modification précédente. Si $\rho_i^{corr}$ est supérieur à $\rho_i^{inter}$, un affichage à plusieurs passes est utilisé (voir annexe A), comme nous l'avons fait lors de la première modification des textures.

Le résultat de la texture corrigée pour le sol d'une de nos scènes est illustré sur la figure 4.12 (b). En (d), les textures ont été modifiées et reprojetées sur les polygones. Il est à noter que les ombres ont été relativement bien enlevées des textures, qui représentent, comme attendu, un éclairage indépendant de la visibilité.

À partir de ces nouvelles textures, correspondant à un éclairage indépendant de la visibilité, de nouvelles valeurs de réflectance $\rho_i^{final}$ sont calculées pour chacun des éléments dans les régions d'ombre de la scène. Avec ces nouvelles réflectances, un calcul de radiosité est effectué en conservant les liens et la subdivision pour mettre à jour la radiosité $B_i^{final}$, et le facteur d'affichage $D_i^{final} = \rho_i^{final}(E_i^{final} + \bar{E}_i^{final})$ (voir l'équation (4.5)). Le nouveau facteur d'affichage correspond à l'éclairage des nouvelles textures. Sur la figure 4.13, la radiosité $B_i^{final}$ est montrée en (a), et le facteur d'affichage $D_i^{final}$ est montré en (b).

### 4.3.3 Reprojection des ombres

Nous avons créé les textures indépendantes des valeurs de visibilité. Nous devons maintenant reprojeter les ombres afin d'avoir une représentation de l'éclairage initial de la scène.

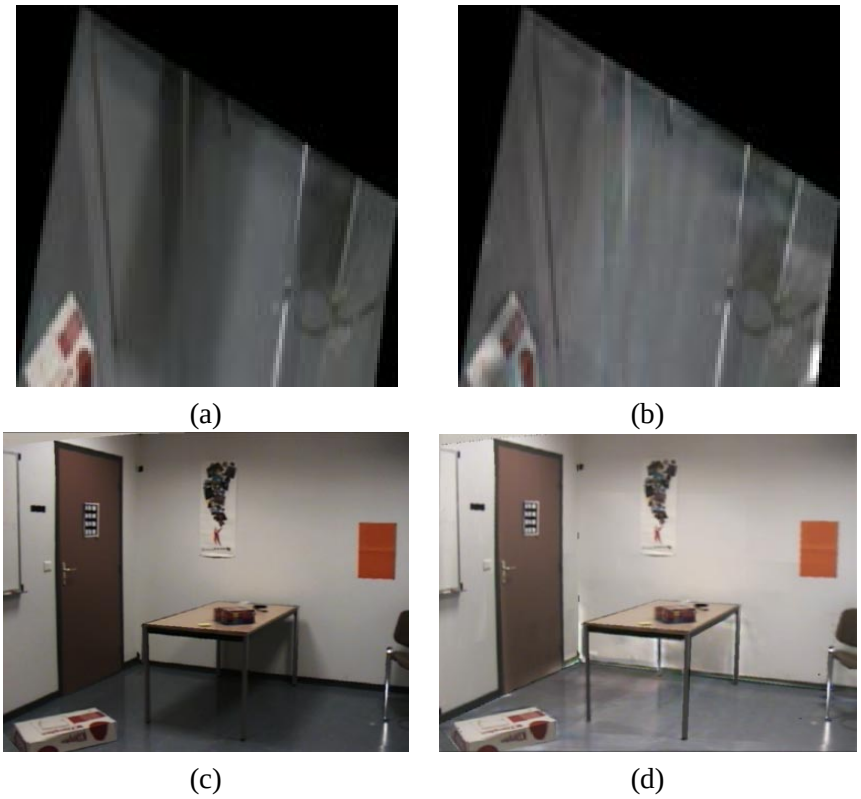



FIG. 4.12: (a) Texture d'origine pour le sol d'une de nos scènes. (b) Texture corrigée par les deux passes de l'algorithme de modification de textures. Comme attendu, cette texture correspond à un éclairage indépendant de la visibilité. (c) Textures d'origine projetées sur les polygones de la scène. (d) Textures finales corrigées $T^{final}$, projetées sur les polygones de la scène.

Nous utilisons pour cela l'équation d'affichage (4.3), avec les nouvelles valeurs de radiosité $B_i^{final}$, de facteur d'affichage $D_i^{final}$, et de texture $T_i^{final}$. Dans les régions d'ombre, la radiosité $B_i^{final}$ est inférieure au facteur d'affichage $D_i^{final}$. Le rapport $\frac{B_i^{final}}{D_i^{final}}$ est inférieur à 1, ce qui a pour effet d'assombrir la texture. La projection des ombres ainsi réalisée produit l'image 4.14 (b). Cet éclairage simulé est très similaire à l'éclairage contenu dans les textures d'origine en (a).

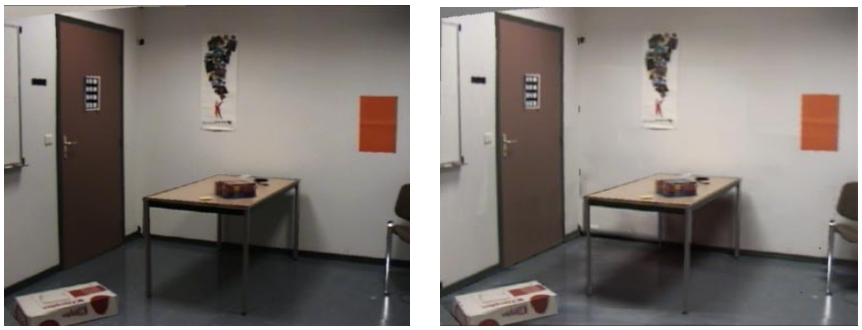
Il n'est souvent pas nécessaire de conserver la subdivision utilisée pour la création des nouvelles textures. Ce maillage est souvent trop fin dans les régions d'ombre, et pas suffisamment dans les régions directement éclairées. De plus nous avons attaché tous les liens aux feuilles de la hiérarchie ce qui peut être un handicap lors de la modification interactive de l'éclairage. Dans certains cas cependant, le maillage peut être conservé.

Dans la majorité des cas, nous finissons l'étape de pré–calcul par la génération d'une nouvelle solution de radiosité complète, utilisant un critère de raffinement BFV [HSA91]. Comme attendu, le maillage est moins fin dans les régions d'ombre, plus fin dans les régions éclairées, et il y a moins de liens actifs. La radiosité et le facteur d'affichage sont calculés en deux passes indépendantes.



(a) (b)

FIG. 4.13: (a) Radiosité $B^{final}$ correspondant à la nouvelle texture corrigée $T^{final}$. (b) Facteur d'affichage $D^{final}$ correspondant à la nouvelle texture corrigée $T^{final}$.

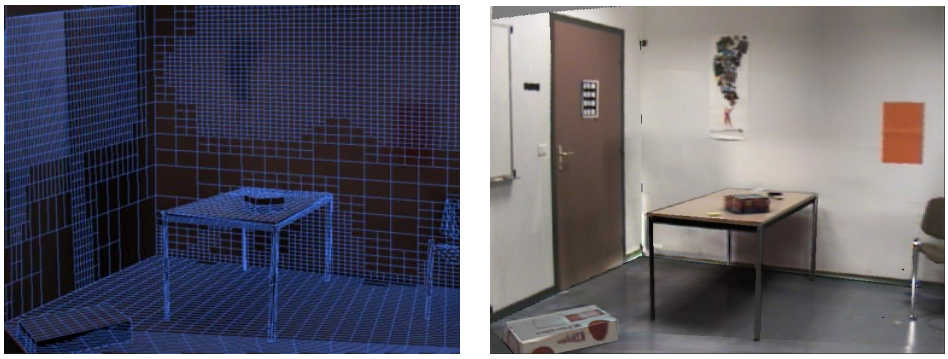


(a) (b)

FIG. 4.14: (a) Scène originale vue dans notre système avec les textures d'origine. (b) Éclairage reprojeté sur les nouvelles textures $T^{final}$, correspondant à un éclairage indépendant de la visibilité. Les ombres sont projetées en utilisant le rapport d'affichage (équation (4.3)) appliqué sur les nouvelles valeurs de radiosité et de facteur d'affichage.

Dans un premier temps, tout le maillage et tous les liens sont enlevés. Les valeurs de réflectance, de radiosité et de facteur d'affichage sont initialisées à zéro. Une première solution de radiosité est ensuite calculée en utilisant le critère BFV. Cette solution produit un nouveau maillage, illustré sur la figure 4.15 x(a). Les valeurs de réflectance pour chacun des éléments de radiosité sont calculées durant la subdivision, et correspondent à la nouvelle texture $T^{final}$. Après avoir attaché tous les liens aux feuilles de la hiérarchie, nous calculons un nouveau facteur d'affichage, correspondant comme précédemment à une valeur de radiosité totale, c'est–à–dire sans prise en compte de la visibilité.

Ensuite, nous conservons le maillage, les valeurs de réflectance associées, et le facteur d'affichage nouvellement calculé pour chacun des éléments. Les liens sont détruits, et les valeurs de radiosité annulées. Une nouvelle solution de radiosité est calculée, en utilisant les mêmes critères de raffinement que ceux de la précédente. De nouveaux liens hiérarchiques sont également créés. La reprojection des ombres, en utilisant la radiosité et le facteur d'affichage ainsi calculés, est montrée sur la figure 4.15 (b).



(a) (b)

FIG. 4.15: (a) Après la modification des textures, une nouvelle solution de radiosité est calculée, en utilisant un critère de raffinement BFV. Le maillage résultant est moins fin dans les régions d'ombre que celui utilisé pour la modification des textures. (b) Reprojection des ombres sur les nouvelles textures, en utilisant la nouvelle solution de radiosité basée sur le critère de raffinement BFV.

## 4.4 Éclairage commun et ré–éclairage interactif

Après avoir modifié les textures originales pour créer de nouvelles textures sans ombres, et après avoir ré–initialisé la solution de radiosité à partir de ces nouvelles textures, nous pouvons modifier interactivement l'intensité des sources de lumière réelles, et ajouter des objets et des lampes virtuelles. Nous expliquons dans cette section quels algorithmes nous utilisons pour réaliser ces modifications interactivement. Les résultats sont présentés ensuite, dans la section 4.5.

### 4.4.1 Insertion d'objets et de lampes virtuels

Pour insérer des objets virtuels et les déplacer dans la scène, nous utilisons l'algorithme de radiosité hiérarchique dynamique de Drettakis et Sillion [DS97], déjà utilisé dans le cadre de la réalité augmentée par Drettakis *et al.* [DRB97]. Nous avons adapté cet algorithme pour l'insertion des lampes virtuelles. L'influence des lampes virtuelles est très importante. C'est pourquoi nous avons forcé le raffinement en créant directement des liens entre la source de lumière et les polygones visibles depuis le point de vue de la scène, au lieu d'avoir ces liens au niveau des clusters. Ensuite, nous utilisons l'algorithme de radiosité hiérarchique dynamique, en autorisant la création d'autant de liens que le demande le raffinement. L'insertion d'une lampe virtuelle s'accompagne d'un éclairement général de la scène. Notre rendu est réalisé par le matériel d'accélération graphique, en affichant les polygones subdivisés et texturés de la scène, plutôt que pixel par pixel [FGR93]. Pour éclaircir les textures par le facteur d'affichage, nous utilisons un rendu en multi–passes, comme nous l'expliquons en annexe A.

### 4.4.2 Ré–éclairage interactif

Pour réaliser un ré–éclairage interactif de la scène, nous conservons le maillage et les liens établis dans la solution de radiosité calculée précédemment. Lors de la modification de l'intensité d'une source de lumière, la radiosité est mise récursivement à zéro, puis une nouvelle solution de radiosité est calculée itérativement en conservant la subdivision et les liens. La radiosité est accumulée aux feuilles en utilisant les liens (*gather*), et la solution est rendue cohérente par un *push–pull*. La solution itérative est arrêtée lorsque l'éclairage global est stable.

Comme nous ne modifions que l'intensité des sources de lumière, la subdivision et les liens sont toujours cohérents, même après la modification. Dans certains cas, conserver la même hiérarchie peut amener à une subdivision trop fine, lorsque les lampes sont éteintes. Comme l'utilisateur peut vouloir rallumer ces lampes, nous préférons conserver la subdivision.

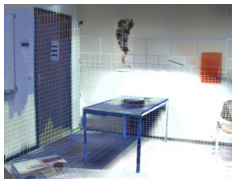
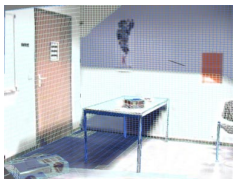
| | | | |
|---|---|---|---|
| Raffinement |  |  |  |
| Diminution de l'intensité de la lampe |  |  |  |
| Temps de modification | 0.2 sec. | 0.3 sec. | 0.7 sec. |
| Temps d'affichage | 0.2 sec. | 0.2 sec. | 0.6 sec. |
| Nombre de feuilles/liens | 3486/11246 | 5781/16807 | 8247/50787 |

FIG. 4.16: Modification interactive de l'intensité d'une lampe virtuelle. Le temps de mise à jour dépend du niveau de subdivision et du nombre de liens actifs établis.

Cet algorithme est interactif car nous évitons de calculer un nouveau raffinement, et en particulier de nouvelles valeurs de visibilité et de facteurs de forme, ce qui serait coûteux (en temps de calcul). Le temps de mise à jour dépend du niveau de subdivision et du nombre de liens. Il ne varie pas beaucoup si on modifie plusieurs intensités de lampe en même temps. Des exemples de temps de calcul sont montrés sur la figure 4.16. Comme prévu, ces temps augmentent en fonction du nombre de feuilles de la hiérarchie des éléments, ainsi que du nombre de liens. Dans ce tableau, nous mettons en évidence le coût (en temps passé) de l'affichage réalisé en plusieurs passes pour permettre l'éclaircissement des textures. Ce coût dépend du nombre d'éléments à afficher.

Cependant, la qualité de la solution varie peu lorsqu'on augmente la subdivision. L'ef-

fet général est le même pour les trois exemples. De plus, les temps restent interactifs, même dans le cas d'une forte subdivision.

## 4.5 Résultats

Nous avons testé nos algorithmes sur plusieurs scènes. Pour chaque scène, nous présentons des résultats interactifs de ré–éclairage et d'insertion de lampes et d'objets virtuels. Les temps sont calculés sur une station de travail *SGI Onyx2 Infinite Reality*, avec un processeur R10000, à 195 Mhz[‡].

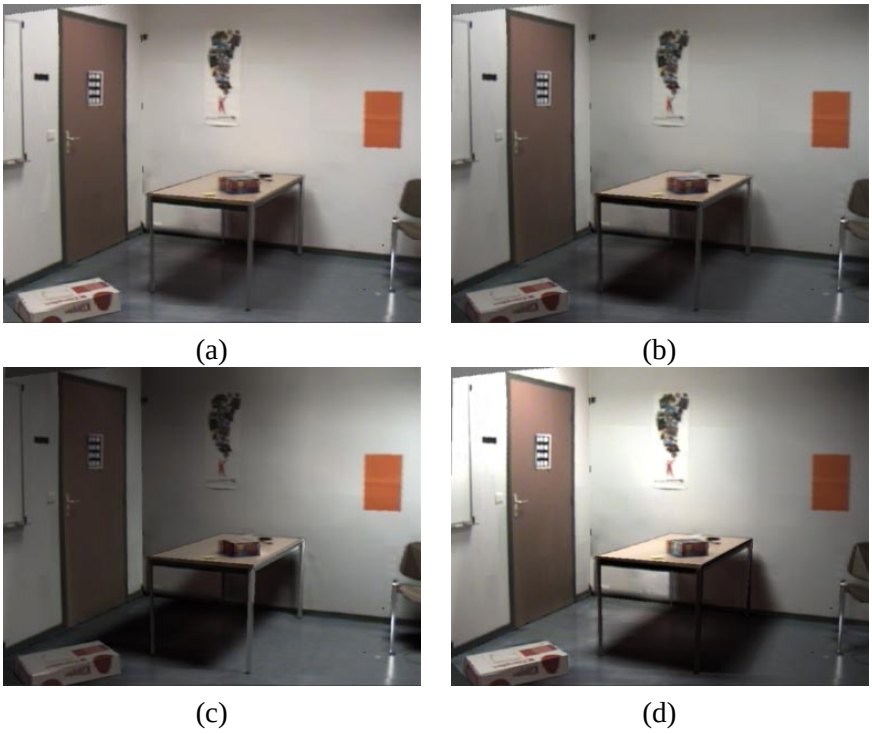


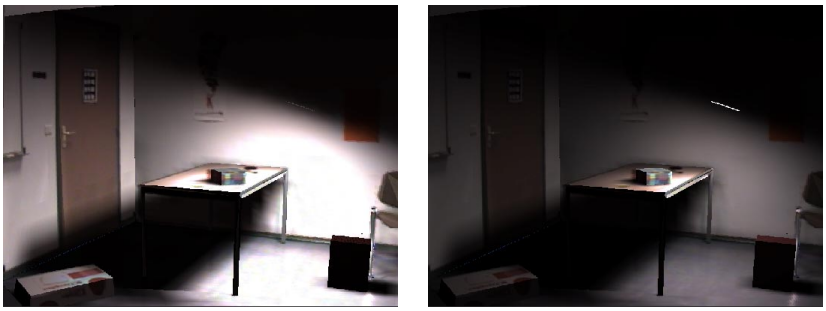

FIG. 4.17: (a) Scène avec son éclairage original reprojeté sur les textures. (b) Les deux lampes du fond ont été éteintes. (c) La lampe de gauche a été éteinte, et l'intensité de la lampe de droite doublée. (d) La lampe de droite a été éteinte, et l'intensité de la lampe de gauche doublée.

La première scène testée est présentée sur la figure 4.17 (a), sous son éclairage d'origine. Nous éteignons d'abord les deux lampes du fond (C et D) (voir figure 4.3). Comme le montre le résultat en (b), la scène est globalement plus sombre, mais il n'y a pas de modification perceptible dans les ombres. Nous éteignons ensuite la lampe A à gauche, et nous doublons l'intensité de la lampe B à droite. Le résultat est montré en (c). L'ombre de la table due à la lampe de gauche a disparu, et la partie de la scène éclairée par cette lampe est plus sombre. Le résultat est meilleur que celui obtenu sans modification de texture en

[‡]Les résultats sont également montrés en vidéo sur internet :
`http ://www-imagis.imag.fr/Membres/Celine.Loscos/CAR/car.html`.

utilisant les algorithmes de Fournier *et al.* [FGR93] ou de Drettakis *et al.* [DRB97] (voir figure 4.5). Nous ré–éclairons ensuite la lampe de gauche, en doublant son intensité initiale et nous éteignons la lampe de droite (voir figure 4.17 (d)). Comme attendu, l'ombre de la table due à la lampe de droite a disparu et la partie de la scène éclairée par cette lampe s'est assombrie. Avec la subdivision utilisée pour cet exemple, le temps total de mise à jour pour chaque modification est de 0.8 secondes, temps d'affichage compris.

Une source virtuelle peut être insérée dans cette scène en utilisant l'algorithme présenté dans la section 4.4.1. L'insertion de la lampe virtuelle prend 7.8 secondes. Dans la scène montrée sur la figure 4.18 (a), toutes les lampes réelles sont éteintes laissant la scène éclairée uniquement par la lampe virtuelle. Les ombres dues aux lampes réelles ont disparu. L'éclairage est cohérent ; par exemple, les ombres de la table dues à la lampe virtuelle sont projetées sur le sol et sur les murs. Nous pouvons également modifier l'intensité de la lampe virtuelle, comme nous le montrons sur la figure 4.18 (b), en utilisant le même algorithme que pour les lampes réelles (voir section 4.4.2).



(a) (b)

FIG. 4.18: (a) Insertion d'une lampe virtuelle. Toutes les lampes réelles ont été éteintes. La scène n'est éclairée que par la source de lumière virtuelle. (b) L'intensité de la lampe virtuelle est diminuée.

Avec ce nouvel éclairage, un objet virtuel est inséré en 1.42 secondes dans la scène et peut être déplacé interactivement. Dans la figure 4.19 (a), on voit la subdivision relative à l'insertion de la lampe virtuelle et de l'objet virtuel. En (b), (c), et (d), l'objet se déplace à raison de 3 images par secondes (temps d'affichage compris). Les ombres de l'objet dynamique, dues à la fois à la lampe virtuelle et aux lampes réelles, sont mises à jour en utilisant l'algorithme de Drettakis et Sillion [DS97]. On peut également remarquer dans ces images que l'effet sp'eculaire du pied de la chaise sur le sol est conservé car il est contenu dans les textures d'origine. Même si l'effet spéculaire n'est parfois plus cohérent avec l'éclairage, il permet de maintenir un effet visuel réaliste dont l'œil humain ne perçoit pas l'inexactitude. De plus un tel effet serait difficle à simuler avec des algorithmes de radiosité.

Nous avons également testé notre méthode sur une autre scène, reconstruite avec l'outil *Rekon* [POF98]. Cette scène est montrée sur la figure 4.20 (a) où les ombres réelles ont été reprojetées sur les textures modifiées, montrées en (b). Comme pour la scène précédente, les ombres ont été correctement enlevées des textures qui représentent un éclairage indé-



(a) (b)

(c) (d)

FIG. 4.19: (a) Insertion d'un objet virtuel en 1.42 secondes, et la subdivision conséquente. (b), (c), (d) L'objet virtuel (boîte) est déplacé interactivement en 3 images par seconde.

pendant de la visibilité.



(a) (b)

FIG. 4.20: (a) Deuxième scène testée. Les ombres ont été reprojetées en utilisant notre système de radiosité. (b) Textures modifiées, représentant un éclairage indépendant de la visibilité.

Nous avons modifié l'intensité des lampes réelles de cette scène, et comparé le résultat avec des photographies prises dans les mêmes conditions d'éclairage. Dans la figure 4.21, la première colonne de gauche correspondant aux images (a), (b), (c), et (d), sont des photographies. La colonne du milieu (images (e), (f), (g) et (h)) correspond aux résultats simulant les mêmes conditions d'éclairage par notre système. L'effet général est similaire, mais les couleurs sont légèrement différentes. Dans la colonne de droite (images (i), (j) et (k)), nous avons corrigé les images en appliquant une correction gamma [War94a]. Nous aurions également pu appliquer un « Tone Mapping » [TR93]. Ces transformations perme-

ttent de transformer l'image pour faire ressortir les contrastes comme le ferait l'œil. Les résultats sont très similaires aux photographies originales. Nous avons d'abord éteint les lampes du fond (figures (a), (e) et (i)), puis nous avons éteint la lampe de gauche, laissant la scène éclairée uniquement par la lampe de droite (figures (b), (f), et (j)). Ensuite, nous avons éteint la lampe de droite et rallumé celle de gauche (figures (c), (g), (k)).

Sur la dernière ligne, nous avons comparé l'insertion d'une lampe virtuelle dans la scène avec toutes les autres lampes éteintes (en (h)). La lampe virtuelle a été placée au même endroit qu'une lampe utilisée pour les photographies (en (d)).

Nous pensons que la différence entre l'image simulée et l'image photographiée est due à une erreur dans l'estimation de l'intensité des sources. Deux raisons justifient cette observation. La première est qu'un simple gamma suffit pour rendre l'image photographiée et l'image simulée similaires (voir figure 4.21). La deuxième est que nous obtenons un très bon résultat pour la lampe virtuelle, pour laquelle l'intensité a été fixée à la main.

## 4.6 Conclusion

Dans ce chapitre, nous avons présenté une solution pour résoudre le problème de ré–éclairage de scènes réelles. Dans notre approche, nous utilisons un modèle texturé pour représenter la scène réelle. Les textures sont extraites depuis une photographie de la scène prise sous des conditions d'éclairage fixes. La résolution du problème de ré–éclairage est difficile car l'éclairage réel, y compris les ombres, est déjà inclus dans les textures. Ces effets d'éclairage doivent être correctement identifiés avant d'être modifiés.

Nous avons présenté une solution permettant une modification interactive de l'éclairage, pour des sources de lumières réelles et virtuelles. L'algorithme est composé de trois parties. Dans la première, un modèle 3D texturé de la scène réelle est reconstruit, et le système de radiosité initialisé avec les algorithmes développés par Fournier *et al.* [FGR93] et Drettakis *et al.* [DRB97]. Dans une deuxième phase de pré–calcul, nous préparons la scène pour que l'intensité des lampes réelles puisse être modifiée. Dans un premier temps, un raffinement basé sur la texture délimite finement les zones d'ombre. Les textures sont ensuite modifiées en incluant la lumière bloquée par des objets réels, puis en se basant sur une couleur de texture de référence. Les textures ainsi obtenues représentent une solution de radiosité calculée sans occlusion. À la fin de cette deuxième partie de l'algorithme, un nouveau raffinement utilisant un critère standard peut–être effectué, et les ombres sont reprojetées sur les textures, par un affichage utilisant le matériel d'accélération graphique des stations de travail. Les images simulées à la fin de cette étape sont très similaires aux photographies originales.

Ensuite, des algorithmes de mise à jour interactive sont appliqués pour modifier l'éclairage de la scène en changeant l'intensité des lampes réelles, et en insérant des objets et des lampes virtuelles. Les résultats sont convaincants, et la comparaison avec des photographies valide nos résultats.

Cependant, notre méthode présente certaines limitations que nous aimerions améliorer.

Dans un premier temps, l'affichage en multi–passes limite les temps d'interactivité. Une optimisation de l'affichage général de notre système devrait améliorer ce problème.

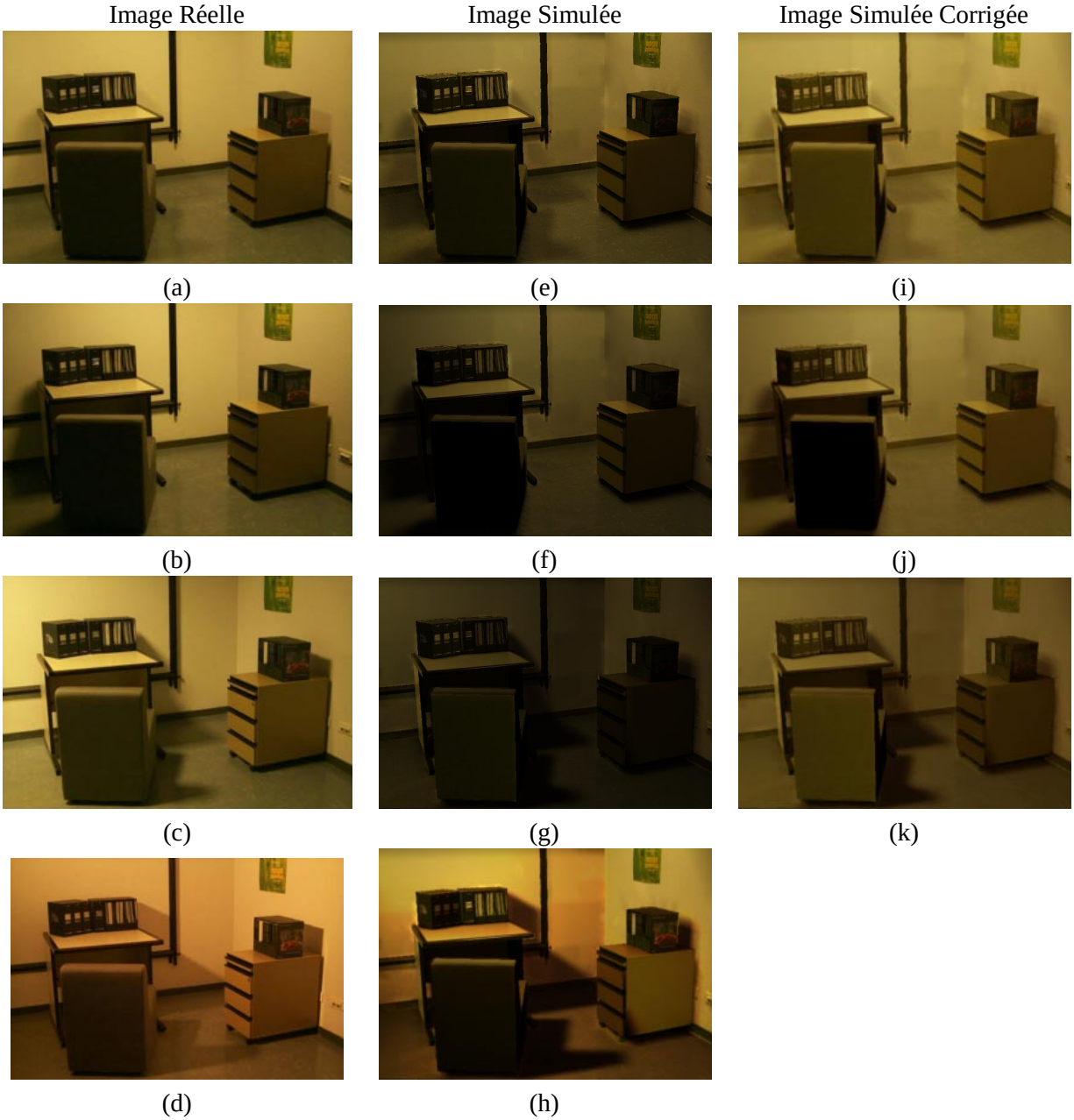



FIG. 4.21: (a), (b), (c), (d) sont des photographies prises sous différentes conditions d'éclairage. (e), (f), (g) et (h) sont des images simulées selon les mêmes conditions d'éclairage. (i), (j), (k) et (l) sont les même images que dans la colonne du centre, mais sur lesquelles nous avons appliqué une correction gamma.

De plus, les algorithmes utilisés pour mettre à jour la radiosité lors de la modification de l'intensité des lampes, pourraient être optimisés en utilisant des techniques dans l'esprit de celles de Drettakis et Sillion [DS97].

Une limitation importante de notre système est l'obligation de visualiser la scène depuis un point de vue fixe. Un utilisateur apprécierait de pouvoir se déplacer dans la scène pour en évaluer l'éclairage. Une solution pourrait être développée en utilisant les techniques de texture indépendantes du point de vue [POF98], ou de mise à jour interactive des textures [DBY98].

Enfin, la qualité des textures obtenues par notre méthode est limitée par la connaissance que nous avons de la scène. Travailler depuis une seule photographie et permettre quand même le ré–éclairage s'est avéré très difficile. Les difficultés rencontrées se situent surtout pour les régions d'ombre. Dans ces régions, la qualité de la simulation de l'éclairage est très importante pour retrouver une couleur de texture correspondant à un éclairage indépendant de la visibilité. De plus, la qualité des couleurs dans ces régions est limitée. La phase de correction basée sur la couleur d'un élément totalement éclairé nous a paru indispensable. En travaillant pixel par pixel, plutôt que sur les éléments de radiosité, les manipulations de textures pourraient être plus fines, et les valeurs de réflectance mieux estimées. Ceci permettrait notamment d'enlever des objets réels.

Dans le chapitre suivant, nous tenons compte des observations faites sur cette solution, pour en proposer une nouvelle, pour laquelle les données dans les régions d'ombre seront plus précises.

# Chapitre 5

# Modification des propriétés d'une scène réelle connue sous différents éclairages

Dans le chapitre précédent, nous avons présenté une méthode permettant le ré–éclairage interactif de scènes réelles. Cette méthode est basée uniquement sur l'utilisation de la radiosité hiérarchique, dont la structure de subdivision de polygone nous permet d'utiliser le matériel d'accélération graphique pour l'affichage. Mêmes si les résultats sont satisfaisants, il arrive parfois que les textures ne soient pas correctement corrigées, et que certains artefacts soient perceptibles, notamment dans les régions d'ombre originales. De plus, les paramètres radiométriques ne sont pas toujours estimés correctement, à cause d'une part des manipulations de textures au niveau des éléments de radiosité plutôt que sur chaque pixel, et d'autre part de l'utilisation d'une seule photographie pour l'estimation.

Dans l'approche présentée dans ce chapitre, nous essayons de retrouver des paramètres radiométriques plus fiables, ce qui nous permet d'étendre les possibilités d'interaction à l'enlèvement virtuel d'objets réels par exemple. Une idée simple pour résoudre ce problème de fiabilité est de prendre plusieurs photographies (au lieu d'une seule pour la solution précédente), afin d'avoir chaque zone de la scène sous éclairage direct sur au moins une photographie. Dans ce chapitre, nous décrivons une méthode permettant de retrouver les propriétés radiométriques d'une scène réelle en utilisant plusieurs photographies prises sous le même point de vue, mais sous des éclairages différents. Comme pour la méthode précédente, notre objectif est d'obtenir une méthode simple et rapide, utilisable par un utilisateur ayant un matériel non professionnel. Ce que nous attendons de cette méthode sont des résultats comparables à la réalité, sans exiger des valeurs exactes. De plus, il nous paraît indispensable de conserver l'aspect interactif lors de la phase de manipulation virtuelle. Notre objectif est toujours de permettre à un utilisateur de modifier *interactivement* et de façon virtuelle la géométrie et l'éclairage de scènes réelles.

Contrairement à la méthode précédente, nous travaillons pixel par pixel, pour estimer les réflectances et effectuer le rendu. Nous utilisons la méthode de radiosité uniquement pour calculer l'éclairage indirect. Le maillage utilisé est un maillage bien plus grossier que celui qui était nécessaire pour la méthode précédente, ce qui permet d'avoir un temps

de calcul très rapide pour la radiosité. Le travail pixel par pixel présente plusieurs avantages. Pour l'affichage, ceci nous permet d'éviter un affichage multi–passes utilisé pour la première méthode. De plus, cette nouvelle méthode nous permet d'enlever des objets réels en comblant l'information manquante par des textures générées et appliquées pixel par pixel, et se prête mieux aux améliorations (voir le chapitre 7).

## 5.1 Description générale de la méthode

Cette nouvelle approche comporte quatre étapes :

1. reconstruction géométrique et capture des différentes conditions d'éclairage pour un point de vue unique,
2. estimation de la réflectance pour chaque condition d'éclairage,
3. combinaison, par moyenne pondérée, des estimations précédentes, pour obtenir une estimation plus fiable de la réflectance,
4. initialisation du système d'éclairage pour permettre les modifications interactives.

Nous utilisons tout d'abord deux ensembles distincts de photographies. Le premier ensemble sert à retrouver un modèle géométrique simplifié de la scène réelle. Le deuxième ensemble permet d'estimer les propriétés de réflectance des objets réels contenus dans cette scène. Les photographies de cet ensemble sont du même point de vue, mais les conditions d'éclairage sont changées pour chacune d'elles, en déplaçant une source de lumière unique et connue. L'étape d'acquisition est développée dans la section 5.2. Nous appelons les photographies du deuxième ensemble des *images de luminance*. Cette appellation n'est pas strictement justifiée car il s'agit d'images « brutes » RGB. Nous levons cette restriction dans le chapitre suivant 6. Nous disposons pour chaque image de la géométrie de la scène réelle, ainsi que du modèle de la source de lumière. Par conséquent, nous avons accès à certaines informations telles qu'une valeur approximée de radiosité pour chaque pixel, les informations concernant la source de lumière, et tous les facteurs dépendants de la géométrie. À partir de ces informations, nous calculons une *image de réflectance* pour chaque position de la source de lumière. Nous n'avons cependant pas d'information sur l'éclairage indirect. Pour cette raison, la réflectance estimée par image est considérée comme approximative et inexacte dans les zones d'ombre, où l'influence de l'éclairage indirect est très importante.

Nous effectuons ensuite une moyenne pondérée des images individuelles de réflectance. La pondération se fait à partir de la confiance qu'on a pixel par pixel en ces estimations individuelles. Des filtres nous permettent d'évaluer cette confiance avant le calcul de la réflectance finale. Toute cette phase d'estimation de la réflectance est décrite dans la section 5.3. Nous supposons que la scène est diffuse. En particulier, nous supposons que les réflectances des objets sont diffuses, et que les lampes émettent de façon diffuse.

Le modèle géométrique et les propriétés de réflectance sont ensuite utilisés pour initialiser un système d'éclairage qui simule l'éclairage original de la scène (voir la section 5.4). Le rendu virtuel utilise les paramètres estimés, et re–projette les ombres des objets réels dues à la lampe réelle. L'image ainsi produite est comparable à la photographie originale.

Pour permettre une bonne définition des ombres, l'éclairage direct est calculé pixel par pixel en utilisant un lancer de rayons depuis le point de vue fixe. En revanche, l'éclairage indirect est simulé par une solution de radiosité hiérarchique, améliorée par une structure d'accélération pour des modifications dynamiques. L'éclairage indirect est indispensable pour avoir une simulation réaliste de l'éclairage. Cependant l'exactitude pixel par pixel de cet éclairage indirect ne nous a pas semblé nécessaire, et nous avons préféré une approximation par petites régions ; en revanche, la définition des ombres directes depuis les sources de lumière est visuellement très importante. Le choix de deux représentations différentes pour chaque type d'éclairage nous permet d'avoir à la fois une connaissance très fine de l'éclairage direct, tout en ayant une solution de radiosité grossière simulant l'éclairage indirect.

Lorsque ces quatre premières étapes de pré–calcul sont terminées, tout est en place pour modifier la géométrie et l'éclairage de la scène réelle modélisée. Les algorithmes de modification interactive adaptés à cette méthode sont présentés dans la section 5.5.

## 5.2 Acquisition

Afin de reproduire et de modifier l'éclairage d'une scène réelle, nous avons besoin d'un modèle géométrique en trois dimensions de cette scène ainsi qu'une connaissance de ses propriétés radiométriques. Pour cela, nous avons besoin d'acquérir les données de la scène réelle. Cette acquisition se fait en deux étapes.

Une première consiste à prendre quelques photographies qui servent à reconstruire un modèle géométrique en trois dimensions de la scène réelle sur laquelle nous voulons travailler. Nous utilisons pour la reconstruction l'outil *Rekon* [POF98]. Cet outil est décrit en détail dans le chapitre 3 section 3.2.2. Les photographies doivent être prises sous des angles de vue différents pour permettre un bon calibrage. L'ensemble de ces photos, typiquement entre sept et neuf, ayant servi à la reconstruction d'une de nos scènes est présenté sur la figure 5.1. Le modèle extrait est montré sur la figure 5.2. Les coordonnées des murs sont mesurées et leur modèle reconstruit à la main. Il est important de noter que le modèle extrait est un modèle simplifié ; la scène réelle n'est pas reconstruite dans tous ses détails. Ceci serait beaucoup trop coûteux en temps de modélisation et ne s'avère pas nécessaire dans notre but de ré–éclairage interactif.

La deuxième étape d'acquisition doit nous fournir un ensemble de données à partir desquelles nous pourrons estimer la réflectance diffuse des objets réels. Pour réaliser cette estimation, nous désirons avoir une bonne connaissance de l'éclairage de la scène. Pour cela, nous prenons un ensemble de photographies, prises depuis un seul et même point de vue, mais sous des éclairages différents. Pour faciliter l'estimation, nous n'avons qu'une seule source de lumière manipulable. Notre objectif est de permettre à un designer de réaliser cette étape d'acquisition dans un temps raisonnable, et en une seule fois. Le résultat ne doit pas dépendre de la qualité exhaustive de l'acquisition. De plus, le matériel utilisé doit être facilement manipulable et peu coûteux. Nous avons donc utilisé une lampe de jardin, très facile à déplacer et à modéliser. Cette lampe est illustrée sur la figure 5.3. Afin d'avoir un éclairage le plus diffus possible, nous avons couvert la lampe par une feuille de papier



FIG. 5.1: Photographies servant à la reconstruction 3D de la scène réelle.

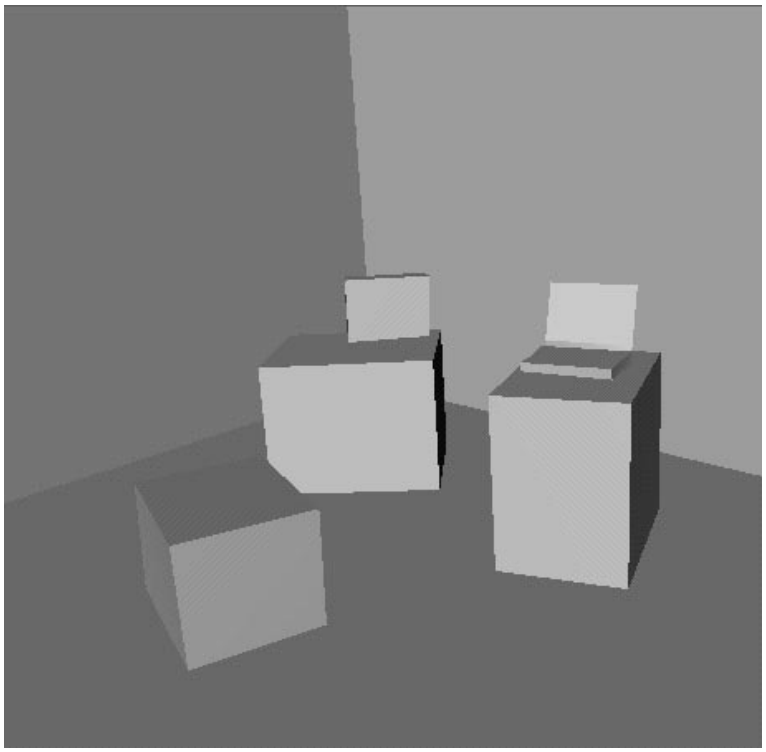


FIG. 5.2: Modèle 3D reconstruit.



FIG. 5.3: Lampe utilisée pour éclairer les scènes lors de nos tests. Elle est recouverte d'un papier de soie, permettant de se rapprocher du caractère d'émission diffuse.



FIG. 5.4: Images de luminance servant à l'estimation de la réflectance.

de soie. La lampe est placée dans une position différente pour chacune des photographies. Cette position ainsi que l'inclinaison de la lampe, sont mesurées à la main. Elle fournit le seul éclairage de la pièce. La figure 5.4 montre 7 photographies, appelées « images de luminance ». On voit clairement que les ombres sont différentes sur chacune de ces images. Pour chaque pixel, nous essayons d'avoir au moins une image où l'objet lui correspondant est éclairé directement par la lampe.

## 5.3 Estimation de la réflectance

Une fois la géométrie reconstruite, nous sommes prêts à procéder à l'estimation radiométrique de la scène réelle. Ceci commence par l'estimation de la réflectance dont nous présentons la méthode dans cette section.

### 5.3.1 Approche générale

L'estimation de la réflectance se fait d'abord pour chacune des images de luminance. Puis, pour chaque pixel, une valeur de confiance est calculée. Cette valeur reflète notre confiance en la précision de l'estimation de la réflectance pour ce pixel. Ensuite, une valeur finale de réflectance est calculée par une moyenne des valeurs estimées sur chacune des images de luminance, pondérées par la valeur de confiance associée.

Si nous supposons que toutes les surfaces de la scène sont diffuses, il existe une relation simple entre la luminance $L$ en un point 3D $P$, la réflectance $\rho$, et la lumière incidente en ce point :

$$L = \rho \left( \hat{B} + \sum_i F_i V_i E_i \right) \tag{5.1}$$

où $F_i V_i E_i$ est l'éclairage provenant directement de la source de lumière $i$, et $\hat{B}$ est l'éclairage indirect provenant du reste de la pièce par inter–réflexions. $E_i$ est l'exitance de la source $i$, c'est–à–dire l'énergie émise, $F_i$ est le facteur de forme entre la source $i$ et le point $P$, et $V_i$ est la valeur de visibilité entre la source $i$ et le point $P$. Le facteur de forme définit la portion d'énergie arrivant sur une surface, et la visibilité décrit la portion d'énergie non arrêtée par des obstacles (voir chapitre 2).

#### Structure de données

Pour estimer la réflectance et effectuer le rendu, nous travaillons dans l'espace de l'image. Ce choix permet d'avoir une estimation de la réflectance plus précise, en gardant les détails sans avoir à manipuler des polygones texturés. De plus, la gestion dans l'espace image permet d'utiliser des structures de données efficaces. A chacun des pixels, nous associons une structure augmentée stockant l'information nécessaire, ce qui évite des calculs inutiles. De plus, cette base de pixels nous permet de réaliser des localisations dans l'espace image (2D) plutôt que dans l'espace de la scène 3D.

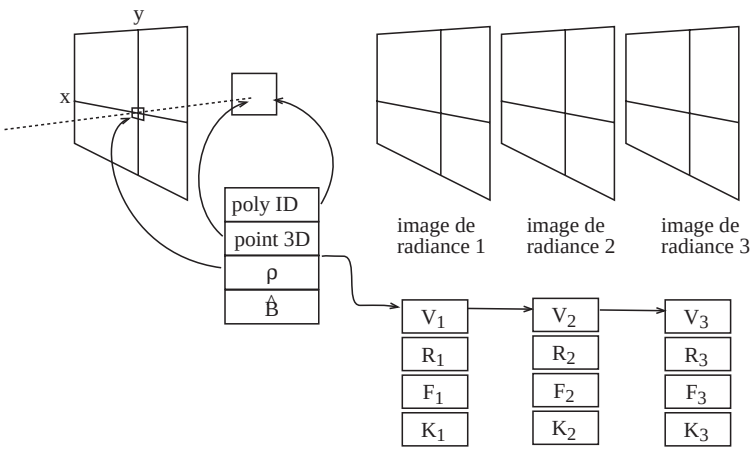



FIG. 5.5: Structure de données pour l'estimation de la réflectance.

Pour l'estimation de la réflectance, nous utilisons une structure de données pixel par pixel, pour le point de vue donné par les images de luminance. Nous créons une structure de données décrite par la figure 5.5. Dans cette structure, nous stockons pour chaque pixel :

– le point 3D $P$ re–projeté au centre du pixel,

– un numéro d'identification (ID) unique pour chaque polygone,

– la réflectance finale $\rho$ en ce point,

– une liste d'information pour chacune des images de luminance.

Pour chaque image de luminance, nous stockons pour chacun des pixels :

– le facteur de forme $F_i$ entre la source de lumière $i$ et le point 3D $P$,

– le facteur de visibilité $V_i$ depuis la source de lumière $i$ et le point $P$, déterminant la quantité de lumière parvenant sans obstacle en ce point,

– l'éclairage indirect $\hat{B}$ arrivant en ce point,

– la réflectance estimée $\rho_i$ en ce point,

– le coefficient $K_i$ représentant la justesse d'estimation de la réflectance.

Le numéro d'identification est déterminé en utilisant un « item buffer » [WHG84]. Un exemple est illustré par la figure 5.6, où chacune des couleurs différentes sont associées à un numéro d'identification. Le point 3D $P$ est calculé en intersectant une droite passant par le point de vue, et le centre du pixel correspondant, avec le polygone identifié par l'« item buffer »*.

Le facteur de forme $F_i$ est calculé en utilisant des techniques standard de calcul entre point et polygone [BRW89] (voir la section 2.2.1). La visibilité $V_i$ est la fraction du polygone de la source de lumière visible depuis le point $P$. La valeur de $V_i$ est comprise

*Cependant des erreurs de précision numérique peuvent survenir, entraînant notamment une non–intersection entre le rayon et le polygone. Ce phénomène apparaît surtout sur les contours des polygones. Dans ce cas–là, nous déplaçons le rayon vers les pixels voisins pour trouver une intersection.

entre ‘0’ lorsque le polygone est entièrement invisible depuis le point $P$, et ‘1’ lorsqu’il est complètement visible. L’estimation se fait par lancer de rayons, entre la source et le point. La valeur de visibilité est égale au pourcentage de rayons n’ayant rencontré aucun obstacle. Le nombre de rayons que l’on utilise varie entre 4 et 64, selon la rapidité et la qualité des calculs désirées.

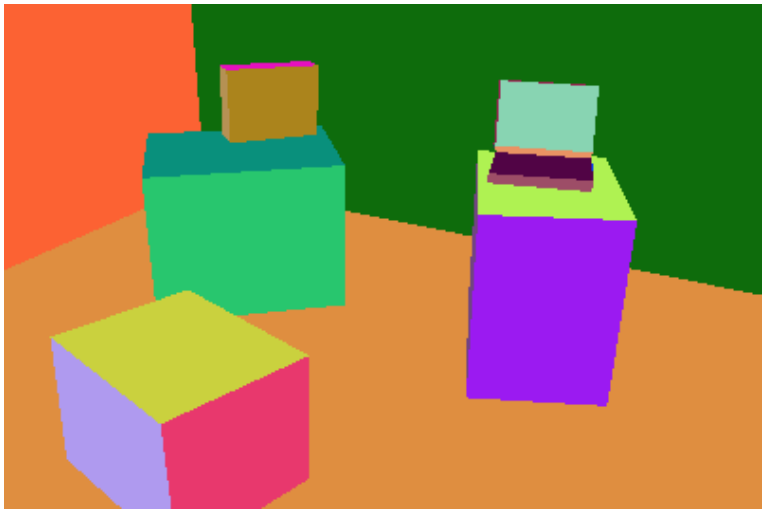


FIG. 5.6: Numérotation des polygones en utilisant un « item buffer ».

La connaissance géométrique de la scène nous permet de calculer certains des coefficients nécessaires à l’estimation de la réflectance. Nous n’avons cependant pas de connaissance radiométrique pour la luminance $L$, l’exitance de la source $E_i$ et l’irradiance indirecte $\hat{B}$. Nous utilisons alors une approximation pour chacune de ces valeurs.

La scène n’est éclairée que par une seule lampe qui est la même pour toutes les images de luminance. Si son exitance est inexacte, la répercussion de cette erreur se fait dans la solution de radiosité. Cependant, comme il n’y a qu’une seule source, l’erreur sur la solution de radiosité n’est qu’à un facteur multiplicatif près. L’exitance $E_i$ de la source est donc fixée par l’utilisateur[†]. Nous ne connaissons pas non plus la valeur d’irradiance indirecte $\hat{B}$. Cette valeur est fixée à la moyenne des luminances pour les pixels de l’image de luminance, divisée par la réflectance moyenne (ceci est basé sur l’équation (2.7)). Cette réflectance moyenne est également déterminée par l’utilisateur (comme pour le chapitre 4). La luminance $L$ est calculée directement pour chacun des pixels de l’image par :

$$L = T^{-1}(C_i) \tag{5.2}$$

où $T$ est la fonction de transformation de la caméra, et $C_i$ la couleur du pixel correspondant au point $P$ dans l’image de luminance associée à la lampe $i$. Nous avons dans un premier temps travaillé sur des images RGB avec une fonction de transformation identité, n’ayant aucun effet sur la couleur. Nous discutons ce choix dans le chapitre suivant (chapitre 6).

[†]Nous montrons dans le chapitre suivant comment retrouver une valeur d’exitance à partir d’images « High Dynamic Range ».

Nous estimons grâce à tous ces paramètres une valeur de réflectance par pixel et par image de luminance, en utilisant l'équation (5.1). Cette estimation n'est pas forcément valable pour tous les pixels à cause de nos approximations sur les valeurs radiométriques, ainsi que les approximations sur la reconstruction géométrique (la scène n'est pas entièrement modélisée). Pour surmonter ces limitations, nous allons créer une image de réflectance en faisant une moyenne pondérée des images de réflectance associées à *chaque* condition d'éclairage. Une valeur de confiance est alors associée à chacun des pixels, et elle servira de valeur de pondération. Cette valeur de confiance est représentée par le coefficient de validité $K_i$. Une opération de filtrage permet de déterminer et d'étendre les valeurs de confiance. Nous réalisons alors un filtre associé à chacune des images de luminance, qui sera composé de coefficients de validité $K_i$, reflétant la confiance que nous avons dans l'estimation de la réflectance pour chacun des pixels. Une valeur finale de réflectance est alors calculée à partir des valeurs estimées pour chacune des images de luminance, pondérées par les coefficients de validité. Nous détaillons toutes ces étapes dans les paragraphes suivants.

### 5.3.2 Estimation de la réflectance pour chaque image de luminance

Pour chacune des images de luminance capturées, la scène n'est éclairée que par une seule lampe. Les valeurs $F_i$ et $V_i$ peuvent être calculées (d'une façon approximative) car nous avons une représentation géométrique de la scène. Comme nous l'avons expliqué précédemment, nous ne connaissons pas exactement l'exitance $E_i$ de la source, ni la réflectance $\rho$ et l'éclairage indirect $\hat{B}$ au point $P$. Les approximations déjà décrites sont utilisées.

Pour chacune des images de luminance associée à une position différente $i$ de la lampe, nous estimons, pour chaque pixel de l'image correspondant à un point 3D de la scène, une réflectance $\rho_i$ :

$$\rho_i = \frac{T^{-1}(C_i)}{F_i V_i E_i + \hat{B}}. \tag{5.3}$$

Le réflectance obtenue n'est représentative de la réalité qu'à un facteur près étant donné que nous ne connaissons pas l'exitance de la source lumineuse.

Les résultats de cette estimation sont montrés par la figure 5.7. Dans la colonne de gauche, sont représentées les images de luminance, et dans la colonne du milieu, les images de réflectance associées. Il est à noter que la qualité des résultats n'est à cette étape pas satisfaisante. L'estimation est notamment assez mauvaise dans les régions d'ombre. Ceci est dû à l'approximation faite sur l'éclairage indirect (voir le chapitre 6 pour des améliorations sur le traitement de l'éclairage indirect).

### 5.3.3 Filtrage des valeurs de réflectance

L'éclairage indirect n'est pas le même pour tous les pixels ; l'utilisation du terme ambiant pour représenter l'éclairage indirect est donc une approximation insuffisante. De plus, la couleur du pixel n'est pas une valeur de luminance. D'une part, les valeurs de luminance réelles ont été transformées par des fonctions de la caméra. D'autre part, le bruit

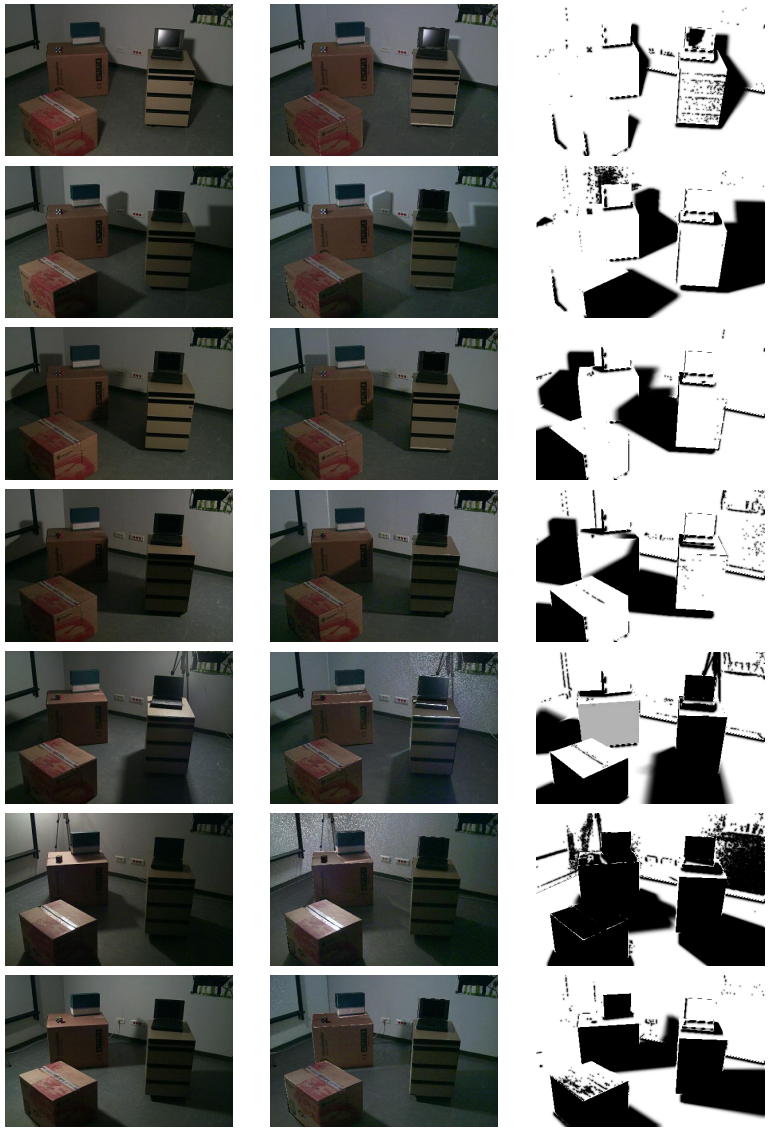


FIG. 5.7: À gauche, les images de luminance (photographies). Au milieu, les images de réflectance associées à chacune des images de luminance. À droite, les images de coefficients de validité associées (après filtrage). Les pixels en blanc équivalent à une grande confiance.

introduit par la compression et la transformation de la lumière en valeurs RGB font que ces valeurs de luminance sont dans certains cas incorrectes. Cette méconnaissance à la fois de l'éclairage indirect et d'une valeur correcte des luminances réelles, entraîne une mauvaise connaissance de la réflectance, en particulier dans les zones d'ombre où il n'y a pas d'éclairage direct. Finalement, l'hypothèse d'avoir une scène entièrement diffuse ne permet pas de tenir compte des effets spéculaires de certaines surfaces, comme le reflet d'objets sur le sol de notre scène ou les taches de lumière sur l'écran de l'ordinateur portable sur les images 5.7.

Pour éviter que tous ces artefacts ne se retrouvent dans la réflectance finale, nous utilisons une moyenne pondérée de ces images de réflectances. Les valeurs de pondération sont déterminées par un processus de filtrage, pour éviter les problèmes dus aux approximations. Pour cela, nous associons à chaque pixel $(x,y)$ de chaque image de luminance $i$ un facteur de validité $K_i(x,y)$ représentatif de la confiance que l'on porte à l'estimation de la réflectance pour ce pixel. Les valeurs $K_i(x,y)$ sont comprises entre un (lorsque nous avons très confiance dans la validité de l'estimation) et zéro (lorsque nous n'avons aucune confiance dans la validité de l'estimation). Le processus complet de filtrage est décrit par la figure 5.8.

**Algorithme de filtrage**
Initialisation : $K_i(x,y) = V_i(x,y)$
Fitre par valeur minimum : $K_i(x,y) = \min\left(V_i\left(x_x,y_y\right)\right)_{5\times5}$
Lissage :
**si** $moy\left(K_i\left(x_x,y_y\right)\right) < K_i(x,y)$
**alors** $K_i(x,y) = \text{moy}\left(K_i\left(x_x,y_y\right)\right)$
Filtrage des valeurs atypiques :
**si** $|\rho_i - moy(\rho_n)| > \varepsilon$
**alors** $K_i = 0$
Lissage :
**si** $moy\left(K_i\left(x_x,y_y\right)\right) < K_i(x,y)$
**alors** $K_i(x,y) = \text{moy}\left(K_i\left(x_x,y_y\right)\right)$

FIG. 5.8: Algorithme de filtrage.

Pour représenter le fait que l'estimation soit souvent erronée dans les régions d'ombre, les valeurs $K_i$ sont initialisées à partir des valeurs de visibilité $V_i$. Un exemple de valeurs de visibilité est illustré par la figure 5.9 (a). La confiance est ainsi très élevée dans les zones d'éclairage direct sans obstruction, et basse dans les zones d'ombre. Le long des zones d'ombre, la précision de la visibilité dépend beaucoup de l'exactitude de la géométrie et donc de la qualité de la reconstruction géométrique. Une première étape de filtrage consiste à prendre en compte cette inexactitude, de façon indépendante pour chacune des images de luminance. Les valeurs basses des facteurs de validité sont étendues aux pixels voisins en utilisant un filtre par valeur de minimum. Le pixel sur lequel est centré le filtre prendra la valeur minimale des valeurs à l'intérieur du support du filtre ($(x_x,y_y)$ dans la figure 5.8). Dans nos tests, nous utilisons par exemple un filtre de 5x5 ce qui reflète notre estimation de

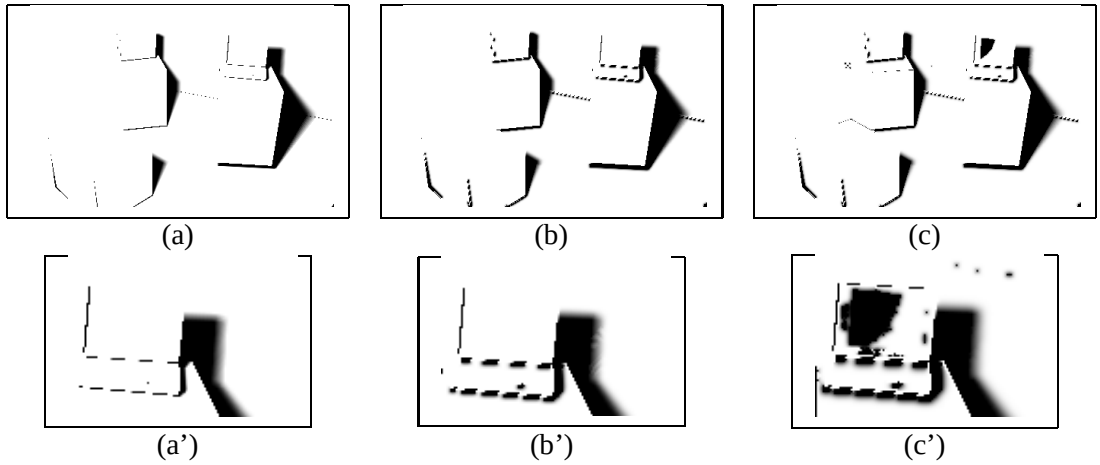



FIG. 5.9: Résultats à chaque étape du filtrage. (a) Initialisation des valeurs de validité par les valeurs de visibilité. (b) Élargissement des zones des valeurs basses. (c) Détection des valeurs singulières. (a'), (b') et (c') sont un zoom sur les images (a), (b) et (c).

la précision de la reconstruction à deux pixels près. La taille du filtre peut donc être ajustée si l'on considère que l'erreur est plus ou moins grande. Pour éviter des changements trop abrupts dans les valeurs de validité, un filtre est appliqué pour lisser les transitions. Ce filtre est également de taille 5x5. Le principe est de remplacer la valeur au centre du filtre par la moyenne des valeurs à l'intérieur du filtre si cette valeur est inférieure à la valeur initiale. Il n'est appliqué cependant que si tous les pixels observés dans le filtre appartiennent au même polygone. Les valeurs de validité ainsi étendues et lissées sont illustrées sur la figure 5.9 (b).

Les valeurs de réflectance très éloignées des autres valeurs estimées pour chacune des images de luminance sont détectées dans une deuxième étape de filtrage. Ceci permet d'enlever les taches de lumière dues à la spécularité d'un objet, comme la spécularité de l'écran du portable, ou d'enlever par exemple les pieds de la lampe utilisée, visibles sur deux des images de la colonne de gauche de la figure 5.7. Nous utilisons pour cela un nouveau filtre qui détecte les valeurs singulières (« outliers »). Pour chacun des pixels, la valeur médiane des réflectances est calculée sur les réflectances ayant une valeur de validité supérieure à un seuil, par exemple de 0.75. Ceci permet d'éliminer du calcul de la médiane les valeurs qui n'ont pas un degré de confiance suffisant pour être prises en compte dans le calcul final. Aux valeurs de réflectance dont l'écartement à la valeur médiane est supérieur à un certain seuil, on associe un facteur de validité de zéro. Ce seuil est déterminé par l'utilisateur. Il ne doit pas être trop bas, car les valeurs de réflectance initiales sont calculées comme nous l'avons vu de façon approximative et ne sont pas égales les unes aux autres. Dans nos tests, nous avons utilisé un seuil de 0.3. À nouveau, pour éviter des discontinuités abruptes, un filtre (de 3x3) lissant les contours est appliqué.

À la fin de cette étape, nous avons les valeurs finales de validité qui serviront à pondérer les valeurs de réflectances pour calculer la réflectance moyenne. Cette opération est expliquée dans la section suivante 5.3.4. Ces valeurs sont illustrées sur la figure 5.9 (c).

Les images composées des coefficients de validité sont présentées dans la colonne de droite de la figure 5.7 pour chacune des images de luminance. Les valeurs sont basses dans les régions d'ombre, et basses pour les valeurs singulières comme pour la tache de spécularité visible sur la première ligne d'images.

### 5.3.4 Estimation finale de la réflectance

Nous avons estimé une réflectance diffuse pour chacun des pixels des images de luminance. À chacune de ces valeurs, nous avons associé une valeur de confiance $K_i$. Nous avons vu que l'estimation est approximative sur chacune des images prises indépendamment les unes des autres. Par contre, en combinant ces valeurs, nous pouvons avoir une estimation convaincante, diminuant ainsi les artefacts. Pour cela, nous calculons une moyenne sur chacun des pixels, en pondérant les valeurs de réflectance par leur coefficient de validité $K_i$. Pour chaque pixel, la réflectance associée $\rho$ est :

$$\rho = \frac{\sum_i K_i \rho_i}{\sum_i K_i} \tag{5.4}$$

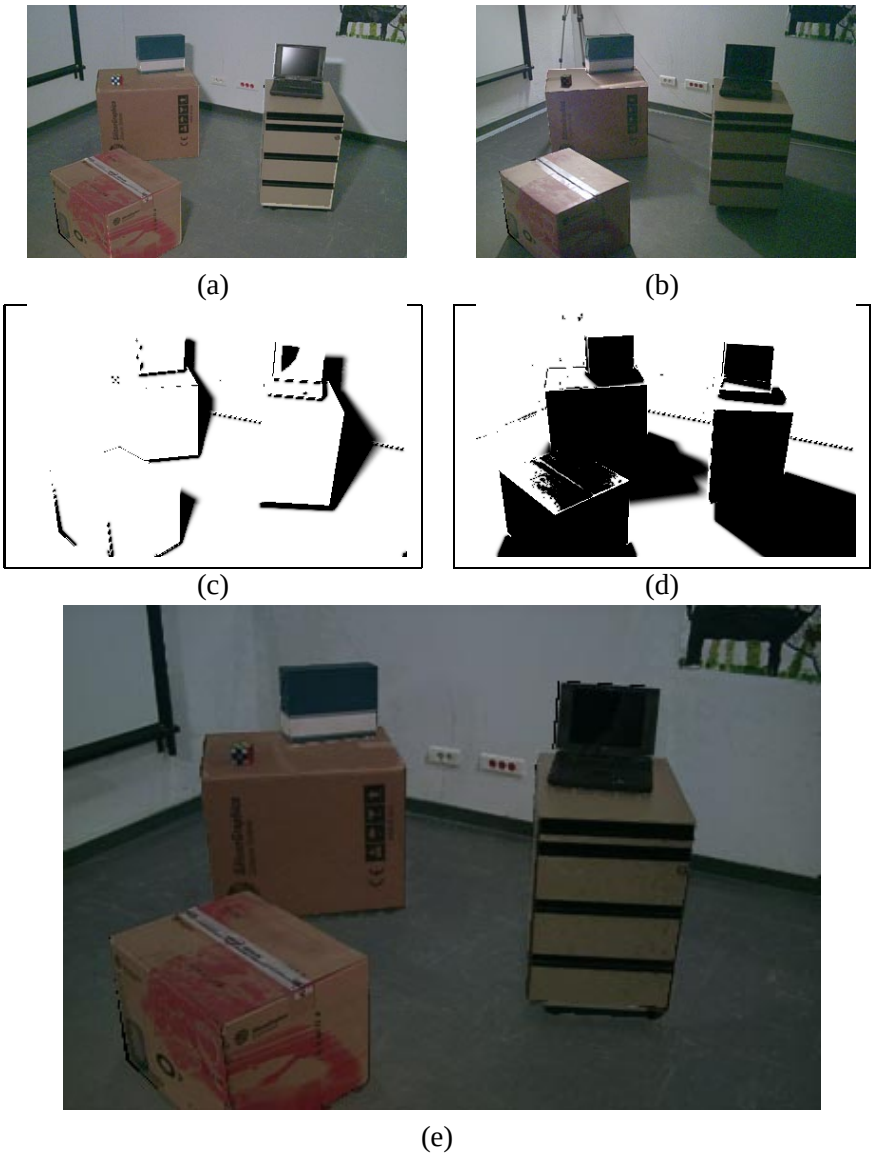
Les résultats sont montrés sur la figure 5.10. Les images (a) et (b) sont des images de réflectances estimées pour deux des images de luminance, auxquelles sont associées des valeurs de validité, montrées respectivement en (c) et (d). En (e) est montrée la réflectance finale estimée par une moyenne pondérée des sept images de réflectance (voir figure 5.7). Alors que les réflectances sont estimées d'une façon moins précise dans les images (a) et (b) (comme pour l'effet spéculaire sur le portable, ou dans les régions d'ombre), l'image de réflectance (e) finale contient très peu d'artefacts.

## 5.4 Initialisation du système d'éclairage

Lorsque les réflectances sont calculées, le système d'éclairage peut être initialisé. L'éclairage est calculé en combinant deux méthodes. L'éclairage direct est calculé pixel par pixel et l'éclairage indirect par une solution de radiosité optimisée. Ce choix permet d'avoir une bonne définition de l'éclairage direct, dont les variations sont souvent abruptes, dans les régions d'ombre par exemple. En revanche, les variations de l'éclairage indirect sont souvent moins significatives et une approximation par une solution de radiosité est souvent suffisante. Deux structures sont ainsi initialisées pour calculer l'éclairage. Ceci nous permet d'avoir une bonne définition des contours des ombres, tout en ayant un rendu et une mise à jour interactifs.

### 5.4.1 Éclairage direct

L'éclairage direct est calculé pixel par pixel. Ce calcul est stocké au fur et à mesure dans une structure qui servira à réaliser un rendu accéléré. Pour cela, nous utilisons une deuxième structure de données où nous stockons à nouveau de l'information pixel par pixel :



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

FIG. 5.10: (a) et (b) Réflectances estimées pour deux images de luminance. (c) et (d) Valeurs de validité associées à ces deux mêmes images de luminance. (e) Réflectance finale.

– le point 3D $P$ re–projeté au centre du pixel,
– le numéro d'identification (ID) du polygone correspondant,
– l'éclairage indirect $\hat{B}$ arrivant en ce point,
– la réflectance $\rho_i$ en ce point,
– une liste d'informations pour chacune des sources de lumière $s$ de la scène :
  – le facteur de forme $F_s$ entre la source de lumière $s$ et le point 3D $P$,
  – le facteur de visibilité $V_s$ depuis la source de lumière $s$ et le point $P$,
  – l'exitance $E_s$ de la source $s$.

La structure de données complète est illustrée figure 5.11. Cette deuxième structure de données n'était pas indispensable. Nous aurions pu travailler avec une structure unique, groupant les informations de cette structure et de la première structure illustrée par la figure 5.5.

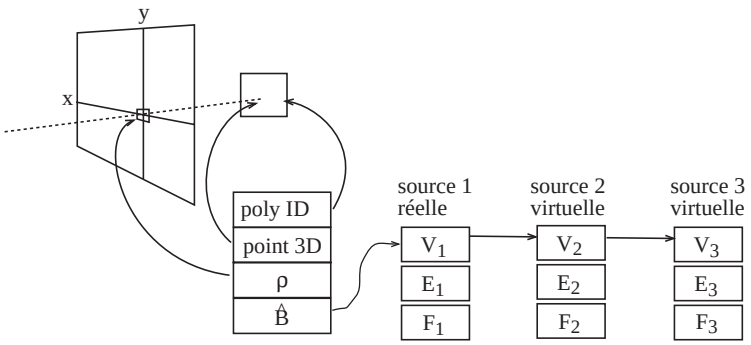



FIG. 5.11: Structure de donnée utilisée pour le rendu.

Cette structure est initialisée pixel par pixel. Les sources de lumière présentes dans la scène ne sont plus celles utilisées pour l'estimation de la réflectance. Lors de l'initialisation du système, une seule source est conservée. Cette source correspond à la source 1 dans la structure de données sur la figure 5.11. L'utilisateur choisit une image de référence, et la source de lumière associée. Les sources dont l'indice est supérieur à un sont des sources de lumière virtuelles ajoutées dans la scène. Le nombre total de sources $n_s$ variera selon les manipulations de l'utilisateur.

Pour chaque pixel, les champs de réflectance sont initialisés, et la liste des sources de lumière comprend un élément, avec l'exitance de la source, les facteurs de forme et la visibilité. Cette initialisation se fait directement à partir de la table ayant servi à estimer les réflectances (voir section 5.5).

### 5.4.2 Éclairage indirect

Pour permettre le calcul de l'éclairage indirect, nous utilisons un système de radiosité optimisé. Tout d'abord, pour chaque polygone visible sur l'image de référence (depuis

le point de vue choisi), la réflectance est initialisée. Pour cela, les polygones sont pré–subdivisés. Pour chaque carreau, la réflectance est rendue égale à la moyenne des réflectances sur les pixels correspondant à partir de l'image de réflectance finale. Pour les carreaux non visibles associés à un polygone visible, la réflectance est initialisée à la moyenne des réflectances de ce polygone. L'initialisation de la réflectance dans le système de radiosité pour ces éléments visibles depuis l'œil est illustrée figure 5.12.

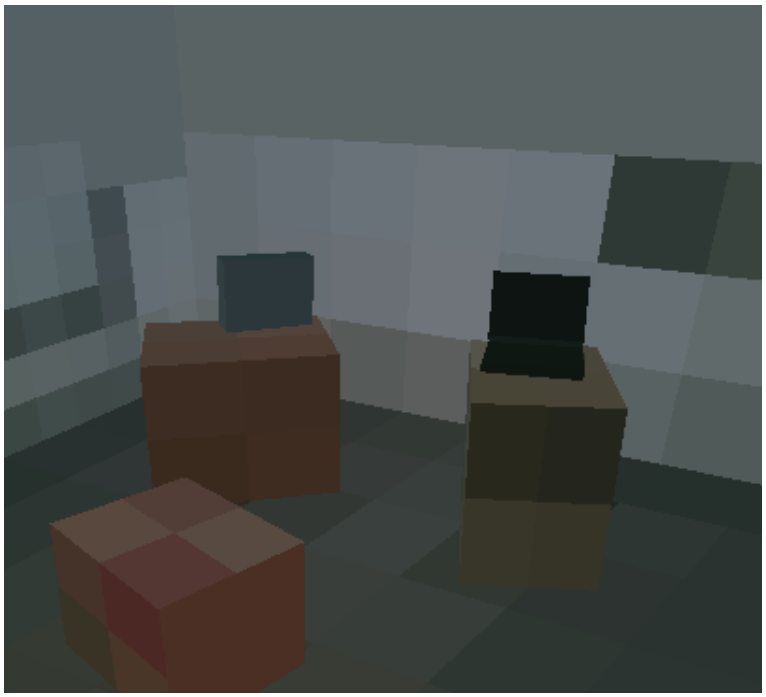


FIG. 5.12: Éléments de la radiosité dont la réflectance est initialisée à partir de l'image de réflectance (voir figure 5.10 (e)).

Pour les polygones non visibles sur l'image de référence, un traitement spécial est effectué. Ces polygones sont montrés sur la figure 5.13. Des photographies additionnelles, montrées sur la figure 5.14, sont prises lors de la capture. En utilisant l'équation (5.3), nous retrouvons pour un échantillon de pixels des valeurs de réflectance qui sont ensuite moyennées. Cette valeur moyenne sert à initialiser la réflectance du polygone. Par exemple, sur la figure 5.15, la texture en (a) sert à estimer une réflectance approximative pour le mur en (b). Les polygones pour lesquels nous n'avons pas d'image de luminance comme base de calcul, sont initialisés à une valeur de réflectance fixée par l'utilisateur.

Lorsque l'ensemble des réflectances de la scène est initialisé, une solution de radiosité peut être calculée. La subdivision est préfixée sur un échantillonnage grossier, car nous n'avons pas besoin d'avoir une définition précise des contours des ombres. La solution de radiosité est calculée par des algorithmes basés sur la radiosité hiérarchique avec « clustering » (voir section 2.2.1). Cette solution de radiosité nous fournit un éclairage global. Pour en extraire l'éclairage indirect, nous procédons en deux étapes. Dans la première étape,

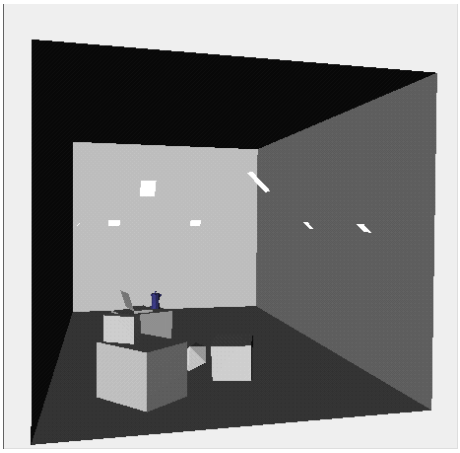
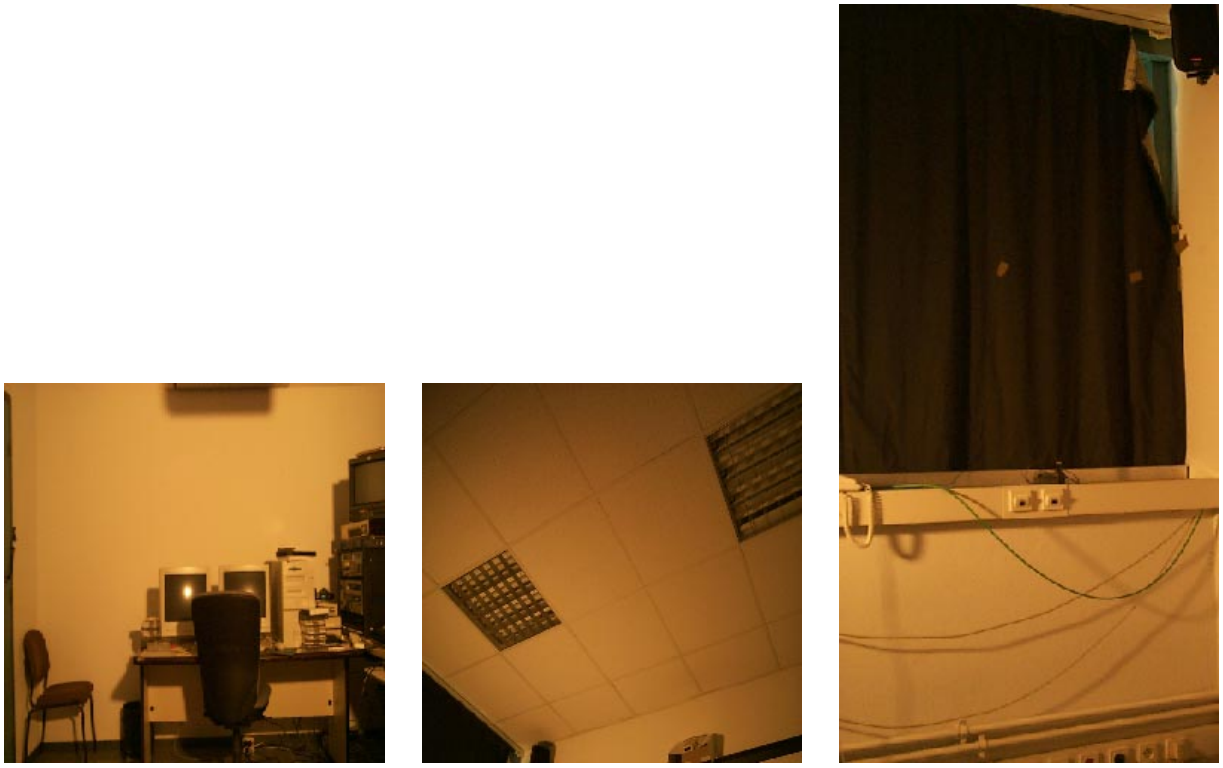
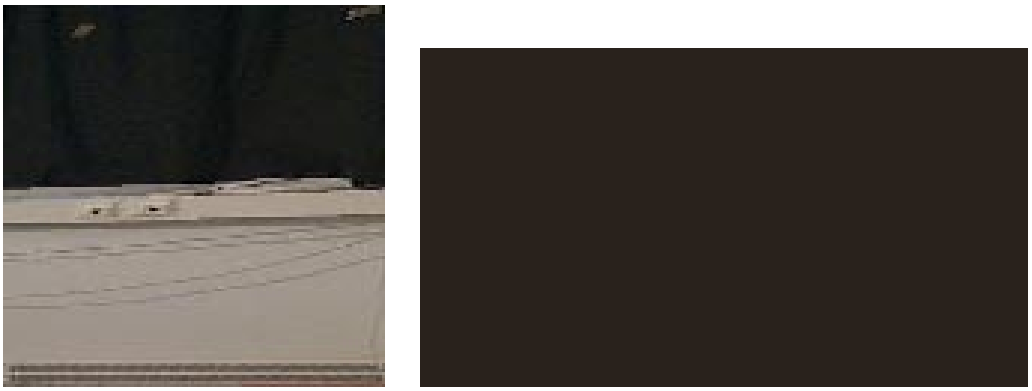


FIG. 5.13: Polygones invisibles depuis l'image originale. Dans cette scène, les polygones invisibles reconstruits sont les murs et le plafond.



(a) (b) (c)

FIG. 5.14: Photographies additionnelles permettant d'avoir une texture plus significative pour estimer la réflectance des murs, non visibles sur l'image d'origine. (a) Mur de droite, (b) Plafond, (c) Mur de derrière.



(a) (b)

FIG. 5.15: (a) Texture servant à l'estimation plus approximative de la réflectance d'un mur non visible depuis le point de vue choisi (celui des images de luminance). (b) Réflectance du mur ainsi estimée.

nous calculons la radiosité globale ; dans la deuxième, nous ne calculons que l'éclairage direct (une seule itération dans la solution). L'éclairage indirect est ensuite estimé pour chaque élément (et pour chaque sommet d'élément), en soustrayant l'éclairage direct de l'éclairage global.

L'éclairage indirect ainsi calculé est ensuite utilisé pour mettre à jour la structure d'affichage, composée en partie par la structure représentant l'éclairage direct (figure 5.11). On utilise pour cela le matériel d'accélération graphique des stations de travail. Les valeurs de l'éclairage indirect sont affichées sur un tampon de mémoire auxiliaire, avec un lissage par affichage avec des valeurs aux sommets et interpolation. Cet affichage est récupéré pixel par pixel pour initialiser $\hat{B}$ dans la structure de données.

En prévision d'éventuelles modifications, une structure de « shaft » est créée [HW94]. Suivant l'idée de la méthode de Drettakis et Sillion [DS97], un shaft est associé à chaque lien, et pour chaque shaft, la liste des polygones intersectés est maintenue. Ceci nous permettra par la suite d'optimiser les changements de visibilité et de localiser les modifications dans la structure des liens.

### 5.4.3 Rendu

Le rendu est réalisé pixel par pixel en utilisant la structure d'affichage 5.11. Lors de l'affichage, la couleur $C$ est calculée pour chaque pixel par l'équation suivante :

$$C = \rho \left( \hat{B} + \sum_{s=1..n_s} F_s V_s E_s \right) \tag{5.5}$$

où $F_s, V_s, E_s$ sont respectivement le facteur de forme, la visibilité, et l'exitance de la source $s$, $n_s$ est le nombre total de sources dans la scène, $\rho$ la réflectance associée au pixel, et $\hat{B}$ l'irradiance indirecte. Avant toute insertion de source virtuelle, il n'y a qu'une seule

source, représentant la source originale. Pour accélérer l'affichage, les composantes de l'équation (5.5) sont groupées et pré–calculées. Typiquement l'affichage d'une image 300 par 400 pixels prends moins d'un dixième de seconde.

La figure 5.16 (b) est le résultat d'un rendu après initialisation de tous les paramètres de la scène originale, et en utilisant l'équation (5.5). Cette image ne contient plus d'informations de la photographie originale, mais n'utilise que celle fournie par l'image de réflectance, dans laquelle il n'y a plus les effets d'éclairage. Les ombres sont à nouveau projetées grâce au terme de visibilité $V_s$. La qualité du rendu, tenant compte des détails, est due à la qualité de la réflectance $\rho$, différente pixel par pixel. L'image simulée n'est pas identique à la photographie originale montrée figure 5.16 (a). En effet, la réflectance estimée est supposée parfaitement diffuse, et en conséquence, les effets spéculaires ne sont pas affichés. De plus, la reconstruction des objets non visibles depuis le point de vue est très approximative ce qui implique des erreurs dans le calcul de l'éclairage indirect. Cette image simulée est cependant convaincante car elle est similaire et cohérente par rapport à l'originale, ce qui la rend acceptable pour l'utilisation voulue.



(a) (b)

FIG. 5.16: (a) Photographie originale. (b) Reprojection après simulation de l'éclairage.

## 5.5 Ré–éclairage et modification de la géométrie

Lorsque les réflectances sont estimées et le système d'éclairage initialisé, nous pouvons modifier interactivement sur ordinateur les propriétés des scènes réelles reconstruites. Les modifications permises par le système sont au niveau de la géométrie et de l'éclairage. Nous pouvons ajouter, déplacer et enlever des objets virtuels. Nous pouvons également enlever des objets réels. De plus, nous pouvons modifier l'intensité des sources de lumière et ajouter des nouvelles sources virtuelles.

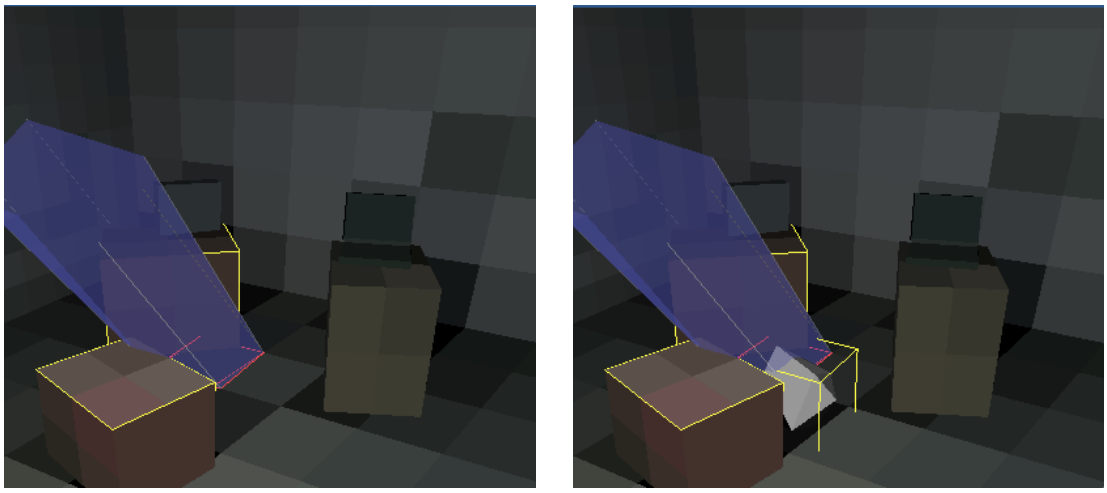
### 5.5.1 Modification de la géométrie

Nous désirons pouvoir interactivement ajouter, déplacer ou enlever des objets. Si nous refaisions entièrement le calcul de radiosité et de mise à jour de la table d'affichage, cette

modification géométrique prendrait de l'ordre de plusieurs minutes. Pour permettre un temps de mise à jour interactif, nous voulons localiser les modifications. Pour cela, nous utilisons une structure de shafts qui ne nous permet non seulement d'accélérer les calculs de visibilité par une localisation spatiale, mais aussi d'identifier les pixels directement affectés par des modifications de positions d'ombres (éclairage direct).

### Insertion d'un objet virtuel

Lors de l'insertion d'un objet virtuel, la solution de radiosité hiérarchique est mise à jour par l'algorithme suivant. Tout d'abord, l'objet est attaché à la racine de la hiérarchie (≪ cluster ≫ englobant toute la scène). Ensuite, des liens sont établis entre cet objet et chacune des lampes. Puis des itérations de raffinement sont faites pour établir les liens aux niveaux satisfaisant le critère BF. Lors de cette passe de modification de la structure de radiosité, les shafts modifiés sont marqués, de façon à pouvoir plus tard localiser rapidement les changements. Un shaft peut être marqué ≪ modifié ≫ soit parce qu'il vient d'être créé, soit parce qu'il y a eu un changement de visibilité (l'objet nouvellement inséré intersecte ce shaft). Dans ce dernier cas, les polygones de l'objet intersectant le shaft sont insérés dans la liste des polygones intersectés associée au shaft. Le nouvel objet inséré est testé pour chacun des shafts, et de nouveaux calculs de visibilité dans la solution de radiosité sont effectués seulement s'il y a intersection. Un exemple de modification des shafts est illustré sur la figure 5.17. En (a), le shaft est initialisé. La liste des polygones intersectés est montrée par des traits clairs. Lors de l'insertion d'un objet virtuel en (b), le cluster englobant l'objet virtuel intersecte ce shaft et est inséré à la liste des polygones intersectés. Ce cluster est dessiné en traits clairs.



(a) (b)

FIG. 5.17: (a) Shaft associé à un élément de la hiérarchie, avec en traits clairs la liste des polygones qu'il intersecte. (b) Un objet virtuel en gris est inséré dans la scène. Le cluster dans lequel il est inclu intersecte le shaft et est inséré à la liste des objets intersectés.

Une fois la nouvelle solution de radiosité mise à jour, l'éclairage indirect est calculé et la table d'affichage est mise à jour. La mise à jour se fait en deux passes différentes successives.

Tout d'abord, les changements visuels dus directement à l'insertion de l'objet, sans tenir compte de l'éclairage, sont localisés et mis à jour. Pour cela, la boîte englobante de l'objet inséré est projetée sur la fenêtre de vue. Ensuite, nous calculons le rectangle englobant de cette projection pour déterminer sur quels pixels les modifications doivent avoir lieu. Lors de l'insertion d'un objet, les pixels ainsi localisés sont visibles en gris clair sur la figure 5.18 (a). Lors du parcours de ces pixels potentiellement modifiés, les opérations suivantes ont lieu :

- Un nouvel « item buffer » est réalisé pour mettre à jour le numéro d'identification des polygones dans la zone de modification.
- Si le numéro d'identification est différent de celui anciennement affecté au pixel, la réflectance est remplacée par la réflectance du nouvel objet. Un nouveau point 3D est aussi calculé, en utilisant l'intersection d'un rayon depuis le point de vue avec cet objet (ce calcul est déjà utilisé dans la section 5.3.1).
- De plus, pour chacune des structures de source de lumière, un nouveau facteur de forme entre chaque source et le nouveau point 3D est calculé. Si le polygone associé au point 3D n'a pas été marqué modifié (c'est-à-dire qu'il n'a pas de shaft associé marqué modifié), une nouvelle valeur de visibilité est calculée. Ce cas peut apparaître plutôt lors du déplacement d'un objet ou lors de son enlèvement, quand des pixels deviennent à nouveau visibles depuis l'œil, mais pour lesquels il n'y pas de modification d'éclairage direct. Dans le cas où le polygone est marqué « modifié », la visibilité est seulement calculée lors de la deuxième passe, que nous allons présenter dans la suite. Cette distinction évite une redondance des calculs.



(a) (b)

FIG. 5.18: Un objet virtuel est inséré dans la scène. (a) Les pixels localisés en gris clair sont ceux sur lesquels il y a des changements visuels potentiels à cause de la projection de l'objet virtuel sur l'écran. (b) Les pixels localisés en gris clair sont ceux pour lesquels il y a des changements de visibilité potentiels, dus aux ombres.

La deuxième passe consiste à mettre à jour les valeurs de visibilité pour les pixels af-

fectés par une modification d'éclairage direct. Lorsque l'objet est inséré dans la scène, certains shafts sont touchés, car l'objet virtuel les intersecte. L'objet intersecte donc des volumes source–récepteur, et dans ce cas, il y a un changement de visibilité, et notamment de nouvelles ombres. Les polygones récepteurs associés aux shafts touchés sont marqués de telle façon qu'on puisse rapidement les identifier dans la table de pixels. Pour mettre à jour la visibilité dans la table de pixels, nous parcourons tous les pixels de la table. Si le polygone associé à un pixel est marqué modifié, alors nous réalisons un nouveau calcul de visibilité pour chacune des sources associées à un shaft marqué modifié. Dans ce cas, la liste des objets intersectés associée au shaft permet d'accélérer les calculs. Ensuite, les nouveaux pixels sont affichés. Cette deuxième partie dans la mise à jour comprend des calculs de visibilité qui sont très coûteux. La localisation permet d'éviter des calculs inutiles compromettant les possibilités d'interactivité.

#### Déplacement d'un objet virtuel

Lors du déplacement d'un objet, le processus est pratiquement identique. Les différences se situent d'abord lors de la modification de la solution de radiosité, quand la hiérarchie des liens associés à l'objet est modifiée. Dans un souci d'efficacité, nous avons conservé l'objet et les structures associées dans la hiérarchie, plutôt que de les enlever puis les rajouter. Les liens attachés à l'objet virtuel sont conservés, mais les informations de visibilité et de facteur de forme associées sont mises à jour. Par contre, les shafts associés à ces liens sont détruits puis recréés car ils ont une composante géométrique. Puis la boîte englobante est projetée sur la fenêtre d'affichage, dans les positions avant et après le déplacement. Le rectangle contenant les pixels à traiter a donc une taille supérieure à celui calculé pour l'insertion de l'objet. La mise à jour des réflectances utilise la réflectance originale des objets réels, qui a été conservée. La suite de l'algorithme pour la mise à jour de la visibilité est identique à celle de l'insertion d'un objet. Les shafts intersectés par la position avant et après déplacement permettent d'identifier les régions potentiellement affectées.

#### Enlèvement d'un objet virtuel

L'algorithme est pratiquement identique pour enlever un objet que pour en insérer un. Au lieu d'insérer un nouvel objet dans la hiérarchie, l'objet en est enlevé. Tous les liens et les shafts attachés à cet objet sont détruits, l'objet enlevé de la hiérarchie (il n'est plus attaché à la racine) et une nouvelle solution de radiosité est calculée. L'algorithme de mise à jour de la table d'éclairage direct reste inchangé.

Ces algorithmes sont présentés sur la figure 5.19. Ils sont valables pour ajouter, déplacer ou enlever un objet virtuel. Les algorithmes utilisés pour manipuler les objets réels sont équivalents, mais comportent des différences importantes car nous ne connaissons pas ce qu'il y a derrière un objet réel enlevé.

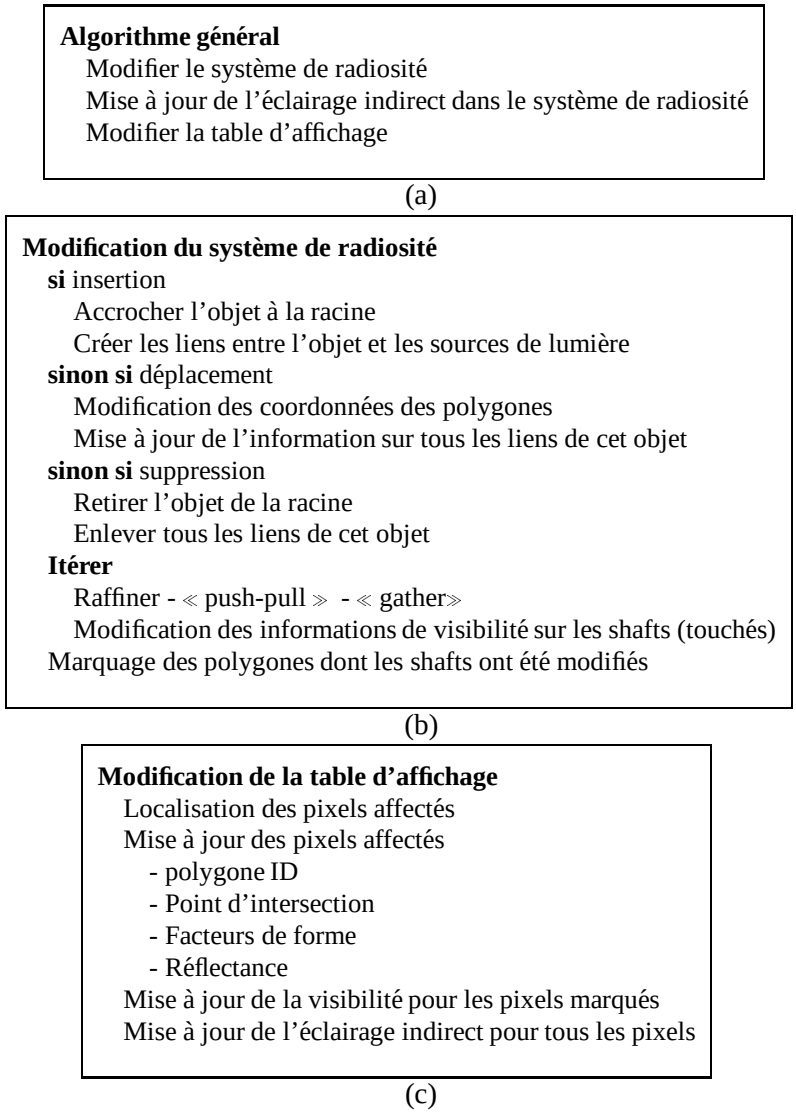



FIG. 5.19: (a) Algorithme général. (b) Algorithme de mise à jour du système de radiosité. (c) Algorithme de mise à jour de la table d'affichage, pour l'insertion, la suppression, ou le déplacement d'un objet virtuel.

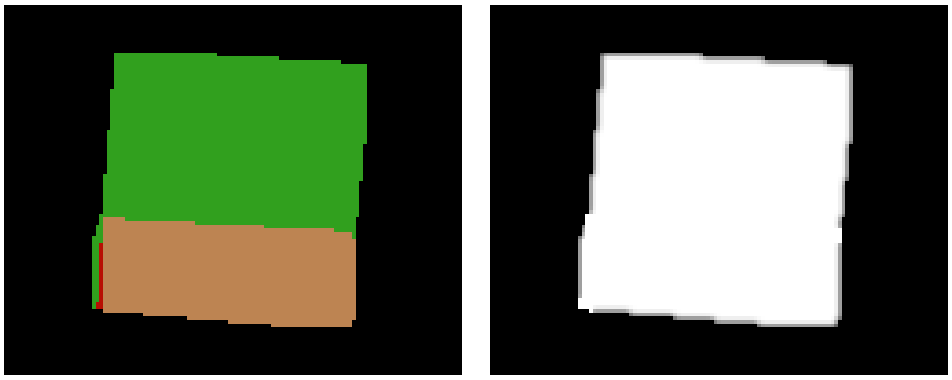
**Enlèvement d'un objet réel**

Nous avons vu comment enlever un objet virtuel. L'enlèvement d'un objet réel est un peu plus difficile car la connaissance des réflectances de la scène n'est valide que depuis un seul point de vue. Si nous enlevons un objet contenu dans la photographie initiale, des objets cachés vont devenir visibles depuis l'œil. Enlever un objet réel va donc produire des *trous*, c'est–à–dire des régions non connues car invisibles dans les images réelles. Pour pallier ce problème, nous utilisons un algorithme de remplissage par génération de texture. Cet algorithme n'est pas très coûteux mais n'est cependant pas interactif. Pour cette raison, nous le réalisons en pré–calcul avant la phase d'interactivité. La technique de génération de textures que nous utilisons est celle de Heeger et Bergen [HB95]. Nous utilisons en particulier l'implantation de El–Maraghi [EM]. Ces techniques de génération de textures sont décrites dans la section 5.5.2. Nous nous sommes restreints à des textures simples. Cependant, des algorithmes de génération de textures plus complexes pourraient être utilisés dans le cadre de notre problème [De 97].

Pour synthétiser les textures qui servent à masquer les régions inconnues, nous utilisons l'image de réflectance. L'avantage de l'image de réflectance est qu'elle ne contient plus d'effets dus à l'éclairage (ou qui sont au moins très diminués), comme les ombres ou la variation due aux facteurs de forme. Pour remplir la région non connue par une nouvelle texture, nous nous sommes inspirés de l'algorithme d'Igehy et Pereira [IP97]. L'application de l'algorithme est cependant différent car nous travaillons directement à partir d'une texture privée des effets d'éclairage, et nous connaissons la géométrie de la scène. Nous extrayons un échantillon de texture de réflectance sur les polygones ayant des régions potentiellement masquées. L'extraction est manuelle. Cet échantillon est ensuite passé en paramètre au programme de génération de texture, ainsi que la taille de la région à couvrir. La taille de cette dernière est calculée automatiquement, comme expliqué par la suite. Cette texture sert dans la phase interactive à initialiser les nouvelles réflectances. Ces textures générées sont stockées pour les objets marqués « enlevables » par l'utilisateur.

Lors de la phase d'interaction, l'utilisateur choisit l'objet à enlever. Les régions masquées par cet objet sont trouvées par un « item buffer ». Les nouveaux numéros d'identification sont calculés dans la zone masquée par l'objet. À nouveau, cette zone est déterminée par la projection de la boîte englobante de l'objet sur la fenêtre de vue. Dans les régions ainsi identifiées, deux masques sont créés. Le premier est un masque de pondération, le deuxième un masque de texture. Le masque de texture est rempli par les numéros d'identification que nous venons de calculer. Ce masque est utilisé pour déterminer quelle texture utiliser pour remplir le trou. Un exemple est illustré par la figure 5.20 (a). Le masque de pondération sert à déterminer la zone à traiter. Les pixels marqués '1' sont ceux à remplir, les pixels marqués '0' sont à laisser tels quels. Le masque est étendu à quelques pixels autour de la zone à modifier pour éviter des problèmes dus à une inexactitude géométrique, de façon à ne plus avoir les couleurs du contour de l'objet enlevé. Un exemple est montré sur la figure 5.20 (b). Les bords du masque, c'est–à–dire les frontières 0/1, sont rendus flous en établissant un fractionnement des valeurs (c'est–à–dire entre 0 et 1).

Dans la région identifiée par la projection de la boîte englobante de l'objet réel, les numéros d'identification des polygones maintenant visibles servent à remplir le masque



(a) (b)

FIG. 5.20: Masques utilisés pour enlever l'ordinateur portable (voir la figure 5.21). (a) Masque représentant les numéros d'identification des polygones masqués par l'objet réel enlevé. (b) Masque indiquant en blanc la région touchée par l'enlèvement de l'objet. La région en noir n'est pas modifiée par le remplissage de texture.

de texture. Le calcul des nouvelles réflectances se fait en utilisant les deux masques. Les couleurs des pixels sont remplies :

- depuis les valeurs des textures générées, sélectionnées à partir du masque de texture, si les valeurs du masque de pondération est '1',
- depuis les anciennes valeurs de réflectance, si les valeurs du masque de pondération est '0',
- par une combinaison des deux, textures générées et anciennes réflectances, si les valeurs du masque de pondération est entre '0' et '1'.

Le fractionnement des valeurs sur les bords du masque est réalisé par un filtre appliqué sur chaque pixel. La valeur au centre du fitre est égale à la moyenne pondérée des pixels voisins.

Ensuite, le même algorithme que celui utilisé pour les objets virtuels est appliqué. Un exemple de génération d'une nouvelle image de réflectance est montré figure 5.21 (b). Le résultat après simulation de l'éclairage est montré en (c).

### 5.5.2 Génération de textures

De nombreux algorithmes existent pour générer des textures à partir d'échantillons. La difficulté du problème incite à définir des algorithmes adaptés à certains types de texture. La technique de génération de texture de Heeger et Bergen [HB95] par exemple fonctionne mieux sur des textures stochastiques, comme une texture de granite par exemple, que sur des textures déterministes, comme une image de mur en briques. En revanche, cette méthode a été testée sur un grand nombre d'échantillons, et a donné pour la majorité de bons résultats. L'algorithme est basé à la fois sur une pyramide directionnelle inversable (« self-inverting steerable pyramide »), et sur une procédure d'ajustement des couleurs. El–Maraghi a implanté cette méthode et son programme est accessible sur internet [EM].



(a) (b) (c)

FIG. 5.21: L'ordinateur portable est enlevé virtuellement de la scène. (a) Scène originale. (b) Les réflectances sont mises à jour après un remplissage par génération de textures des régions précédemment cachées par l'ordinateur. (c) En utilisant ces nouvelles valeurs de réflectance, l'éclairage est simulé en modifiant localement l'éclairage direct.

Dans notre algorithme d'enlèvement d'objets réels, nous nous sommes inspirés des techniques utilisées par Igehy et Pereira [IP97]. Leur objectif est d'enlever une partie de l'image, et de remplir le trou laissé par une texture. Le résultat doit paraître réaliste. Pour atteindre cet objectif, ils modifient l'algorithme de Heeger et Bergen [HB95] de génération de texture, en y incluant des procédures spécifiques. Les données de l'algorithme de Heeger et Bergen sont un échantillon et un bruit. Igehy et Pereira ajoutent en entrée un masque et une image originale. Le masque est créé à la main. Ses valeurs sont de '0' dans la partie à enlever, des valeurs fractionnées autour, et de '1' dans la partie à conserver. L'échantillon de texture est également choisi par l'utilisateur. Contrairement à notre approche où la texture est synthétisée puis composée à l'image, la composition se fait au fur et à mesure de la génération de texture. Les limitations de cet algorithmes sont celles de l'algorithme de génération de texture de Heeger et Bergen qui ne traite que les textures stochastiques. De plus, la région à modifier ne doit correspondre qu'à une seule surface, et les différences d'éclairage ne sont pas prises en compte.

Il existe une autre méthode efficace d'enlèvement de « défauts », comme par exemple un fil électrique ou une inscription sur un mur qu'on ne voudrait pas voir. Cette méthode est développée par Hirani et Totsuka [HT96]. Elle tient compte de l'image originale ce qui permet un résultat cohérent au niveau de l'éclairage. De plus, cette méthode semble permettre de travailler sur n'importe des types variés de texture, y compris les textures déterministes. L'algorithme modifie l'image bruitée en s'appuyant sur une image modèle. Les deux images sont dans une premier temps transformées par des opérations de transformation de Fourier (FFT). Les amplitudes et les phases ainsi déterminées vont servir à créer une nouvelle image, avec une transformée de Fourier Inverse (IFFT), où le bruit sera diminué. Cette nouvelle image est ensuite composée avec l'image originale, où un masque binaire détermine les régions où les pixels de l'image générée doivent être utilisés. Cette image composée sert ensuite de nouvelle entrée à l'algorithme itératif. Les exemples décrits par les auteurs utilisent typiquement 10 itérations pour enlever le bruit. Pour des textures déterministes, l'image contenant le défaut est découpée, et des échantillons adaptés à chaque découpage sont choisis. Les régions à modifier et les échantillons modèles associés sont choisis à la main. Cet algorithme présente l'avantage sur le précédent de pou-

voir travailler sur des types variés de texture, et d'enlever des défauts même à cheval sur différentes surfaces. De plus, la composition permet de conserver les aspects d'éclairage, s'il est similaire dans le reste de l'image. Par contre, cet algorithme ne peut fonctionner que si les défauts sont de petite taille par rapport à l'image. Cet algorithme pourrait éventuellement être appliqué pour étendre l'enlèvement d'objet dans n'importe quelle zone de la scène, où les textures seraient plus variées.

### 5.5.3 Modification de l'éclairage

Nous avons auparavant décrit les algorithmes nous permettant d'ajouter, de déplacer ou d'enlever des objets virtuels, et réels. Nous décrivons maintenant les algorithmes permettant de modifier l'éclairage, en changeant l'intensité des sources de lumière, ou en ajoutant des lampes virtuelles.

#### Modification de l'intensité des sources

Lorsqu'on modifie l'intensité d'une source de lumière, deux opérations doivent être réalisées :

- le calcul de l'éclairage indirect avec la nouvelle valeur d'exitance, pour mettre à jour le terme d'éclairage indirect $\hat{B}$ dans la structure de données. Pour cela, on utilise le système de radiosité puis on fait un rendu dans un tampon mémoire, comme précédemment (voir section 5.4.3). Comme il n'y a aucun changement de visibilité, seules des itérations de « gather » et « push–pull » sont réalisées par le système de radiosité (voir la section 2.2.1 à propos de la radiosité). Ce calcul est très rapide étant donnée la subdivision grossière choisie.
- le changement de l'exitance de la source en question dans la structure de données, servant à calculer l'éclairage direct.

Le temps mis pour la modification dépend de la taille de l'image et de la rapidité de la solution de radiosité. Il n'y a aucun calcul supplémentaire en ce qui concerne la visibilité ou la structure de la solution de radiosité.

#### Addition de sources de lumière virtuelles

Lors de l'insertion d'une source de lumière, il y a plus d'opérations à faire. Il faut d'abord modifier la structure de liens dans le système de radiosité hiérarchique, et créer les « shafts » associés aux nouveaux liens. Ceci nous permet à la fois de calculer un éclairage indirect tenant compte de la nouvelle lampe virtuelle, mais également d'avoir une structure efficace pour faciliter les calculs de visibilité.

Comme lors de l'insertion d'un objet virtuel, un test est réalisé pour connaître les pixels visuellement modifiés, en projetant la boîte englobante de la source sur la fenêtre de vue. Très souvent dans nos scènes, la source de lumière ajoutée est hors du champ visuel.

En ce qui concerne l'éclairage, pour chaque pixel, chaque champ de la nouvelle structure de données associée à la nouvelle lampe $s$ est mis à jour :

- l'exitance $E_s$,
- la visibilité $V_s$ entre la source et le point 3D associé à ce pixel. Pour cela, nous utilisons les « shafts » créés lors de la solution de radiosité. Si un shaft ne contient aucun polygone dans sa liste de polygones intersectés, il n'y a pas de calcul de visibilité à faire et le facteur de visibilité est mis à '1' (complètement visible). Sinon, un calcul de visibilité est réalisé par un lancer de rayons entre la source et le point 3D. Comme nous sommes dans une session d'interaction, nous n'utilisons que quatre rayons, ce qui introduit de « l'aliasing » dans la pénombre. Nous limitons les calculs d'intersection entre rayons et polygones aux polygones de la liste associée au « shaft ».
- le facteur de forme $F_s$ entre la source $s$ et le point 3D associé au pixel. Ce facteur de forme est calculé comme lors de la phase d'initialisation.

Une nouvelle structure de lampe est ensuite ajoutée dans la liste des structures associées aux lampes dans la structure de données d'affichage (et éclairage direct) (voir la figure 5.11).

## 5.6 Résultats

Nous avons implanté et testé ces algorithmes interactifs sur plusieurs scènes, sur une station de travail *SGI Onyx2 Infinite Reality*, avec un processeur R10000, à 195 Mhz. Les résultats sont montrés sur des vidéos accessibles sur internet[‡]. La première scène testée correspond à celle présentée tout au long du chapitre. L'image originale est présentée sur la figure 5.22 (a), la reprojection virtuelle simulée par notre programme est en (b). Cette image est de 400 par 300 pixels. Nous avons tout d'abord enlevé l'ordinateur portable posé sur un meuble à tiroir en (c). Cet enlèvement d'objet prend 1.2 secondes. Ce temps ne prend pas en compte la génération de texture qui a été réalisée en pré–calcul (30 secondes pour chaque texture générée). En revanche, les masques pour remplacer l'ancienne réflectance par la nouvelle sont calculés dans cette phase d'interactivité. Nous avons ensuite inséré deux objets, une sphère (polyèdre) en (d) comprenant 10 polygones en 1.54 secondes, et une cafetière en (e) comprenant 200 polygones en 5.65 secondes. La cafetière correspond à un objet réel reconstruit auparavant en utilisant le système de reconstruction à base d'images *Rekon* [POF98]. Nous l'avons placée à l'endroit où se trouvait l'ordinateur portable. Nous avons ensuite déplacé la sphère, en (f) et (g). La mise à jour se fait en moins de 1 seconde par image. Le déplacement de la cafetière met un temps supérieur (3 secondes par image).

Sur la figure 5.23, nous avons ensuite ajouté une première lampe à droite dans la scène en (a), puis une deuxième à gauche dans la scène en (b). Chaque insertion de lampe met 12 secondes. Ce temps est supérieur à celui réalisé pour l'insertion d'un objet car nous devons calculer des facteurs de forme et des facteurs de visibilité pour tous les pixels de l'image. Enfin, nous avons éteint la lampe créant l'éclairage original en (c). L'extinction de la lampe se fait en 0.4 secondes. Le déplacement d'un objet comme la sphère (en (d) et (e)) met un peu plus longtemps avec les nouvelles lampes, typiquement 4.5 secondes, car une mise à jour des paramètres est nécessaire pour chacune des lampes.

[‡]http ://www-imagis.imag.fr/Membres/Celine.Loscos/relight.html

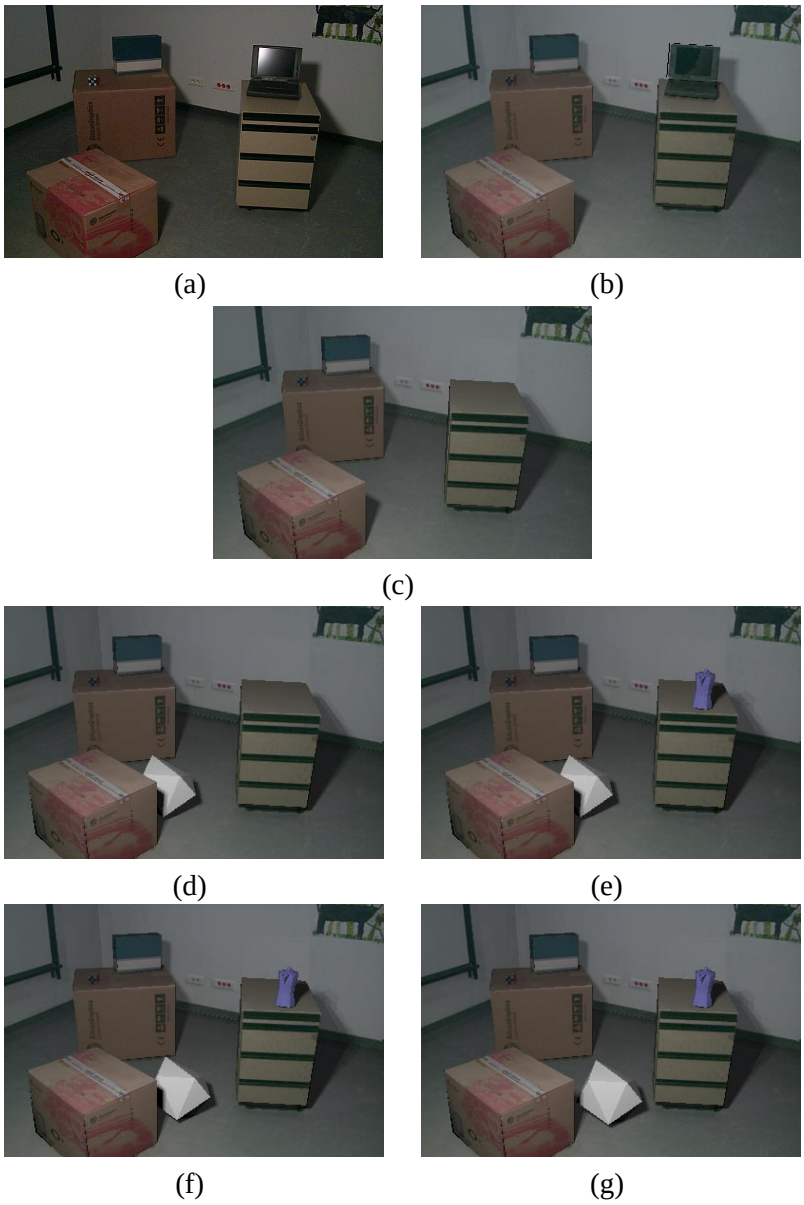
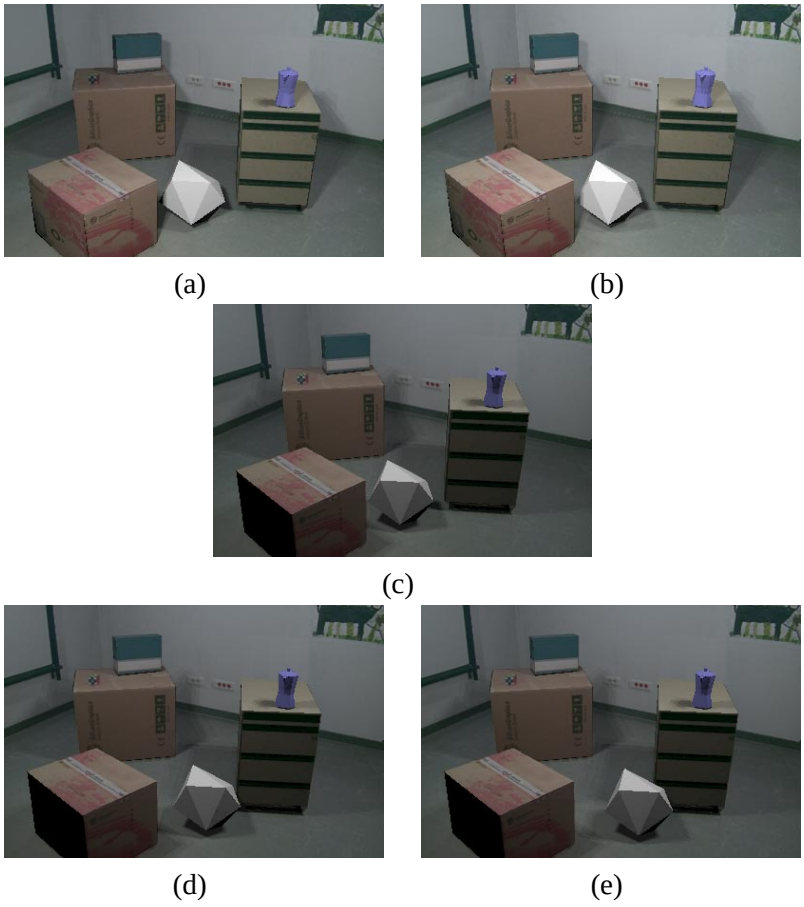



FIG. 5.22: Modifications de la géométrie pour la première scène testée.



(a) (b)

(c)

(d) (e)

FIG. 5.23: Modifications de l'éclairage pour la première scène testée.

Nous avons testé une autre scène dont l'image originale (512 par 480 pixels) est présentée par la figure 5.24 (a). La réflectance estimée pour cette scène est présentée en (b). Nous avons utilisé six positions de lampes différentes pour calculer cette image de réflectance. La simulation d'éclairage par le système est présentée en (c). A nouveau, cette image virtuelle est similaire à celle d'origine. Avec cette scène, nous voulions montrer que nous pouvions réaliser des modifications très difficiles à faire dans la réalité. Nous avons donc enlevé la porte virtuellement, en (d). Cette opération prend 3.3 secondes, sans compter le temps de génération de textures calculée antérieurement. Cette opération met plus longtemps que l'enlèvement du portable dans la scène précédente. La taille de la zone à traiter est en effet plus grande, de même que la taille de l'image. L'affichage sur cette scène prend 0.17 secondes (le temps d'affichage pour la première scène prend 0.1 secondes). Nous avons ensuite ajouté une chaise virtuelle en (e), ce qui a pris 5.37 secondes. Cette chaise comporte 185 polygones. Le temps de mise à jour est à nouveau supérieur à celui du temps d'insertion des objets dans la scène précédente, car la zone à traiter a une taille supérieure. Enfin nous avons inséré une lampe virtuelle dans cette scène en (f), projetant les ombres de la chaise sur le mur où se trouvait la porte. Le temps mis pour l'insertion est de 7.6 secondes.

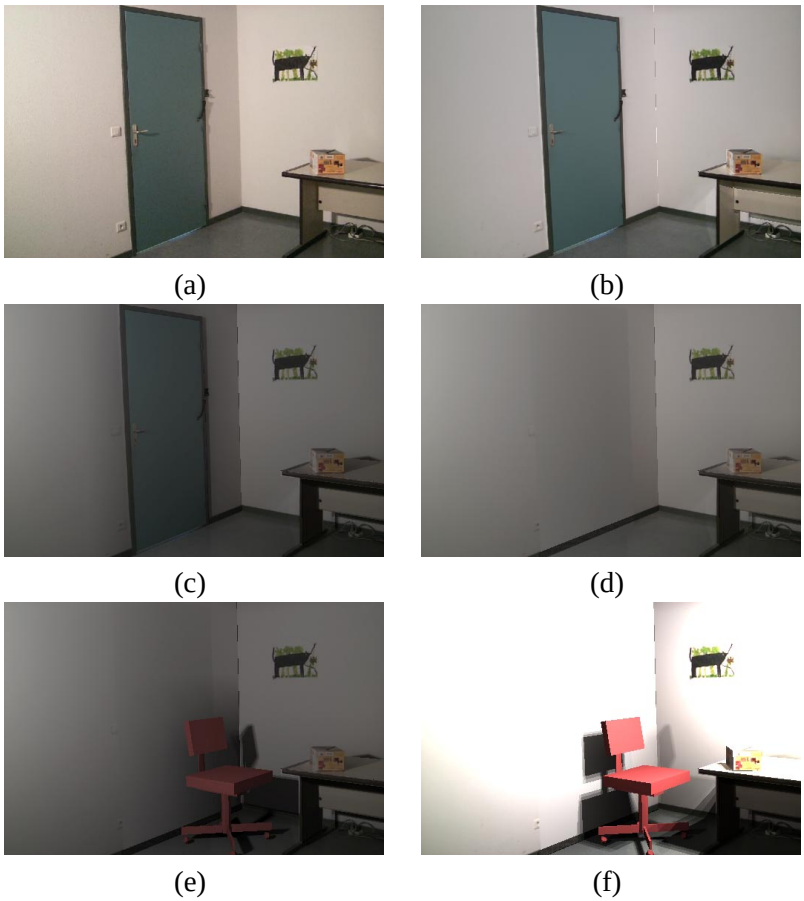



FIG. 5.24: Résultats sur une deuxième scène testée.

## 5.7 Conclusion

Nous avons présenté dans ce chapitre une nouvelle approche pour ré–éclairer et modifier géométriquement les scènes réelles. Notre approche est basée sur un algorithme comprenant une phase de pré–calcul dans laquelle les propriétés de réflectance diffuses des surfaces réelles sont approximativement retrouvées. Nous utilisons pour cela une séquence de photographies, appelées images de luminance, prises sous un même point de vue, mais sous des éclairages différents. Après avoir reconstruit un modèle 3D simplifié de la scène réelle, les images de réflectance associées à chaque image de luminance sont créées en utilisant les informations géométriques fournies par le modèle, et les informations de couleur contenues dans les photographies. Ces images de réflectance contiennent des imperfections, notamment dans les régions d'ombre. Elles sont combinées en calculant une moyenne des pixels pondérée par la confiance que nous avons en chacun des pixels. Cette confiance a été calculée par un algorithme de filtrage, utilisant les valeurs de visibilité, et en détectant les valeurs singulières. L'image obtenue par combinaison des images de réflectance est une image de réflectance, dont les couleurs sont homogènes et les effets de lumières sont estompés.

Les valeurs de réflectance sont associées à chaque pixel d'une structure de données augmentée. Le rendu de l'éclairage se fait en combinant un lancer de rayons pour l'éclairage direct, et une méthode de radiosité pour l'éclairage indirect. Le rendu se fait pixel par pixel en utilisant la structure de données augmentée.

Lorsque le pré–calcul est terminé, nous pouvons modifier de façon réaliste la géométrie et l'éclairage de la scène réelle en temps interactif. Des algorithmes adaptés sont utilisés pour localiser efficacement les pixels affectés par les modifications.

Dans cette méthode nous avons levé le problème du traitement des régions d'ombre en les compensant par une meilleure connaissance de l'éclairage de la scène. Ceci nous a permis d'estimer les réflectances de façon plus précise que dans la première méthode présentée au chapitre 4. Les temps de mises à jour lors des modifications sont cependant équivalents ou plus lents. De plus nous sommes toujours restreints par un point de vue fixe. Par contre, les structures de données utilisées nous permettent d'enlever automatiquement les objets réels de la scène de façon convaincante. Cette méthode s'apprête plus à des extensions, et le contrôle pixel par pixel permet une estimation des paramètres et une simulation plus fines.

Nous pensons que l'estimation des réflectances pourraient être améliorée par l'utilisation d'images HDR, et par une itération sur le calcul de la réflectance, en utilisant au fur et à mesure un nouveau calcul de l'éclairage indirect. Nous discutons de ces améliorations dans le chapitre suivant.

# Chapitre 6

# Calibrage photométrique

Nous avons présenté dans les chapitres précédents deux nouvelles méthodes qui permettent, à partir de données simples, de ré–éclairer et de remodéliser *interactivement* des scènes réelles. Les données utilisées par ces systèmes sont des photographies numériques d'environnements intérieurs. Ces photographies sont utilisées pour retrouver un modèle géométrique de la scène réelle, ainsi que les propriétés radiométriques décrivant l'éclairage réel (perceptible sur ces photographies). Malgré la simplicité des données, les méthodes de ré–éclairage produisent des résultats réalistes et convaincants. La précision de l'estimation est suffisante pour que les résultats soient convaincants, dans le cadre d'une visualisation d'effets d'éclairages (sans mesures précises). La qualité des résultats peut cependant être améliorée avec un meilleur calibrage des données. Les valeurs de réflectance obtenues dans les deux chapitres précédents ne sont pas comparables directement avec des valeurs mesurées dans la réalité. Dans certains cas cependant, une meilleure estimation de la réflectance pourraient améliorer la qualité des résultats comme par exemple pour éviter des discontinuités lors de l'enlèvement d'un objet (voir l'exemple de l'enlèvement de la porte sur la figure 5.24(d)).

Dans ce chapitre, nous allons présenter plusieurs améliorations sur la qualité des données fournies par les appareils photographiques. L'application de ces nouvelles données à nos méthodes de ré–éclairage permet d'obtenir des résultats de meilleure qualité.

Nous proposons une nouvelle méthode de création des images de luminance à partir de photographies prises avec notre appareil semi–automatique (Kodak DC260). Cet appareil ne permet pas un contrôle manuel du temps d'exposition. Les images de luminance ainsi obtenues sont fiables à un facteur près. Pour corriger les valeurs de luminance, nous avons développé un algorithme de recalage des images, en nous basant sur les équations de la radiosité.

Nous utilisons ces images de luminance dans les méthodes de ré–éclairage présentées précédemment, ce qui nous permet d'obtenir des résultats de meilleure qualité car le contrôle de la couleur des pixels est plus fin.

Nous mesurons également les propriétés de la lampe utilisée pour la méthode présentée dans le chapitre 5. Ceci nous permet en particulier une meilleure estimation de la réflectance.

Grâce aux données calibrées, nous avons amélioré le processus d'estimation de la réflectance dans la méthode du chapitre 5 en utilisant un algorithme basé sur une mise à jour

itérative de l'éclairage indirect et des valeurs de réflectance. À la fin du processus, la réflectance diffuse obtenue est plus homogène et plus fiable.

## 6.1 Calibrage photométrique

La réalité contient des effets très complexes, et très différents les uns des autres. Lorsqu'on essaie de la capturer en la photographiant, on ne peut restituer tous les effets existants. Cette limitation est accentuée par le matériel utilisé. Les appareils photographiques transforment énormément les données qu'ils captent. Ceci provient d'une part de leur mécanisme, et d'autre part des fonctions qui y sont intégrées. Nous décrivons dans les sections suivantes les différentes propriétés observées pour l'appareil semi–automatique que nous avons choisi pour nos recherches.

### 6.1.1 Propriétés contrôlables et non contrôlables de notre appareil photographique

L'appareil photographique Kodak DC260 permet un contrôle limité de la capture. En particulier, nous ne pouvons pas contrôler le temps d'exposition, en dessous d'une demie seconde. En utilisant des scripts (voir Annexe B), nous pouvons contrôler un paramètre *EV*, prenant neuf valeurs possibles, entre -2 et 2. La valeur $EV = 0$ correspond aux paramètres choisis *automatiquement* par l'appareil, dont le temps d'exposition.

### 6.1.2 Glaire et vignettage

La réflexion de la lumière sur les lentilles ne suit pas forcément un chemin direct jusqu'au capteur. Un effet visuel facilement identifiable est l'effet de glaire, comme lorsqu'un rayon de soleil frappe une caméra et s'effiloche en gouttes de lumière. Un autre effet est ce qu'on appelle le vignettage. Les bords de la photographie sont plus sombres que le centre. Le phénomène est le plus visible lorsque l'ouverture du diaphragme est grande. Pour éviter ce phénomène, nous choisissons de travailler avec une ouverture de diaphragme petite.

## 6.2 Méthode pour créer des images de luminance à partir de photographies

Lors de la capture, nous nous sommes aperçus que les photographies numériques ne répondaient pas toujours à nos attentes. Le premier problème est la limitation de la gamme des couleurs, de valeurs de 0 à 255 pour chacune des composantes rouge, verte et bleue. Un deuxième problème bien plus gênant est la saturation des couleurs. Les zones sombres ou trop claires de l'image ne comportent pas des valeurs fiables pour les exploiter dans nos systèmes de ré–éclairage. Toutes les valeurs au–dessus de 255 sont fixées arbitrairement à 255, et au–dessous de 0.5 à 0.

Pour éviter la saturation des couleurs, il faut prendre des photographies à des temps d'exposition plus longs pour les régions sombres, et plus courts pour les régions claires. Une méthode [DM97], décrite ci–dessous, existe pour combiner plusieurs images d'un même

point de vue, prises à des temps d'exposition différents. La combinaison produit une image codée par des flottants variant de 0 à la valeur maximale des flottants. L'étendue est donc bien supérieure à l'intervalle utilisé pour les images RGB. Les couleurs sont bien représentées qu'elles soient sombres ou claires. Cependant, la mise en oeuvre de cette méthode exige l'utilisation d'un appareil « haut de gamme » avec des temps d'exposition contrôlables.

### 6.2.1 Méthodes existantes

Debevec et Malik [DM97] ont développé un algorithme pour convertir des données photographiques en *images de luminance*. Ces images sont codées en flottants. Plusieurs photographies sont prises depuis un même point de vue, mais à des temps d'exposition différents. La capture est réalisée par un appareil complètement manuel, qui permet de contrôler et de connaître l'ouverture du diaphragme et le temps d'exposition. Les images sont combinées en fonction du temps d'exposition qui leur est associé. L'image ainsi créée est appelée une *image de luminance*. Lors du traitement, l'algorithme déduit aussi la fonction de transformation de la caméra qui avait été appliquée sur ces images. En effet, pour obtenir une meilleure présentation visuelle des couleurs, la caméra transforme les couleurs qu'elle reçoit sur ces capteurs. Cette fonction est fixe pour une distance focale donnée. Les couleurs obtenues après l'application de la fonction inverse de la caméra, correspondent à l'éclairage réel. C'est pourquoi, en utilisant cette méthode, les valeurs des pixels sont assimilées à des luminances.

L'algorithme de Debevec et Malik [DM97] est basé sur le concept photographique de la réciprocité : l'exposition $X$ est définie par le produit de l'irradiance $E$ par le temps d'exposition $\Delta t$*. Pour l'image numérisée, les couleurs obtenues ont été transformées par la fonction de la caméra, et représentées par des pixels. Pour une couleur $Z$ d'un pixel, variant de $Z_{min}$ à $Z_{max}$, l'exposition $X$ est égale à $f^{-1}(Z)$. En combinant ces deux équations, pour un index $i$ dans l'image et un temps d'exposition indexé en $j$, on obtient :

$$f^{-1}(Z_{ij}) = E_i \Delta t_j. \tag{6.1}$$

En appliquant une transformation logarithmique, on obtient :

$$g(Z_{ij}) = \ln E_i + \ln \Delta t_j, \text{ avec } g = \ln f^{-1}. \tag{6.2}$$

Les deux inconnues de cette équation sont $g$ et $E_i$, si le temps d'exposition $\Delta t_j$ est donné lors de la capture. La couleur $Z_{ij}$ est récupérable directement sur les images numérisées. Pour déterminer ces deux inconnues, les auteurs utilisent une résolution aux moindres carrés. En particulier, le système d'équations linéraires est résolu avec une décomposition en valeurs singulières. Cette méthode de résolution est robuste, mais coûteuse en temps de calcul. Par conséquent, la résolution n'est faite que sur quelques pixels choisis par l'utilisateur. Les pixels choisis sont bien répartis dans l'image, et représentatifs d'une grande variété des couleurs. Pour obtenir la résolution du système, il faut choisir $N$ pixels, tels que

*Nous conservons dans cette section les nomenclatures utilisées dans l'article de Debevec et Malik [DM97]

$N(P-1) > (Z_{max} - Z_{min})$, où $P$ est le nombre de photographies. Par exemple, si $Z_{max} = 255$, $Z_{min} = 0$, $P = 11$, $N$ doit être supérieur à 25.

Les solutions obtenues pour $g$ et $E$ sont à un facteur d'échelle près. Pour simplifier la résolution et fixer le facteur, une contrainte est ajoutée au système, telle que $g(Z_{mid}) = 0$, où $Z_{mid}$ est la valeur au milieu de l'intervalle $[Z_{min}, Z_{max}]$. De plus, on considère que les valeurs proches des bornes ne seront pas très fiables pour évaluer la fonction $g$. Une fonction de pondération $w$ est donc introduite dans le système :

$$w(z) = \begin{cases} z - Z_{min} & \text{pour } z \leq \frac{1}{2}Z_{min} + Z_{max} \\ \\ Z_{max} - z & \text{pour } z > \frac{1}{2}Z_{min} + Z_{max} \end{cases} \tag{6.3}$$

L'équation résolue aux moindres carrés par décomposition en valeurs singulières est :

$$O = \sum_{i=1}^{N} \sum_{j=1}^{P} \left( w(Z_{ij}) \left[ g(Z_{ij}) - \ln E_i - \ln \Delta t_j \right] \right)^2 + \lambda \sum_{z=Z_{min}+1}^{Z_{max}-1} \left[ w(z) g''(z) \right]^2 \tag{6.4}$$

où $\lambda$ est un terme permettant de contrôler le lissage de la fonction $g$.

Une fois que la fonction $g$ est retrouvée, on applique pour chacun des pixels $i$, l'équation suivante :

$$\ln E_i = \frac{\sum_{j=1}^{P} w(Z_{ij})(g(Z_{ij}) - \ln \Delta t_j)}{\sum_{j=1}^{P} w(Z_{ij})} \tag{6.5}$$

pour retrouver l'irradiance $E_i$ en chaque pixel.

Les valeurs de $g$ et $E$ ainsi déterminées peuvent servir de base pour recalculer une nouvelle solution. Le logarithme de la fonction inverse d'une caméra utilisée par les auteurs de l'article original, est montrée pour chacune des composantes rouge, verte et bleue dans la figure 6.1.

### 6.2.2 Adaptation de la méthode précédente au cas d'un appareil photographique semi–automatique

Nous voulons utiliser les images de luminance ainsi construites pour remplacer les images RGB que nous utilisions dans les deux méthodes présentées aux chapitres 4 et 5. Cependant, avec les propriétés de notre appareil photographique (Kodak DC260), nous ne pouvons pas utiliser cet algorithme tel quel car nous ne pouvons pas contrôler directement le temps d'exposition.

Nous avons résolu le problème en utilisant la fonction $EV$ de notre appareil. Nous avons déduit un temps d'exposition relatif à un temps donné $t$ pour $EV = 0$. Ces temps d'exposition sont pour chaque valeur de $EV$ :

| $EV$ | -2 | -1.5 | -1 | -0.5 | 0 | +0.5 | +1 | +1.5 | +2 |
|---|---|---|---|---|---|---|---|---|---|
| $\Delta t$ | t/4 | $t/2\sqrt{2}$ | t/2 | $t/\sqrt{2}$ | t | $\sqrt{2}t$ | 2t | $2\sqrt{2}t$ | 4t |

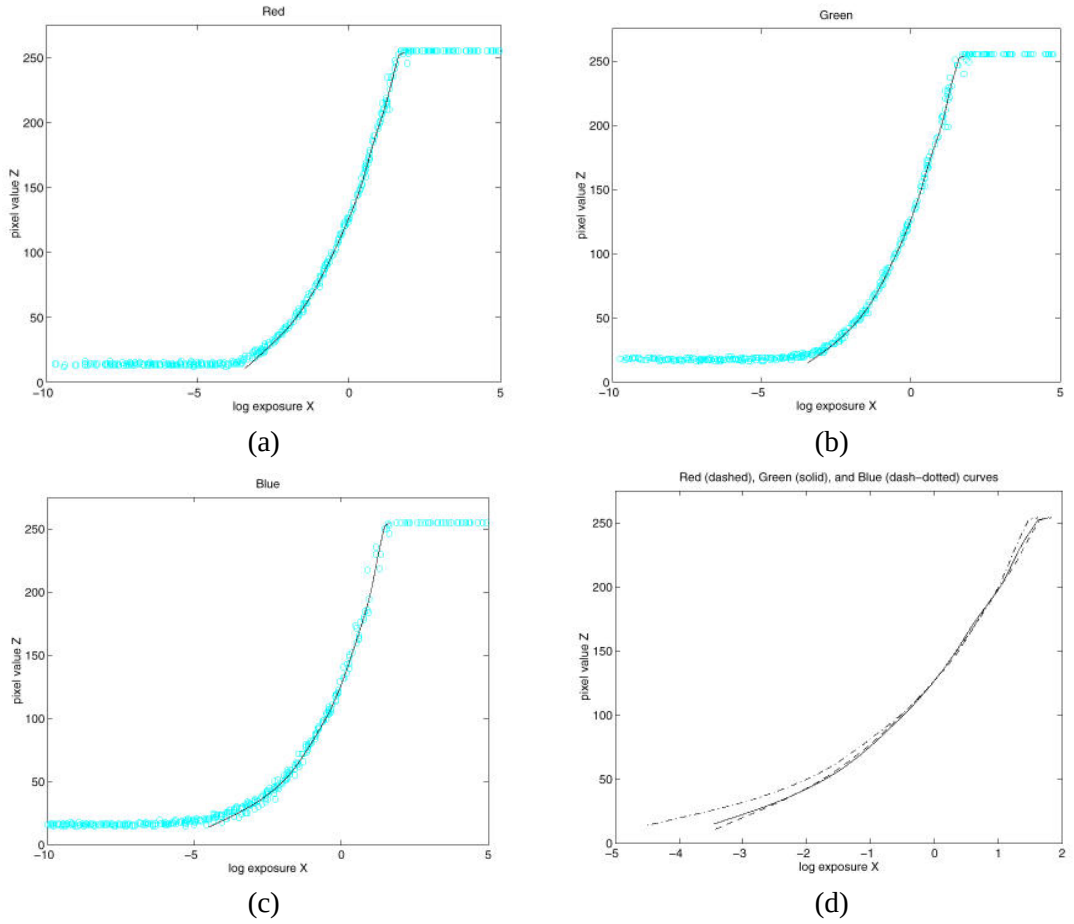



FIG. 6.1: Fonction retrouvée de la caméra de Debevec et Malik [DM97]. Le logarithme de la fonction inverse de la caméra est donné en abscisse et les couleurs des pixels (de 0 à 255) en ordonnée. (a), (b), (c) Fonctions de caméra pour chacune des couleurs rouge, verte et bleue. (d) Superposition des trois courbes.

Pour obtenir ces temps relatifs, nous nous sommes basés sur les principes photographiques en utilisant un facteur $2^{EV}$, entre chaque pas de temps d'exposition.

Pour obtenir les images pour chacune des expositions *EV*, nous avons écrit un script pour l'appareil donné en annexe B. Grâce à ce script, les photographies sont prises automatiquement. L'approche est la suivante. L'appareil est posé sur un trépied pour éviter tout mouvement entre chaque photographie. Nous fixons le zoom en angle maximal, et nous choisissons une ouverture de diaphragme. Le flash et l'équilibrage des couleurs sont désactivés. Ensuite, l'appareil prend automatiquement une photographie, en *EV* par défaut à zéro. Nous récupérons la valeur de la distance focale utilisée, et nous la fixons pour que l'appareil n'utilise pas une distance focale différente pour la série de photographies. Chacune des neuf photographies est prise avec un *EV* différent. L'appareil peut cependant par le mouvement des lentilles produire des images qui sont un peu décalées les unes des autres

(pas plus d'un pixel en général). Nous les alignons par une translation manuelle simple des pixels en prenant une photographie comme référence.

En utilisant les données adaptées comme nous les avons décrites ci–dessus, nous avons implanté la méthode de Debevec et Malik [DM97]. Les temps donnés en entrée du programme sont définis relativement à cette table de conversion où $t$ prend une valeur arbitraire. Nous avons typiquement utilisé $t = 10$. Pour calculer une fonction de la caméra, et tester notre algorithme, nous avons utilisé une série de photographies contenant beaucoup de contraste d'éclairage, présentée sur la figure 6.2.

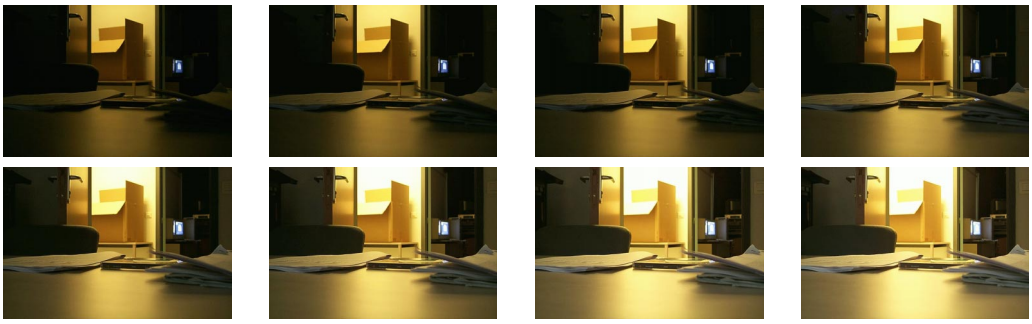


FIG. 6.2: Photographies qui nous ont servi à calculer la fonction de la caméra. Chacune correspond à un EV différent, de -2 à 1.5. Nous avons choisi ces photographies car elles contiennent beaucoup de contraste de lumière.

La fonction retrouvée de la caméra est pour chaque composante rouge, verte et bleue, comme nous le montrons sur la figure 6.3.

Cette adaptation souffre cependant de quelques limitations. La première provient du manque de contrôle. Nous n'avons pas la garantie que la plage de capture soit suffisamment étendue pour éviter les saturations, bien qu'elle soit meilleure que pour une seule image prise en $EV = 0$ (par défaut).

Une deuxième limitation est que les valeurs d'irradiance $E$ ne sont pas cohérentes d'une image à une autre, lorsque les conditions d'éclairage changent. En effet, l'appareil photographique choisit le temps d'exposition en $EV = 0$ pour chaque nouvelle photographie. Comme nous l'expliquons dans la section 6.3.2, nous avons développé une méthode pour recaler les images entre elles. Cette amélioration est nécessaire pour la méthode de ré–éclairage présentée dans le chapitre 5, où les conditions d'éclairage changent. Nous avons ainsi adapté l'algorithme de création d'images de luminance à notre appareil photographique numérique, dont le contrôle est restreint. Les résultats obtenus sont suffisamment bons pour être utilisables dans nos systèmes. L'utilisation ainsi que l'adaptation adéquate pour chacune des méthodes de ré–éclairage sont présentées dans les sections suivantes.

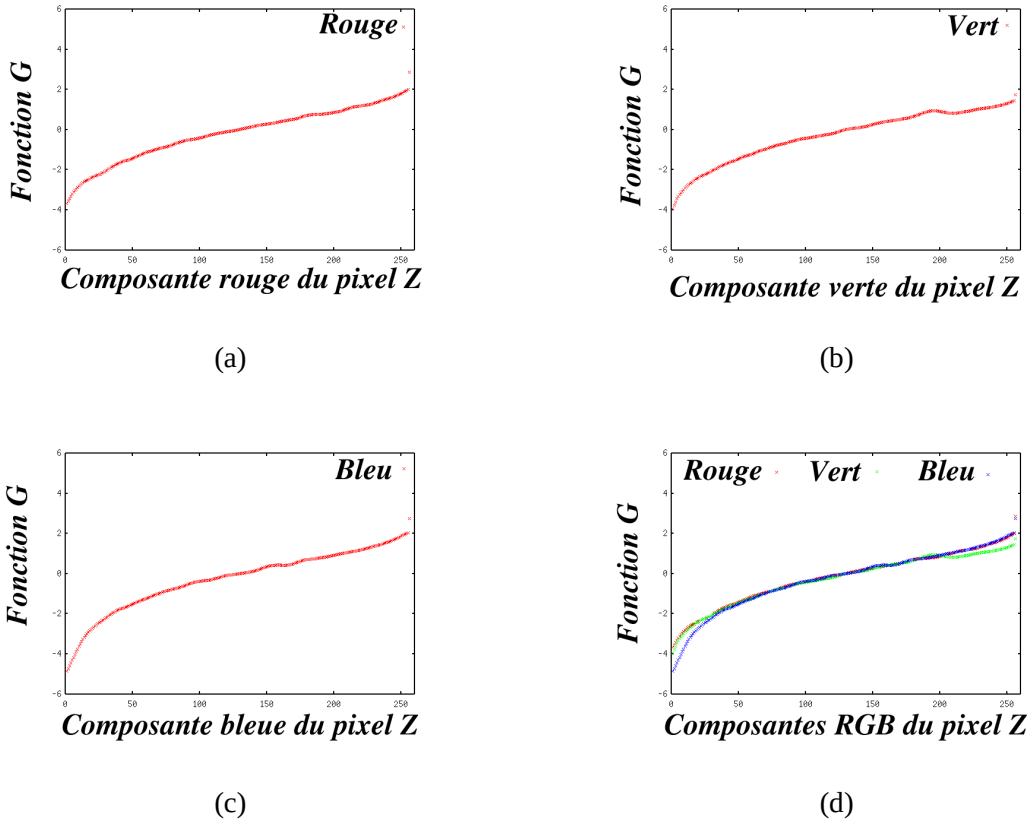



FIG. 6.3: Fonction de la caméra que nous avons retrouvée. Les couleurs des pixels sont données en abscisse, le logarithme de la fonction inverse de la caméra est en ordonnée. (a), (b), (c) Courbes de la fonction de la caméra respectivement pour les couleurs rouge, verte et bleue. (d) Superposition des trois courbes.

## 6.3 Utilisation d'images de luminance pour les méthodes de ré–éclairage développées

Nous avons utilisé les images de luminance dans les méthodes de ré–éclairage que nous avons présentées précédemment. Pour les deux méthodes, l'utilisation de ces images ont impliqué certaines modifications que nous expliquons dans la suite. Nous comparons ensuite les nouveaux résultats obtenus avec ces images de luminance, à ceux obtenus avec les images RGB.

### 6.3.1 Application à la méthode basée sur la connaissance d'un seul éclairage

Nous avons utilisé des images de luminance pour la première méthode présentée (voir le chapitre 4). Notons que dans ce cas, nous n'avons pas besoin de recalage des images

car les conditions d'éclairage ne changent pas. Dans la méthode initiale, nous utilisons un modèle texturé de la scène réelle, dont les textures sont extraites d'une photographie RGB. Ensuite, en utilisant les équations de radiosité, nous transformons les textures pour qu'elles représentent un éclairage calculé sans occlusion.

Dans l'approche modifiée, nous avons utilisé une image de luminance plutôt qu'une image RGB pour extraire les textures. Pour cela, nous avons modifié l'implantation de l'outil de reconstruction géométrique à base d'images *Rekon* (voir la section 3.2.2). Les textures extraites sont elles aussi des images de luminance.

Comme précédemment, le système est initialisé en utilisant la couleur contenue dans les textures. Les textures sont représentées par des flottants. Nous transformons ces textures pour qu'elles ne tiennent plus compte des occlusions (voir la section 4.3). Les textures modifiées sont montrées sur la figure 6.4 (b). En (a), ce sont les textures RGB modifiées. L'amélioration est globalement légère. Cependant, en regardant sur des endroits plus précis (en (c) et (d)), il y a beaucoup moins d'artefacts sur les textures de luminance modifiées que sur les RGB.

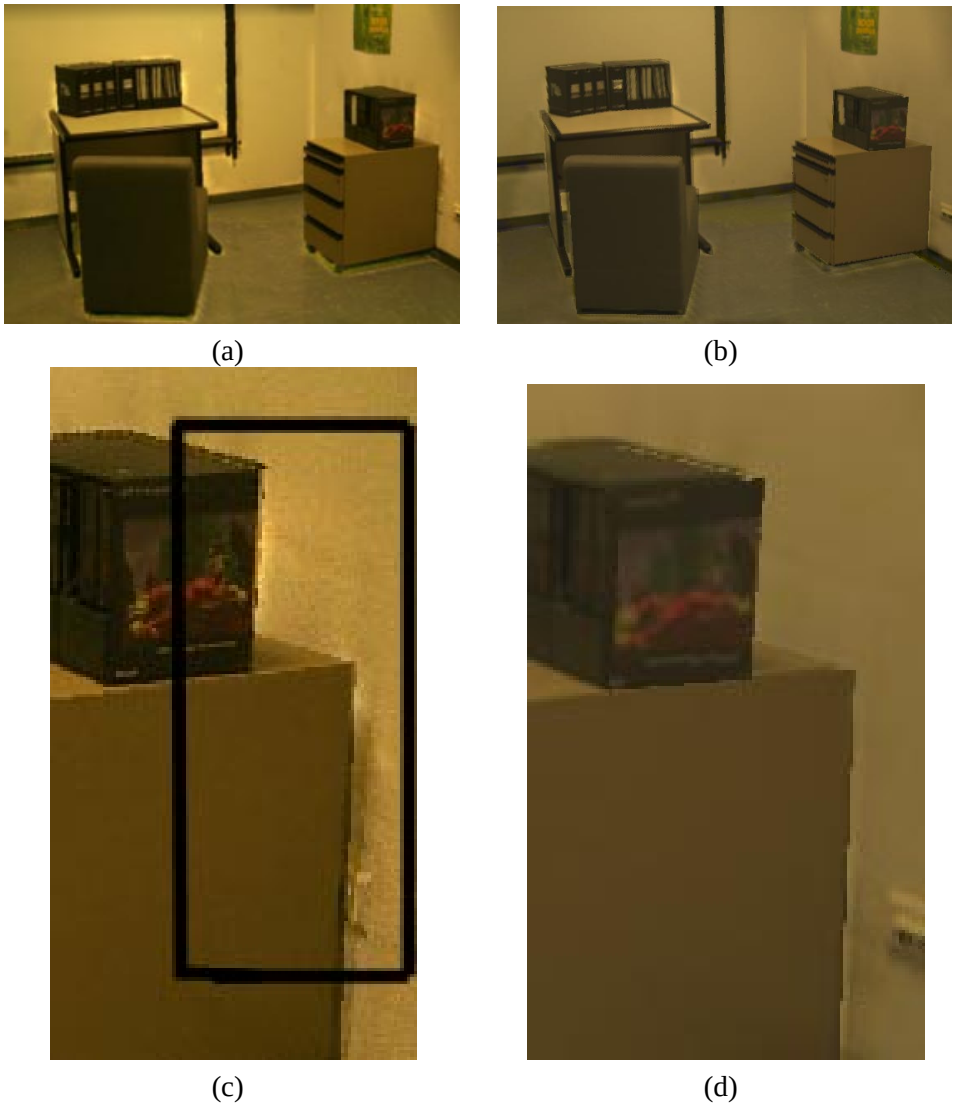
Nous avons comparé les résultats de ré–éclairage et de remodélisation. Nous avons appliqué un gamma correctif sur les résultats obtenus par les images RGB et par les images de luminance. Ils sont montrés sur la figure 6.5. Sur la colonne de gauche, ce sont les photographies originales. Sur la colonne de droite, ce sont les résultats obtenus avec les images RGB. Sur la colonne du droite, ce sont les résultats obtenus avec les images de luminance.

Pour montrer les différences entre les images simulées avec des textures RGB et celles simulées avec des textures de luminance, nous avons fait un zoom sur des images présentées sur la figure 6.5. Les images (a) et (b) de la figure 6.6 correspondent à un éclairage simulé avec une seule lampe allumée à droite. Les images (c) et (d) correspondent à un éclairage simulé avec une seule lampe allumée à gauche. Comme le montrent ces images, le ré–éclairage est de bonne qualité pour les deux utilisations de textures. Les ombres dues aux lampes réelles sur la photographie originale ont disparu. Cependant, le ré–éclairage simulé avec des textures de luminance contient un peu moins d'artefacts.

### 6.3.2 Application à la méthode basée sur la connaissance de plusieurs éclairages

Nous avons également amélioré la méthode de ré–éclairage basée sur plusieurs éclairages connus, présentée au chapitre 5. Cette méthode utilise en entrée une série d'images qui sont des photographies prises depuis le même point de vue, mais sous des éclairages différents. Nous avons réalisé plusieurs modifications lors de l'adaptation de cette méthode aux images de luminance.

Nous avons d'abord mesuré l'intensité de la lampe utilisée grâce aux images de luminance. Cette intensité est donnée en entrée au système lors du calcul de la réflectance. Ensuite, nous avons pris de nouvelles photographies pour obtenir une série d'images de luminance qui servent de nouvelles données au système. Comme noté précédemment, les images peuvent être incohérentes entre elles, car l'appareil s'adapte automatiquement à chaque nouvel éclairage. Nous avons développé une nouvelle méthode pour recalibrer les images entre elles. Enfin, nous améliorons les valeurs de réflectance en itérant sur l'estimation de la réflectance, en utilisant un éclairage indirect calculé à partir d'une solution



(a) (b)

(c) (d)

FIG. 6.4: Nouvelles textures corrigées à partir d'images de luminance. (a) Texture RGB corrigée représentant un éclairage sans occlusion. (b) Texture de luminance représentant un éclairage sans occlusion. (c) et (d) Zoom sur les images (a) et (b). La correction sur l'image de luminance en (d) contient moins d'artefacts qu'en (c).




FIG. 6.5: Résultats de ré–éclairage en utilisant des images de luminance. Dans la première colonne à gauche, ce sont les photographies de la scène réelle. Dans la colonne du milieu, ce sont les résultats de modifications d'éclairage simulées à partir de textures RGB. Dans la colonne de droite, ce sont les résultats de modifications d'éclairage simulées à partir de textures de luminance.

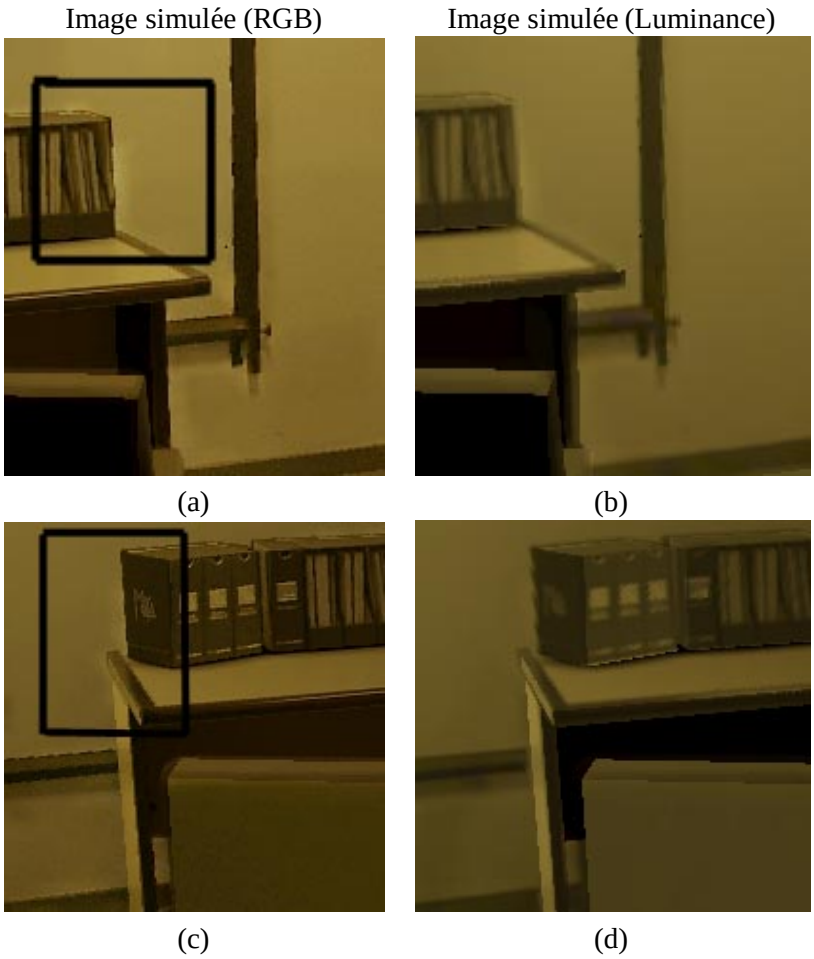



FIG. 6.6: Résultats de ré–éclairage en utilisant des images de luminance. Nous avons fait un zoom sur deux des éclairages modifiés présentés sur la figure 6.5 pour montrer les différences de résultats. (a) et (b) La scène est éclairée par une seule lampe à droite. (c) et (d) La scène est éclairée par une seule lampe à gauche.

de radiosité.

Nous décrivons dans les sections suivantes ces trois améliorations.

#### Mesure des propriétés de la lampe utilisée

Nous avons utilisé des images de luminance pour mesurer l'intensité lumineuse de la lampe que nous utilisons pour la capture d'images, comme précédemment. Nous montrons une des neuf photographies sur la figure 6.7.

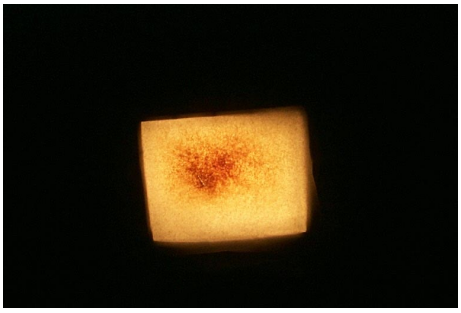


FIG. 6.7: Photographie de la lampe allumée ayant servi à la capture. Nous calculons à partir de cette photographie la distribution spectrale en RGB de son intensité.

À partir de l'image de luminance, nous calculons une moyenne des pixels correspondant à la zone couverte par la lampe (17.5x14 $cm^2$). C'est une lampe halogène de 500 Watts. L'exitance retrouvée pour la lampe en question est $(37.4584, 13.774, 3.40891)$. La composition du spectre de la lampe a donc une prédominance de rouge.

Nous avons également vérifié son aspect diffus. Pour cela, nous plaçons la lampe face à un mur blanc, sur lequel nous disposons des repères. Nous repérons notamment la projection du centre de lampe sur le mur. La photographie du mur est montrée sur la figure 6.8 (a). La capture d'images a été réalisée comme nous l'avons décrit dans la section 6.2.2, pour créer une image de luminance de cette photographie. La géométrie du mur a été reconstruite, et la position de la lampe mesurée. Nous utilisons cette géométrie pour enlever l'effet du facteur de forme sur l'image, en divisant la couleur de chaque pixel par le facteur de forme entre ce pixel et la source de lumière. L'effet que nous attendons est une texture de mur uniforme ne contenant plus les degradés de couleur dus aux facteurs de forme. Ce n'est pas exactement le résultat que nous obtenons par simulation (voir figure 6.8 (b)), bien que l'image semble plus homogène et l'effet des facteurs de forme atténué. Nous avons pris pour chaque image la ligne du milieu de l'image pour comparer les résultats. En (c), la courbe des pixels met en évidence l'effet de l'éclairage, et en particulier celui des facteurs de forme. En (d), les pixels correspondent à l'image corrigée. L'effet des facteurs de forme est enlevé, et les pixels sont plus alignés. Les quelques points s'éloignant de la courbure sont dues aux punaises posées sur le mur pour nous servir de repère. Ces différences de valeurs ne nous permettent pas de conclure que la lampe soit directionnelle. Nous estimons donc à partir de ces résultats que notre hypothèse sur le caractère diffus de la lampe est raisonnable.

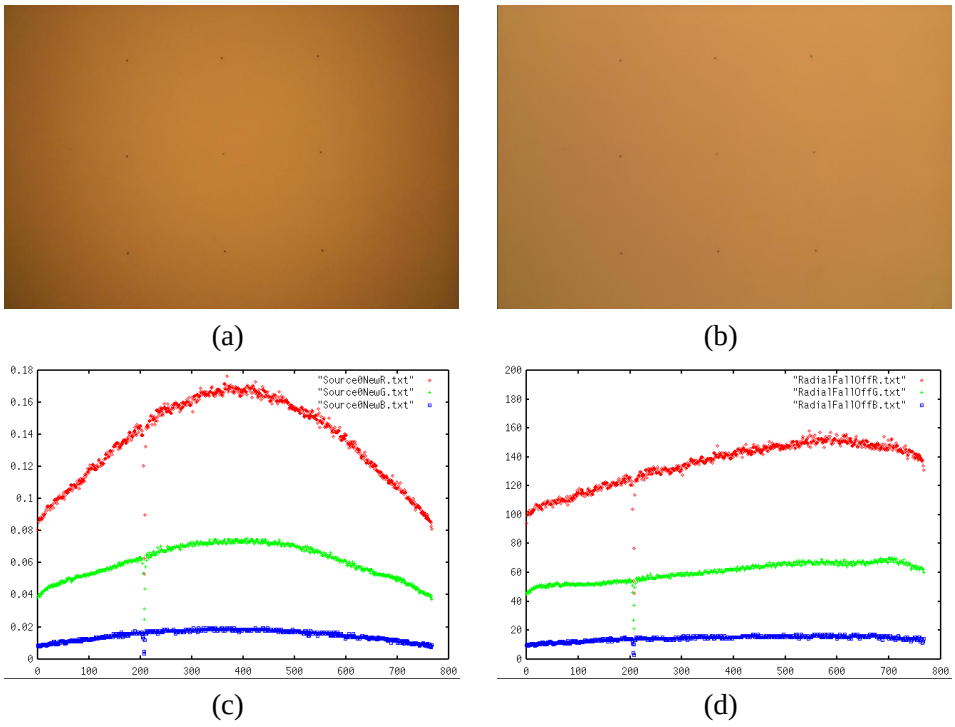



FIG. 6.8: Mesure de la propriété de diffusion de la lampe servant à la capture. (a) Photographie du mur éclairé de face par la lampe. (b) Nous avons enlevé les effets de facteurs de forme. L'image est plus homogène. (c) et (d) Variation des couleurs des pixels pour la ligne du milieu des images (a) et (b) respectivement.

### Recalage des images

Nous avons expliqué précédemment que les images de luminance ne sont pas cohérentes entre elles dans leur façon de représenter la luminance lorsque l'éclairage change, à cause des réglages automatiques de l'appareil (notamment la sélection automatique des réglages *EV*). Nous avons développé une nouvelle méthode qui permet de recalibrer les images entre elles, en appliquant un facteur de correction sur chacune des images de luminance. Pour calculer ce facteur de correction, nous utilisons les équations de la radiosité. L'algorithme général est décrit par la figure 6.9.

Nous partons du principe que pour une image de référence choisie, une réflectance estimée servira de référence pour les autres images. Pour chaque image $I_j$, la luminance $L_{ij}$ du pixel $i$ peut être définie par les équations de radiosité, et en particulier :

$$L_{ij} = \rho_{ij} F_{is_j} E_{s_j} + B_j \tag{6.6}$$

où $\rho_{ij}$ est la réflectance associée au pixel $i$ sur l'image $I_j$, $F_{is_j}$ est le facteur de forme entre la source $s_j$ et le point 3D correspondant au pixel $i$, $E_{s_j}$ est l'exitance de la source de lumière $s_j$, et $B_j$ est la radiosité indirecte. Comme les images sont prises sous le même point de vue, la réflectance $\rho_{ij}$ associée au pixel $i$ doit être la même pour toutes les images. De plus, la

**Algorithme de recalage**
Sélection d'une image de référence $I_1$
Pour toutes les images $I_j$, $j = 1..n$
Affectation des pixels visibles comme valides
Pour l'image de référence $I_1$
Calcul d'une réflectance de référence $\rho^{ref}$
Pour toutes les autres images $I_j$, $j = 2..n$
Pour tous les pixels valides $i$
nouvelle couleur de pixel $L_{ij}^{ref}$
Pour toutes les autres images $I_j$, $j = 2..n$
Calcul du facteur $a_j$ de minimisation entre $L_j^{ref}$ et $L_j$
Pour toutes les autres images $I_j$, $j = 2..n$
Application du facteur de correction sur tous les pixels
$L_{ij}^{new} = a_j L_{ij}$
Création de nouvelles images $I_j$, $j = 2..n$

FIG. 6.9: Algorithme de recalage.

source de lumière est la même dans toutes les images, mais elle est placée à des positions différentes. L'exitance $E_{s_j}$ est la même, quelque soit $j$.

La radiosité indirecte $B_j$ est calculée pour chacune des images. Nous partons du principe que les pixels dans les ombres ne sont éclairés que par de l'irradiance indirecte. La couleur observée dans les ombres est l'irradiance indirecte multipliée par la réflectance des objets. Nous allons donc plutôt considérer comme radiosité indirecte la moyenne des couleurs des pixels dans l'ombre. Pour chaque image $I_j$, la radiosité indirecte $B_j$ est mise à jour par cette moyenne. Ceci nous permet par rapport au terme ambiant employé dans la méthode de ré–éclairage du chapitre 5, d'avoir une pseudo–radiosité indirecte. Cette approximation de l'éclairage indirect est plus significative maintenant que nous travaillons avec des images de luminance.

En nous inspirant de l'équation 6.6, nous allons d'abord calculer une réflectance de référence, qui nous permettra de calculer une couleur de pixel de référence. La réflectance de référence est calculée sur une image qui servira de référence tout au long du recalage. Notons cette image $I_1$. Pour le calcul de recalage, nous préférons ne pas considérer les pixels dans les ombres, pour lesquels la précision de l'éclairage indirect est très importante pour la justesse des calculs. Nous pouvons sélectionner les pixels directement éclairés par les sources de lumière grâce à notre connaissance de la géométrie de la scène. Nous considérons les pixels ainsi sélectionnés comme *valides*. Ils sont montrés dans la figure 6.10 (a). Nous faisons de même pour les autres images en écartant également les pixels invalides sur la première image. Sur les images montrées en (b), (c), (d), (e), (f), (g) et (h), les pixels retenus sont les pixels qui ne sont pas noirs.

En utilisant l'équation 6.6, nous calculons une réflectance de référence pour chacun

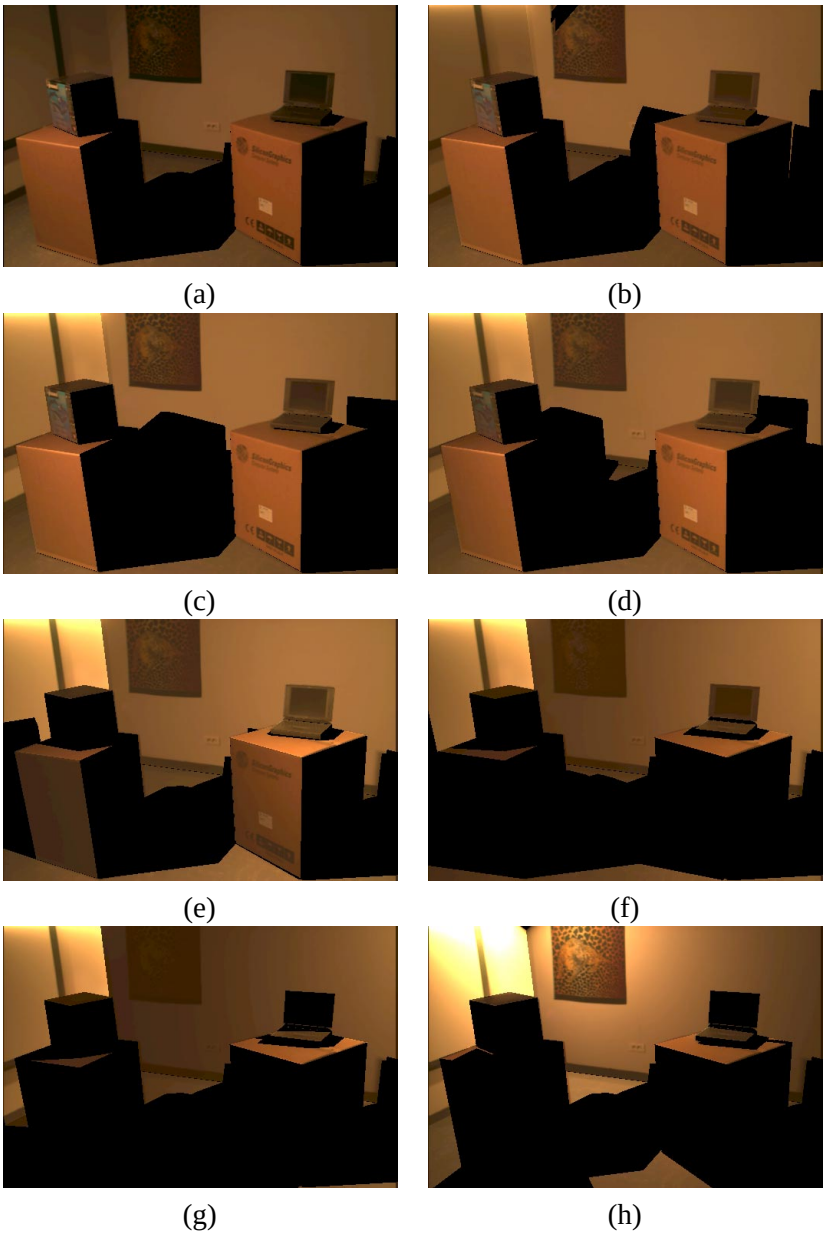



FIG. 6.10: Images obtenues après la première étape du recalage. Sur ces images les pixels considérés non valides sont montrés en noir. (a) Image de référence. (b), (c), (d), (e), (f), (g) et (h) Images correspondant à la couleur désirée pour les pixels valides.

des pixels $i$ sélectionnés sur l'image de référence $I_1$ :

$$\rho_i^{ref} = \frac{L_{i1} - B_1}{F_{is_1} E_{s_1}}. \tag{6.7}$$

La réflectance ainsi calculée pour les pixels valides est montrée sur la figure 6.11.



FIG. 6.11: Réflectance de référence calculée à partir de l'image de référence.

Nous utilisons cette réflectance pour calculer une nouvelle valeur de couleur du pixel pour chacune des images de luminance, autre que l'image de référence. Pour chaque pixel $i$, et pour chaque image $I_j$ ($2 \leq j \leq n$), nous utilisons l'équation suivante pour calculer une nouvelle couleur $L_{ij}^{ref}$ de pixel :

$$L_{ij}^{ref} = \rho_i^{ref} F_{is_j} E_s + B_j. \tag{6.8}$$

La couleur du pixel idéal est montrée pour chacune des images sur la figure 6.10.

À partir de cette nouvelle couleur de pixel, nous calculons la correction $a_j$ à appliquer pour que les images soient cohérentes entre elles. Cette correction est calculée en utilisant une résolution aux moindres carrés de :

$$\varepsilon = \sum_i \| a_j L_{ij} - L_{ij}^{ref} \|^2. \tag{6.9}$$

La correction $a_j$ est égale à :

$$a_j = \frac{\sum_i L_{ij} L_{ij}^{ref}}{\sum_i L_{ij}^2}. \tag{6.10}$$

Le facteur $a_j$ est ainsi calculé en utilisant les pixels considérés comme valides sur chacune des images $I_j$. Une fois ces facteurs obtenus, nous transformons les images de luminance en appliquant à chacun des pixels (valides et non valides) l'opération $a_j L_{ij}$. Les images obtenues sont recalées entre elles. Elles sont montrées sur la figure 6.12, avant et après le recalage. Nous ne montrons que cinq images des huit utilisées.

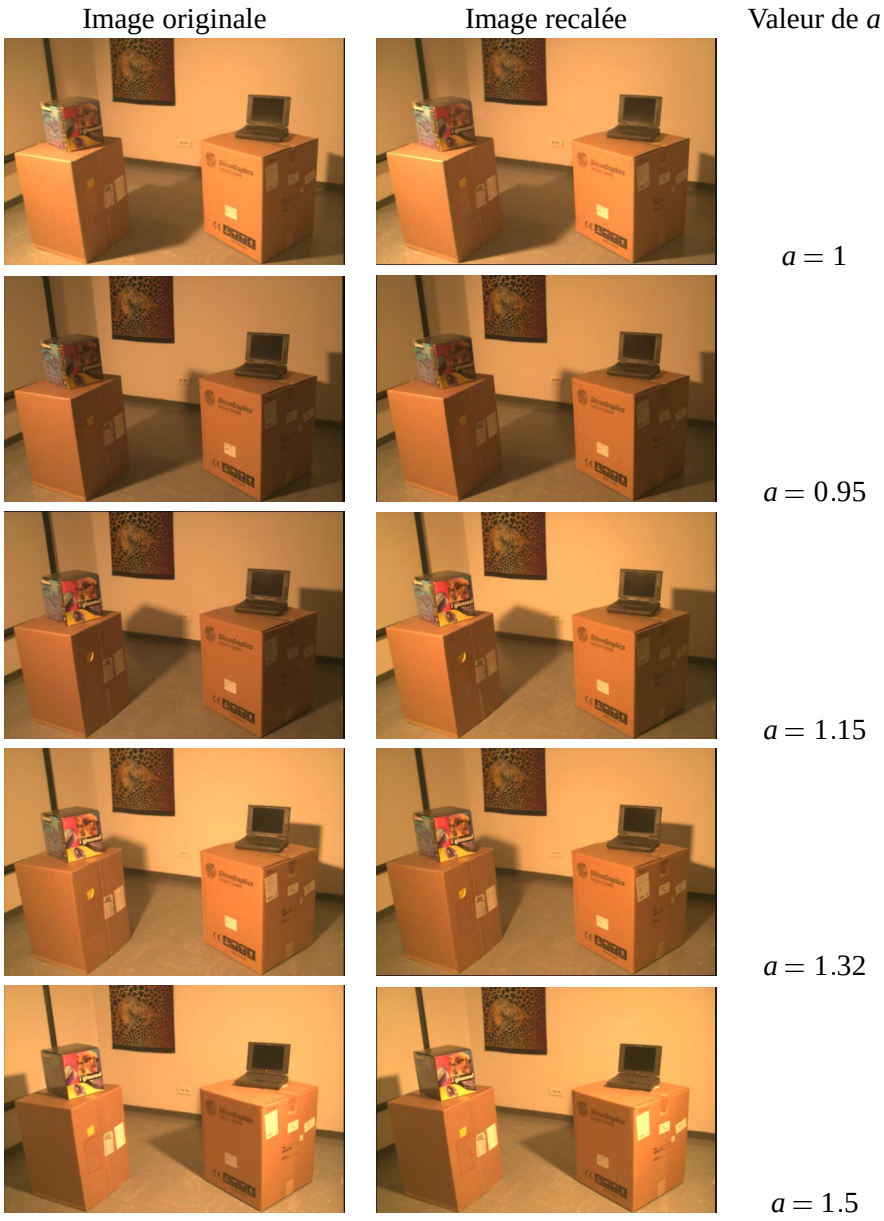



FIG. 6.12: Images originales et images recalées.

**Amélioration de l'estimation de la réflectance en affinant la solution par itérations**

En utilisant les images de luminance recalées, nous recalculons une nouvelle image de réflectance, avec le processus décrit dans le chapitre 5. Comme nous l'avons fait pour le recalage des images, nous utilisons un nouveau terme ambiant représentant l'éclairage indirect, différent pour chacune des images, et calculé à partir des pixels dans les régions d'ombre. L'image de réflectance obtenue après ce processus est montrée sur la figure 6.13 (b). Nous la comparons avec une image de réflectance en (a) obtenue avec des images RGB. En comparant une ligne de pixel sur la figure 6.13 (c) et (d), nous ne pouvons pas conclure que l'image de réflectance obtenue avec des images calibrées soit plus homogène que celle obtenue avec des images RGB. Cependant, on peut remarquer que le bruit a diminué. La ligne de pixels correspond au mur du fond. Les trois courbes sur chaque image correspondent aux composantes rouge, verte et bleue.

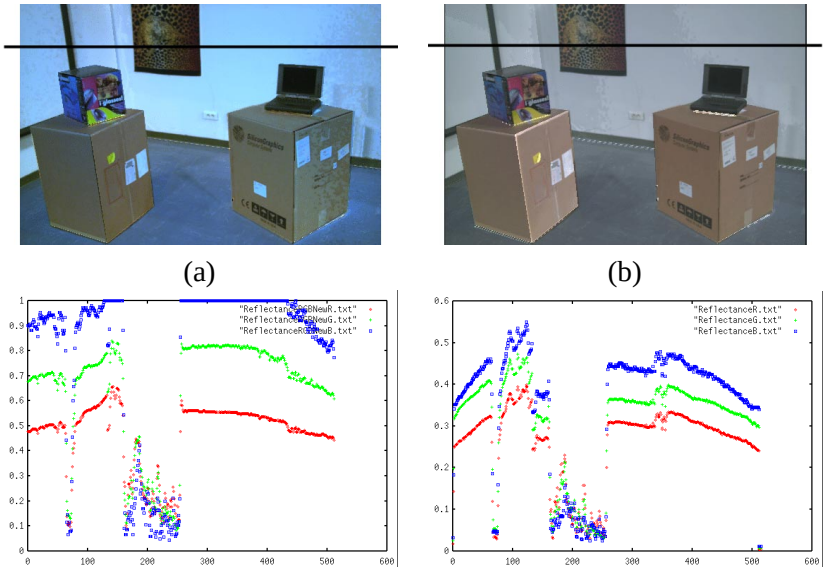



FIG. 6.13: (a) Réflectance estimée avec des images RGB. (b) Réflectance estimée avec des photographies calibrées. (c) et (d) Affichage des valeurs des pixels d'une même ligne (marquée d'un trait noir) sur les images (a) et (b). Les valeurs alignées en (c) correspondent à une saturation des valeurs (au–dessus de 255). Dans ces diagrammes, les indices des pixels sont en abscisse et les valeurs des couleurs en ordonnée.

Nous avons également comparé les variations pour chaque composante rouge, verte et bleue, d'une même ligne de pixels d'images de réflectance obtenues avec des images de luminance non calibrées et avec des images de luminance calibrées. Les deux variations de pixels sur l'image sont présentées sur la figure 6.14. Elles ne sont pas très différentes l'une de l'autre.

L'influence de l'éclairage indirect dans l'éclairage global est très importante. Cette importance se perçoit notamment dans les zones d'ombres qui sont également bien éclairée. Une meilleure précision de l'éclairage indirect est donc indispensable pour obtenir des

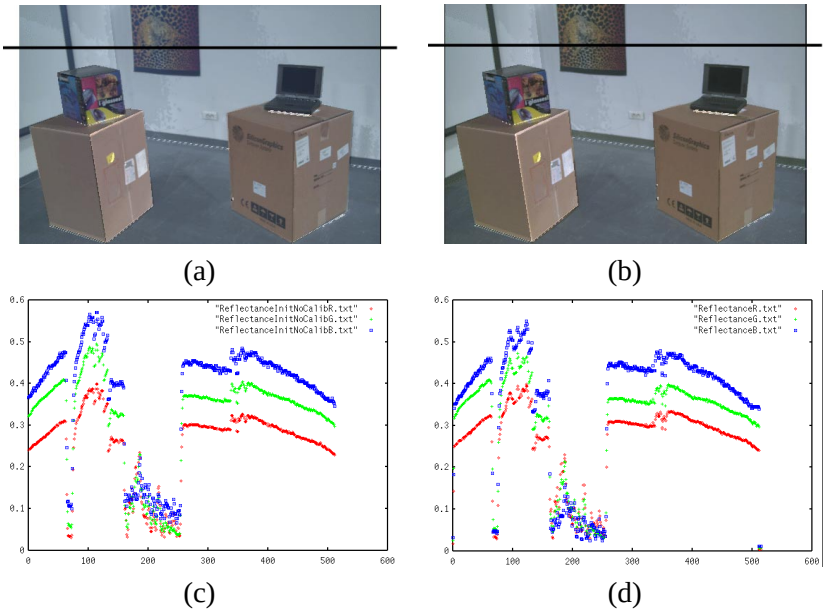



FIG. 6.14: (a) Réflectance estimée avec des images de luminance non calibrées. (b) Réflectance estimée avec des images de luminance calibrées. (c) et (d) Affichage des valeurs des pixels d'une même ligne (marquée d'un trait noir) sur les images (a) et (b). Dans ces diagrammes, les indices des pixels sont en abscisse, et les valeurs des couleurs en ordonnées.

meilleurs résultats dans l'estimation de la réflectance. Pour améliorer la solution, nous réalisons des itérations en mettant à jour successivement l'irradiance et la réflectance, comme nous le décrivons dans l'algorithme présenté sur la figure 6.15. Ceci nous permet d'affiner à la fois l'éclairage global et les valeurs de réflectance.

**Algorithme d'estimation des réflectances**
Estimation des réflectances avec un terme ambiant
Itérer
Pour $j = 0..NbImages$
Calculer l'éclairage indirect
Mise à jour des réflectances pour l'image $j$
Combinaison des réflectances individuelles
Mise à jour des réflectances dans le système de radiosité
Calcul d'un éclairage indirect pour la position $j = 0$ de lampe.

FIG. 6.15: Nouvel algorithme d'estimation des réflectances.

Le terme ambiant est remplacé par une valeur d'éclairage indirect différente pour chaque pixel, calculée par une solution de radiosité hiérarchique. Après quelques itérations, nous obtenons une nouvelle image de réflectance. Cette nouvelle réflectance sert à initialiser une dernière fois le système de radiosité. Un nouvel éclairage indirect est ensuite simulé pour la position de la lampe correspondant à la première photographie originale.

L'éclairage est ensuite affiché, et le processus interactif présenté au chapitre 5 peut être utilisé.

Les réflectances obtenues par ce processus itératif sont moins bruitées et ont des valeurs plus homogènes (moins de variations). La réflectance obtenue après dix itérations est montrée sur la figure 6.16 (b), en comparaison à celle obtenue avant les itérations en (a). Nous comparons les valeurs de réflectance en (c) et (d), pour une même ligne de pixels pour les images (a) et (b) respectivement. La réflectance obtenue après itération est visuellement plus homogène, ce qui est mis en évidence par la courbe de pixels en (d).

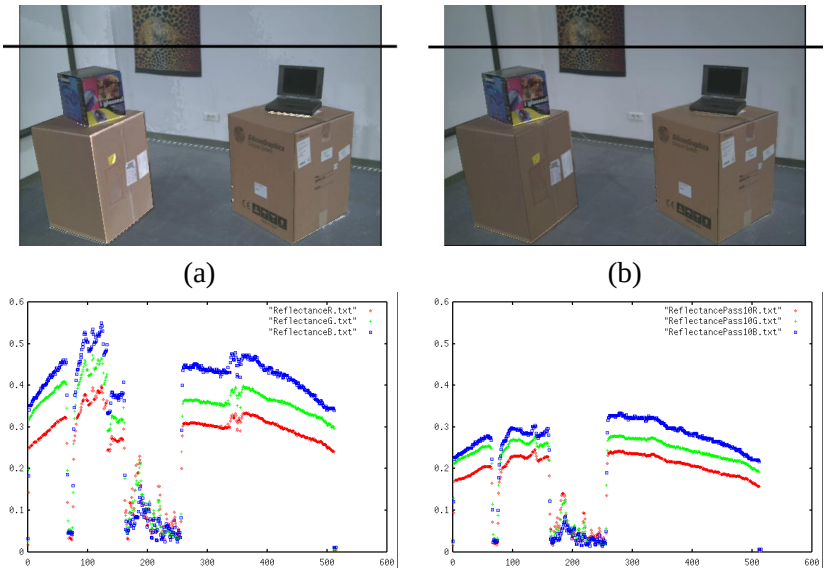
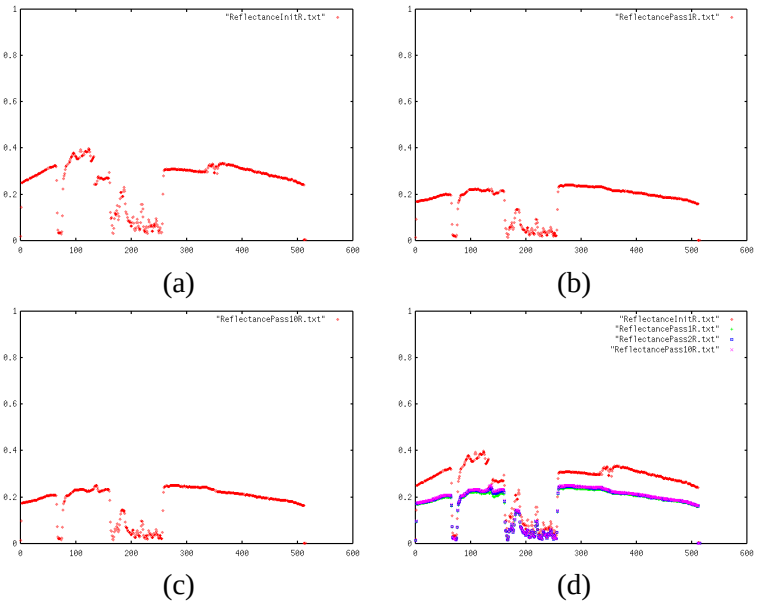



FIG. 6.16: (a) Réflectance estimée avec des images de luminance calibrées avant les itérations. (b) Réflectance estimée avec des images de luminance calibrées après dix itérations. (c) et (d) Affichage des valeurs des pixels d'une même ligne (marquée d'un trait noir) sur les images (a) et (b). Dans ces diagrammes, les indices des pixels sont en abscisse, et les valeurs des couleurs en ordonnée.

De plus, l'estimation de la réflectance est convergente. Dès la première itération, les valeurs de réflectance sont très proches des valeurs stabilisées dès la quatrième itérations. Nous montrons l'évolution des réflectances au cours des itérations sur la figure 6.17, pour la composante rouge. En (a), la courbe correspond à la réflectance avant les itérations. En (b), elle correspond à la première itération et en (c) à la dixième. En (d), nous avons superposé les courbes après chaque itération. La courbe la plus haute correspond à la réflectance avant les itérations ; la courbe la plus basse correspond à la première itération ; la courbe juste au–dessus correspond à la deuxième itération ; et la courbe encore juste au–dessus correspond à la dixième. Les trois dernières courbes sont très proches l'une de l'autre.

Enfin, nous montrons la reprojection de l'éclairage avec la réflectance obtenue après itérations sur la figure 6.18 (b). Elle est de meilleure qualité que la reprojection avec une réflectance obtenue avec des images RGB montrée en (a), car l'image de réflectance ini-




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

FIG. 6.17: Courbes d'évolution des valeurs de réflectance pour une même ligne de pixels au cours des itérations (voir le texte). Les indices des pixels sont en abscisse, et les valeurs des couleurs en ordonnées. Seule la composante rouge est montrée sur cette figure. (a) Première estimation de la réflectance. (b) Première itération pour l'estimation de la réflectance. (c) Dixième itération pour l'estimation de la réflectance. (d) Superposition des courbes. Les courbes convergent rapidement vers celle correspondant à la dixième itération.



(a) (b) (c)

FIG. 6.18: (a) Reprojection des ombres avec la réflectance calculée avec des images RGB. (b) Reprojection des ombres avec la nouvelle réflectance. (c) Modification interactive de l'intensité de la source de lumière originale.

tiale contient moins d'artefacts. En (c), nous avons modifié les composantes de l'intensité de la lampe, modifiant ainsi interactivement l'éclairage.

***Résultats sur une deuxième scène***

Nous avons appliqué l'algorithme de calibrage et le processus d'itérations sur des photographies montrées dans un exemple sur la figure 5.24 du chapitre 5. Ces photographies n'ont pas été prises à des temps d'exposition différents. Pour chaque position de la lampe, nous avons une seule photographie correspondant à un *EV* de zéro (choix automatique du temps d'exposition adéquat par l'appareil photographique). Pour chaque photographie, nous avons cependant créé une image de luminance. Même si nous n'obtenons pas une meilleure connaissance des couleurs, ces images de luminance ne contiennent plus la fonction de la caméra. Nous avons ensuite recalé les images avec l'algorithme de recalage présenté à la page 147. Enfin, nous avons estimé une première réflectance à partir de ces images de luminance calibrées. Cette réflectance est montrée sur la figure 6.19 (b). Après six itérations pour améliorer l'estimation de l'éclairage indirect, nous obtenons une nouvelle image de réflectance montrée en (c). L'image de réflectance estimée à partir d'images RGB non calibrées est montrée en (a). Nous montrons pour une même ligne de pixel l'évolution de l'estimation par des courbes montrées en (d), (e) et (f).

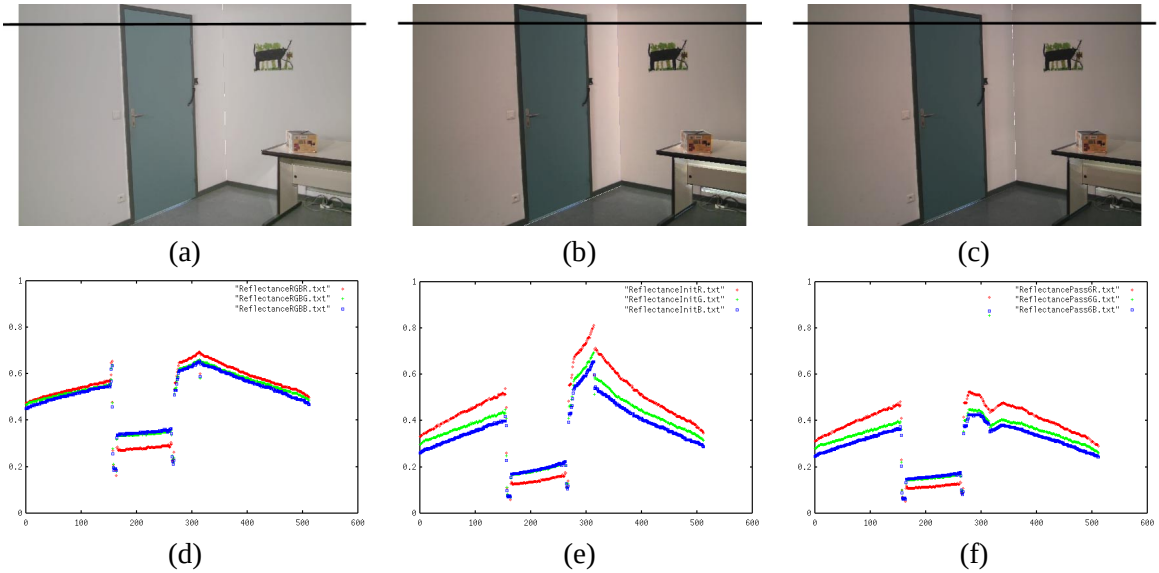



FIG. 6.19: (a) Estimation de la réflectance avec des photographies RGB. (b) Estimation de la réflectance avec des images de luminance recalées et sans itérations sur l'éclairage indirect. (c) Estimation de la réflectance avec des images de luminance recalées et avec itérations sur l'éclairage indirect. (d), (e) et (f) Courbe d'une même ligne de pixels (marquée en noir) respectivement pour les images (a), (b), et (c). Dans ces diagrammes, les indices des pixels sont en abscisse, et les valeurs des couleurs en ordonnée. Pour chaque image, nous montrons les trois composantes rouge, verte et bleue.



(a) (b)

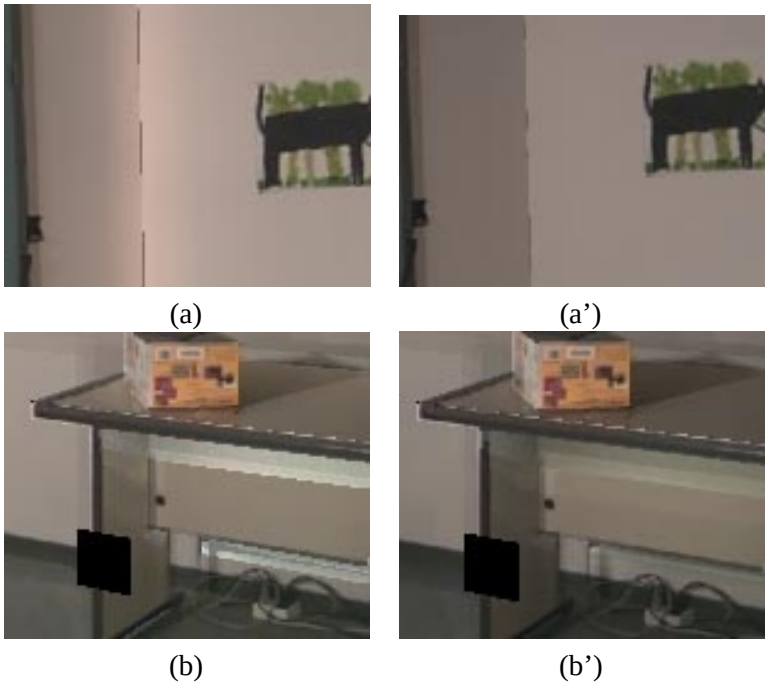
FIG. 6.20: Ajout d'une lampe virtuelle (en noir) pour éclairer le dessous du bureau. (a) Ré–éclairage en utilisant la réflectance estimée sans itérations (voir figure 6.19 (b)) (b) Ré–éclairage en utilisant la réflectance estimée sans itérations (voir figure 6.19 (c))

La réflectance estimée après le processus d'itérations en (c) est de bien meilleure qualité que celle estimée avant. Ceci se voit notamment dans les zones non directement éclairées lors de la capture des photographies. Par exemple, nous n'avons pas de connaissance de l'éclairage indirect sous le bureau. La réflectance est de moins bonne qualité à cet endroit sur les images (a) et (b).

Nous montrons l'importance d'une bonne estimation de la réflectance à cet endroit sur la figure 6.20. Dans cette figure, nous avons placé une petite lampe sous la table. Cette lampe est représentée par un polygone noir (car elle est vue de derrière). La zone éclairée est une zone où l'estimation de la réflectance est difficile. L'image 6.20 (a) correspond à un ré–éclairage en utilisant la réflectance montrée sur la figure 6.19 (b) ; on voit que l'utilisation d'un terme ambiant pour l'estimation de la réflectance est insuffisant. L'image (b) correspond à un ré–éclairage à partir de la réflectance montrée sur la figure 6.19 (c). Le ré–éclairage sur l'image (b) est plus réaliste. Les artefacts d'estimation de la réflectance en dessous du bureau sont diminués. Cependant, ils restent perceptible. Nous n'avions pas au départ une connaissance suffisante des couleurs à cet endroit de la scène. Nous n'avions en effet pas pris de photographies à des temps d'exposition différents. Le dessous du bureau est sombre sur toutes les images d'entrée, et les couleurs sont très bruités et nulles pour certains pixels. Sur la figure 6.21, nous montrons trois zoom sur ces images pour montrer la différence d'estimation. Nous montrons à gauche les images correspondant à une estimation de réflectance avant les itérations, et à gauche, les images correspondant à une réflectance estimée après le processus d'itérations. Sur les deux premières images (a) et (a'), on voit l'arrête du mur. La réflectance estimée pour l'image (a) était trop élevée ce qui provoque un éclaircissement non réaliste à cet endroit de l'image. Sur les images (b) et (b'), nous montrons l'estimation sur le dessous du bureau. Bien que le résultat soit meilleur en (b'), il reste des artefacts en–dessous de la table. Par contre, le mur perçu en–dessous du support du bureau est mieux estimé.

### 6.3.3 Rendu photographique

Le rendu obtenu à partir des images de luminance semble sans contraste. Pour rendre de façon plus contrastée, il faudrait que nous appliquions des méthodes d'affichage plus



(a) (a’)

(b) (b’)

FIG. 6.21: (a) et (a’) Zoom sur le coin du mur. La réflectance à cet endroit a été surestimée pour l’image (a). La réflectance est de meilleure qualité après la processus d’itérations sur l’éclairage indirect. (b) et (b’) Zoom sur le dessous du bureau. L’estimation radiométrique en (b’) est de meilleure qualité sous le bureau, malgré quelques artefacts restant (voir le texte). On peut noter que l’éclairage simulé est de meilleure qualité en (b’) pour la région de transition entre le mur et le support du bureau.

adéquates. La solution la plus simple serait s'appliquer la fonction de la caméra pour rendre l'image tel que l'aurait fait l'appareil photographique [DM97]. Des méthodes de rendu perceptuel pourraient également être appliquées, comme un « tone mapping » [TR93, Sch94, LRP97, TT99], une correction gamma [War94a].

## 6.4 Discussion

Nous avons présenté dans ce chapitre une nouvelle méthode de calibrage de photographies fournies par un appareil dont le contrôle manuel est limité (notamment au niveau du temps d'exposition). Nous avons pour cela appliqué à nos données photographiques un algorithme de création d'images de luminance [DM97]. Nous avons ainsi montré que créer des images de luminance avec des appareils photographiques de « bas de gamme » est possible. Ces nouvelles données améliorent la qualité des résultats des deux méthodes de ré–éclairage présentées dans cette thèse.

Pour compenser les imprécisions dû à une mauvaise connaissance du temps d'exposition, nous avons développé un algorithme de recalage d'images nécessaire pour calibrer plusieurs photographies de différents éclairages utilisées dans la deuxième méthode de ré–éclairage (voir le chapitre 5). L'estimation des réflectance a également été améliorée grâce à une meilleure précision de l'éclairage indirect obtenue par un processus itératif. La qualité de l'estimation des réflectances après ce processus est améliorée. Nous pourrions évaluer cette qualité en comparant les enlèvement d'objets réels pour l'estimation des réflectance par images calibrées et non calibrées.

Nous avons montré dans les chapitres précédents 4 et 5 que des données simples non transformées permettaient d'obtenir de résultats dont la qualité était suffisante pour certaines utilisation. Cependant, l'utilisation d'images de luminance a permis pour les deux méthodes de ré–éclairage d'obtenir des résultats dont la qualité visuelle peut être meilleure, et dont l'estimation de la réflectance est plus précise.

# Chapitre 7

# Comparaisons et améliorations

Nous avons présenté dans cette thèse deux approches différentes pour ré–éclairer les scènes réelles, ainsi qu'une amélioration de ces méthodes par un meilleur calibrage photométrique.

Les méthodes présentées sont différentes l'une de l'autre. Nous comparons dans une première section 7.1 les avantages et les inconvénients de chacune, ainsi que les résultats obtenus. Nous proposons ensuite des algorithmes pour accélérer et étendre les possibilités d'interaction de la méthode basée sur plusieurs éclairages connus. Enfin, nous proposons une troisième méthode combinant les avantages de chacune des deux méthodes de ré–éclairage.

## 7.1 Discussion sur les méthodes présentées

Nous récapitulons dans cette section les différences entre les deux méthodes de ré–éclairage présentées dans les chapitres 4 et 5. Les avantages et les limitations de chacune des méthodes sont résumés dans le tableau 7.1.

### 7.1.1 Différences algorithmiques

Les deux méthodes ont les mêmes besoins au niveau de la reconstruction géométrique de la scène réelle. Par contre, la première méthode utilise une seule photographie dont des textures sont extraites. L'estimation des propriétés radiométriques ne se fait qu'à partir de ces textures, ce qui limite la connaissance à un seul éclairage de la scène réelle et un seul point de vue. L'éclairage utilisé est l'éclairage présent dans la scène réelle. La deuxième méthode utilise plusieurs photographies prises depuis un même point de vue et représentant des éclairages différents. Ces éclairages sont obtenus en déplaçant une lampe unique connue, en différentes positions. L'estimation des propriétés radiométriques se fait en combinant les différentes propriétés estimées pour chacune des images.

L'estimation des propriétés radiométriques est également très différente. Dans la première méthode, l'exploitation des données se fait de façon relative, et une réflectance très approximative est estimée. Les textures sont corrigées pour correspondre à un éclairage

| | Première Méthode | Deuxième Méthode |
|---|---|---|
| Données pour la reconstruction géométrique | identiques | identiques |
| Données pour l'estimation radiométrique | Une seule photographie de la scène réelle, dont l'éclairage n'est pas choisi. | Plusieurs photographies de la scène réelle sont prises depuis le même point de vue, et sous un éclairage connu différent. |
| Estimation des propriétés radiométriques | Estimation d'une réflectance approximative calculée pour chaque élément de subdivision à partir de textures corrigées. | Estimation plus précise d'une réflectance diffuse pixel par pixel. |
| Calcul de l'éclairage | Solution de radiosité hiérarchique optimisée. | Pixel par pixel pour l'éclairage direct, et une solution de radiosité hiérarchique optimisée pour l'indirect. |
| Affichage | Le matériel graphique est utilisé pour afficher chaque élément de radiosité texturé (affichage en multi–passes si nécessaire). | L'affichage pixel par pixel en utilisant l'information stockée par pixel dans une structure de données. |
| Modification de l'intensité des lampes | Algorithme de radiosité sans nouveau raffinement. | Algorithme de radiosité sans nouveau raffinement pour l'éclairage indirect. Mise à jour de l'éclairage direct pixel par pixel, sans nouveau calcul de visibilité. |
| Insertion de lampes virtuelles | Modification de la solution de radiosité hiérarchique par un algorithme de mise à jour. | Modification de la solution de radiosité hiérarchique pour la mise à jour de l'éclairage indirect. Calcul pixel par pixel de l'éclairage direct dû à la lampe insérée. |
| Modification de la géométrie | Insertion et déplacement d'objets virtuels. | Insertion, déplacement et enlèvement d'objets virtuels. Possibilité d'enlèvement d'objets réels. Le déplacement et l'ajout d'objets réels est relativement facile. |
| Qualité visuelle des résultats | Le ré–éclairage et l'insertion d'objets virtuels sont réalistes. | Le ré–éclairage et l'insertion d'objets virtuels sont réalistes. Cependant, les effets spéculaires ne sont pas simulés, ce qui manque au niveau du réalisme. |
| Temps de mise à jour | | |
| Ré–éclairage | 0.4 à 0.8 secondes | environ 0.4 secondes |
| Ajout d'objets virtuels | environ 1.5 secondes | 1 à 5 secondes |
| Ajout de lampes virtuelles | environ 8 secondes | 7 à 15 secondes |
| Déplacement d'objets virtuels | 0.3 secondes | de 1 à 3 secondes |

FIG. 7.1: Comparaison des deux méthodes de ré–éclairage.

indépendant des occlusions des objets réels entre eux. Cette correction est réalisée sur des éléments de subdivision, déterminés par un algorithme de raffinement qui combine la connaissance de la couleur des textures, et les valeurs de visibilité par rapport aux sources de lumière. Nous nous contentons d'une estimation approximative des réflectances et des valeurs de radiosité car l'affichage de l'éclairage est réalisé de façon relative à l'éclairage original. Les nouveaux éclairages créés sont une correction de l'éclairage original. La deuxième méthode estime au contraire une réflectance diffuse de façon plus précise et moins heuristique, pixel par pixel. L'affichage est ensuite réalisé pixel par pixel en utilisant la réflectance estimée pour simuler un nouvel éclairage. Les solutions d'éclairage sont complètement indépendantes de l'éclairage original. La deuxième méthode s'apparente plus à une méthode d'illumination inverse, où les propriétés de l'éclairage sont complètement retrouvées.

Les algorithmes interactifs sont aussi très différents. Les algorithmes de la première méthodes utilisent la radiosité hiérarchique [DS97]. Cette solution permet un rendu quasiment indépendant de la taille de l'image affichée. Le rendu est réalisé polygone par polygone en utilisant le matériel d'affichage graphique. La complexité d'affichage est ainsi réduite aux nombres d'éléments de subdivision de la radiosité. Au contraire, les algorithmes d'affichage et de modification de la deuxième méthode sont dépendants de la taille des images, et du nombre de pixels affectés par les modifications. L'interactivité n'est pas garantie pour une image de haute résolution et pour des modifications importantes dans l'image (insertion de lampes virtuelles, ou déplacement de gros objets).

Malgré les différences évoquées ci–dessus, la qualité des résultats est équivalente. La qualité du ré–éclairage est similaire. L'insertion d'un objet virtuel se fait dans des temps similaires. Par contre, le déplacement des objets virtuels est trois fois plus rapide pour la première méthode. L'insertion des lampes virtuelles peut également être plus rapide pour la première méthode, car l'algorithme crée de nouveaux liens avec la source de lumière et met à jour localement la radiosité. Les calculs de facteurs de forme et de visibilité sont ainsi limités aux éléments subdivisés. Dans la deuxième méthode, tous les pixels de l'image sont affectés par l'insertion d'une lampe virtuelle, et nécessitent une mise à jour individuelle. Pour chacun des pixels, de nouveaux facteurs de forme et de visibilité relatifs à la lampe virtuelle sont calculés.

En revanche, l'acquisition d'une image de réflectance dans la deuxième méthode permet de faire des modifications qui nous paraissent plus difficilement réalisables dans la première méthode, tel que l'enlèvement d'objets réels. La connaissance d'une texture privée des effets d'éclairage permet de générer de nouvelles textures, privées elles aussi des effets d'éclairage. Les zones masquées par les objets réels peuvent être remplies par ces nouvelles textures, et l'éclairage re–simulé à partir de ces nouvelles réflectances. L'effet obtenu est réaliste, et l'objet réel est complètement enlevé de la scène réelle. Nous expliquons dans la section suivante comment exploiter les avantages de cette image de réflectance pour permettre le déplacement des objets réels, ou l'insertion d'objets réels présents dans une autre scène. De plus, la structure de données stockées pixel par pixel, peut servir de base pour utiliser des algorithmes parallèles. La deuxième méthode devrait après ces améliorations obtenir des résultats plus rapides, et permettre une plus grande pos-

sibilité d'interaction.

Les deux méthodes ont l'inconvénient commun de n'estimer que des réflectances diffuses. La qualité des résultats s'en ressent, car les surfaces parfaitement diffuses sont très peu présentes dans les scènes réelles. Pourtant, le rendu par modulation de l'éclairage original dans la première méthode permet de conserver les effets spéculaires dus à l'éclairage originale. Bien que l'éclairage soit modifié, ces spécularités accentuent l'effet réaliste des scènes modifiées. Souvent, ces effets sont incohérents par rapport à l'éclairage simulé. Pourtant, l'œil humain ne perçoit pas toujours ces incohérences, et « croit » x à ces effets spéculaires, comme le montrent les résultats dans le chapitre 4. Dans la deuxième méthode, les effets spéculaires sont enlevés par le calcul combiné des réflectances diffuses. Comme nous ne simulons pas ces effets spéculaires, l'œil humain perçoit qu'il manque quelque chose pour que le réalisme soit complet. Une estimation grossière de la composante spéculaire pour chaque objet, en utilisant une méthode similaire à Yu *et al.* [YDMH99], pourrait améliorer le réalisme. Les effets spéculaires directs pourraient être affichés pixel par pixel. Le rendu cohérent d'effets spéculaires dans la première méthode serait beaucoup plus compliqué, à cause du rendu polygone par polygone.

### 7.1.2 Utilisation appropriée de chaque méthode

Un des grands avantages de la première méthode est qu'elle permet de travailler avec des données non ciblées, comme des données vidéos quelconques d'une pièce d'intérieur. Les différents points de vue sur les images peuvent servir à reconstruire un modèle réel, et pour un point de vue choisi, le ré–éclairage peut être réalisé. De plus, l'utilisation d'un algorithme de radiosité permet un affichage polygone par polygone, qui est rapide et quasi–indépendant de la taille de l'image affichée. Cette méthode pourrait très bien être utilisée dans des environnements de réalité virtuelle sur des grands écrans d'immersion (de type CAVE).

La deuxième méthode nécessite une capture plus spécifique, ce qui réduit l'utilisation de données non ciblées. Par contre, la réflectance est mieux estimée, et le contrôle dans le résultat de manipulation est plus fin. L'utilisateur peut manipuler la scène de façon plus précise, et a de plus amples possibilités de manipulations, comme l'enlèvement d'objets réels. Nous pensons aussi que cette méthode se prêtera plus facilement à des extensions comme la prise en compte des effets spéculaires ou le déplacement de point de vue (voir la section 8.2).

## 7.2 Amélioration de la méthode basée sur la connaissance de plusieurs éclairages

La deuxième méthode de ré–éclairage présentée comporte des avantages significatifs sur certains points. Le premier est la création d'une image de réflectance. Nous pouvons grâce à l'image de réflectance, déplacer des objets réels, en gardant un éclairage réaliste. Le deuxième avantage porte sur l'organisation des structures de données et des algorithmes, à

partir desquels nous pouvons développer des algorithmes de parallélisation, pour accélérer les mises à jour lors de la phase d'interaction.

### 7.2.1 Déplacement des objets réels

Nous pensons que nous pouvons facilement déplacer les objets réels de la scène. Nous savons déjà les enlever en remplaçant les trous créés par leur suppression par des textures adéquates. Nous savons également localiser de manière efficace les modifications d'éclairage dues au déplacement des objets virtuels. Cet algorithme peut s'appliquer aux objets réels. Cependant, nous ne savons pour l'instant traiter que les objets virtuels sans texture, alors que les objets réels en ont une. À partir de l'image de réflectance, nous extrayons une texture pour chacun des objets réels que nous voulons déplacer. Cette texture est détordue, par rapport au point de vue utilisé, et replaquée à la nouvelle position de l'objet. La structure de données associée à l'éclairage direct est mise à jour avec ces nouvelles valeurs de réflectance, comme pour les objets virtuels. Nous pouvons également étendre l'algorithme à l'insertion d'objets réels pris dans d'autres scènes.

Que ce soit pour le déplacement d'objets réels ou l'insertion d'objets réels, il faut cependant faire attention aux limites dues à la projection des textures. Comme les textures sont extraites depuis un point de vue donné, il ne faut pas trop s'en écarter, pour éviter les distorsions et les incohérences de perspective dans le cas où tous les objets de la scène ne sont pas modélisés. Si nous utilisons des textures indépendantes du point de vue, la qualité visuelle sera inférieure car ces textures ont tendance à être un peu floues à cause de la combinaison des images lors de leur création.

### 7.2.2 Parallélisme

La parallélisation des systèmes permet de répartir les calculs afin d'accélérer le temps de calcul général. Il n'est malheureusement pas toujours simple d'appliquer des algorithmes parallèles. La deuxième méthode de ré–éclairage présente l'avantage d'avoir des structures de données et des calculs associés à des pixels. Il paraît possible d'accélérer les mises à jour lors de la manipulation d'objets virtuels et réels, ou lors de l'insertion de lampes virtuelles. En effet, les calculs se font pixel par pixel, le résultat du calcul pour un pixel ne dépend pas de celui des autres. Les calculs peuvent être répartis et les résultats sont identiques à un calcul séquentiel.

Cependant, un simple découpage de l'image ne suffirait pas. Lors de la manipulation des objets, les calculs sont localisés dans une région de l'image (voir section 5.5). Avant la répartition des calculs sur plusieurs processeurs, nous devons parcourir la table des pixels afin de déterminer quels sont les pixels affectés par la modification. Les pixels ainsi sélectionnés sont répartis sur chacun des processeurs qui effectuent les calculs de mise à jour de la visibilité, et éventuellement des facteurs de forme. Lors de l'insertion d'une source virtuelle, tous les pixels sont affectés. Il n'y a pas besoin de sélectionner les pixels avant de les répartir. Un simple découpage de l'image suffit.

Nous pouvons en revanche avoir des problèmes de synchronisation des processus. Il

faudra veiller à ce que le lancement des calculs soit bien géré, car il s'agit de boucles interactives de l'ordre d'un dixième de seconde, et la synchronisation peut s'avérer coûteuse.

## 7.3 Combinaison des avantages des deux méthodes

Nous avons vu que nous pouvons utiliser l'image de réflectance obtenue dans la deuxième méthode pour extraire des textures. Pour la première méthode, des textures de réflectances peuvent donc servir de textures à un modèle. Ceci permet d'avoir une estimation plus juste des échanges d'énergie, tout en conservant les avantages de l'affichage par le matériel graphique, et des algorithmes de mise à jour lors des modifications des propriétés de la scène réelle. L'enlèvement d'objets réels devient possible aussi dans cette méthode, de même que l'insertion d'objets réels appartenant à une autre scène.

Il faut cependant réaliser quelques modifications au niveau de l'affichage (voir section 4.3.3). L'affichage des textures se fait par un facteur de modulation représentant le rapport entre le nouvel éclairage et l'éclairage original sans occlusion. Si nous utilisons des textures de réflectances, il faudra multiplier la texture par l'irradiance accumulée sur un élément de radiosité. Il reste à voir si la qualité de l'éclairage ainsi simulé sera suffisant, car il s'agira d'une approximation. Si cette irradiance est supérieure à un, nous pouvons toujours utiliser un affichage multi–passe.

## 7.4 Discussion

Nous avons étudié dans ce chapitre les avantages et les inconvénients de chacune des méthodes de ré–éclairage présentées dans cette thèse. Nous pouvons conclure que les méthodes ont des avantages significatifs dans des domaines différents. La seconde méthode semble cependant permettre de plus amples extensions. Son estimation de la réflectance peut servir de base aux algorithmes de la première méthode.

Les améliorations présentées dans ce chapitre ne sont pas encore implantées. Cependant le déplacement des objets réels nous parait important car il correspond au besoin qu'aurait un designer désirant ré–arranger une pièce existante.

# Chapitre 8

# Conclusion

Nous avons décrit dans cette thèse les recherches que nous avons menées pour permettre le ré–éclairage et la remodélisation de scènes réelles d'intérieur, dans des temps *interactifs*. Pour chacune des nouvelles méthodes présentées, nous nous sommes appliqués à utiliser des données simples et à développer des algorithmes interactifs, tout en conservant l'aspect qualité et réalisme de l'éclairage.

Ce travail combine de nombreux domaines de la synthèse d'images, comme la simulation interactive de l'éclairage, l'illumination inverse et l'éclairage commun pour la réalité augmentée. Dans le chapitre 2, nous avons décrit brièvement les méthodes de simulation de l'éclairage ainsi que chacune des notions permettant de comprendre les différentes méthodes existantes pour l'illumination inverse et l'éclairage commun. Il y a peu de travaux sur le ré–éclairage, qui est un problème difficile. Les méthodes de ré–éclairage de la littérature sont pour l'instant loin d'être interactives. Pourtant pour de nombreuses applications, l'interactivité, voir le temps réel, sont indispensables. Nous nous sommes donc fixé comme objectif de développer des méthodes permettant un ré–éclairage de qualité dans des temps interactifs.

Nous avons décrit dans le chapitre 3 la mise en place nécessaire à nos recherches. Les méthodes que nous proposons utilisent en effet une représentation géométrique des scènes réelles, obtenue par des outils de capture et de reconstruction. Nous avons expliqué chacune des étapes ainsi que les difficultés rencontrées lors de ce travail d'acquisition. Nous espérons que ce chapitre servira de point de départ à d'autres chercheurs travaillant sur des problèmes similaires.

Nous avons présenté deux nouvelles méthodes de ré–éclairage et de remodélisation. Pour chacune des méthodes, nous avons d'abord montré qu'elles pouvaient être utilisées avec des données d'entrée simples. Nous avons ensuite montré que nous pouvions améliorer la qualité des données, et ainsi améliorer la qualité des résultats. Cette extension nécessite cependant une phase de capture et de préparation plus importante. Les résultats montrent que nous avons atteint nos objectifs, qui étaient de permettre à un concepteur d'intérieur de manipuler virtuellement et interactivement un environnement réel, après une séance de capture simple.

## 8.1 Contributions

Dans cette thèse, nous avons apporté trois contributions principales : deux méthodes de ré–éclairage et de remodélisation interactives, et une méthode de calibrage photométrique adaptée à des données fournies par du matériel non spécialisé. Les méthodes ont été comparées dans le chapitre 7.

### 8.1.1 Méthode de ré–éclairage basée sur la connaissance d'un éclairage unique

L'objectif de la première méthode développée est de permettre un ré–éclairage de qualité, pour une scène réelle connue d'après une seule photographie. La résolution du problème est basée sur les algorithmes de radiosité hiérarchique. Grâce aux méthodes de radiosité, on peut simuler un éclairage global de qualité, dans des temps interactifs en utilisant des méthodes incrémentales. De plus, ces méthodes de radiosité nous permettent d'utiliser le matériel graphique, qui fournit un affichage rapide et quasiment indépendant de la résolution de l'image produite. Ces méthodes d'éclairage commun ne permettent cependant pas tels quels le ré–éclairage des scènes réelles, car l'éclairage original y est perceptible.

La solution que nous avons proposée est de produire une nouvelle représentation texturée de la scène réelle correspondant à un éclairage non bloqué par des objets réels, c'est–à–dire sans ombre. Pour modifier la représentation initiale de la scène réelle, nous délimitons d'abord de façon précise les régions à modifier, avec un nouveau critère de raffinement associant la couleur de texture attendue aux valeurs de visibilité par rapport aux sources de lumière. Nous calculons un facteur de modification à partir des équations de radiosité. Pour corriger d'éventuelles erreurs lors de la modification, nous appliquons ensuite un autre facteur de correction basé sur la couleur désirée.

Cette représentation sans ombre sert de base lors de la manipulation de la scène. L'éclairage est simulé et affiché en modulant la représentation sans ombre par un facteur d'affichage. Ce facteur a pour effet d'assombrir les régions d'ombre, et d'éclaircir les régions recevant plus de lumière qu'initialement. En utilisant des méthodes adaptées aux algorithmes de radiosité, nous pouvons modifier interactivement l'éclairage original, en modifiant l'intensité des sources de lumière ou en insérant de nouvelles lampes virtuelles. Nous pouvons également insérer des objets virtuels, et les déplacer en temps interactif (en trois images par seconde pour des scènes typiques).

### 8.1.2 Méthode de ré–éclairage basée sur la connaissance de plusieurs éclairages

La première méthode développée est basée sur la connaissance d'un seul éclairage. Comme la connaissance initiale est limitée, l'estimation des propriétés radiométriques est approximative. Ces approximations sont compensées par le fait que le nouvel éclairage simulé est une correction de l'éclairage original.

Nous voulions avoir une meilleure estimation des paramètres radiométriques, pour obtenir des solutions indépendantes de l'éclairage original. Pour cela, nous avons utilisé plusieurs photographies, représentant la même scène réelle sous des éclairages différents. Alors que la géométrie de la scène est simplifiée, nous retrouvons une réflectance diffuse plus

fiable pour chaque objet de la scène. Nous calculons pour cela, pixel par pixel, une estimation de la réflectance pour chacune des photographies. Une réflectance finale est ensuite calculée en pondérant chacune des réflectances par un coefficient reflétant sa validité. Une réflectance est considérée valide, si elle n'est pas dans une région d'ombre sur l'image originale et si sa valeur n'est pas trop éloignée des autres réflectances.

Pour avoir une simulation de qualité tout en étant efficace, nous utilisons une structure de données par pixel pour calculer l'éclairage direct, et une solution de radiosité pour calculer l'éclairage indirect. En exploitant l'association des deux structures, nous avons développé des algorithmes interactifs pour modifier l'intensité des lampes, ajouter de nouvelles sources virtuelles, ajouter et déplacer des objets virtuels. Nous pouvons aussi enlever des objets réels, en comblant les régions précédemment masquées par des morceaux de textures générées à partir des textures de réflectances (indépendantes de l'éclairage). Nous avons également montré que nous avons la possibilité de déplacer des objets réels. La rapidité de la mise à jour lors de modifications peut être accélérée en parallélisant les calculs.

### 8.1.3 Calibrage photométrique

La qualité des résultats obtenus peut être améliorée par un pré–traitement des données photographiques. Nous avons étudié les différentes étapes de la phase d'acquisition pour mieux calibrer les données. Nous avons adapté une méthode de création d'images de luminance à partir de photographies, pour un appareil photographique dont le contrôle manuel est limité. À la fin de ce processus, l'information de couleur de ces images n'est pas limitée par des saturations lumineuses, et les couleurs sont indépendantes de la fonction de la caméra.

Les images ainsi obtenues servent à créer de nouvelles textures pour la première méthode. Les résultats obtenus après ces différentes étapes de calibrage contiennent moins d'artefacts.

Pour pouvoir utiliser les images de luminance dans la deuxième méthode, il faut d'abord les recaler les unes par rapport aux autres, car l'appareil photographique choisit automatiquement un temps d'exposition de référence. Le recalage est fait en modulant les images de luminance par un facteur de correction, estimé par des équations de radiosité. Ces nouvelles images ont des valeurs de luminance plus cohérentes entre elles. Enfin, nous avons mesuré les propriétés de la lampe utilisée dans la deuxième méthode, en vérifiant sa propriété diffuse, et en mesurant son spectre. Ces différents calibrages donnent une meilleure qualité des données. Nous avons ainsi pu améliorer la fiabilité de la réflectance estimée, en itérant le processus d'estimation.

## 8.2 Perspectives

Les méthodes que nous avons développées permettent le ré–éclairage de scènes d'intérieur. Dans cette thèse, nous ne prenons en compte que les propriétés diffuses. De plus, la scène ne peut être visualisée que depuis un point de vue fixe.

### 8.2.1 Spécularité

Dans ce document, nous avons plusieurs fois noté la restriction de nos méthodes aux scènes aux propriétés diffuses. Nous aimerions étendre les méthodes que nous avons proposées aux objets spéculaires et aux lampes directionnelles.

#### Effets spéculaires

##### *Première méthode de ré–éclairage*

Retrouver un terme spéculaire pour les surfaces réelles dans la première méthode de ré–éclairage (voir chapitre 4) nous semble délicat si nous n'utilisons pas plus de données photographiques. En effet, depuis une seule photographie représentant l'éclairage, nous ne pouvons pas prédire si les différences de couleurs sont dues à une tache de spécularité ou à une variation de texture. Pour déterminer la spécularité des objets, une première solution serait de faire intervenir l'utilisateur. Il pourrait définir si une surface est spéculaire ou non, et ajuster à la main une composante spéculaire. On pourrait également prendre plus de photographies depuis des points de vue différents, et comparer les différentes textures des objets. Ces composantes spéculaires pourraient être ajustées en utilisant un algorithme équivalent à celui de Debevec [Deb98]. Pour effectuer le rendu de ces surfaces spéculaires, nous pourrions utiliser une méthode s'appliquant aux solutions de radiosité comme celle de Stamminger *et al.* [SSG+99], ou superposer une solution de lancer de rayons spéculaire, étant donné que le point de vue est fixe.

##### *Deuxième méthode de ré–éclairage*

Si nous voulons effectivement prendre en compte les effets spéculaires dans le système d'éclairage présenté au chapitre 5, il faudra faire attention à ce que nos données soient suffisantes, et que la capture soit orientée dans ce sens. En prenant plus de photographies, l'estimation des réflectances pourrait se faire en utilisant une méthode similaire à celle de Yu *et al.* [YDMH99]. Dans un premier temps, nous pouvons limiter le rendu aux effets spéculaires directs et isotropiques. Le rendu pourrait se faire pixel par pixel en stockant la composante spéculaire associée à chaque pixel de l'image dans la structure de données associée à l'éclairage direct. Il serait cependant préférable d'inclure des effets également dans l'éclairage indirect. Comme pour la première méthode, une solution grossière pourrait être réalisée en utilisant par exemple les algorithmes de Stamminger *et al.* [SSG+99].

Pour chacune des intégrations des effets spéculaires, il faudra veiller à ce que la scène soit suffisamment bien modélisée pour que les effets soient corrects. Dans les reconstructions actuelles, la géométrie est très approximative, notamment pour les surfaces courbes. Ceci n'est pas perceptible dans les scènes actuelles, où ces approximations sont compensées par les textures. Par contre, en simulant des effets spéculaires, ces erreurs dans la géométrie seraient plus perceptibles.

#### Lampes directionnelles

Nous aimerions aussi adapter les deux méthodes aux lampes directionnelles, très présentes dans la réalité. Nous avons décrit dans la section 6.3.2 un processus pour vérifier l'émission diffuse des lampes. Ce processus pourrait être exploité pour déterminer les propriétés directionnelles des lampes. Il faudra ensuite prendre en compte cette propriété directionnelle pour l'estimation des réflectances, et dans la simulation de l'éclairage, en utilisant un échantillonnage des lampes approprié.

### 8.2.2 Changement de point de vue

Dans nos systèmes actuels, l'utilisateur ne peut pas changer de point de vue. Nous pouvons imaginer combien il est important de pouvoir se déplacer pour se rendre compte des modifications et des effets d'éclairage.

Pour les deux méthodes de ré–éclairage que nous avons présentées, le déplacement du point de vue nous paraît un problème difficile. Pour chacune des méthodes, on pourrait appliquer les mêmes algorithmes pour différents points de vue. Une méthode d'interpolation, ou une structure de textures comme dans Debevec *et al.* [DBY98] pourraientt être utilisées pour le rendu. Cette approche nous paraît cependant trop lourde et nous aimerions développer de nouveaux algorithmes pour résoudre le problème. De plus, elle ne résout pas le problème de modélisation supplémentaire qui sera inévitablement nécessaire.

En ce qui concerne la première méthode, nous pourrions travailler avec des textures indépendantes du point de vue pour retrouver les propriétés radiométriques. Le rendu classique ne serait pas suffisant avec de telles textures, qui ont tendance à être un peu floues et à ne pas représenter correctement les effets de parallaxe, lorsque tous les objets de la scène ne sont pas modélisés. Il pourrait être *corrigé* pour chaque point de vue, en utilisant les textures dépendantes du point de vue original (non modifiées) pour rendre les effets de perspectives.

#### Nouvelle solution

Au lieu d'étendre la deuxième méthode, nous pensons qu'une nouvelle voie à explorer serait de combiner différentes techniques existantes. Pour retrouver les propriétés de réflectance des surfaces pixel par pixel, les algorithmes de Yu *et al.* [YDMH99] pourraient être appliqués. Le rendu pourrait être réalisé en combinant une méthode de lancer de rayons dynamique, comme celle proposée par Walter *et al.* [WDP99], avec une solution de radiosité pour l'éclairage indirect, comme nous le faisons dans notre méthode. Le rendu dynamique de Walter *et al.* n'est cependant pas de qualité suffisante pour un rendu à chaque image en temps interactif, et la vitesse de mise à jour dépend de la taille de l'image.

### 8.2.3 Scènes d'extérieur

Nous avons présenté dans le chapitre 2 des méthodes permettant le ré–éclairage de scènes d'extérieur. Ces méthodes sont cependant limitées par la difficulté de l'estimation

de la réflectance pour de tels environnements, éclairés par le soleil et par le ciel. Permettre un ré–éclairage de qualité pour des environnements extérieurs est un problème difficile, à la fois à cause de la complexité de l'éclairage, mais aussi à cause de la complexité de la géométrie et des réflectances des environnements. Nous aimerions cependant trouver des solutions pour adapter nos systèmes à des scènes d'extérieur, en prenant en compte la possibilité d'utiliser des solutions de radiosité en extérieur comme par exemple celle de Daubert *et al.* [DSSD97].

### 8.2.4 Vers le temps réel...

Nous prévoyons de nombreuses applications, encourageant la poursuite de nos recherches. Les applications sont diverses, comme en urbanisme architectural, en cinéma ou en réalité augmentée.

Un premier exemple d'application est l'aménagement de l'urbanisme, où un architecte par exemple voudrait se rendre compte de l'impact sur l'environnement de la construction d'un nouveau bâtiment. À partir de photographies, il pourrait virtuellement détruire les anciennes constructions et les remplacer par des nouvelles. Il étudierait à la fois comment les constructions s'intègrent à l'environnement existant, et quel serait leur impact sur l'éclairage. Il serait utile de visualiser si les bâtiments reçoivent de la lumière du soleil, ou si ils en empêchent d'autres d'être suffisamment éclairés.

Les méthodes de modification des environnements existants peuvent également intéresser les cinéastes. Imaginons qu'un réalisateur veuille faire un film sur le début du siècle. Il pourrait tourner son film dans un environnement actuel. Ensuite, le film pourrait être modifié en enlevant des lampadaires, des voitures actuelles, ou une statue récente. Les effets d'éclairage seront correctement modifiés. De telles manipulations sont déjà réalisées mais nécessitent un travail pénible et manuel de retouche d'image. Elles seraient facilitées par un processus plus automatique.

Enfin, nous aimerions aller vers des systèmes de réalité augmentée et de réalité virtuelle, où un utilisateur en immersion pourrait modifier virtuellement la scène réelle. Il pourrait entrer dans une pièce, et mettre des lunettes d'immersion. Il faudrait qu'il croit qu'il continue de voir la réalité, alors qu'il ne verrait qu'une représentation virtuelle de la scène réelle. L'utilisateur pourrait alors manipuler virtuellement l'environnement et son éclairage.

# Annexe A

# Affichage multi–passes

Nous avons présenté dans le chapitre 4, une méthode de ré–éclairage basée sur des algorithmes de radiosité hiérarchique. L'affichage se fait élément par élément, en multipliant la texture par un facteur d'affichage. Nous utilisons le matériel graphique et notamment la librairie OpenGL®. L'affichage se fait typiquement en initialisant la fonction *glColor* en passant en paramètre les trois composantes du facteur d'affichage. Cette fonction nécessite des paramètres de couleur compris entre zéro et un. Si le facteur d'affichage est supérieur à un, il est automatiquement mis à un. Par cet affichage, nous n'obtenons pas l'éclairage désiré si le facteur est supérieur à un.

Pour éviter cette limitation, nous avons utilisé une méthode d'affichage en multi–passes. L'algorithme d'affichage est décrit sur la figure A.1. Pour tenir compte du fait que le facteur est supérieur à un, nous allons superposer $n$ affichages, où $n-1$ est la partie entière du facteur d'affichage. La $n^{\text{ème}}$ passe correspond au reste du facteur d'affichage moins sa partie entière. La texture sera modulée par le facteur d'affichage non limité à un. Un exemple de l'importance de cet affichage en multi–passes est montré sur la figure A.2. Elle montre le résultat d'une insertion de lampe virtuelle dans une scène réelle. Nous n'avons pas utilisé d'affichage en multi–passes en (a), alors que nous l'avons utilisé en (b). La différence est clairement visible. Il manque de la luminosité en (a). Toutes les valeurs du facteur d'affichage ont été limitées à un. En (b), l'éclairage affiché rend bien compte de l'éclaircissement des textures dû à l'insertion de la lampe virtuelle.

Des problèmes similaires ont été rencontrés par d'autres chercheurs.
Debevec *et al.* [DBY98] utilisent une approche similaire pour combiner des textures pondérées, et n'en créer qu'une seule. Soler et Sillion [SS98] utilisent également un affichage multi–passes pour moduler correctement les textures avec l'éclairage direct.

```
Affichage Multi–passes
   glBegin(GL_BLEND)
   glTexEnvf(GL_TEXTURE_ENV,
         GL_TEXTURE_ENV_MODE,
         GL_MODULATE)
   NumIter = 1
   MaxIter = AfficherScène(NumIter)
   glBlendFunc(GL_ONE, GL_ONE)
   tant que NumIter < MaxIter faire
      AfficherScène(NumIter)
      glBlendFunc(GL_ONE, GL_ONE)
      NumIter++
   glDisable(GL_BLEND)
```

```
AfficherScène ( NumIter )
   MaxIter = 0
   pour chaque feuille
      if(NumIter == 0) alors
         Maxiter = Max (Maxiter, B_i / D_i)
      glTexImage2D( …, Texture)
      pour chaque sommet
         color = (B_i / D_i) - NumIter
      si color > 1 alors color = 1
      si color < 0 alors color = 0
      glColor(color)
      retourner Maxiter
```

FIG. A.1: Algorithme d'affichage en multi–passes.



(a) (b)

FIG. A.2: (a) Insertion d'une lampe virtuelle, sans utiliser un affichage multi–passes. (b) Utilisation d'un affichage multi–passes (5 passes en 0.3 secondes).

# Annexe B

# Programme de capture automatique de photographies

## B.1 Propriétés contrôlables de l'appareil photographique Kodak DC260

L'appareil photographique numérique Kodak DC260 que nous utilisons n'est pas contrôlable manuellement pour toutes ses fonctions. Certaines propriétés sont contrôlables par simple choix dans des menus, ou directement par pression des boutons. En appuyant directement sur des boutons, on contrôle la position du zoom et la prise des photographies. La mise au point est faite automatiquement sur le centre de la scène capturée. Par contre, par une demi–pression, on peut choisir une mise au point pour un autre cadrage, et la conserver en revenant sur le cadrage choisi.

Dans les fonctions du menu, on trouve :

- le contrôle des propriétés des images comme la taille, la résolution ou la cadence de capture (en rafale ou en instantané),
- le contrôle du flash,
- le contrôle de la balance des couleurs en choisissant un type d'éclairage. Cette balance automatique peut être désactivée,
- la possibilité de choisir manuellement la distance focale parmi des valeurs pré–définies,
- un choix sur le type d'exposition (automatique ou longue durée). L'exposition longue durée varie entre une demie seconde et seize secondes. Le temps d'exposition ne peut être contrôlable en–dessous d'une demie seconde.

D'autres propriétés peuvent être directement contrôlables par des scripts. Parmi les propriétés contrôlables par script, on retrouve la distance focale. D'autres propriétés non contrôlables directement sur les menus, comme l'ouverture du diaphragme, peuvent être contrôlées depuis les scripts. En revanche, le temps d'exposition n'est pas contrôlable en–dessous d'une demie seconde.

Nous pouvons fixer une ouverture, mais pas le temps d'exposition. Malheureusement l'appareil choisit automatiquement certains paramètres pour que la photographie aient les « meilleures » couleurs. Même si l'ouverture est fixée, il compensera par le choix automatique d'un temps d'exposition. Ceci veut dire que pour une même scène prise sous un point de vue légèrement différent, les couleurs ne seront pas cohérentes. L'appareil photographique se sera automatiquement adapté au nouveau niveau d'éclairage perceptible dans le cadrage.

Nous ne pouvons donc pas contrôler le paramètre conjugué ouverture/temps. L'appareil possède cependant un mode appelé *EV*. Ce n'est pas le mode *EV* traditionnellement utilisé en photographie. Le mode traditionnel permet justement de calculer le paramètre conjugué ouverture/temps, car il existe un relation entre la valeur de *EV*, le temps d'exposition et l'ouverture du diaphragme. Celui du Kodak DC260 est relatif à la photographie prise en mode automatique. Quand $EV = 0$, la photographie correspond à une ouverture et un temps d'exposition le mieux adapté d'après les critères de l'appareil. Les valeurs de *EV* peuvent varier mais elles seront toujours relatives à l'image choisie automatiquement en $EV = 0$. Les *EV* sont pré–définies et distribuées uniformément autour de 0. Les différentes valeurs sont : $-2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2$. Ceci nous permet de prendre neuf photographies avec des temps d'exposition différents.

Nous avons essayé de savoir si l'appareil faisait varier l'ouverture ou le temps d'exposition pour des *EV* différents. Nous avons fixé l'ouverture avant la modification de l'*EV*, et nous avons lu la valeur d'ouverture après la prise de photographies. Comme elle était restée la même que celle que nous avions fixée arbitrairement sur plusieurs prises, nous en avons déduit que c'était le temps d'exposition qui variait.

## B.2 Script de capture de photographies sur une plage étendue

En utilisant la capacité de programmation de l'appareil photographique et le contrôle des valeurs de *EV,* nous avons écrit un script qui prend automatiquement neuf photographies à des *EV* différents.

```
name "Exposure Bracketing"
mode 0
menu "Kodak Scripts"
label "My Exposure Bracketing"


# Variable declaration and definition
declare u : uTemp, uError, uStorage, uImageTaken, uImageAvail,
  uSsiz, uCameraType
declare u : uSelectStop, uSelectCapture, uSelect
declare i : iFlagOn, iFlagOff, iInitialEV, iImageRaw,
  iInitialAper
declare i : iErrorCode, iBracketValue, iTemp, iInitialShut
declare i : iAper

declare s : sProductName, sVendorName, sCameraType, sTemp,
  sStop
declare b : bCameraType
declare f : fTemp

# Variable initialization
iErrorCode = 0
iFlagOn = 1
iFlagOff = 0
uCameraType = 0


# Get camera name
  GetProductInfo ("ptid", sProductName)
  GetProductInfo ("vdid", sVendorName)
  if sProductName == "KODAK DIGITAL SCIENCE DC220"
    uCameraType = 220
    sCameraType = "DC220"
  end

  if sProductName == "KODAK DIGITAL SCIENCE DC260"
    uCameraType = 260
    sCameraType = "DC260"
  end
  if uCameraType == 0
    iErrorCode = 10
    goto ErrorProc
  end

DisplayLine ("Nine (9) pictures will be captured at different
  levels of exposure.")
Wait (3000)

# Get initial camera settings
  GetCameraState ("xcmp", iInitialEV)
  GetCameraState ("ssiz", uSsiz)
```

```
  GetCameraState("aper", iInitialAper)
  DisplayLine("Initial aperture is ", iInitialAper, " .")

# Check available memory
  uStorage = 1
  iImageRaw = -1
  GetStorageStatus (uStorage, uImageTaken, uImageAvail,
  iImageRaw)

  if uImageAvail < 4
    if uSsiz > 2
      iErrorCode = 20
      goto ErrorProc
    end
    if uCameraType == 220
      SetCameraState ("ssiz", 3)
      GetStorageStatus (uStorage, uImageTaken, uImageAvail,
        iImageRaw)
      if uImageAvail < 4
        iErrorCode = 20
        uError = 4 - uImageAvail
        goto ErrorProc
      end
    end
    if uCameraType == 260
      SetCameraState ("ssiz", 2)
      GetStorageStatus (uStorage, uImageTaken, uImageAvail,
        iImageRaw)
      if uImageAvail < 4
        SetCameraState ("ssiz", 3)
        GetStorageStatus (uStorage, uImageTaken,
          uImageAvail, iImageRaw)
        if uImageAvail < 4
          iErrorCode = 20
          uError = 4 - uImageAvail
          goto ErrorProc
        end
      end
    end
    Alert ("Due to insufficient memory, the camera’s
      resolution and/or quality has been reduced.")
  end



# Ask what aperture
  SetOption (1, "3", iFlagOff)
  SetOption (2, "4", iFlagOff)
  SetOption (3, "5.6", iFlagOff)
  SetOption (4, "8", iFlagOff)
  SetOption (5, "11", iFlagOff)
```

```
  SetOption (6, "16", iFlagOff)
  SetOption (10, "Return", iFlagOff)
  GetOption (uSelectStop)

  if uSelectStop == 1
    iAper = 300
  end
  if uSelectStop == 2
    iAper = 400
  end
  if uSelectStop == 3
    iAper = 560
  end
  if uSelectStop == 4
    iAper = 800
  end
  if uSelectStop == 5
    iAper = 1100
  end
  if uSelectStop == 6
    iAper = 1600
  end
  if uSelectStop == 10
    goto GetCaptureType
  end

  SetCameraState("aper", iAper)

# Start Bracket Exposure
  DisplayClear ()
  SetOption (1, "Begin Capture", iFlagOn)
  SetOption (10, "Return", iFlagOff)
  GetOption (uSelect)
  if uSelect == 10
    goto GetCaptureType
  end

  DisplayClear ()
    DisplayLine ("Nine consectutive pictures will automatically
      be taken.")
  Wait (5000)
  Alert ("Position the camera to take the desired picture.
    Press CONTINUE to start capturing the exposure bracket.")
  Wait (1000)

  SetCaptureMode (still)


  iBracketValue = -200
  DisplayClear()
  DisplayLine("EV is ", iBracketValue)
```

```
  Wait(2000)
  uError = SetCameraState ("xcmp", iBracketValue)
  if uError != 0
    iErrorCode = 50
    goto ErrorProc
  end

Capture :


  uError = StartCapture ()
  if uError == 12
    # system is busy
    Wait (1000)
    Display (".")
    goto Capture
  end
  if uError != 0
    iErrorCode = 60
    goto ErrorProc
  end
  iBracketValue = iBracketValue + 50

  if iBracketValue > 200
    goto FinishCapture
  end
  DisplayClear()
  DisplayLine("EV is ", iBracketValue)
  Wait(2000)

  # take picture at upper bracket
  uError = SetCameraState ("xcmp", iBracketValue)
  if uError != 0
    iErrorCode = 50
    EndCapture()
    goto ErrorProc
  end
  goto Capture

FinishCapture :
  DisplayClear()
  DisplayLine("Stop capture.")
  Wait(2000)

  EndCapture ()

ErrorProc :
  if iErrorCode > 1
    DisplayClear ()
    if iErrorCode == 10
      Alert ("Error : camera not supported.")
```

```
    end
    if iErrorCode == 20
      Alert ("Insufficient memory : ", uError, "
        pictures must be removed to perform exposure
        bracket.")
    end
    if iErrorCode == 50
      Alert ("Error : set camera state <", uError, ">.")
    end
    if iErrorCode == 60
      Alert ("Error : start capture <", uError, ">.")
    end
  end

Finish :
  # set camera to initial values
  SetCameraState ("xcmp", iInitialEV)
  SetCameraState ("aper", iInitialAper)
  SetCameraState ("ssiz", uSsiz)
  if iErrorCode == 0
    Alert ("Bracket capture is complete. Switch to review
      mode to view the pictures.")
  end

Done :
  exitscript
```

# Bibliographie


[Azu97] Ronald T. Azuma. A survey of augmented reality. In *Presence : Teleoperators and Virtual Environments 6*, pages 355–385, August 1997.

[Bli77] James F. Blinn. Models of light reflection for computer synthesized pictures. volume 11, pages 192–198, July 1977.

[BP96] Normand Brière and Pierre Poulin. Hierarchical view-dependent structures for interactive scene manipulation. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 83–90, August 1996.

[BR97] Sylvain Bougnoux and Luc Robert. TotalCalib : a fast and reliable system for off-line calibration of images sequences. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, 1997. The Demo Session.

[BRW89] D. R. Baum, H. E. Rushmeier, and J. M. Winget. Improving radiosity solutions through the use of analytically determined form-factors. In *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23, pages 325–334, July 1989.

[BWCG86] Daniel R. Baum, John R. Wallace, Michael F. Cohen, and Donald P. Greenberg. The back-buffer algorithm : An extension of the radiosity method to dynamic environments. *The Visual Computer*, 2(5) :298–306, September 1986.

[Can] Canoma. http ://www.metacreations.com/products/canoma/.

[CCWG88] Michael F. Cohen, Shenchang Eric Chen, John R. Wallace, and Donald P. Greenberg. A progressive refinement approach to fast radiosity image generation. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 75–84, August 1988.

[CF92] Norman Chin and Steven Feiner. Fast object-precision shadow generation for areal light sources using BSP trees. In David Zeltzer, editor, *Computer Graphics (1992 Symposium on Interactive 3D Graphics)*, volume 25, pages 21–30, March 1992.

[CG85] Michael F. Cohen and Donald P. Greenberg. The Hemi-Cube : A radiosity solution for complex environments. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19, pages 31–40, August 1985.

[Che90] Shenchang Eric Chen. Incremental radiosity : An extension of progressive radiosity to an interactive image synthesis system. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 135–144, August 1990.

[CLSS97] P. H. Christensen, D. Lischinski, E. J. Stollnitz, and D. H. Salesin. Clustering for glossy global illumination. *ACM Transactions on Graphics*, 16(1) :3–33, January 1997.

[CT82] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics*, 1(1) :7–24, January 1982.

[DBY98] Paul Debevec, George Borshukov, and Yizhou Yu. Efficient view-dependent image-based rendering with projective texture-mapping. In *Rendering Techniques '98 (9th Eurographics Workshop on Rendering)*, pages 105–116. Springer-Verlag, June 1998.

[De 97] Jeremy S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 361–368. ACM SIGGRAPH, Addison Wesley, August 1997. ISBN 0-89791-896-7.

[Deb98] P.E. Debevec. Rendering synthetic objects into real scenes : Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *SIGGRAPH '98 Conference Proceedings*, Annual Conference Series, pages 189–198, July 1998.

[DM97] P.E. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH '97 Conference Proceedings*, Annual Conference Series, pages 369–378, August 1997.

[DRB97] G. Drettakis, L. Robert, and S. Bougnoux. Interactive common illumination for computer augmented reality. In *Rendering Techniques '97 (8th Eurographics Workshop on Rendering)*, pages 45–56. Springer-Verlag, June 1997.

[DS97] G. Drettakis and F. Sillion. Interactive update of global illumination using a line-space hierarchy. In *SIGGRAPH '97 Conference Proceedings*, Annual Conference Series, pages 57–64, August 1997.

[DSSD97] Katja Daubert, Hartmut Schirmacher, Francois Sillion, and George Drettakis. Hierarchical lighting simulation for outdoor scenes. In Julie Dorsey and Phillip Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 229–238, New York, NY, 1997. Springer Wien. ISBN 3-211-83001-4.

[DTM96] P.E. Debevec, C.J. Taylor, and J. Malik. Modeling and rendering architecture from photographs : A hybrid geometry- and image-based approach. In *SIGGRAPH '96 Conference Proceedings*, Annual Conference Series, pages 11–20, July 1996.

[EM] T. El-Maraghi. An implementation of Heeger and Bergen's texture analysis/synthesis algorithm with source code, http ://www.cs.toronto.edu/~tem/2522/texture.html.

[Fai98] Mark D. Fairchild. *Color Appearance Models,*. Addison Wesley, 1998.

[Fau92] Olivier Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig. In G. Sandini, editor, *Proceedings of the 2nd European Conference on Computer Vision*, volume 588 of *Lecture Notes in Computer Science*, pages 563–578, Santa Margherita Ligure, Italy, May 1992. Springer-Verlag.

[Fau93] O. Faugeras. *Three-Dimensional Computer Vision — A Geometric Viewpoint*. MIT Press, 1993.

[FGR93] A. Fournier, A.S. Gunawan, and C. Romanzin. Common illumination between real and computer generated scenes. In *Proc. of Graphics Interface '93*, pages 254–262, May 1993.

[FLR+97] Olivier Faugeras, Stéphane Laveau, Luc Robert, Gabriella Csurka, Cyril Zeller, Cyrille Gauclin, and Imed Zoghlami. 3-d reconstruction of urban scenes from image sequences. *CVGIP : Image Understanding*, 1997.

[Fra99] Marie-Claude Frasson. Reconstruction interactive de scènes tridimensionnelles à partir d'images. M.sc. thesis, Département d'Informatique et Recherche Opérationnelle, Université de Montréal, March 1999.

[FT86] Olivier Faugeras and Giorgio Toscani. The calibration problem for stereo. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 15–20, Miami Beach, FL, June 1986. IEEE.

[FYT94] David A. Forsyth, Chien Yang, and Kim Teo. Efficient radiosity in dynamic environments. In *Fifth Eurographics Workshop on Rendering*, pages 313–323, Darmstadt, Germany, June 1994.

[GH96a] S. Gibson and R. J. Hubbold. Efficient hierarchical refinement and clustering for radiosity in complex environments. *Computer Graphics Forum*, 15(5) :297–310, 1996. ISSN 0167-7055.

[GH96b] Simon Gibson and R. J. Hubbold. Efficient hierarchical refinement and clustering for radiosity in complex environments. *Computer Graphics Forum*, 15(5) :297–310, December 1996.

[GSG90] David W. George, Francois X. Sillion, and Donald P. Greenberg. Radiosity redistribution for dynamic environments. *IEEE Computer Graphics and Applications*, 10(4) :26–34, July 1990.

[GTGB84] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modelling the interaction of light between diffuse surfaces. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 212–22, July 1984.

[HB95] D.J. Heeger and J.R. Bergen. Pyramid-Based texture analysis/synthesis. In *SIGGRAPH '95 Conference Proceedings*, Annual Conference Series, pages 229–238, August 1995.

[HDSD99] Jean-Marc Hasenfratz, Cyrille Damez, Francois Sillion, and George Drettakis. A practical analysis of clustering strategies for hierarchical radiosity. In *Computer Graphics Forum (Proc. Eurographics '99)*, volume 18, pages 221–232, September 1999.

[Hec92] Paul Heckbert. Discontinuity Meshing for Radiosity. In *Third Eurographics Workshop on Rendering*, pages 203–226, Bristol, UK, May 1992.

[HSA91] Pat Hanrahan, David Salzman, and Larry Aupperle. A rapid hierarchical radiosity algorithm. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 197–206, July 1991.

[HT96] Anil N. Hirani and Takashi Totsuka. Combining frequency and spatial domain information for fast interactive image noise removal. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 269–276. ACM SIGGRAPH, Addison Wesley, August 1996.

[HTSG91] Xiao D. He, Kenneth E. Torrance, Francois X. Sillion, and Donald P. Greenberg. A Comprehensive Physical Model for Light Reflection. In *Computer Graphics (ACM SIGGRAPH '91 Proceedings)*, volume 25, pages 175–186, July 1991.

[HW94] E. A. Haines and J. R. Wallace. Shaft Culling for Efficient Ray-Traced Radiosity. In *Photorealistic Rendering in Computer Graphics (Proceedings of the 2nd Eurographics Workshop on Rendering)*, New York, NY, 1994. Springer-Verlag.

[IP97] H. Igehy and L. Pereira. Image replacement through texture synthesis. In *Proceedings of the 1997 IEEE International Conference on Image Processing*, 1997.

[JNP+95] Pierre Jancène, Fabrice Neyret, Xavier Provot, Jean-Philippe Tarel, Jean-Marc Vézien, Christophe Meilhac, and Anne Verroust. Res : computing the interactions between real and virtual objects in video sequences. In *Second IEEE Workshop on networked Realities*, pages 27–40, Boston, Massachusetts (USA), October 1995. http ://www-rocq.inria.fr/syntim/textes/nr95-eng.html.

[Kaj86] James T. Kajiya. The Rendering Equation. In *Computer Graphics (ACM SIGGRAPH '86 Proceedings)*, volume 20, pages 143–150, August 1986.

[LdlFK+96] F. Leymarie, A. de la Fortelle, J. Koenderink, A. Kappers, M. Stavridi, B. van Ginneken, S. Muller, S. Krake, O. Faugeras, L. Robert, C. Gauclin, S. Laveau, and C. Zeller. Realise : Reconstruction of reality from image sequences. In *International Conference on Image Processing*, volume 3, pages 651–654, Lausanne (Switzerland), 1996.

[LDR98] C. Loscos, G. Drettakis, and L. Robert. Interactive modification of real and virtual lights for augmented reality. In *SIGGRAPH '98 Technical Sketch (Visual Proceedings)*, July 1998.

[LDR99] C. Loscos, G. Drettakis, and L. Robert. Interactive virtual relighting of real scenes. 1999. soumis à révision dans un journal international.

[LFD99a] C. Loscos, M.-C. Frasson, and G. Drettakis. Interactive virtual relighting and remodeling of real scenes. In *Poster Session, Second IEEE and ACM International Workshop on Augmented Reality (IWAR)'99*, October 1999.

[LFD+99b] C. Loscos, M.-C. Frasson, G. Drettakis, B. Walter, X. Granier, and P. Poulin. Interactive virtual relighting and remodeling of real scenes. In *Rendering Techniques '99*, pages 329–340, New York, NY, June 1999. Springer Wien.

[LHO90] Yuncai Liu, Thomas S. Huang, and Faugeras Olivier. Determination of Camera Location from 2-D to 3-D Line and Point Correspondences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1) :28–37, January 1990.

[LRP97] Greg Ward Larson, Holly Rushmeier, and Christine Piatko. A visibility matching tone reproduction operator for high dynamic range scenes. *IEEE Transactions on Visualization and Computer Graphics*, 3(4) :291–306, October–December 1997.

[LSG94] Dani Lischinski, Brian Smits, and Donald P. Greenberg. Bounds and Error Estimates for Radiosity. In *Computer Graphics Proceedings, Annual Conference Series, 1994 (ACM SIGGRAPH '94 Proceedings)*, pages 67–74, 1994.

[LTG92] Daniel Lischinski, Filippo Tampieri, and Donald P. Greenberg. Discontinuity Meshing for Accurate Radiosity. *IEEE Computer Graphics and Applications*, 12(6) :25–39, November 1992.

[LTG93] Daniel Lischinski, Filippo Tampieri, and Donald P. Greenberg. Combining hierarchical radiosity and discontinuity meshing. In *Computer Graphics Proceedings, Annual Conference Series, 1993*, pages 199–208, 1993.

[Mar98] Stephen R. Marschner. *Inverse Rendering in Computer Graphics*. PhD thesis, Program of Computer Graphics, Cornell University, Ithaca, NY, 1998.

[MCS90] J.F. Murray-Coleman and A.M. Smith. The automated measurement of brdfs and their apllication to luminaire modeling. *The Illuminating Engineering Society*, Winter 1990.

[MG97] Stephen R. Marschner and Donald P. Greenberg. Inverse lighting for photography. In *IS&T/SID, Fifth Color Imaging Conference*, pages 262–265, November 1997.

[MNP+99] D.K. McAllister, L. Nyland, V. Popescu, A. Larsa, and C. McCue. Real–time rendering of real world environments. In *Rendering Techniques '99*, pages 145–160, New York, NY, June 1999. Springer Wien.

[MS94] Stefan Muller and Frank Schoffel. Fast radiosity repropagation for interactive virtual environments using a shadow-form-factor-list. In *Fifth Eurographics Workshop on Rendering*, pages 325–342, Darmstadt, Germany, June 1994.

[MYTG94] Vannuary Meas-Yedid, Jean-Philippe Tarel, and André Gagalowicz. Calibration métrique faible et construction interactive de modèles 3D de scènes. In *Congrès Reconnaissance des Formes et Intelligence Artificielle*, Paris, France, 1994. AFCET.

[NHIN86] E. Nakamae, K. Harada, T. Ishizaki, and T. Nishita. A montage method : The overlaying of the computer generated images onto a background photograph. In *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 207–214, August 1986.

[Nus28] W. Nusselt. Grapische Bestimmung des Winkelverhaltnisses bei der Warmestrahlung. *Zeitschrift des Vereines Deutscher Ingenieure*, 72(20) :673, 1928.

[Nyl] Lars S. Nyland. Capturing dense environmental range information with a panning, scanning laser rangefinder, http ://www.cs.unc.edu/˜ibr/projects/rangefinder/how_it_works/.

[Pho] Photomodeler. http ://www.photomodeler.com.

[Pho75] Bui-T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6) :311–317, June 1975.

[POF98] P. Poulin, M. Ouimet, and M.-C. Frasson. Interactively modeling with photogrammetry. In *Rendering Techniques '98 (9th Eurographics Workshop on Rendering)*, pages 93–104. Springer-Verlag, June 1998.

[Rea] http ://www.realviz.com/.

[Ren] Renoir. http ://www.integra.co.jp/eng/products/renoir/index.htm.

[Rob95] L Robert. Camera calibration without feature extraction. *Computer Vision, Graphics, and Image Processing*, 63(2) :314–325, March 1995. also INRIA Technical Report 2204.

[SAG94] Brian Smits, James Arvo, and Donald Greenberg. A clustering algorithm for radiosity in complex environments. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 435–442, July 1994.

[Sam90] Hanan Samet. *Applications of Spatial Data Structures*. Addison-Wesley, Reading, Massachusetts, 1990.

[SAS92] Brian E. Smits, James R. Arvo, and David H. Salesin. An Importance-Driven Radiosity Algorithm. In *Computer Graphics (ACM SIGGRAPH '92 Proceedings)*, volume 26, pages 273–282, July 1992.

[Sch93] Peter Schröder. Numerical Integration for Radiosity in the Presence of Singularities. In *Fourth Eurographics Workshop on Rendering*, number Series EG 93 RW, pages 177–184, Paris, France, June 1993.

[Sch94] Christophe Schlick. Quantization techniques for visualization of high dynamic range pictures. In *Fifth Eurographics Workshop on Rendering*, pages 7–18, Darmstadt, Germany, June 1994.

[SDS95] F. Sillion, G. Drettakis, and C. Soler. A clustering algorithm for radiance calculation in general environments. In *Eurographics Rendering Workshop 1995*. Eurographics, June 1995.

[Sha97] E. Shaw. Hierarchical radiosity for dynamic environments. *Computer Graphics Forum*, 16(2) :107–118, 1997.

[SHC+96] Andrei State, Gentaro Hirota, David T. Chen, Bill Garrett, and Mark Livingston. Superior augmented reality registration by integrating landmark tracking and magnetic tracking. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 429–438, August 1996.

[SKv+92] Mark Segal, Carl Korobkin, Rolf van Widenfelt, Jim Foran, and Paul E. Haeberli. Fast shadows and lighting effects using texture mapping. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 249–252, July 1992.

[SP94] François Sillion and Claude Puech. *Radiosity and Global Illumination*. Morgan Kaufmann, San Francisco, 1994.

[Spa63] Ephraim M. Sparrow. On the Calculation of Radiant Interchange Between Surfaces. In Warren Ibele, editor, *Modern Developments in Heat Transfer*. Academic Press, New York, NY, 1963.

[SS98] Cyril Soler and Francois Sillion. Automatic calculation of soft shadow textures for fast, high-quality radiosity. In *Rendering Techniques '98*, 9th EG workshop on Rendering, Vienna, Austria, June 1998. Springer Verlag.

[SSG+99] Marc Stamminger, Annette Scheel, Xavier Granier, Frederic Perez-Cazorla, George Drettakis, and Francois Sillion. Efficient glossy global illumination with interactive viewing. In *Graphics Interface '99*, pages 50–57, San Francisco, CA, June 1999. Morgan Kaufmann.

[SSI99] Imari Sato, Yoichi Sato, and Katsushi Ikeuchi. Acquiring a radiance distribution to superimpose virtual objects onto a real scene. *IEEE Transactions on Visualization and Computer Graphics*, 5(1) :240–254, January–March 1999. ISSN 1077-2626.

[SWI97] Y. Sato, M.D. Wheeler, and K. Ikeuchi. Object shape and reflectance modeling from observation. In *SIGGRAPH '97 Conference Proceedings*, Annual Conference Series, pages 379–387, August 1997.

[TR93] Jack Tumblin and Holly E. Rushmeier. Tone Reproduction for Realistic Images. *IEEE Computer Graphics and Applications*, 13(6) :42–48, November 1993.

[TS67] K. E. Torrance and E. M. Sparrow. Theory for off-specular reflection from roughened surfaces. *Journal of Optical Society of America*, 57(9), 1967.

[TT99] Jack Tumblin and Greg Turk. LCIS : a boundary hierarchy for detail–preserving contrast reduction. In *SIGGRAPH '99*, 1999.

[War92] Gregory J. Ward. Measuring and modeling anisotropic reflection. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 265–272, July 1992.

[War94a] Greg Ward. A contrast-based scale factor for luminance display. In Paul Heckbert, editor, *Graphics Gems IV*, pages 415–421. Academic Press, Boston, 1994.

[War94b] Gregory J. Ward. The RADIANCE lighting simulation and rendering system. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 459–472, July 1994.

[WDP99] Bruce Walter, George Drettakis, and Steven Parker. Interactive rendering using the render cache. In *10th Eurographics Workshop on Rendering*, pages 19–30, June 1999.

[WEH89] John R. Wallace, Kells A. Elmquist, and Eric A. Haines. A Ray Tracing Algorithm for Progressive Radiosity. In *Computer Graphics (ACM SIGGRAPH '89 Proceedings)*, volume 23, pages 315–324, July 1989.

[WG96] S. Weik and O. Grau. Recovering 3-D object geometry using a generic constraint description. In *ISPRS96–18th Congress of the International Society for Photogrammetry and Remote Sensing*, July 1996.

[WHG84] H. Weghorst, G. Hooper, and D. P. Greenberg. Improved computational methods for ray tracing. *ACM Transactions on Graphics*, 3(1) :52–69, January 1984.

[Whi80] Turner Whitted. An improved illumination model for shaded display. *Communications of the ACM*, 23(6) :343–349, June 1980.

[Wil78] Lance Williams. Casting curved shadows on curved surfaces. In *Computer Graphics (SIGGRAPH '78 Proceedings)*, volume 12, pages 270–274, August 1978.

[WNDS99] Mason Woo, Jackie Neider, Tom David, and Dave Shriner. *OpenGL(r) 1.2 Programming Guide*. Addison-Wesley, 1999. Third Edition : The Official Guide to Learning OpenGL, Version 1.2.

[WSI97] M.D. Wheeler, Y. Sato, and K. Ikeuchi. Consensus surfaces for modeling 3D objects from multiple range images. In *DARPA Image Understanding Workshop*, 1997.

[YDMH99] Y. Yu, P.E. Debevec, J. Malik, and T. Hawkins. Inverse global illumination : Recovering reflectance models of real scenes from photographs. In *SIGGRAPH '99*, 1999.

[YM98] Y. Yu and J. Malik. Recovering photometric properties of architectural scenes from photographs. In *Computer Graphics (ACM SIGGRAPH '98 Proceedings)*, July 1998.

[ZDFL95] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence Journal*, 78 :87–119, October 1995.

[ZFD97] I. Zoghlami, O. Faugeras, and R. Deriche. Using geometric corners to build a 2D mosaic from a set of images. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, Puerto Rico, June 1997. IEEE.