THÈSE

présentée par

Jérôme Blanc

pour obtenir le grade de docteur

de l'Institut National Polytechnique de Grenoble

(Arrêté ministériel du 30 Mars 1992)

Spécialité : informatique

Synthèse de nouvelles vues d'une scène 3D à partir d'images existantes

---

Date de soutenance : 27 janvier 1998

Composition du jury :	Président :	Claude Puech
	Rapporteurs :	Gérard Giraudon
		Jean-Claude Paul
	Examinateurs :	Roger Mohr
		Luc Robert

Résumé

La synthèse d'images a pour but de calculer des vues aussi réalistes que possible d'une scène tridimensionnelle définie par un modèle géométrique 3D, augmenté de certaines informations photométriques : couleurs, textures, matériaux, et nature de leurs interactions avec la lumière.

Classiquement, pour ces applications, il est nécessaire d'effectuer une première étape de modélisation manuelle de la scène synthétisée. Cette étape est longue et fastidieuse, et peut demander plusieurs hommes-mois pour des scènes que l'on veut très réalistes, donc complexes : scènes comportant un grand nombre d'objets, de couleurs et de textures ; paysages naturels. Or, nous constatons qu'il n'est pas toujours nécessaire d'atteindre un haut niveau de détail pour produire des images réalistes, car la texture des objets synthétiques peut suffire à faire illusion.

Aussi, nous proposons de modéliser une scène existante, non pas à partir d'une description mathématique explicite, mais à partir de quelques photographies, ou d'une séquence vidéo. Cette technique est appelée *synthèse de nouvelles vues d'une scène* 3D *à partir de vues existantes*, et constitue un domaine innovant de la vision par ordinateur, et de son application à la synthèse d'images. Formellement, le but en est le suivant : à partir de quelques images d'une scène tridimensionnelle, nous voulons synthétiser de nouvelles images de cette scène, sous un angle de vue inédit. Les applications de ce procédé sont toutes celles de la synthèse d'image, et de la Réalité Virtuelle : simulation, entraînement, commerce, loisirs, ainsi que des applications en compression vidéo. L'apport d'une telle technique est de supprimer totalement l'étape de modélisation propre à tous les systèmes de synthèse d'images.

Pour cela, nous faisons appel à des méthodes classiques de vision par ordinateur, concernant la perception tridimensionnelle. Nous recensons les solutions applicables, nous les adaptons, nous les intégrons, puis nous évaluons la qualité du processus global, sur des critères quantitatifs et qualitatifs. Des tests sont menés sur des données synthétiques (images de synthèse), et des données réelles (photographies).

Abstract

The aim of image synthesis is to compute realistic views of a three-dimensional scene, defined by a geometric 3D model, along with some photometric information: colour, texture, material, and their interaction with light.

For these applications, it is usually necessary to perform a first step consisting in manually modelling the synthesized scene. This is a long and tedious task, and may need several man-months for realistic, hence complex scenes: scenes made of many objects, colours or textures; natural landscapes. We notice however that achieving a high level of detail is not always necessary, as the synthetic objects texture may be enough to give illusion.

Therefore, we propose to model an existing scene, not from an explicit mathematical description, but from a few photographs, or a video sequence. This technique is called *image synthesis of a* 3D *scene from existing views*, and makes up an innovative topic of computer vision, and its application to image synthesis. More formally, our aim is the following: knowing some views of a three-dimensional scene, we want to synthesize new views of this scene, under a new point of view. Applications of this technique are the ones of image synthesis and Virtual Reality: simulation, training, marketing, entertainment, as well as video stream compression. The contribution of our technique is to totally suppress the modelling stage, common to every image synthesis system.

For this, we make use of classical methods of computer vision regarding three-dimensional perception. We list the applicable solutions, we adapt them, we integrate them, then we evaluate the quality of the whole process, on a quantitative and qualitative basis. Tests are led on synthetic data (synthetic images), and real data (photographs).

Remerciements

Je tiens tout d'abord à remercier Roger Mohr pour m'avoir accueilli dans son laboratoire, au sein du projet Movi, et de m'avoir proposé un sujet de dea, puis de thèse, presque sur mesure. Son enthousiasme, son soutien, sa confiance et sa disponibilité m'ont permis de travailler dans des conditions optimales pendant ces quatre courtes années. Merci d'avoir su, avec tes encouragements et tes idées, me guider tout au long de ce travail, et de m'avoir également permis de m'exprimer et de poursuivre mes propres intuitions.

Je remercie vivement les personnes qui m'ont fait l'honneur d'avoir participé à mon jury : mes rapporteurs Mm. G. Giraudon et J-C. Paul pour leurs commentaires constructifs sur le manuscrit, ainsi que M. L. Robert, et M. C. Puech pour l'avoir présidé. Je souhaite bonne chance à L. Robert et à son entreprise naissante !

Merci aux personnes qui se sont intéressées à ce travail et avec qui j'ai pu collaborer : D. Canu, de Matra Ms2i, qui a soutenu ce projet ; au sein de l'équipe Movi, B. Boufama, qui m'a initié à ce travail, P. Bobet pour notre collaboration fructueuse, ainsi que É. Malevergne, G. Robert et S. Livatino qui, par leurs recherches et leurs réalisations, ont fait avancer cette tâche. Pour nos discussions amicales et enrichissantes, merci à G. le Mestre et D. Pelé, au ccett (Rennes), et à L. Oisel et L. Morin, à l'irisa, qui travaillent ou ont travaillé sur des préoccupations très proches des nôtres. Bonne continuation à M. Lhuillier, qui poursuit sur le même sujet.

Merci aussi au personnel de l'inria Rhône-Alpes d'avoir assuré nos exceptionnelles conditions de travail, tout particulièrement notre assistante D. Herzog, qui a su garder le sourire malgré toutes les gaffes que j'ai pu commettre : billets d'avion perdus, avions ratés, matériel volé... !

Il est difficile de citer toutes les personnes que j'ai côtoyées et qui ont pu m'aider. Je tiens néanmoins à remercier nos «géomètres» P. Sturm, B. Triggs et L. Quan pour avoir répondu à mes nombreuses questions. L'ambiance de travail a été détendue, amicale, chaleureuse même, et j'en suis reconnaissant à tous les membres du projet Movi. Merci en particulier à mon ami et co-bureau B. Lamiroy pour nos nombreux fous rires, et à notre troisième co-bureau G. Olague pour avoir supporté nos fous rires avec une patience stupéfiante. Merci à C. Gauclin pour ses nombreuses lectures et relectures du manuscrit, ainsi qu'à L. Lamiroy.

De tout cœur enfin, je remercie ma famille, et mes parents, qui par leur soutien, leur amour et leur tolérance, m'ont permis d'accomplir ce travail dans la sérénité.

Stéphane, ces années de thèse ont été des années merveilleuses à tes côtés. Tu m'as soutenu dans les moments où j'en avais besoin ; tu as toujours été présent pour me réconforter et me donner ton amour ; puisse la vie t'apporter autant de joies et de bonheurs que ceux que tu m'as si tendrement offerts...

Table des matières

1 Introduction **1**

1.1 Objectifs . . . 1
1.1.1 Intérêts . . . 1
1.1.2 Cadre formel . . . 2
1.2 Deux problématiques . . . 3
1.3 Applications . . . 4
1.3.1 Entraînement . . . 4
1.3.2 Simulation . . . 5
1.3.3 Commerce, conservation, loisirs . . . 5
1.3.4 Compression de données . . . 6
1.3.5 Classification . . . 7
1.4 Plan du rapport . . . 7

2 État de l'art **9**

2.1 Approches non géométriques . . . 9
2.1.1 *Morphing Interpolation* . . . 9
2.1.2 Combinaisons linéaires . . . 10
2.1.3 Utilisation de vecteurs propres . . . 11
2.2 Approches géométriques . . . 11
2.2.1 Morphing exact . . . 11
2.2.2 Utilisation de la géométrie épipolaire . . . 12
2.2.3 Utilisation des relations trilinéaires . . . 13
2.2.4 Modèle 3D explicite . . . 16
2.2.5 Autres approches . . . 18
2.3 Mosaïques . . . 19
2.3.1 Mosaïques sans information 3D . . . 19
2.3.2 Mosaïques avec information 3D . . . 20
2.3.3 Relation avec le transfert classique . . . 21
2.4 Conclusion . . . 21

3 Appariement **23**

3.1 Introduction . . . 23
3.2 État de l'art de l'appariement . . . 25

3.2.1 Hypothèses de base . . . 26
3.2.2 Mesures de ressemblance . . . 28
3.2.3 Algorithmes d'appariement . . . 41
3.3 Comment évaluer? . . . 55
3.3.1 Scènes planes . . . 55
3.3.2 Scènes quelconques . . . 56
3.3.3 Autres méthodes . . . 56
3.3.4 Images synthétiques . . . 57
3.4 Nos méthodes d'appariement . . . 64
3.4.1 Appariement dense / appariement épars . . . 64
3.4.2 Schéma des opérations . . . 65
3.4.3 Choix d'un algorithme . . . 66
3.4.4 Proposition de nouveaux algorithmes . . . 67
3.4.5 Choix d'une mesure . . . 70
3.4.6 Proposition de nouvelles mesures . . . 70
3.4.7 Aspects algorithmiques . . . 73
3.4.8 Calcul de la géométrie épipolaire . . . 75
3.4.9 Régularisation . . . 78
3.4.10 Affinage d'appariements . . . 79
3.5 Évaluation . . . 83
3.5.1 Évaluation sur images de synthèse — 1 . . . 83
3.5.2 Évaluation sur images de synthèse — 2 . . . 114
3.5.3 Évaluation sur images de synthèse — 3 . . . 125
3.5.4 Évaluation sur images réelles . . . 132
3.6 Conclusion . . . 139

4 Transfert **143**
4.1 Introduction . . . 143
4.2 Critère de qualité . . . 143
4.3 Étalonnage . . . 145
4.3.1 Nécessité d'un étalonnage fort . . . 146
4.4 Mosaïques . . . 147
4.4.1 Tests sur images synthétiques . . . 147
4.4.2 Tests sur images réelles . . . 158
4.4.3 Remarques finales sur les mosaïques . . . 160
4.5 Construction d'une représentation 3D . . . 161
4.5.1 Calcul des matrices de projection . . . 161
4.5.2 Reconstruction robuste . . . 164
4.5.3 Construction d'un maillage . . . 168
4.5.4 Calcul des textures . . . 172
4.6 Évaluation sur images de synthèse — 1 . . . 175
4.6.1 Reconstruction 3D . . . 175
4.6.2 Synthèse d'images à partir de points . . . 177
4.6.3 Synthèse d'images à partir de triangles . . . 183

4.7 Évaluation sur images de synthèse — 2 . . . 186
4.7.1 Reconstruction 3D . . . 186
4.7.2 Synthèse d'images à partir de points . . . 187
4.7.3 Synthèse d'images à partir de triangles . . . 191
4.8 Évaluation sur images de synthèse — 3 . . . 194
4.8.1 Reconstruction 3D . . . 194
4.8.2 Synthèse d'images à partir de points . . . 195
4.8.3 Synthèse d'images à partir de triangles . . . 198
4.9 Évaluation sur images réelles . . . 202
4.9.1 Synthèse d'images à partir de points . . . 202
4.9.2 Synthèse d'images à partir de triangles . . . 206
4.10 Conclusion . . . 210

5 Conclusion **213**
5.1 Contribution . . . 213
5.2 Conclusions . . . 215
5.3 Futurs développements . . . 216

Chapitre 1

Introduction

1.1 Objectifs

Imaginons qu'à partir de quelques photographies du *Titanic*, nous puissions le visualiser en trois dimensions sur un écran, lui faire subir des rotations, des transformations, et l'incruster dans une scène de synthèse. Pour cela, nous n'aurions pas eu besoin d'un accès physique à l'objet, aujourd'hui disparu, ni à ses plans, ou à d'autres formes de mesure ou de modélisation. Seules quelques photos auraient été nécessaires.

Ceci est un exemple de *synthèse d'images à partir de vues réelles*, qui constitue le but de notre travail.

1.1.1 Intérêts

Les techniques classiques de synthèse d'images s'attachent toutes à produire des vues d'une scène de modèle 3D connu. La scène à synthétiser (par exemple, le *Titanic*) doit avoir été entièrement modélisée au préalable.

Phase de modélisation Lors de cette modélisation, l'utilisateur doit décrire à l'aide d'un formalisme approprié les caractéristiques géométriques (forme) et photométriques (couleurs, textures planes) de l'objet. Il le fait généralement à la main, assisté par un logiciel spécialisé lui permettant de décrire rapidement et de placer les objets dans le volume de la scène.

Phase de rendu Le système de synthèse d'images génère ensuite des images d'un réalisme maximal vis-à-vis du modèle de la scène, et des lois physiques de propagation de la lumière, prenant en compte les réflexions, réfractions, diffusions, et interactions avec les

matériaux composant la scène. Ceci est le calcul de rendu, et constitue l'axe essentiel de la recherche actuelle en synthèse d'images.

Bien que ces méthodes donnent des résultats très satisfaisants et de très haute qualité (photographique et cinématographique), elles sont inapplicables en l'état à des scènes *réelles*, qui sont vastes et complexes comme par exemple un paysage forestier, ou une ville entière. Les modèles décrivant ces scènes devraient en effet contenir un grand nombre de détails pour être réalistes (cas de la forêt), ou tout simplement un trop grand volume de données (cas de la ville), difficilement gérable en pratique. Cela pose deux types de problèmes.

1. Les méthodes de synthèse d'image sont gourmandes en temps de calcul. Si la scène est trop complexe, calculer une seule image peut déjà demander plusieurs heures de calcul sur des machines spécialisées. Ceci peut être considéré comme un problème secondaire, car la puissance de calcul des machines évolue exponentiellement avec le temps. Cependant, des calculs temps-réel peuvent être dès à présent nécessaires pour, par exemple, la simulation à retour d'effort, ou les applications de la réalité virtuelle.

2. Une scène complexe demande une modélisation fastidieuse, pouvant se chiffrer en hommes-années. Ainsi, il est hors de question de modéliser chaque bâtiment et chaque rue d'une ville entière à la main, ou de modéliser les branches d'un arbre dépouillé en hiver. En revanche, la somme de données décrivant les détails d'une forêt au printemps est le plus souvent inutile, car un réalisme suffisant peut être obtenu par un modèle d'arbre relativement simple, pourvu d'une texture de feuillage faisant illusion.

Pour remédier à ces problèmes, ce que nous proposons est de définir la scène non pas par un *modèle tridimensionnel*, mais par des *vues (bidimensionnelles)* réelles de cette scène. Par «vues réelles», nous entendons par exemple des photographies de la scène, ou des vidéos. La scène doit donc exister réellement, et être physiquement accessible à la prise de vues.

Nous pensons ainsi supprimer l'étape de modélisation (elle est intrinsèque), et accélérer l'étape de rendu. En effet, les vues disponibles de la scène contiennent les informations géométriques nécessaires, et les informations de texture et de couleur sous une forme déjà «rendue», en quelque sorte, puisque les objets sont éclairés par une source de lumière physique (réelle). Ces informations pourront donc être reprises et transformées par notre système de synthèse d'images, afin de générer de nouvelles vues bidimensionnelles de cette même scène.

1.1.2 Cadre formel

Nous décrivons ci-dessous plus formellement en quoi consiste notre étude.

Le but est le suivant : étant données N images d'une même scène rigide et statique, prises depuis N points de vue, on désire synthétiser une $N+1^{\text{e}}$ image de cette même scène (mêmes conditions d'illumination), sous un angle de vue inédit. L'image synthétique devra

paraître aussi naturelle (réaliste) que possible. Nous nous autorisons éventuellement à utiliser des informations sur les dispositifs de prise de vues, comme par exemple les positions des N caméras (paramètres extrinsèques), ou leurs caractéristiques optiques (paramètres intrinsèques).

Les N images seront *a priori* traitées comme N photographies non ordonnées ; elles pourront éventuellement constituer une séquence de photos, ou même une séquence vidéo, et on pourra dans ce cas tirer parti de la continuité temporelle de la séquence (on indiquera dans les chapitres suivants ce qui devrait être changé, en particulier l'appariement).

Pour une synthèse d'images réaliste vis-à-vis des lois de la physique, notre système doit pouvoir, à partir des photographies fournies, capturer des informations de relief et de structure sur la scène 3D observée. Il nous faut donc au minimum $N = 2$ photos de la même scène. Certains systèmes proches fonctionnent à partir d'une seule photo, et jouent sur des déformations de l'image (techniques de *morphing*) pour donner l'illusion d'un changement de point de vue ; les images calculées par de tels systèmes ne sont cependant pas physiquement valides, c.-à-d. elles ne représentent pas l'objet tel qu'il serait réellement observé depuis les nouveaux points de vue.

Nous nous situons donc dans un domaine à mi-chemin entre la synthèse d'images et la vision par ordinateur ; notre recherche ne porte pas sur la synthèse d'images proprement dite, mais sur les techniques de vision par ordinateur pouvant être appliquées ou adaptées à notre problème.

1.2 Deux problématiques

Dans les techniques de synthèse d'images à partir de vues existantes, nous confondons en fait deux problématiques, qui sont :

$\mathcal{P}1$: synthétiser de nouvelles images ;

$\mathcal{P}2$: calculer une représentation tridimensionnelle de la scène.

Une telle séparation des problématiques est aussi proposée par L. Robert à l'INRIA [Rob 97]. Le but de notre travail est bien de résoudre $\mathcal{P}1$, et non $\mathcal{P}2$.

Cependant, la résolution de $\mathcal{P}2$ entraîne celle de $\mathcal{P}1$, car si l'on dispose d'un modèle 3D de la scène, il est aisé d'en générer de nouvelles vues. De plus, un modèle tridimensionnel permet des traitements géométriques plus généraux, comme des déformations, des simplifications, des augmentations ou des incrustations d'autres objets. Une représentation 3D peut en effet être manipulée par toutes sortes d'outils standard, comme des logiciels de synthèse d'images, ou des éditeurs de modèles.

Aussi, ces avantages nous pousseront à résoudre $\mathcal{P}2$. Nous allons donc tenter de générer une représentation tridimensionnelle d'une scène décrite par des images, tout en gardant à l'esprit que, même si le résultat est imparfait, l'essentiel est bien pour nous d'obtenir de nouvelles images, et de résoudre le problème $\mathcal{P}1$. Nous montrerons d'ailleurs qu'un modèle imprécis est tout à fait suffisant pour générer des images d'une qualité acceptable. Un produit comme QuickTime VR de chez Apple, par exemple, permet de visualiser sous

des angles multiples une scène élaborée à partir de photos, sans devoir reconstruire un modèle tridimensionnel au préalable. En visualisant un film QuickTime VR, l'utilisateur est immergé dans la scène ; il peut simuler des mouvements de tête dans les 4 directions, pour inspecter les parties de la scène situées en dehors de son champ de vision direct. Nous reviendrons sur ces techniques dites de «mosaïque». QuickTime VR est décrit en 2.3.1.

1.3 Applications

Les applications de cette technique sont toutes les applications usuelles de la synthèse d'images pour la Réalité Virtuelle : entraînement, simulation, commerce, loisirs. On peut aussi l'appliquer à la compression de données, et des recherches sont en cours dans ce sens.

1.3.1 Entraînement

Pour toutes sortes d'interventions en milieu hostile : nucléaire, militaire, spatial, il est nécessaire de préparer et d'entraîner les hommes à évoluer dans leur futur milieu d'action. Ceci recouvre les simulateurs de vol et d'entraînement militaires, dont le but est de former les pilotes à leur mission ; en les plongeant dans une situation simulée le plus fidèlement possible, ils apprennent à repérer la topographie du site et l'emplacement des objectifs. Ceci recouvre également des missions d'intervention dans le nucléaire civil : des simulateurs ont pour but d'entraîner les agents à intervenir rapidement en cas d'incident, en leur permettant de se familiariser à la géographie du lieu à l'aide d'une représentation graphique tridimensionnelle réaliste.

Dans le cas des simulateurs de vol, le modèle de la scène est construit à partir de vues aériennes ou satellitaires. Les images constituent des couples stéréo, ou sont de simples vues monoculaires. Le travail de modélisation consiste à reconstruire la topologie du lieu observé (ainsi que les bâtiments), à partir de ces photos ; ce travail est encore très souvent manuel ou semi-manuel, et demande un temps considérable. Aussi, beaucoup de travaux ont pour but d'automatiser cette tâche, et nous établirons des liens entre ces systèmes automatiques, et notre approche. Une revue de ces travaux est présentée dans [Bla 97].

Il faut remarquer que les images disponibles ne sont pas toujours bien adaptées à la tâche de reconstruction 3D. Ainsi, il est difficile d'obtenir un couple stéréo d'images satellitaires, car il faut que le satellite survole deux fois le même site, sous un angle différent, aux mêmes heures et sous les mêmes conditions météo, et à des instants assez proches (pour que les deux vues de la scène soient aussi semblables que possible). Aussi, certains satellites sont équipés de systèmes de prise de vues stéréoscopiques. Notons que dans le cas d'images monoculaires, une reconstruction tridimensionnelle automatique est bien sûr impossible, et il faut disposer d'autres sources d'information : cartographiques, ou suppositions sur la hauteur des bâtiments, par exemple. Enfin, la résolution, excellente pour des images aériennes (mais il faut que le site soit survolable à basse ou moyenne altitude), est moins bonne dans le cas d'images satellitaires : 1 pixel représente souvent 1 mètre au sol (20 cm pour les meilleurs satellites militaires d'observation).

Le cadre est un peu différent dans le cas des simulateurs civils (p. ex. pour le nucléaire), car les images sont disponibles en aussi grande quantité et précision que nécessaire. Ainsi,

lors de la construction d'une centrale nucléaire, les architectes prennent plusieurs milliers de photographies du site, et les archivent sur un vidéodisque. Ces images permettent ensuite au personnel de la centrale de se familiariser avec tous les recoins du site, rendus désormais inaccessibles par les radiations, afin de pouvoir intervenir sans délai en cas d'incident, en se repérant facilement pour agir vite et subir une exposition minimale. De tels systèmes de formation sont opérationnels, mais ils ne présentent qu'une collection d'images fixes, depuis des points de vue figés. On pourrait imaginer les étendre pour synthétiser les images intermédiaires, ce qui permettrait de visualiser les déplacements de façon continue (images animées), rendant ainsi l'immersion de l'opérateur plus naturelle.

1.3.2 Simulation

La construction de modèles de villes à grande échelle, à partir de vues aériennes par exemple, a d'autres applications que les simulateurs de vol. Par exemple, la très forte croissance des services de téléphonie mobile oblige les opérateurs à couvrir rapidement des zones urbaines denses. Il est donc nécessaire de savoir où placer les relais hertziens pour une couverture optimale, et ceci est réalisé par simulation, à partir d'un modèle 3D de la ville. Comme il est fastidieux de produire de tels modèles manuellement, des systèmes automatiques de génération à partir d'images aériennes ont ici tout leur sens. Ceci reprend bien sûr la problématique $\mathcal{P}2$ (et non $\mathcal{P}1$), et bien que ce ne soit pas le but premier de notre travail, nous lierons notre approche aux travaux existant dans le domaine.

La simulation d'environnements complexes est aussi nécessaire en robotique. Pour la simulation de la marche d'un robot martien, on pourrait envisager de modéliser le sol martien à partir de photographies, ce qui fournirait un environnement synthétique de complexité réaliste, tout en évitant une saisie manuelle fastidieuse.

L'intérêt de telles simulations a été démontré dans des contextes industriels, parfois de façon spectaculaire. Ainsi au CERN, à Genève, la simulation en images de synthèse du site de construction du futur accélérateur de particules LHC dans le tunnel existant a permis aux ingénieurs de mieux appréhender sa configuration, et d'éviter la construction d'un puits supplémentaire, économisant plusieurs millions de dollars [Bal 96].

Enfin, les problèmes de reconnaissance d'objets, de saisie et d'asservissement, sont aussi liés à l'extraction automatique de modèles à partir de vues.

1.3.3 Commerce, conservation, loisirs

Avec Internet s'est développée la possibilité d'effectuer des transactions financières à distance, et de pratiquer le commerce électronique. Aussi, les chaînes de distribution commencent à rendre disponibles leurs catalogues sur le web, agrémentés de photos et de visualisations plus ou moins animées ou interactives de leurs produits. Certaines galeries marchandes virtuelles ont été créées (IBM, The Internet Mall, la Redoute...) où l'utilisateur pourra à terme se déplacer dans les magasins, afin de visualiser puis de choisir les produits. Certaines implémentations partielles sont actuellement basées sur QuickTime VR (hypermarchés Leclerc).

La conservation est aussi une application de la capture automatique de modèles à partir

d'images. Nous regroupons sous ce terme les activités cartographiques et de préservation du patrimoine. Le bâtiment Otto Wagner à Vienne fait l'objet d'une telle étude, détaillée dans [Str 95], qui consiste à pouvoir visualiser tous les aspects du bâtiment, décrit à l'aide de quelques photos.

On peut encore citer le domaine des loisirs, et de toutes les applications exigeant un grand nombre d'images : visite de musées virtuels, visite de maisons ou d'équipements ménagers pour la vente, de lieux de vacances dans une agence de voyage, jeux vidéo. La visite virtuelle du Louvre, actuellement diffusée sur cédérom, utilise des films QuickTime VR créés à partir de photos.

Enfin, le procédé peut être utilisé en complément des techniques traditionnelles de synthèse d'images, pour en accélérer le calcul. Après avoir constitué un modèle tridimensionnel de la scène synthétique (nous avons vu que ce procédé pourrait lui-même être automatisé), les logiciels de synthèse d'images procèdent ensuite au calcul proprement dit des images synthétiques. Ces calculs de rendu consistent à déterminer la couleur de chacun des pixels constituant les nouvelles images. Les algorithmes de lancer de rayon par exemple, largement employés pour leur réalisme, déterminent le trajet de chaque rayon lumineux atteignant le plan de la caméra virtuelle. On doit tenir compte des interactions avec les matériaux rencontrés, dont la nature est décrite par un modèle mathématique parfois complexe, des phénomènes de réflexion, de réfraction, et de diffusion de la lumière. De tels calculs peuvent demander un temps considérable. On peut pourtant imaginer, à partir de quelques vues complètement synthétisées, de produire d'autres vues de la même scène, à moindres frais, en utilisant notre procédé. Sans espérer atteindre la qualité d'images réellement calculées par *ray tracing*, on peut cependant produire rapidement de nouvelles vues de la scène.

1.3.4 Compression de données

À partir de quelques images d'une scène, nous pouvons calculer d'autres vues de cette scène. Cela a une application immédiate en compression de données : la compression à très fort taux de séquences vidéo. Des recherches sont actuellement poursuivies au CCETT et à l'IRISA, à Rennes, afin d'utiliser la synthèse d'images à partir de vues pour la compression vidéo. Nous avons nous-mêmes exposé cette possibilité dans [Bla 95]. Le principe est de calculer une représentation géométrique grossière des objets filmés, ainsi que les transformations qu'ils subissent : changements de position, ou de point de vue. Pour transmettre le film, on transmet alors en une seule fois cette représentation géométrique. Chaque image successive est ensuite entièrement décrite par les paramètres décrivant le point de vue courant, ce qui constitue un volume de données très faible et indépendant de la taille des images. Connaissant la représentation géométrique de la scène et la position d'observation, le récepteur peut alors reproduire chacune des images.

Notons que de façon similaire, la norme de compression vidéo à très bas débit MPEG4, en cours de définition, explore les techniques de compression par modèles : les objets composant une image y seront décrits de façon individuelle par leur forme 2D, leur texture, peut-être même leur modèle 3D.

1.3.5 Classification

Parmi les applications envisagées, certaines requièrent un modèle exact et complet de la scène observée, ce qui relève du problème $\mathcal{P}2$; pour d'autres au contraire, un modèle exact n'est pas nécessaire, et nous retrouvons clairement la problématique $\mathcal{P}1$. Nous pouvons classifier les applications de la manière suivante :

Applications relevant de $\mathcal{P}1$: formation (familiarisation à un environnement, p. ex. paysages des simulateurs de vol), commerce, préservation, loisirs, compression vidéo, synthèse d'images et visualisation en général.

Applications relevant de $\mathcal{P}2$: simulateurs (s'il y a interaction avec la scène), applications cartographiques, études de terrain et d'impact (radiotéléphonie), robotique.

1.4 Plan du rapport

Pour synthétiser des vues réalistes d'une scène tridimensionnelle, il est indispensable de disposer d'informations 3D sur cette scène sous une certaine forme (implicite ou explicite), comme par exemple la position dans l'espace de certains points. Il est clair que de telles informations ne peuvent être déduites qu'à partir de l'observation d'au moins 2 vues de cette scène, prises de points de vue distincts. Sur ces $N \geq 2$ vues, nous devrons d'abord procéder à un appariement, afin d'établir des correspondances entre les projections des points 3D dans les N images. Les positions relatives de ces projections dans les images nous permettront d'inférer les informations de relief nécessaires, et d'alimenter les algorithmes de synthèse d'image.

Certaines méthodes de synthèse ne nécessitent pas d'information 3D. Comme dans QuickTime VR, toutes les images peuvent (doivent) être prises du même point, et seule l'orientation des vues peut changer. Ces méthodes de synthèse seront également abordées.

Notre rapport suivra l'enchaînement logique des opérations.

1. Nous présenterons d'abord un état de l'art sur le problème de la synthèse d'images à partir de vues existantes, et les approches connexes, sous ses aspects $\mathcal{P}1$ et $\mathcal{P}2$ (chapitre 2).

2. Ensuite, nous aborderons les problèmes d'appariement visuel, et les solutions envisagées *dans le cadre de notre étude*. Nous ne prétendons pas résoudre le problème de l'appariement d'une façon générale, mais explorer et évaluer les solutions adaptées à notre problème (chapitre 3).

3. Enfin, nous exposerons et évaluerons les méthodes de synthèse utilisées dans notre système (chapitre 4).

Sauf mention contraire, nous nous placerons systématiquement dans le cadre de caméras perspectives suivant le modèle sténopé ; nous utiliserons des notations en coordonnées homogènes. Nous supposerons connues les notions géométriques nécessaires (une introduction pourra être trouvée dans [Moh 93]).

Chapitre 2

État de l'art

La synthèse d'images à partir de vues existantes est un domaine très récent. Nous avons commencé nous-mêmes à nous y intéresser en 1994, et à l'époque, nous ne connaissions qu'une seule autre personne travaillant sur ce sujet, de façon indépendante (S. Laveau [Lav 96], à l'INRIA de Sophia-Antipolis). Aujourd'hui les techniques sont devenues plus matures, et de nombreuses publications apparaissent chaque année. Ci-dessous, nous recensons les principaux travaux, et tentons de les organiser. Dans ces travaux, l'opération de synthèse de nouvelles vues est parfois appelée *transfert*: connaissant les N projections d'un point de l'espace dans N images, il s'agit de le *transférer* dans une $(N+1)^{\text{e}}$ image, c.-à-d. de calculer la position où il se projetterait dans cette nouvelle image.

2.1 Approches non géométriques

Nous plaçons dans cette catégorie les systèmes de synthèse d'images comme le *view morphing*. Ils fournissent des images assez réalistes, mais qui ne sont pas géométriquement correctes. Aussi, le réalisme est limité à une certaine zone géométrique. Plus l'observateur s'éloigne de ce *domaine de validité*, plus l'image synthétique se dégrade, et devient inacceptable en termes de géométrie.

2.1.1 *Morphing Interpolation*

Le *morphing* est très populaire en synthèse d'images pour le grand public, car il permet justement des effets spectaculaires et disproportionnés. L'utilisateur dispose d'une seule image, sur laquelle il place des points de contrôle, qui sont considérés comme des attaches fixes sur la surface élastique de l'image. Ces points peuvent ensuite être déplacés pour étirer ou comprimer l'image dans certaines zones.

Cela peut être appliqué à la synthèse de nouvelles vues: il suffit de disposer de deux vues, une vue initiale et une vue finale, sur lesquelles on place des points de contrôle en

correspondance. En interpolant les positions de ces points de contrôle, on peut synthétiser des images intermédiaires, et dans une certaine mesure, extrapoler. Un exemple de vue interpolée par un morphing est donné en figure 2.1.

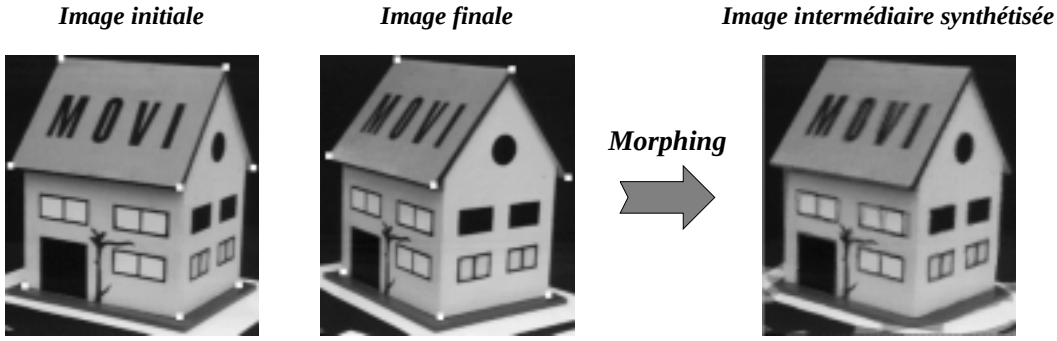


FIG. 2.1: *On peut générer de nouvelles vues grâce au morphing. L'aspect flou provient du fait que le morpher combine les textures des images de référence.*

La vue synthétisée est géométriquement d'assez bonne qualité, en n'utilisant que 8 points de contrôle. Le *morpher* construit un maillage triangulaire des points de contrôle, puis applique des déformations affines des textures à l'intérieur de chaque triangle. Cette approximation est acceptable, car la scène est polyédrique, donc composée de plans (une transformation homographique serait cependant plus adaptée).

2.1.2 Combinaisons linéaires

L'idée de T. Werner, à l'Université de Prague, est assez semblable au *morphing interpolation*. Dans [Wer 94, Wer 95], il expose un système de synthèse de nouvelles vues dont l'hypothèse de base est que la position de la projection d'un point dans une image est une combinaison linéaire des projections du même point dans les autres images. Une telle modélisation avait été proposée par S. Ullman et R. Basri en 1991, pour la reconnaissance d'objets [Ull 91]. Si le point P de l'espace se projette dans N images en $(p_i)_{i=1..N}$, alors la position de p_{N+1} dans une $N+1^{\text{e}}$ image est donnée par l'équation 2.1.

$$p_{N+1} = \sum_{i=1}^{i=N} \alpha_i p_i \tag{2.1}$$

Une première étape de mise en correspondance permet d'évaluer les α_i. Cette relation est évidemment fausse en général. Dans le cas de caméras orthographiques, T. Werner propose une relation exacte, mais affine, donnant la position d'un point p_3 dans une troisième vue connaissant ses positions p_1 et p_2 dans deux vues de référence (éq. 2.2).

$$\left\{ \begin{array}{rcl} p_3^x & = & \alpha_1\ p_1^x + \alpha_2\ p_2^x + \alpha_3 \\ p_3^y & = & \alpha_4\ p_1^y + \alpha_5\ p_2^y + \alpha_6 \end{array} \right. \tag{2.2}$$

L'approche est alors géométriquement plus correcte. Les résultats sont d'ailleurs bons, mais les angles de vue choisis sont très proches des vues de référence. Cela ressemble beaucoup à la formulation des relations trilinéaires pour la synthèse de nouvelles vues (voir 2.2.3), mais ces dernières sont exactes dans le cas général : caméras perspectives, déplacement quelconque.

2.1.3 Utilisation de vecteurs propres

Comme de nombreux auteurs [Sir 87, Tur 91, Mur 95, Hut 96, Pog 96], D. Casasent à CMU représente les différents aspects d'un objet 3D sous différents points de vues par leurs coordonnées dans un espace de caractéristiques à N dimensions. Cet espace peut par exemple être l'espace des vecteurs propres des aspects de l'objet : chaque image de l'objet est considérée comme un vecteur de pixels ; si l'on dispose d'une base comptant un grand nombre d'images, une analyse en composantes principales ou une décomposition SVD fournit les vecteurs propres (les «images-propres») de la base.

Il propose ensuite dans [Cas 97] d'utiliser cette description pour générer de nouvelles vues. En effet, un point quelconque dans l'espace des paramètres représentant une image, combinaison linéaire des images-propres, on pourrait dans une certaine mesure générer des vues interpolées par ce moyen : ce sont simplement des points intermédiaires dans l'espace des paramètres.

2.2 Approches géométriques

La majorité des approches s'attachent à calculer des vues physiquement valides, c.-à-d. semblables à celles qui seraient réellement vues par une caméra placée à l'endroit donné. Cela nécessite des informations géométriques sur le relief de la scène, donc un appariement, et au minimum un étalonnage faible [1] (géométrie épipolaire connue).

2.2.1 Morphing exact

À l'université du Wisconsin à Madison, S.M. Seitz et C.R. Dyer modifient dans [Sei 95] (cas de caméras affines) puis [Sei 96] (cas général) l'utilisation habituelle du morphing. Ils montrent que si les images de référence respectent la contrainte d'ordre, ou *monotonicité*, on peut obtenir des vues interpolées physiquement valides. L'idée est de rectifier les deux images, d'interpoler la nouvelle vue (intermédiaire) linéairement par morphing, et de dé-rectifier ensuite cette vue synthétique, en la ramenant dans le plan de la caméra virtuelle (figure 2.2).

1. Celui-ci peut parfois apparaître sous une forme implicite.

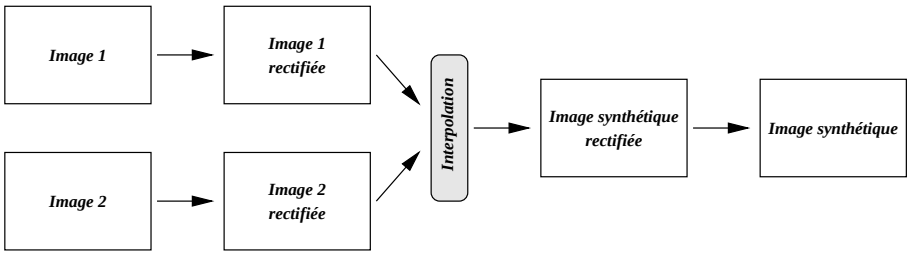


FIG. 2.2: *Schéma de production de nouvelles vues par morphing physiquement valide.*

Le contrôle du positionnement de la caméra virtuelle, qui est étalonnée, se fait en plaçant à la main les 4 points de contrôle de la transformation, pour chaque nouvelle vue synthétique.

Une étape d'appariement dense est nécessaire pour pouvoir transférer dans l'image synthétique chaque couple apparié dans les images de référence. Ceci est réalisé par programmation dynamique, ce qui est cohérent avec la contrainte de monotonicité.

2.2.2 Utilisation de la géométrie épipolaire

S. Laveau fait dans [Lav 94b, Lav 94a] du transfert par intersection d'épipolaires. En effet, connaissant des points appariés p_1 et p_2, images d'un point P dans les images 1 et 2, et la matrice fondamentale $F_{1,3}$ (resp. $F_{2,3}$) liant les images 1 et 3 (resp. les images 2 et 3), alors le point p_3 transféré dans l'image 3 se trouve nécessairement sur l'épipolaire conjuguée à p_1 dans l'image 3, ainsi que sur l'épipolaire conjuguée à p_2 dans l'image 3 ; il est donc à l'intersection des deux :

$$p_3 = F_{1,3}\ p_1 \wedge F_{2,3}\ p_2 \tag{2.3}$$

Le contrôle du positionnement de la caméra virtuelle se fait en plaçant à la main 5 points décrivant le plan de la caméra virtuelle et son centre optique. Aucun étalonnage fort n'est nécessaire, puisque les matrices fondamentales suffisent.

L'équation 2.3 n'est pas applicable si les lignes $F_{1,3}\ p_1$ et $F_{2,3}\ p_2$ sont parallèles ou confondues, donc en particulier pour tous les points P du plan trifocal; de plus, les équations deviennent numériquement instables à l'approche du plan trifocal, et l'image synthétisée est inutilisable dans cette zone (figure 2.3).

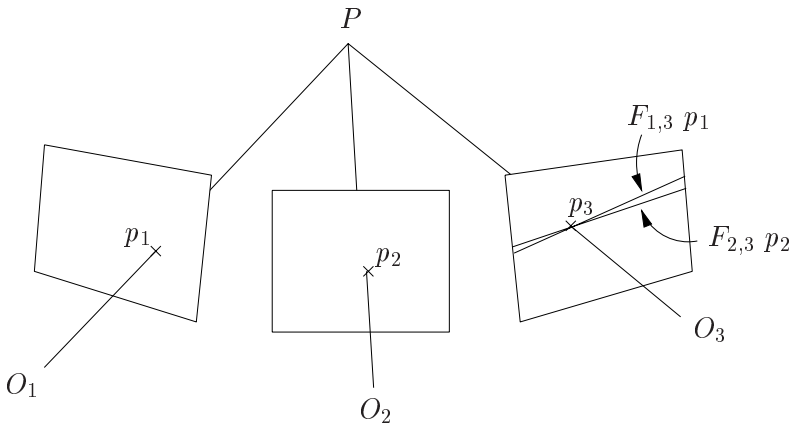


FIG. 2.3: *Le transfert par intersection des épipolaires fonctionne mal pour les points P proches du plan trifocal* $(O_1O_2O_3)$.

Nous montrons nous-mêmes dans [Bla 94] quelques exemples où il est impossible de synthétiser l'image à l'aide de cette méthode. Nous la comparions à la méthode suivante, utilisant les relations trilinéaires.

2.2.3 Utilisation des relations trilinéaires

Les relations trilinéaires ont été établies pour la première fois par A. Shashua, de l'Université de Jerusalem. Il fournit dans [Sha 94] une démonstration complexe de ces relations, qui lient les coordonnées (x_1, y_1), (x_2, y_2) et (x_3, y_3) d'un triplet de points (p_1, p_2, p_3) en correspondance dans 3 images. Ces relations sont données par les équations 2.4, 2.5, 2.6 et 2.7.

$$\begin{cases} \alpha_1+\alpha_2x_1+\alpha_3x_3+\alpha_4y_1+\alpha_5y_2+\alpha_6x_1x_3+ \\ \quad \alpha_7y_1y_2+\alpha_8x_1y_2+\alpha_9x_3y_1+\alpha_{10}x_3y_2+\alpha_{11}x_3y_1y_2+\alpha_{12}x_1x_3y_2=0 & (2.4) \\ \alpha_{13}+\alpha_{14}x_1+\alpha_{15}y_1+\alpha_{16}y_2+\alpha_3y_3+\alpha_{17}y_1y_2+ \\ \quad \alpha_9y_1y_3+\alpha_{10}y_2y_3+\alpha_{18}x_1y_2+\alpha_6x_1y_3+\alpha_{12}x_1y_2y_3+\alpha_{11}y_1y_2y_3=0 & (2.5) \\ \alpha_{19}+\alpha_{20}x_1+\alpha_5x_2+\alpha_{21}x_3+\alpha_{22}y_1+\alpha_8x_1x_2+ \\ \quad \alpha_{23}x_1x_3+\alpha_{10}x_2x_3+\alpha_7x_2y_1+\alpha_{24}x_3y_1+\alpha_{11}x_2x_3y_1+\alpha_{12}x_1x_2x_3=0 & (2.6) \\ \alpha_{25}+\alpha_{26}x_1+\alpha_{16}x_2+\alpha_{27}y_1+\alpha_{21}y_3+\alpha_{18}x_1x_2+ \\ \quad \alpha_{24}y_1y_3+\alpha_{17}x_2y_1+\alpha_{23}x_1y_3+\alpha_{10}x_2y_3+\alpha_{12}x_1x_2y_3+\alpha_{11}x_2y_1y_3=0 & (2.7) \end{cases}$$

Les α_i sont les 27 coefficients représentant la géométrie relative des 3 images. Chez la plupart des auteurs, ils sont regroupés dans un tenseur $3 \times 3 \times 3$. Ce *tenseur trilinéaire* est l'équivalent pour 3 images de la matrice fondamentale 3×3 pour deux images. De même que les coefficients de la matrice fondamentale sont liés par une condition de rang,

les coefficients du tenseur ne sont pas indépendants ; le positionnement relatif de 3 caméras perspectives, sans autre hypothèse, n'est en effet décrit que par 18 paramètres. Le tenseur peut être calculé par les mêmes méthodes que la matrice fondamentale, dont nous reparlerons en 3.4.8.

Nous pouvons utiliser les seules équations 2.4 et 2.5 pour calculer la position du point (x_3, y_3) connaissant le tenseur et les positions de (x_1, y_1) et (x_2, y_2) dans les deux premières images. Nous avons montré dans [Bla 94] que cela donnait de meilleurs résultats en général que la méthode de transfert par intersection des épipolaires, ce qui est logique puisqu'il n'y a pas de point P de l'espace où ces équations sont dégénérées. Les premières applications recensées par A. Shashua sont d'ailleurs le transfert d'images (le «transfert trilinéaire»), et la reconnaissance.

En revanche, nous avons remarqué que le transfert utilisant seulement les équations 2.6 et 2.7 donnait de très mauvais résultats. Nous avons expliqué en détail ce comportement avec P. Bobet dans [Bob 96], et nous en reparlons ci-après ; il était dû à la position relative de nos images.

2.2.3.1 Nature des relations trilinéaires

R. Hartley, L. Quan, puis B. Mourrain et O. Faugeras dans [Fau 95c, Fau 95b], donnent une démonstration plus simple de la relation trilinéaire, qui permet de mieux comprendre sa nature et son comportement.

Si un point P se projette dans 3 images en (x_1, y_1), (x_2, y_2) et (x_3, y_3) *via* les matrices de projection M_1, M_2 et M_3, alors ceci peut s'écrire :

$$\left\{ \begin{array}{lcl} (x_1\ y_1\ 1)^T & \cong & M_1\ P \\ (x_2\ y_2\ 1)^T & \cong & M_2\ P \\ (x_3\ y_3\ 1)^T & \cong & M_3\ P \end{array} \right. \tag{2.8}$$

Si on décompose les matrices de projection en 3 vecteurs-lignes a, b, c :

$$\forall\ i \in 1, 2, 3,\ M_i\ = \left(\begin{array}{|c|} \hline a_i \\ \hline b_i \\ \hline c_i \\ \hline \end{array} \right) \tag{2.9}$$

alors l'équation 2.8 peut être aussi notée :

$$\left(\begin{array}{c} x_1\ c_1 - a_1 \\ y_1\ c_1 - b_1 \\ x_2\ c_2 - a_2 \\ y_2\ c_2 - b_2 \\ x_3\ c_3 - a_3 \\ y_3\ c_3 - b_3 \end{array} \right) \left(\begin{array}{c} X \\ Y \\ Z \\ 1 \end{array} \right) = C_{6\times4}\ P_{4\times1}\ = 0_{6\times1} \tag{2.10}$$

La matrice C est nécessairement de rang inférieur à 4, donc tous ses sous-déterminants 4×4 sont nuls. Si on prend par exemple la première, la deuxième, la troisième et la cinquième ligne de C, on obtient :

$$\begin{vmatrix} x_1\, c_1 - a_1 \\ y_1\, c_1 - b_1 \\ x_2\, c_2 - a_2 \\ x_3\, c_3 - a_3 \end{vmatrix} = 0 \tag{2.11}$$

Si l'on développe l'équation 2.11, on retrouve l'une des équations trilinéaires. On peut trouver ainsi les 4 relations trilinéaires donnant x_3 et y_3 en fonction des données x_1, y_1, x_2, y_2, et les α_i sont des combinaisons linéaires des coefficients des matrices M_i.

De cette façon, la nature d'une relation trilinéaire apparaît clairement. L'équation 2.11 par exemple n'est rien d'autre que la *reconstruction implicite* du point P, par l'intersection de la ligne de vue définie par (x_1, y_1) et du plan de vue défini par x_2, et sa reprojection (implicite) sur la troisième image, en un point d'abscisse x_3.

Le comportement erratique du transfert trilinéaire est aussi expliqué : certaines équations calculent implicitement le point P en utilisant les données (x_1, y_1, x_2), d'autres en utilisant (x_1, y_1, y_2), ou (x_1, x_2, y_2), ou encore (y_1, x_2, y_2) (4 possibilités). Si l'on s'appuie sur une équation du type de 2.11, il est maintenant clair que le calcul sera mal conditionné si la ligne de vue définie par (x_1, y_1) et le plan de vue défini par x_2 sont presque parallèles ; la reconstruction implicite de P sera très imprécise, et le transfert sera mauvais. Dans cette configuration, il serait bien plus avantageux de choisir une équation du type suivant :

$$\begin{vmatrix} x_1\, c_1 - a_1 \\ y_1\, c_1 - b_1 \\ y_2\, c_2 - b_2 \\ x_3\, c_3 - a_3 \end{vmatrix} = 0 \tag{2.12}$$

La ligne de vue définie par (x_1, y_1) et le plan de vue défini par y_2 seront presque orthogonaux, et la reconstruction implicite de P d'autant meilleure.

Si on utilise cette méthode de transfert, il faut donc soigneusement choisir les équations à utiliser, car si elles sont toutes algébriquement équivalentes, leur comportement numérique est en revanche totalement différent.

2.2.3.2 Relations trilinéaires, ou reconstruction explicite ?

Le transfert trilinéaire est donc strictement équivalent à une reconstruction projective *implicite* à partir des deux premières images, suivi d'une reprojection sur la troisième image.

Une reconstruction projective *explicite* est beaucoup plus facilement manipulable. Elle peut aisément être construite à partir de plus de 2 images, alors que le transfert trilinéaire est contraint par nature à n'utiliser que deux images de référence. Elle est aussi plus facilement contrôlable, car elle peut progressivement être transformée en une reconstruction

affine, puis euclidienne (selon les informations externes dont on dispose). Une reconstruction euclidienne est ensuite visualisable et transformable par de nombreux programmes. On peut ainsi la visualiser sous n'importe quel angle, ce qui est notre but.

En revanche, pour pouvoir déplacer la caméra virtuelle lors d'un transfert trilinéaire, il faudrait pouvoir modifier les α_i de façon cohérente, ce qui semble beaucoup moins direct.

Aussi, nous nous attachons dans la suite à la capture d'un modèle 3D explicite, sous une forme projective ou euclidienne. Notre travail se situe entièrement dans cette optique.

2.2.4 Modèle 3D explicite

Nous détaillerons dans les chapitres suivants notre propre approche de la synthèse de nouvelles vues par reconstruction d'un modèle 3D explicite. Nous exposons ici les travaux réalisés dans les différentes équipes de recherche.

À l'université de Hannovre, R. Koch effectuait un travail très proche du nôtre, puisqu'il synthétisait de nouvelles images à partir d'un couple stéréoscopique existant. Dans [Koc 94, Koc 95], il proposait l'algorithme suivant :

1. étalonner les caméras ;
2. rectifier le couple stéréo ;
3. réaliser un appariement dense ;
4. calculer un maillage triangulaire des points appariés ;
5. créer un modèle en facettes planes triangulaires et texturées.

L'appariement était calculé par programmation dynamique, puis affiné par des déformations affines de fenêtres de corrélation. Ensuite, les points 3D reconstruits étaient approximés par une surface plane (recherche d'orientations locales cohérentes), puis maillés. L'intégration de nombreuses images par un filtre de Kalman permettait d'affiner les résultats.

Le travail de P. Debevec à Berkeley [Deb 96a, Deb 96b] consiste à modéliser des bâtiments et des motifs architecturaux pour le rendu, sous d'autres points de vue. L'approche est assez manuelle et très adaptée à des modèles en blocs (bâtiments parallélépipédiques ou prismatiques) ; une fois le modèle 3D obtenu, il est texturé, et peut être visualisé sous n'importe quel angle. Nous la citons néanmoins, car l'étape de synthèse est particulièrement soignée. P. Debevec considère en effet que la texture n'est pas statiquement attribuée au modèle, mais qu'elle peut évoluer dynamiquement, en fonction du point de vue : il applique au modèle 3D une pondération des textures des vues de référence les plus proches du point de vue courant. Sur ses images, le gain de qualité est appréciable, une texture bien choisie pouvant améliorer considérablement la perception d'un modèle 3D par trop approximatif. Cependant, le champ d'application de l'approche de P. Debevec sort de notre cadre, et de très nombreuses personnes suivent en fait une procédure similaire à celle de R. Koch, avec quelques variantes, que nous détaillons ci-dessous.

Toujours à Hannovre, W. Niem [Nie 95] réalise un calcul de *shape from occluding contours*, ce qui impose de placer la scène sur une table tournante pour retrouver sa structure 3D. Les appariements denses sont ensuite convertis en triangles texturés. W. Niem remarque que le *texture-mapping* est de mauvaise qualité si on ne choisit pas soigneusement quelle est l'image de référence dont on doit utiliser la texture. Il implémente donc une méthode de *texture-mapping* pondéré, à la manière de P. Debevec. De plus, les triangles sans texture attribuée, qui font apparaître des trous dans l'image synthétique, sont progressivement comblés par lissage avec leurs voisins. L'inconvénient provient bien sûr de la méthode de *shape from occluding contours*. Elle consiste à reconstruire progressivement la surface de l'objet en observant ses contours occultants, et ne permet pas de capturer les zones concaves, car celles-ci ne génèrent pas de contour occultant, et passent inaperçues. Aussi, quelques auteurs effectuent dans une étape ultérieure un appariement dense, pour retrouver les concavités.

T. Kanade à CMU [Kan 95] réalise un appariement dense multi-oculaire, d'où il tire une triangulation connexe, qui est ensuite découpée en surfaces indépendantes le long des lignes de rupture de disparité. L'utilisation de plusieurs images ne sert qu'à désambiguïser les appariements, et nécessite un environnement expérimental fixe, lourd et complexe, constitué de 51 caméras fixées sur un dôme métallique surplombant la scène observée.

S. Laveau dans [Lav 96] réalise un appariement épars sur quelques points des images de référence, puis effectue une triangulation manuellement.

Au CCETT à Rennes, G. le Mestre [Mes 96] et P. Lechat [Lec 97] réalisent un appariement dense binoculaire, puis le fusionnent pour obtenir un appariement multi-oculaire. Sur la carte de disparité obtenue, ils calculent l'histogramme des profondeurs, ce qui leur permet d'estimer des seuils de segmentation. Le nombre de seuils à détecter doit être fixé à l'avance, et si par exemple on décide de détecter 3 seuils, alors on pourra segmenter la scène en 3 plans. Chacun de ces plans est ensuite triangulé indépendamment, puis les maillages sont simplifiés sur des critères de coplanarité, et affinés sur un critère d'énergie pour mieux s'ajuster au modèle 3D sous-jacent. On obtient un modèle en triangles texturés, qu'on peut mettre sous forme VRML.

Dans [Ois 96], L. Oisel à l'IRISA indique très brièvement comment le même schéma pourrait être suivi pour la compression vidéo à très fort taux. Suite à la segmentation en facettes planes et aux différentes erreurs survenant à chaque étape (appariement en particulier), il est nécessaire d'effectuer une reprise sur les zones d'occultation. La segmentation en facettes planes est réalisée par calcul robuste d'homographies sur des zones de l'image.

D. Scharstein à Cornell University [Sch 96a] réalise un appariement dense par une méthode non décrite, et calcule implicitement une reconstruction 3D point par point de la scène (modèle en nuage de points). Il transfère ensuite chaque point pour produire la nouvelle image. Cela évite de générer un modèle en triangles texturés, mais l'image synthétisée peut comporter des trous (voir la section 3.4.1 à ce sujet).

L'idée de S.M. Seitz et C.R. Dyer dans [Sei 97] est assez similaire : une scène peut être décrite par un ensemble de voxels colorés, qu'il suffit de projeter sur le plan-image de la caméra virtuelle. Néanmoins, leur méthode d'appariement est originale, et nous y reviendrons au chapitre suivant.

L. McMillan [McM 95] se place dans le cas particulier d'images de référence prises

toutes selon le même axe vertical. En chaque point de vue, il prend plusieurs images en rotation autour de cet axe, et construit une image cylindrique composée de ces vues. À l'aide de deux telles images de référence cylindriques, il peut synthétiser une troisième vue (cylindrique) pixel par pixel, en utilisant une forme de reconstruction implicite des points 3D.

Il existe enfin des articles plus théoriques, comme ceux de O. Faugeras et L. Robert, qui expliquent dans [Fau 93a, Fau 94] comment transférer dans une troisième image des points, lignes, courbures, ou coniques vus dans seulement deux images. Ils s'appuient uniquement sur la géométrie épipolaire, avec les dégénérescences connues. Dans [Fau 95a], ils proposent une stratification de la représentation des scènes tridimensionnelles. Ainsi, pour synthétiser une scène, on commence par capturer un modèle projectif (calcul d'appariements et de la géométrie épipolaire), puis affine (calcul du plan à l'infini par recherche de structures parallèles), puis euclidien (intégration de contraintes métriques : angles, longueurs). Ceci est appliqué dans le cadre du projet Esprit *Realise* à la reconstruction de bâtiments à partir de photos aériennes. Dans tous ces travaux, les applications à la synthèse de nouvelles images sont préliminaires, et les algorithmes assez peu détaillés. Les auteurs présentent surtout des voies d'intégration de tous les outils de la géométrie projective nécessaires à cette tâche.

2.2.5 Autres approches

Nous classons ici l'approche de M. Levoy, à Stanford. Dans [Lev 96], il adopte une conception complètement différente du problème de synthèse de vues à partir de vues existantes. Pour lui, la scène est décrite par un *light field*, qui est l'ensemble des rayons lumineux traversant le volume de la scène. Ce champ lumineux peut être représenté par une fonction f à 4 dimensions, donnant la radiance en fonction de la position et de la direction d'observation. Les images de référence ne sont alors que des échantillons bidimensionnels de la fonction f. Il suffit donc d'intégrer un grand nombre d'images de référence pour capturer entièrement f.

Pour cela, M. Levoy utilise un système automatique constitué d'une table tournante sur laquelle on place l'objet, synchronisée avec un éclairage tournant, et d'une caméra contrôlable en position (mouvement planaire), tangage et lacet. Ce système robotisé acquiert de l'ordre de 200 à 8000 images, chacune de taille 128×128 à 256×256, en un temps variant de 15 minutes à 4 heures. Cette énorme quantité de données (jusqu'à 1.6 Go pour une scène) peut être ensuite compressée astucieusement, jusqu'à un facteur 100.

Tous les rayons lumineux étant décrits par f, on peut alors synthétiser la même scène sous n'importe quel angle : il suffit en quelque sorte de découper une tranche bidimensionnelle dans la fonction f pour obtenir une nouvelle image. De même, les effets de profondeur de champ sont intrinsèquement modélisés, et rendus de façon fidèle.

Il est d'ailleurs assez logique d'obtenir des résultats de bonne qualité avec une telle quantité de données traitées. De plus, le système d'acquisition limite son utilisation à des scènes de taille raisonnable (objets manipulables).

2.3 Mosaïques

La construction de mosaïques à partir d'images de référence est aussi un domaine très actif de la vision par ordinateur ces trois dernières années. Il s'agit de recoller toutes les images de référence dans le même repère, de façon à reconstituer une seule vue d'ensemble de la scène.

Certaines méthodes simplifient le problème et le réduisent à un aboutement d'images, avec une déformation adéquate des lignes de couture. Elles ne garantissent pas la correction géométrique de l'image produite. D'autres approches fournissent des images géométriquement correctes, parfois dans des cas particuliers (mouvements de caméra panoramiques).

2.3.1 Mosaïques sans information 3D

Chez Apple, S.E. Chen présente le système QuickTime VR dans [Che 95a]. QuickTime VR est limité à la génération de mosaïques à partir d'images de référence prises par une caméra de position fixe (le centre optique est à une position fixe ; seule l'orientation de la caméra peut changer). Dans ces conditions de projection, aucune information de relief ne peut être capturée sur la scène, puisque le centre optique de la caméra est fixe. Cependant, n'importe quel couple d'images est lié par une homographie planaire (matrice 3×3), qui peut être estimée à partir de 4 correspondances binoculaires seulement [Moh 93]. Ces homographies peuvent servir à amener toutes les images dans le même repère. S.E. Chen les transforme dans un repère cylindrique, qui facilite ultérieurement la visualisation rapide. Le procédé est décrit dans le brevet [Che 95b] de QuickTime VR. Cela est donc valable essentiellement pour des mouvements panoramiques (gauche/droite), les mouvements plongeants (haut/bas) étant moins facilement rendus.

D'une point de vue pratique, l'utilisateur doit prendre des images selon les conditions indiquées, par exemple en plaçant l'appareil photo sur un trépied, et effectuer un panoramique circulaire, avec un certain recouvrement des images (de l'ordre de 30 %), en s'assurant que l'axe de rotation passe bien par le centre optique de l'appareil photo. Ensuite, il désigne manuellement quelques éléments de correspondance entre les images (au moins 4), et le système calcule les homographies, puis projette les images de référence dans le repère cylindrique (figure 2.4).

On remarque que le procédé d'appariement pourrait facilement être automatisé. De plus, la relation homographique s'applique également au cas de caméras en mouvement, pourvu que la scène soit plane, et QuickTime VR pourrait être utilisé dans ces cas ; par exemple, si la scène est un paysage éloigné (donc presque plan), il n'est pas nécessaire que les rotations de la caméra se déroulent strictement autour du centre optique.

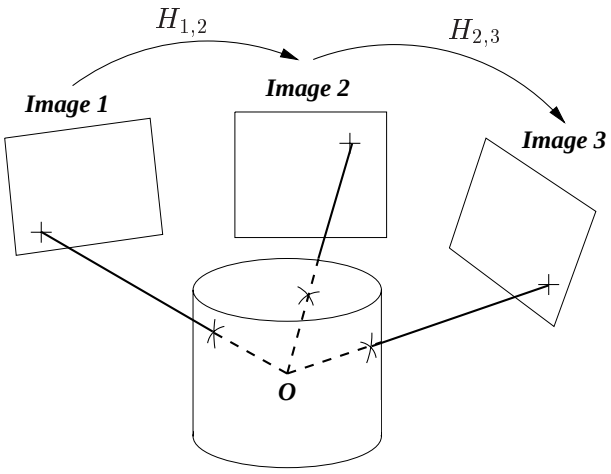


FIG. 2.4: *Dans QuickTime* VR*, toutes les images de référence sont projetées dans le même repère cylindrique.*

2.3.2 Mosaïques avec information 3D

H.S. Sawhney, chez IBM, réalise des mosaïques à modèle complet, c.-à-d. avec information 3D. Dans [Saw 95a, Saw 95b], il estime le mouvement dominant de la séquence d'images, décrit par une transformation affine des coordonnées des pixels 2D. C'est-à-dire il suppose que les points (x_2, y_2) de l'image 2 correspondant aux points (x_1, y_1) de l'image 1 sont tels que :

$$\left\{ \begin{array}{lcl} x_2 & = & ax_1 + by_1 + c \\ y_2 & = & dx_1 + ey_1 + f \end{array} \right. \tag{2.13}$$

Par une minimisation robuste des différences des intensités des pixels $I_2(x_2, y_2) - I_1(x_1, y_1)$, il retrouve les paramètres (a, b, c, d, e, f) du mouvement dominant de l'image. Il estime ensuite l'erreur résiduelle de déplacement pour chaque pixel (la parallaxe), ce qui revient à calculer un appariement dense ; comme pour le calcul du mouvement dominant, cela est réalisé par la minimisation de l'erreur globale de reprojection. Toutes les images peuvent alors être projetées dans le même repère, constituant une seule grande image.

R. Szeliski et S.B. Kang appliquent la même idée : estimation d'un mouvement dominant, puis de la parallaxe résiduelle [Sze 95a], avec un appariement dense à base de splines, comme dans [Sze 95b] (description plus loin en 3.2.2.3).

Enfin, R. Kumar [Kum 94, Kum 95] procède de la même façon, sauf pour la modélisation du mouvement dominant, qui est quadratique :

$$\left\{ \begin{array}{lcl} x_2 & = & ax_1 + by_1 + c + gx_1^2 + hx_1y_1 \\ y_2 & = & dx_1 + ey_1 + f + gx_1y_1 + hy_1^2 \end{array} \right. \tag{2.14}$$

L'équation 2.14 décrit approximativement le mouvement apparent 2D dans les images d'un plan 3D de la scène, dans le cas de petits déplacements. On aurait pu utiliser une

homographie 3×3, toujours valable pour le déplacement (même important) d'un plan, et décrite aussi par 8 paramètres.

2.3.3 Relation avec le transfert classique

Dans les premiers travaux, la construction de mosaïques semblait rester un domaine plutôt séparé du nôtre, bien qu'il concerne aussi la synthèse d'images à partir d'images ; les méthodes, les objectifs, et même le vocabulaire, étaient assez différents.

Nous constatons maintenant une fusion presque complète de ces deux activités. Faire une mosaïque revient à aligner des modèles tridimensionnels sur la même projection, donc à procéder au transfert de plusieurs images de référence dans un repère commun. Il s'agit à chaque fois de procéder à une reconstruction 3D, de façon implicite ou explicite, et de la projeter sur le plan-image d'une caméra virtuelle. Cela est vrai même dans le cas de QuickTime VR, où une reconstruction 3D du modèle est impossible : procéder au transfert homographique d'un point d'une image dans une autre revient en fait à reconstruire la ligne de vue passant par ce point et le centre de la caméra, et à calculer son intersection avec le nouveau plan-image (figure 2.5).

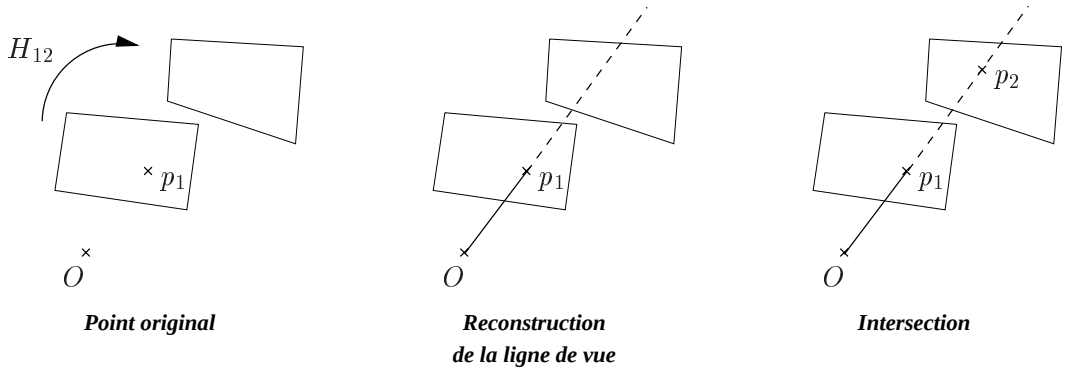


FIG. 2.5: *Le transfert homographique d'un point peut être vu comme la reconstruction de la ligne de vue passant par ce point, suivie de sa reprojection (intersection avec le nouveau plan-image).*

Ce qui ne semblait qu'un axe de recherche connexe est donc entièrement inclus dans le cadre de notre travail. Aussi, nous avons développé nos propres techniques de construction de mosaïques, et elles seront décrites au chapitre 4.

2.4 Conclusion

De notre étude du problème, il ressort que tous les auteurs suivent le même schéma opératoire. Pour synthétiser de nouvelles vues à partir d'images de référence, ils appliquent la procédure suivante :

1. apparier des structures (presque toujours des points) dans les images de référence ;

2. procéder à une forme de reconstruction (projective, affine, ou euclidienne) des structures appariées ;

3. visualiser ces structures sous un nouveau point de vue, c.-à-d. les reprojeter sur le plan-image d'une nouvelle caméra.

Comme précisé en introduction, notre rapport suivra cet enchaînement.

La phase d'appariement constituant un gros problème à elle seule, elle sera étudiée à part, en chapitre 3. Pour les mêmes raisons, ce chapitre contiendra séparément un état de l'art des mesures et des algorithmes d'appariement.

Nous avons vu que les phases de reconstruction et de synthèse pouvaient être intégrées en une seule étape (cas du transfert trilinéaire, et des reconstructions implicites). Cette étape de transfert sera décrite en chapitre 4.

Chapitre 3

Appariement

3.1 Introduction

L'appariement est une étape primordiale de notre travail, ainsi que pour toutes les tâches de la vision stéréoscopique. Il consiste à déterminer quelles sont les projections qui se correspondent dans les images, c.-à-d. quels sont les points 2D représentant les projections d'un même point 3D (voir figure 3.1). Un appariement est donc un n-uplet de points 2D en correspondance, $n \geq 2$.

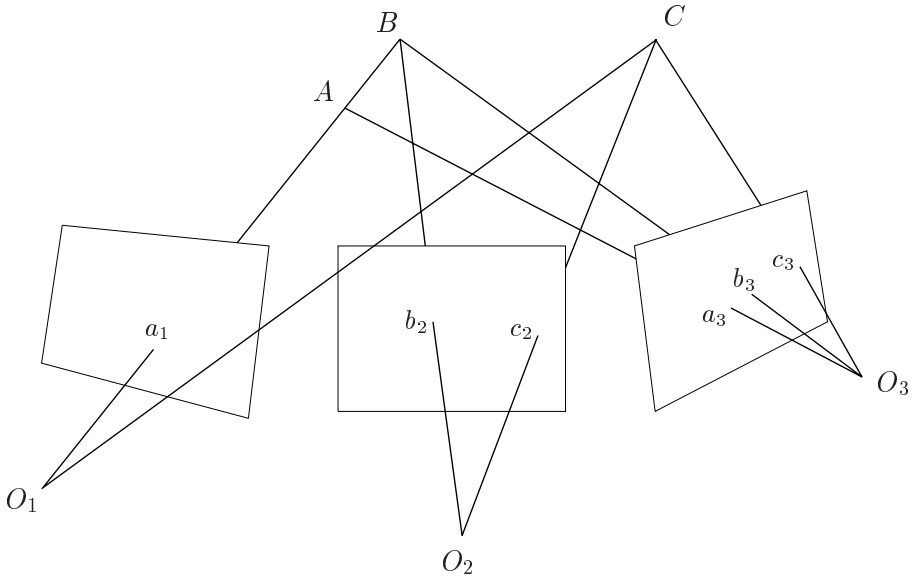


FIG. 3.1: *Définition d'un appariement (voir texte) ; les appariements existants sont* (a_1, a_3), (a_1, b_2, b_3) *et* (c_2, c_3).

Sur la figure 3.1, l'ensemble des appariements est égal à $((a_1, a_3), (a_1, b_2, b_3), (c_2, c_3))$. Les appariements sont indispensables à la perception du relief. Ainsi, si l'on suppose les positions O_1, O_2 et O_3 des trois caméras connues, l'appariement (a_1, b_2, b_3) nous permet de calculer la position du point B dans l'espace.

D'autres approches

Les positions des points 3D de la scène observée peuvent être obtenues de façon plus directe, avec un matériel spécialement adapté.

En restant dans le cadre de l'appariement stéréoscopique, on peut illuminer la scène à l'aide d'une lumière structurée (projection d'une grille, ou d'une ligne, ou d'un point à l'aide d'un laser par exemple), ce qui rend la phase d'appariement triviale : les points appariés sont les projections du seul point illuminé de la scène. Il faut ensuite balayer chaque point de la scène avec le faisceau laser. De tels capteurs sont limités : ils peuvent être employés avec succès sur des objets de taille raisonnable (manipulables), mais ils ne peuvent clairement pas s'appliquer à d'autres types de scènes, telles que des paysages.

D'autres systèmes mesurent directement les distances des points de la scène à partir d'une seule image, par mesure du temps de réflexion d'une onde sonore ou lumineuse envoyée sur l'objet. L'appariement devient alors inutile, puisqu'on obtient immédiatement les coordonnées de tous les points 3D de la scène. Les mêmes remarques que précédemment s'appliquent aussi.

Il est également possible de calculer la structure tridimensionnelle d'un objet en observant uniquement la distribution de l'intensité lumineuse qu'il réfléchit. Ceci est l'objet des techniques de *shape from shading*. Si les positions de la caméra et de la source lumineuse sont connues, ainsi que les lois de réflectance de l'objet observé, il est possible de calculer la forme de l'objet. Par exemple, si l'intensité lumineuse réfléchie en un point est $I = f(\vec{n}, \vec{s}, \vec{r})$, où $\vec{n}$ est la normale à l'objet en ce point, $\vec{s}$ la direction de la source lumineuse (rayon incident), et $\vec{r}$ la direction du rayon réfléchi (direction de vue), alors il est possible de calculer les normales $\vec{n}$ à la surface de l'objet, en tous les points où I, f, $\vec{s}$ et $\vec{r}$ sont connus, donc de calculer une équation de la surface de l'objet (voir figure 3.2). Les vecteurs $\vec{s}$ et $\vec{r}$ sont donnés par les positions de la source lumineuse et de la caméra, I est donné par la caméra, et f est issue d'un modèle de réflexion *a priori*.

Nous avons vu aussi que d'autres approches sont possibles, comme l'utilisation du *shape from occluding contours* (travaux de W. Niem, voir le chapitre précédent), qui oblige à pouvoir placer la scène sur une table tournante, et ne permet pas de toute façon de capturer tous les types de scènes (zones concaves).

▶ Nous ne nous plaçons pas dans le champ d'application de ces travaux. Nous nous situons délibérément en aval du processus d'acquisition d'images, nous réservant la possibilité d'utiliser des photos d'origine inconnue quant à leurs conditions de prise de vue. Nous cherchons à réaliser un système simple, utilisable sans appareillage particulier, sur tout type de scène.

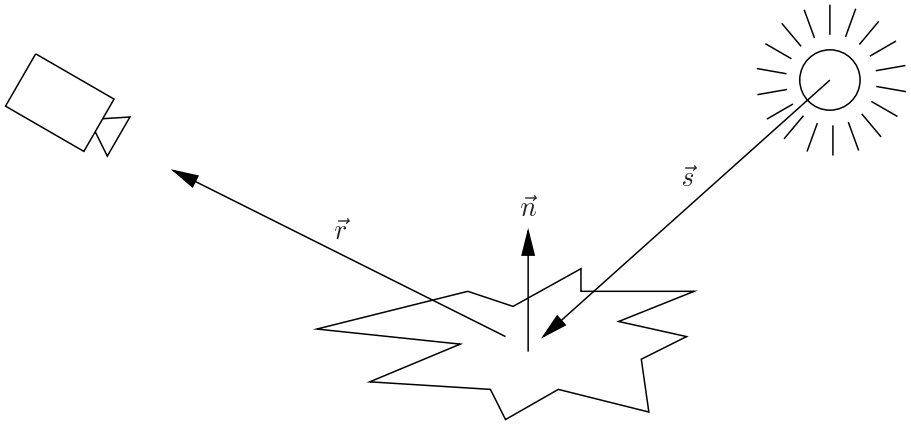


FIG. 3.2: *Principe du shape from shading : calculer $\vec{n}$ connaissant $I = f(\vec{n}, \vec{s}, \vec{r})$.*

Plan de ce chapitre

Nous allons étudier dans ce chapitre des algorithmes permettant de déterminer les appariements entre N images. De très nombreuses méthodes existent déjà, et notre travail sera d'intégrer et *d'évaluer* ces méthodes. Nous élaborerons éventuellement nos propres algorithmes, à des fins de comparaison.

Nous nous plaçons toujours dans le but final de pouvoir synthétiser de nouvelles vues à partir de vues existantes. La qualité et la précision de l'appariement seront donc évaluées en conséquence, et nous verrons qu'il n'est pas toujours nécessaire d'obtenir un appariement parfait pour pouvoir synthétiser des vues de qualité acceptable.

Nous dressons ci-dessous un résumé des méthodes d'appariement existantes (section 3.2). Puis nous expliquerons comment nous évaluerons ces méthodes, (section 3.3), et décrirons précisément les algorithmes utilisés (section 3.4). Nous mènerons l'évaluation en section 3.5, et conclurons sur cette partie en 3.6.

3.2 État de l'art de l'appariement

Il est possible d'apparier des structures plus évoluées que de simples pixels, comme par exemple des segments, des groupes de segments, ou des régions. Cependant, ces structures sont déjà difficiles à détecter en soi, et n'ont pas nécessairement le même aspect dans toutes leurs projections.

Un couple d'images à apparier est présenté en figure 3.3. Un détecteur de contours sur ces deux images peut donner le résultat montré en figure 3.4.

FIG. 3.3: *Un couple stéréo à apparier.*

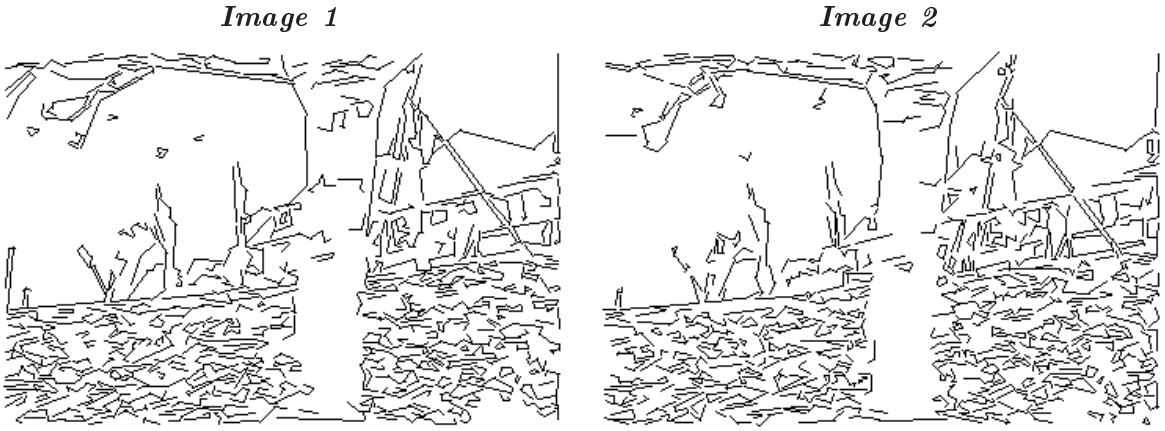


FIG. 3.4: *Contours extraits sur les images de la figure 3.3.*

Les arêtes ne sont pas les mêmes dans l'image 1 et 2 (existence, longueur), et ne présentent pas nécessairement la même topologie (connexité). Des algorithmes pouvant apparier de telles structures existent, mais sont d'un maniement délicat. Dans notre application, une telle complexité ne se justifie pas, et nous choisissons de n'apparier que des *points*, c.-à-d. (au mieux) des *pixels* dans les images.

3.2.1 Hypothèses de base

Si l'on reprend la figure 3.1, trouver un appariement de c_2 dans la troisième image consiste à calculer la position de c_3, sans autres hypothèses. Ceci est un problème insoluble en général, car sans autre information, c_3 peut se trouver n'importe où dans l'image 3, et ni son aspect, ni sa position ne sont liés à ceux de c_2 dans l'image 2.

Aussi, les points c_2 et c_3 étant les projections du même point C, on fait systématiquement l'hypothèse dans les algorithmes d'appariement que les signaux des images 2 et 3

autour des points c_2 et c_3 se ressemblent. Il suffit donc de calculer des mesures de ressemblance entre les voisinages des pixels c_2 dans l'image 2 et de ses correspondants potentiels dans l'image 3, et de conserver les candidats les plus ressemblants. Cette hypothèse de base est bien respectée, sauf si l'une des propositions suivantes est vraie :

Proposition $\mathcal{A}$: le point C appartient à une surface d'une courbure telle qu'elle n'a pas le même aspect depuis les points de vue O_2 et O_3 ;

Proposition $\mathcal{B}$: le point C appartient à un objet d'un matériau tel qu'il n'a pas le même aspect depuis les points de vue O_2 et O_3 ;

Proposition $\mathcal{C}$: le point C est occulté dans une vue ; par exemple, un objet opaque se trouve sur le trajet du rayon lumineux (O_3C).

La proposition $\mathcal{A}$ est toujours vraie (sauf si tous les points de vue sont confondus et que la surface observée est plane, parallèle aux caméras, ce qui n'a plus grand intérêt). Elle est d'autant plus vraie que la surface est tangente aux directions de vue, et que les points de vue sont éloignés. Aussi, les mesures de ressemblance sont d'autant plus robustes qu'elles intègrent un grand nombre de pixels voisins, pour limiter l'influence de ces erreurs locales.

La proposition $\mathcal{B}$ est souvent vraie. Lorsqu'elle est illuminée, une surface parfaitement lambertienne renvoie une lumière d'intensité égale dans toutes les directions. Ainsi, une feuille de papier vue dans N images a le même aspect dans ses N projections, et les pixels de ces images ont la même intensité. L'hypothèse de ressemblance locale est donc respectée. Cependant, les matériaux usuels ne sont pas lambertiens, et peuvent présenter des réflexions spéculaires. De tels reflets ne peuvent pas être appariés, car leur position sur l'objet dépend de la position de l'observateur. Ce problème est complexe, et nous nous tiendrons autant que possible en dehors de cette hypothèse. Des changements de luminosité moins radicaux peuvent être contrebalancés avec succès par des mesures de ressemblance «centrées». De telles mesures mettent en jeu des gradients d'intensité locaux, plutôt que de simples intensités de pixels.

Enfin, la proposition $\mathcal{C}$ est assez souvent vraie ; cela dépend bien sûr du type de scène observée. Dans une scène d'extérieur, des occultations sont presque toujours présentes : personnages, arbres, bâtiments devant un paysage (voir figure 3.5). Elles sont moins sensibles dans des scènes d'intérieur, et si les points de vue sont suffisamment rapprochés. Néanmoins, le problème est assez fréquent pour nécessiter une étude approfondie, et nous détaillerons les mesures robustes aux occultations.

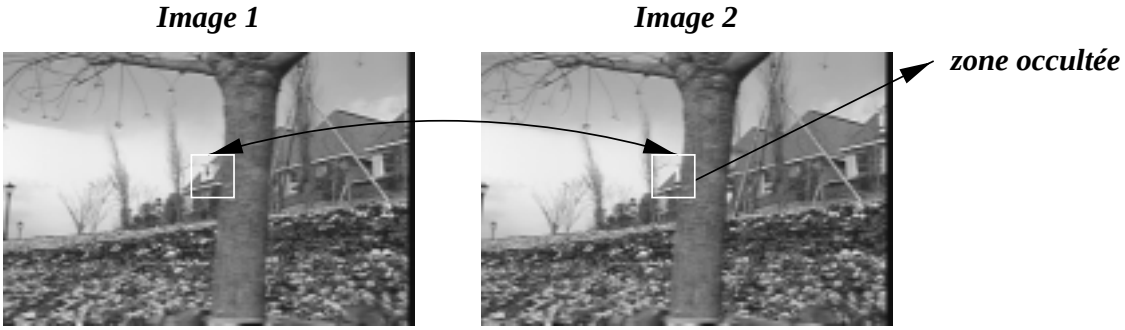


FIG. 3.5: *La zone encadrée de l'image 2 est partiellement occultée par l'arbre dans l'image 1, au premier plan.*

Principe des méthodes d'appariement

Sur la base de ces hypothèses, les méthodes d'appariement s'appuient toutes sur des mesures de ressemblance locale, et sur des contraintes de cohérence «moins locale». Il nous faut donc une *mesure de ressemblance*, capable de donner une distance de ressemblance entre M pixels dans N images (généralement, $N = 2$) ; il nous faut ensuite un *algorithme de mise en correspondance*, appariant au mieux les pixels des deux images, en optimisant un critère global sur les images (prenant en compte par exemple la régularité du champ de disparité observé).

Mesures et algorithmes sont très souvent évalués manuellement, par l'observation des cartes de disparité, en établissant des classements subjectifs. Ils sont parfois évalués quantitativement et automatiquement sur des images synthétiques représentant des plans vus de face de disparités connues (stéréogrammes aléatoires), ou sur des images réelles de plans quelconques. Dans ce dernier cas, on suppose que la méthode évaluée fournit au moins 50 % de bons appariements. À l'aide des appariements obtenus, on peut alors estimer de manière robuste la transformation homographique liant les positions des pixels des plans dans les deux images, et observer la distribution des erreurs de positionnement par rapport à cette homographie supposée correcte (voir 3.3.1).

Nous exposons ci-dessous les mesures de ressemblance existantes (en 3.2.2), puis les algorithmes employés (en 3.2.3).

3.2.2 Mesures de ressemblance

3.2.2.1 Mesures de corrélation standard

Les mesures de ressemblance que nous avons évoquées sont implémentées dans l'immense majorité des cas par des mesures de corrélation.

De telles mesures intègrent les différences des intensités sur des voisinages rectangulaires (généralement carrés) des pixels considérés. Quelques mesures usuelles sont rappelées en tableau 3.1 ; elles mesurent la ressemblance d'un point (u_1, v_1) de l'image 1 de signal I_1, à celle d'un point (u_2, v_2) de l'image 2 de signal I_2, sur un masque carré $(2n+1) \times (2n+1)$ (une fenêtre, ou un *patch*).

Mesure	Expression
SAD	$\sum_{du=-n}^{du=+n} \sum_{dv=-n}^{dv=+n} \lvert I_2(u_2+du, v_2+dv) - I_1(u_1+du, v_1+dv)\rvert$ (3.1)
ZSAD	$\sum_{du=-n}^{du=+n} \sum_{dv=-n}^{dv=+n} \lvert I_2(u_2+du, v_2+dv) - \overline{I_2} - I_1(u_1+du, v_1+dv) + \overline{I_1}\rvert$ (3.2)
SSD	$\sum_{du=-n}^{du=+n} \sum_{dv=-n}^{dv=+n} (I_2(u_2+du, v_2+dv) - I_1(u_1+du, v_1+dv))^2$ (3.3)
ZSSD	$\sum_{du=-n}^{du=+n} \sum_{dv=-n}^{dv=+n} (I_2(u_2+du, v_2+dv) - \overline{I_2} - I_1(u_1+du, v_1+dv) + \overline{I_1})^2$ (3.4)

TAB. 3.1: *Quatre mesures usuelles de corrélation.*

Ces mesures sont assez intuitives ; SAD correspond mathématiquement à la norme L_1, évaluée entre les fonctions I_1 et I_2 en $(2n+1) \times (2n+1)$ points, et SSD au carré de L_2. Ce sont des mesures de l'énergie de la fonction de différence des intensités des masques. ZSAD et ZSSD sont centrées, par soustraction de la moyenne locale des intensités $\overline{I_1}$ et $\overline{I_2}$ sur les masques courants. Ceci permet d'annuler l'effet de changements d'intensité locaux, mais a pour inconvénient de donner de bons scores à tort à des masques très différents. Avec ZSAD ou ZSSD, une fenêtre uniformément blanche correspond parfaitement à une fenêtre uniformément noire. Pour compenser l'effet des changements de luminosité, il est plus correct de modifier *globalement* les intensités des images, puis d'appliquer des mesures non-centrées. Par exemple, si les pixels de l'image 1 ont une intensité de moyenne m_1 et d'écart-type σ_1, et ceux de l'image 2 une intensité de moyenne m_2 et d'écart-type σ_2, alors on peut transformer les intensités I_2 de chaque pixel de l'image 2 selon la formule 3.5.

$$I_2 \mapsto \frac{\sigma_1}{\sigma_2} I_2 + m_1 - m_2 \frac{\sigma_1}{\sigma_2} \tag{3.5}$$

La distribution des intensités des pixels de l'image 2 transformée aura ainsi la même moyenne m_1 et le même écart-type σ_1 que dans l'image 1.

P. Aschwanden a étudié 19 mesures de corrélation dans [Asc 92]. Les 5 critères de test étaient la robustesse :

1. IRIS : à des changements de luminosité ;
2. NOISE : à un bruit gaussien (centré) sur les valeurs des pixels, simulant le bruit électronique d'une caméra ;
3. SALTY : à des occultations de la taille d'un pixel (pixel non informé), supposées simuler les occultations en général ;

4. ZOOM : à des zooms ;

5. FOCUS : à une mauvaise mise au point.

Trois images de test étaient artificiellement bruitées selon ces 5 critères, et P. Aschwanden évaluait les appariements renvoyés par autocorrélation sur ces trois images. La première image comportait des textures, la deuxième des arêtes et des structures linéaires, et la troisième une scène de bureau en situation réelle. Dans l'une des 3 images originales, on choisit 18 points bien contrastés, et pour chacun de ces points, on cherche son correspondant dans la version bruitée de cette même image, à l'aide de l'une des 19 mesures de corrélation, et pour plusieurs tailles de masques. Le correspondant est celui donnant la meilleure mesure de corrélation. Un appariement est considéré comme bon si on tombe à 1 pixel ou moins du point original.

La mesure la plus simple : SAD donne les meilleurs appariements pour tous les critères, sauf IRIS ; pour ce dernier, comme on pouvait s'y attendre, une mesure centrée est indispensable.

On peut facilement contester ces travaux sur le fait que les mesures de corrélation ne sont évaluées que sur des auto-appariements, ce qui ne permet pas de connaître leurs robustesses comparées à des déformations perspectives. L'utilisation habituelle des mesures de corrélation est pourtant de comparer deux images ayant subi une déformation perspective (couple stéréo).

De plus, le processus SALTY ne modélise pas les occultations. SALTY simule en fait le dysfonctionnement de certains pixels du capteur CCD d'une caméra vidéo. Mais ceci ne correspond pas au processus normal d'occultation, où la zone occultée est généralement cohérente : elle correspond à la projection d'un objet interposé entre la scène et la caméra, et ne peut pas être correctement modélisée par la mise à 0 de certains pixels aléatoirement (figure 3.6).

Occultation modélisée par le processus SALTY. Occultation réellement observée.

FIG. 3.6: *La modélisation des occultations dans le processus* SALTY *ne traduit pas la cohérence locale.*

3.2.2.2 Mesures de corrélation robustes

Z.D. Lan répertorie dans sa thèse [Lan 97] de nombreuses mesures de corrélation robustes aux occultations, dont celles de R. Zabih, et de D.N. Bhat. Il apporte également

ses propres innovations.

Mesures de R. Zabih

R. Zabih propose dans [Zab 94] deux mesures de ressemblance non paramétriques. Ces mesures agissent sur des images transformées. La transformation *rank* transforme chaque pixel en une valeur indiquant le nombre de pixels de son voisinage 3×3 qui lui sont inférieurs. La transformation *census* transforme chaque pixel en une chaîne de bits indiquant quels sont les pixels du voisinage qui lui sont inférieurs. Si pour construire la chaîne de bits, on considère ces 8 pixels dans l'ordre suivant : $\begin{array}{|c|c|c|}\hline 8&1&2\\\hline 7&\cdot&3\\\hline 6&5&4\\\hline\end{array}$, alors le masque $\begin{array}{|c|c|c|}\hline 84&119&157\\\hline 85&232&226\\\hline 246&250&175\\\hline\end{array}$ sera transformé en `11110011`, car $119 < 232$, $157 < 232$, $226 < 232$, $175 < 232$, $250 \geq 232$, $246 \geq 232$, $85 < 232$ et $84 < 232$. La transformation *rank* donne la valeur 6, car 6 bits sont à 1. Un exemple est donné en figure 3.7.

Image originale en niveau de gris.

84	119	157	94	222
85	232	226	250	250
246	250	175	250	57
6	17	67	88	51
72	170	147	253	10

Image transformée par *rank*.

-	-	-	-	-
-	6	4	6	-
-	8	3	6	-
-	1	1	4	-
-	-	-	-	-

Image transformée par *census*.

--------	--------	--------	--------	--------
--------	11110011	11001001	11010111	--------
--------	11111111	00011100	00111111	--------
--------	00000010	00000010	01110010	--------
--------	--------	--------	--------	--------

FIG. 3.7: *Exemples de transformations de R. Zabih.*

La mesure de distance sur ces images transformées est une mesure SAD (dans le cas d'une transformation *rank*), ou une distance de Hamming (dans le cas d'une transformation *census*).

Il est clair que de telles mesures sont robustes à des occultations : si un pixel est occulté dans un masque, il y a une probabilité non-nulle pour que le masque transformé ne change absolument pas ; il suffit que la nouvelle valeur du pixel soit du même ordre que l'ancienne valeur (par rapport au pixel central). De plus, même si le masque transformé change, ce changement est minime vis-à-vis de la distance utilisée : les distances SAD ou de Hamming ne changeront au plus que de 1 unité.

R. Zabih a réalisé un test sur deux images synthétiques, représentant 3 plans de disparité fixe, vus de face (pas de déformation perspective). Il compare les cartes de disparité obtenues avec les mesures SSD, *rank* + SAD, et *census* + Hamming. Pour un point de

l'image 1, on choisit comme appariement dans l'image 2 celui qui donne la distance minimale, sans autre vérification.

Les contours obtenus sont bien mieux délimités dans le cas des mesures robustes. Le nombre de faux appariements est aussi considérablement réduit (tableau 3.2).

Mesure	Nombre de faux appariements
SSD	1385
rank + SAD	609
census + Hamming	407

TAB. 3.2: *Tests comparatifs des mesures de R. Zabih, selon [Zab 94].*

Mesure de D.N. Bhat

D.N. Bhat décrit une nouvelle transformation dans [Bha 96]. Elle consiste à numéroter les 9 pixels d'un masque 3×3 dans l'ordre croissant. Si on numérote ces 9 pixels dans l'ordre suivant : $\begin{array}{|c|c|c|}\hline 1&2&3\\\hline 4&5&6\\\hline 7&8&9\\\hline\end{array}$, alors le masque $\begin{array}{|c|c|c|}\hline 84&119&157\\\hline 85&232&226\\\hline 246&250&175\\\hline\end{array}$ sera transformé en `142396578`, car les valeurs apparaissent dans cet ordre : $(84, 85, 119, 157, 175, 226, 232, 246, 250)$.

Ainsi, sur la même image brute qu'en figure 3.7, l'image transformée est celle donnée en figure 3.8.

Image transformée par D.N. Bhat.

---------	---------	---------	---------	---------
---------	142396578	312854679	921734568	---------
---------	789163245	789521346	967841235	---------
---------	456798312	456872139	963457128	---------
---------	---------	---------	---------	---------

FIG. 3.8: *Transformation de D.N. Bhat.*

Sur les images transformées, deux masques en correspondance doivent normalement présenter des pixels dans le même ordre. C'est toujours le cas si les variations des intensités des pixels sont suffisamment faibles d'une image à l'autre. En revanche, si les changements sont plus importants, l'ordre peut être perturbé ; cette perturbation doit être faible pour des masques en correspondance. La mesure de ressemblance utilisée est optimale si la permutation observée est minimale (c.-à-d. proche de l'identité).

D.N. Bhat revendique des résultats meilleurs que ceux de R. Zabih : de 25 % à 50 % de faux appariements en moins.

Mesure de Z.D Lan

Dans [Lan 97], Z.D. Lan utilise une approche plus mathématique pour la gestion des occultations, faisant appel aux statistiques robustes. Le principe est de calculer une corré-

lation traditionnelle, comme ZSSD, mais seulement sur les pixels de la fenêtre qui ne sont pas occultés.

En suivant ses propres notations, on suppose que les valeurs $(x_i)_{i=1..2n+1}$ du masque de l'image 1 et les valeurs $(y_i)_{i=1..2n+1}$ du masque de l'image 2 sont liées par une fonction f_α dépendant de paramètres α, à une erreur ε près :

$$\forall i \in [1..2n+1], y_i = f_\alpha(x_i) + \varepsilon_i \tag{3.6}$$

La procédure est la suivante :

1. on estime α de façon robuste à partir des observations (x_i) et (y_i) ;
2. on détermine les pixels occultés dans les masques ; ce sont ceux ne correspondant pas au modèle décrit par l'équation 3.6 ;
3. on calcule une mesure de corrélation sur la partie non occultée des fenêtres.

Étape 1

Si on suppose que les masques en correspondance sont d'intensité identique pixel par pixel d'une image à l'autre (c'est l'hypothèse faite pour les mesures non centrées comme SSD), alors :

$$f_\alpha(x_i) = x_i \tag{3.7}$$

...et on peut négliger l'étape 1.

Si on suppose au contraire que les pixels des fenêtres en correspondance subissent une translation d'intensité (c'est l'hypothèse faite pour les mesures centrées comme ZSSD), alors :

$$f_\alpha(x_i) = x_i + \alpha \tag{3.8}$$

...et l'étape 1 consiste à déterminer α de façon robuste. On peut pour cela utiliser des méthodes de moindres carrés médians, et calculer la valeur de α avec une confiance arbitraire par tirages aléatoires.

D'autres mesures de corrélation standard supposent un f_α plus complexe (p. ex. transformation affine).

Étape 2

Une fois α déterminé, on peut estimer les paramètres de la distribution de l'erreur ε.

Si on suppose que cette distribution est gaussienne et centrée, alors on peut déterminer son écart-type σ en observant sa médiane $\text{med}(\varepsilon_i)$. La médiane est en effet une mesure robuste, et permet de calculer de façon fiable $\sigma = 1.4826\,\text{med}$.

Si on suppose que la distribution de ε est laplacienne centrée, alors on peut encore déterminer son écart-type $\sigma = 1.4427\,\text{med}$ en observant sa médiane.

En effet, dans le cas de la distribution gaussienne :

$$\int_{-\,\text{med}}^{+\,\text{med}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma}} dx = \frac{1}{2} \Rightarrow \text{erf}(\frac{\text{med}}{\sigma\sqrt{2}}) = \frac{1}{2} \tag{3.9}$$

$$\Rightarrow \sigma \simeq 1.482602219\,\text{med} \qquad \text{(résolution numérique)} \tag{3.10}$$

Dans le cas de la distribution laplacienne :

$$\int_{-\,\text{med}}^{+\,\text{med}} \frac{1}{2\sigma} e^{-\frac{|x|}{\sigma}} dx = \frac{1}{2} \Rightarrow \sigma = \frac{\text{med}}{\ln 2} \tag{3.11}$$

$$\Rightarrow \sigma \simeq 1.442695041\,\text{med} \tag{3.12}$$

Étape 3

Il suffit maintenant de rejeter dans les masques les pixels i pour lesquels ε_i indique une trop faible vraisemblance qu'ils appartiennent au modèle supposé par f_α. On réalise ceci en pondérant les termes de la corrélation standard par un facteur ω_i. La version robuste RZSSD de ZSSD s'écrit donc :

$$\frac{\sum_{du=-n}^{du=+n} \sum_{dv=-n}^{dv=+n} \omega(I_1, I_2, du, dv)\,(I_2(u_2+du, v_2+dv) - \overline{I_2} - I_1(u_1+du, v_1+dv) + \overline{I_1})^2}{\sum_{du=-n}^{du=+n} \sum_{dv=-n}^{dv=+n} \omega(I_1, I_2, du, dv)} \tag{3.13}$$

Dans ses expériences, Z.D. Lan suppose une distribution gaussienne de ε, et affecte les poids suivants :

$$\begin{cases} \omega_i = 0 & \text{si} \quad \varepsilon_i > 2.5\sigma \\ \omega_i = 1 & \text{si} \quad \varepsilon_i \leq 2.5\sigma \end{cases} \tag{3.14}$$

Le seuil de 2.5σ revient à n'accepter que les pixels qui ont une vraisemblance d'au moins 98.76 % d'appartenir au modèle choisi. En effet, la limite k telle que les erreurs ε_i mesurées supérieures à $k\sigma$ aient une vraisemblance τ de ne pas appartenir au modèle est donnée par :

$$\int_{-k\sigma}^{+k\sigma} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma}} dx = \tau \Rightarrow \text{erf}(\frac{k}{\sqrt{2}}) = \tau \tag{3.15}$$

$$\text{donc} \qquad k = 2.5 \Rightarrow \tau = 0.9876 \tag{3.16}$$

Pour une distribution laplacienne, on aurait :

$$\int_{-k\sigma}^{+k\sigma} \frac{1}{2\sigma} e^{-\frac{|x|}{\sigma}} dx = \tau \Rightarrow 1 - e^{-k} = \tau \tag{3.17}$$

$$\text{donc} \qquad k = 2.5 \Rightarrow \tau = 0.9179 \tag{3.18}$$

Il faudrait monter à $k = 4.39$ pour obtenir $\tau = 0.9876$ comme précédemment.

Résultats

En termes d'appariement, RZSSD se révèle plus performante que les mesures *census* ou *rank*, *sur les régions occultées*. Les faux appariements sont alors moins nombreux de 5 % à 20 % par rapport à ces deux méthodes.

Sur les régions non occultées, la meilleure mesure est *census*, ou même simplement ZSSD. Aussi, l'utilisation recommandée est de réaliser une première phase d'appariement à l'aide de mesures standard (non robustes), de détecter sur cette base les possibles régions d'occultation, et d'effectuer une seconde passe d'appariement robuste dans ces régions seulement.

Remarques communes à ces méthodes

▶ Les mesures *rank* et *census* imposent un prétraitement des images qui peut être rapide. La corrélation est ensuite d'un coût équivalent aux mesures classiques. Le prétraitement nécessité par la méthode de rang de R. Zabih est assez lourd : il faut trier la liste des pixels de chaque fenêtre ; la comparaison fait ensuite intervenir des calculs de permutation, potentiellement complexes. Les seules données chiffrées concernent RZSSD, qui présente un surcoût de 250 % à 300 % par rapport à ZSSD, ce qui est tolérable. Remarquons que dans une version «continue», où les ω_i ne valent plus 0 ou 1, mais la vraisemblance $1 - \mathrm{erf}(\frac{\varepsilon_i}{\sigma\sqrt{2}})$ du pixel i, le surcoût de RZSSD passe à plus de 1000 %, ce qui la rend difficilement utilisable.

▶ Ces mesures n'ont jamais été évaluées autrement que sur des plans, donc pour des fenêtres en déformation homographique.

▶ Le défaut principal de ces mesures est de ne pas prendre en compte la cohérence locale des occultations. La mesure RZSSD par exemple ôte du calcul de corrélation tous les pixels «qui ne conviennent pas», c.-à-d. tous les pixels du masque qui ne sont pas déjà suffisamment ressemblants. En effet, on ne contraint pas les pixels rejetés à respecter une disposition cohérente, comme c'est pourtant le cas pour les occultations «naturelles» (cette discussion est similaire à celle sur le processus SALTY). Ainsi, la mesure obtenue est nécessairement bonne : on ne garde que les pixels ressemblants. Pire, si on l'applique à des masques très semblables, alors les erreurs ε sont très faibles, et le σ estimé très petit. Aussi, on rejettera de nombreux points dont l'erreur, bien que faible, sera néanmoins supérieure à 2.5σ. Tout ceci explique pourquoi il vaut mieux ne pas utiliser ces mesures sur des zones non occultées.

Autres méthodes robustes

Les méthodes que nous décrivons rapidement ci-dessous sont assez semblables à l'approche de Z.D. Lan ; mais plutôt que de rejeter certains pixels dans des masques de corrélation carrés, elles tentent d'adapter dès le départ la taille des fenêtres de corrélation, de façon que chaque pixel de la fenêtre soit un pixel «acceptable» au sens de Z.D. Lan.

T. Kanade [Kan 91] élabore une fonction f reliant les dimensions de la fenêtre de corrélation (rectangulaire) et la disparité locale de la scène, à l'incertitude sur la mesure de corrélation. Il est donc nécessaire de partir d'une première estimation d'appariement dense, avec des fenêtres de corrélation fixes et carrées, ce qui permet d'estimer la disparité de la scène en tout point de l'image. Dans une deuxième passe, on tente d'agrandir les masques de corrélation au maximum, sous la contrainte que f ne croisse pas. On trouve ainsi en chaque pixel les dimensions maximales de la fenêtre de corrélation permettant une estimation de la disparité sans augmentation de l'incertitude ; les fenêtres trouvées recouvrent des zones planes, et ne chevauchent pas les ruptures de disparité. On recalcule les appariements à l'aide de ces nouveaux masques de corrélation. Les résultats ne sont pas évalués quantitativement : seules des cartes de disparité sont fournies, et semblent excellentes. La difficulté est le calcul initial de disparité, qui doit être suffisamment fiable. Surtout, cette méthode a pour inconvénient de contraindre les fenêtres à être rectangulaires, donc les frontières d'occultation à être rectilignes, ce qui n'est pas souvent le cas sur des grandes fenêtres (p. ex. 15×15) et sur des scènes d'extérieur réelles.

M. Zahid [Zah 92] et J-L. Lotti [Lot 96] ont développé de façon indépendante des méthodes assez similaires. Les masques de corrélation sont contraints à ne pas franchir les contours de l'image, ce qui permet de supposer qu'ils ne franchissent pas non plus de rupture de disparité, donc qu'ils recouvrent une surface relativement plane. Là encore, les résultats semblent très bons et s'appliquent avec succès au traitement de photos aériennes et satellitaires, où les scènes sont essentiellement constituées de plans vus de face, à différentes hauteurs, présentant des arêtes contrastées et de nettes ruptures de disparité. Ils font appel à de nombreux pré- et post-traitements, comme des filtres, ou des ré-estimations. Un inconvénient est que ces algorithmes s'appuient fortement sur des informations de contour, dont la détection est notoirement instable dans des couples stéréoscopiques : il n'est pas garanti que les contours de l'image 1 se retrouvent dans l'image 2, ni qu'ils présentent la même topologie (connexité par exemple).

3.2.2.3 Mesures de corrélation précises

Les méthodes précédentes fonctionnent sur des masques de pixels, et ne peuvent donc fournir que des appariements précis au mieux au pixel près. Une plus grande précision est souvent nécessaire, surtout lorsqu'on aborde la reconstruction tridimensionnelle.

Il est possible de calculer les corrélations précédentes sur des pixels non-entiers, et il est alors nécessaire de pouvoir calculer l'intensité du signal lumineux en des positions de l'image non-entières. Pour cela, on utilise généralement une interpolation bilinéaire des intensités des pixels entiers les plus proches. Avec les notations de la figure 3.9, l'intensité

retenue pour le pixel $P(x, y)$ est :

$$\begin{array}{rcccccccc} I(P(x,y)) = & (1 & - & \lambda_x) & (1 & - & \lambda_y) & I(P_1) & + \\ & (1 & - & \lambda_x) & & & \lambda_y & I(P_2) & + \\ & & & \lambda_x & (1 & - & \lambda_y) & I(P_3) & + \\ & & & \lambda_x & & & \lambda_y & I(P_4) & \end{array}$$

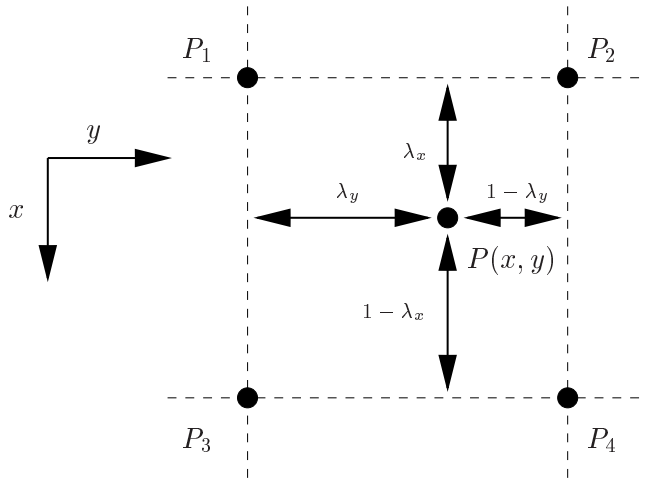


FIG. 3.9: *L'intensité $I(P(x, y))$ est donnée par l'interpolation bilinéaire des intensités de ses 4 voisins entiers.*

Cependant, ces corrélations seront sous-pixelliques sans être forcément précises pour autant.

En effet, dans les corrélations classiques ou robustes, on fait l'hypothèse que les fenêtres carrées $(2n + 1) \times (2n + 1)$ représentent la même portion de scène dans les deux images. Cette hypothèse n'est vraie que si les deux images sont séparées par un mouvement de translation horizontale et frontale ; elle est bien sûr d'autant plus fausse que le mouvement réel en est éloigné. Les mesures obtenues ne sont alors pas très représentatives, et il est impossible d'obtenir un appariement précis. Plusieurs méthodes d'appariement précis s'attachent donc à adapter la taille des fenêtres à la portion de scène observée ; ce sont les méthodes de *déformation de fenêtres*.

D'autres méthodes conservent l'hypothèse d'un mouvement translationnel, et cherchent à déterminer la quantité de ce déplacement avec une précision sous-pixellique. Nous les appelons *méthodes translationnelles*.

Déformation de fenêtres — 1

Dans [Bra 95], P. Brand étudie en détail les possibilités d'appariement sous-pixellique à l'aide de fenêtres déformables. Comme F. Ackermann dans [Ack 84] ou A. Gruen dans [Gru 85], il suppose que les masques sont suffisamment petits pour qu'on puisse considérer que deux masques en correspondance sont déformés par une transformation affine. Cette

transformation affine n'est pas quelconque, car les coins des fenêtres (qui se correspondent), respectent nécessairement la contrainte épipolaire (voir figure 3.10).

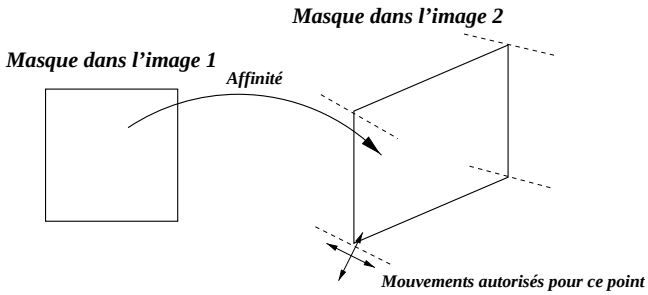


FIG. 3.10: *La fenêtre du masque de l'image 2 est autorisée à se déformer selon certaines contraintes.*

Pour un masque fixé dans l'image 1, trois coins de la fenêtre de l'image 2 sont autorisés à se déplacer le long de leurs lignes épipolaires. Le quatrième coin est implicitement positionné, car la transformation globale de la fenêtre est affine, donc entièrement définie par 3 points. Le déplacement peut également avoir lieu perpendiculairement aux lignes épipolaires, si la géométrie épipolaire calculée est trop peu précise. Il suffit alors de minimiser le score de corrélation entre la fenêtre de l'image 1 et la fenêtre de l'image 2 transformée, en fonction des paramètres de déplacement.

P. Brand revendique une précision de mise en correspondance de 0.05 pixel, validée sur des images de plans. Ceci est équivalent à la méthode de F. Ackermann ; mais ce dernier n'utilisant pas la géométrie épipolaire, il est nécessaire de fournir au système une estimation initiale de la transformation affine des fenêtres, afin de converger correctement. Ce n'est pas le cas chez P. Brand, dont la méthode est plus facilement automatisable.

Cette méthode est itérative et relativement lente (de l'ordre de 1 seconde par point).

Déformation de fenêtres — 2

Chez DEC, R. Szeliski et S.B. Kang [Sze 95b] utilisent des fenêtres de corrélation pouvant subir des déformations bilinéaires. Dans un algorithme de tracking dans une séquence d'images, ils utilisent en effet des splines pour modéliser le flot optique au cours de la séquence. Cela garantit une certaine régularité, et permet de calculer les corrélations sur des fenêtres dont la forme suit les contraintes de disparité locale. Comme prévu, les résultats sont meilleurs que les autres méthodes surtout dans les cas où la corrélation classique ne peut pas fonctionner : rotation de la scène, ou grands changements d'échelle. La modélisation par splines est bien adaptée au tracking, mais ne peut pas être appliquée a un couple stéréo (seulement 2 images) présentant de fortes disparités, cas que nous voulons pouvoir traiter.

Méthodes translationnelles

Dans [Lan 97], Z.D. Lan suppose que les fenêtres en correspondance sont carrées et de même taille (cas translationnel), et qu'il existe localement une relation entre les intensités de chaque pixel de la fenêtre de l'image 2, et un voisinage des pixels de la fenêtre de l'image 1. Par exemple :

$$\begin{aligned} I_2(x_2, y_2) =& a_1 I_1(x_1, y_1) + \\ & a_2 I_1(x_1 + 1, y_1) + a_3 I_1(x_1 - 1, y_1) + \\ & a_4 I_1(x_1, y_1 + 1) + a_5 I_1(x_1, y_1 - 1) \end{aligned} \tag{3.19}$$

La première étape consiste à déterminer les 5 paramètres a_i à partir des observations de I_1 et I_2 en plusieurs endroits. Ensuite, le déplacement sous-pixellique recherché est directement donné par ces coefficients, qui représentent une sorte de pondération barycentrique ; le déplacement recherché est :

$$\left\{ \begin{array}{lcl} dx & = & (a_2 - a_3)/S \\ dy & = & (a_4 - a_5)/S \\ S & = & a_1 + a_2 + a_3 + a_4 + a_5 \end{array} \right. \tag{3.20}$$

La précision obtenue est de 0.1 pixel, validée sur des images de plans, même avec une rotation (30°), et un facteur d'échelle (0.9).

Cette approche utilise le 4-voisinage de chaque pixel ; une autre approche est présentée utilisant le 8-voisinage, avec des résultats similaires.

L'avantage est que cette méthode est non-itérative, et une fois les a_i estimés, le calcul du déplacement est immédiat. Le temps de calcul complet pour un masque 15×15 est de l'ordre de 0.03 seconde.

Combinaison des approches

Enfin, les approches peuvent être combinées pour produire une méthode d'appariement précise et rapide, même en cas de déformation affine des images.

Après une première passe d'appariement standard, on peut calculer les transformations affines locales liant les fenêtres, de manière robuste. Ensuite, on applique la méthode précédente non-itérative, sur des fenêtres redressées. La précision obtenue est de 0.1 pixel, validée sur des images de plans (toujours [Lan 97]).

3.2.2.4 Invariants

Les corrélations sont utilisées depuis longtemps pour l'appariement. L'idée sous-jacente est que le signal est le même pixel à pixel dans les deux masques dont on mesure la ressemblance ; cela est faux en cas de rotations ou de changements d'échelle importants.

D'autres mesures locales du signal-image sont invariantes aux rotations. Par exemple, l'intensité d'un pixel, ou son laplacien, sont des grandeurs qui sont invariantes aux rotations

de l'image. C. Schmid propose dans sa thèse [Sch 96b] d'utiliser 9 tels invariants, regroupés dans un vecteur v, représentant les caractéristiques locales d'un point de l'image :

$$v = \begin{pmatrix} L \\ L_i L_i \\ L_i L_{ij} L_j \\ L_{ii} \\ L_{ij} L_{ji} \\ \varepsilon_{ij}(L_{jkl} L_i L_k L_l - L_{jkk} L_i L_l L_l) \\ L_{iij} L_j L_k L_k - L_{ijk} L_i L_j L_k \\ -\varepsilon_{ij} L_{jkl} L_i L_k L_l \\ L_{ijk} L_i L_j L_k \end{pmatrix} \tag{3.21}$$

L est la fonction de signal-image. La première composante du vecteur est donc l'intensité du point considéré. Les autres composantes utilisent la notation d'Einstein, où les indices i, j, k, l représentent la somme des dérivations par rapport à l'ensemble des variables ; par exemple, la troisième composante est égale à :

$$L_i L_{ij} L_j = \frac{\partial L}{\partial x}\frac{\partial^2 L}{\partial x^2}\frac{\partial L}{\partial x} + 2\frac{\partial L}{\partial x}\frac{\partial^2 L}{\partial x \partial y}\frac{\partial L}{\partial y} + \frac{\partial L}{\partial y}\frac{\partial^2 L}{\partial y^2}\frac{\partial L}{\partial y} \tag{3.22}$$

Enfin, $\varepsilon_{xy} = -\varepsilon_{yx} = 1$, et $\varepsilon_{xx} = \varepsilon_{yy} = 0$.

L'une des difficultés est de pouvoir calculer de façon stable les dérivées du signal-image jusqu'à l'ordre 3. C. Schmid calcule et lisse les dérivées par convolution du signal L avec une dérivée de gaussienne, et l'implémentation actuelle mène à des lissages sur des zones de taille 31×31.

Ensuite, pour mesurer la ressemblance entre deux points p_1 et p_2 décrits par leurs vecteurs d'invariants v_1 et v_2, elle utilise une distance de Mahalanobis :

$$d(p_1, p_2) = \sqrt{(v_2 - v_1)^T \Lambda^{-1} (v_2 - v_1)} \tag{3.23}$$

...où Λ est la matrice de covariance 9×9 des composantes du vecteur. Cette matrice est calculée expérimentalement en de nombreux points, sur de nombreuses images. Il est difficile de procéder de façon plus formelle, car les variances des mesures dépendent essentiellement des points où elles sont faites ; or, si l'on peut facilement modéliser mathématiquement le bruit dans l'image, l'erreur sur le positionnement est, elle, impossible à modéliser *a priori*. Dans notre cas, nous estimerons Λ à partir des points observés dans les images en cours d'appariement, comme nous le verrons en 3.4.6.

Ces vecteurs de caractéristiques ont été construits pour être invariants aux translations et aux rotations de l'image, et on obtient un taux de 90 % de bons appariements sur des images planes avec une rotation quelconque. C. Schmid propose aussi une modification pour rendre ces vecteurs invariants aux changements locaux d'intensité. En revanche, ils ne sont pas invariants aux changements d'échelle de l'image, et il faut alors les calculer à

plusieurs échelles ; ils ne sont pas robustes non plus aux occultations. En outre, leur calcul est assez lent, et cette approche peut difficilement être envisagée si les points à apparier sont trop nombreux (p. ex. pour l'appariement dense).

3.2.3 Algorithmes d'appariement

Nous avons répertorié de nombreuses mesures de ressemblance entre pixels. Pour obtenir une mise en correspondance des images de référence, il nous faut maintenant un algorithme capable de calculer un appariement globalement optimal au sens de cette mesure. Nous distinguerons les algorithmes d'appariement dense, des algorithmes d'appariement épars.

3.2.3.1 Appariement épars

Pour un appariement épars, l'ensemble des points à apparier est défini. On cherche à mettre en correspondance les points $(p_1, ..., p_n)$ dans l'image 1 avec les points $(q_1, ..., q_m)$ dans l'image 2, c.-à-d. à trouver une relation r inversible telle que :

$$\forall i \in [1..n], \exists! j \in [0..m], r(p_i) = q_j \tag{3.24}$$

$$\forall j \in [1..m], \exists! i \in [0..n], r^{-1}(q_j) = r_i \tag{3.25}$$

On note p_0 et q_0 des points virtuels signifiant «pas de correspondance». Sauf mention contraire, la condition d'unicité est respectée dans tous les algorithmes que nous détaillerons, c.-à-d. un point dans une image a, au plus, un seul correspondant dans l'autre image.

Il s'agit donc d'une recherche combinatoire, de complexité plus grande que $\mathcal{O}(A_n^m)$ ou $\mathcal{O}(A_m^n)$. Notons $B(n, m)$ le nombre exact de combinaisons possibles. Deux possibilités se présentent :

1. le point p_1 de l'image 1 peut être apparié à l'un des m points de l'image 2, et il faudra ensuite apparier les points p_2 à p_n avec les $m-1$ points restants, ce qui représente $B(n-1, m-1)$ possibilités à chaque fois ;
2. le point p_1 n'a pas de correspondant, et il faudra ensuite apparier les points p_2 à p_n avec les points q_1 à q_m, soit $B(n-1, m)$ possibilités.

On a donc :

$$\begin{cases} B(n,m) &= mB(n-1,m-1) + B(n-1,m) \\ B(n,1) &= n+1 \\ B(1,m) &= m+1 \end{cases} \tag{3.26}$$

Par exemple pour $n = m = 15$ points à apparier dans les deux images, on obtient $B(15, 15) > 300\ 000$ milliards d'appariements à tester. Une recherche exhaustive est donc

hors de question. Aussi, le principe le plus fréquent est de procéder itérativement : pour le point p_1, on cherche son meilleur correspondant parmi les points q_1 à q_m. Ensuite, on cherche le meilleur correspondant du point p_2 dans les $m-1$ points restants de l'image 2, et ainsi de suite. La complexité tombe à $\mathcal{O}(nm)$, ce qui est acceptable dans le cas de l'appariement épars, où n et m sont souvent de l'ordre de la centaine.

Une autre façon de trouver l'expression de $B(n,m)$ est la suivante. Supposons $n > m$. S'il y a k points de l'image 1 appariés avec des points de l'image 2, alors il y a C_n^{n-k} façons de choisir les $n-k$ points non-appariés. Et à chaque fois, il y a A_m^k façons d'apparier les autres. Donc :

$$B(n,m) = \sum_{k=0}^{k=m} C_n^{n-k}\, A_m^k \tag{3.27}$$

$$= \sum_{k=0}^{k=m} \frac{n!\; m!}{k!\; (n-k)!\; (m-k)!} \tag{3.28}$$

Les deux définitions de B sont équivalentes, et on vérifie de plus que $B(n,m) = B(m,n)$, ce qui permet de calculer B si $n \leq m$.

3.2.3.2 Appariement dense

L'appariement dense consiste à mettre en correspondance autant de pixels que possible dans les deux images. Contrairement à l'appariement épars, les points à mettre en correspondance ne sont pas «connus à l'avance», dans un certain sens. Pour des images classiques de taille 500×500, la recherche itérative n'est pas applicable telle quelle car $nm = 500^4 > 60$ milliards de tests. On utilise donc des contraintes supplémentaires : limite de disparité, contrainte épipolaire, contraintes de cohérence globale (les appariements ne sont cherchés que dans des régions voisines des appariements des voisins). L'appariement dense est le plus souvent binoculaire. S'il est multi-oculaire, il est réalisé par une succession d'appariements binoculaires ramenés dans un référentiel commun, ou par recherche dans un espace 3D, ou par une combinaison de ces deux méthodes. Pour limiter l'espace de recherche, des contraintes multi-linéaires peuvent aussi être imposées (trilinéarités, quadrilinéarités) ; comme nous l'avons vu en 2.2.3, elles sont équivalentes, dans le cas étalonné, à des contraintes sur la position des points 3D reconstruits.

Les méthodes d'optimisation utilisées peuvent être classées en 4 catégories :

1. méthodes directes : on effectue simplement de nombreux appariements épars ;
2. résolution par programmation dynamique ;
3. approches énergétiques et utilisation de modèles de surfaces, de splines ;
4. appariement dans un espace de paramètres.

De plus, chaque implémentation peut inclure une ou plusieurs des caractéristiques suivantes :

- □ approche hiérarchique (multi-échelle) ;
- □ régularisation *a priori* ;
- □ régularisation *a posteriori* ;
- □ calcul en plusieurs passes (p. ex. utilisation de fenêtres de corrélation adaptatives en seconde passe).

Les étapes de régularisation semblent déterminantes pour le succès de l'appariement, à tel point que les corrélations sont parfois réduites à des simples différences d'intensité de pixels (corrélations 1×1).

Nous détaillons ci-dessous les quatre catégories d'algorithmes recensées. Nous nommerons les algorithmes pour pouvoir les référencer dans la suite, et les abréviations utilisées sont résumées en p. 141.

3.2.3.3 Méthodes directes

Nous appelons «méthodes directes» les méthodes d'appariement dense qui consistent simplement à appliquer des méthodes d'appariement épars, mais sur un plus grand nombre de points, avec le plus souvent une contrainte de disparité et la contrainte épipolaire.

L'algorithme le plus basique fonctionne de la façon suivante : pour le point p_1 de l'image 1, on cherche son meilleur correspondant q_{i_1} dans l'image 2 (celui présentant la meilleure corrélation). Ensuite, on cherche pour le point p_2 de l'image 1 le meilleur correspondant q_{i_2} dans l'image 2, parmi les points restants, et ainsi de suite. Cet algorithme est en $\mathcal{O}(nm)$; il n'attribue pas un rôle symétrique aux deux images, et donne un correspondant à chaque point de l'image 1. On peut faire intervenir un seuil pour ne garder que les correspondants suffisamment bons, mais du réglage de ce seuil dépendra la fiabilité de l'algorithme. Nous appelons cet algorithme «Winner Takes All» : WTA.

Une amélioration simple est d'effectuer une vérification croisée des hypothèses d'appariement. Cette méthode est depuis longtemps et très couramment employée. Elle consiste à vérifier que pour q_{i_1} dans l'image 2, son meilleur correspondant est bien p_1 dans l'image 1. On effectue donc deux recherches croisées, l'une de l'image 1 vers 2, et l'autre de l'image 2 vers 1. Si la vérification réussit, alors l'appariement (p_1, q_{i_1}) est accepté ; sinon p_1 n'a pas de correspondant. Les cartes de disparité obtenues comportent donc des zones non renseignées, et ces trous correspondent le plus souvent à des occultations. La méthode est symétrique, et le résultat est largement fiabilisé par rapport à WTA. Nous appelons cet algorithme de vérification croisée «Cross Check Raw» (CCR). Comme pour WTA, un seuil de rejet peut être fixé, et nous appellerons cet algorithme «Cross Check Threshold» (CCT).

Il est curieux de voir que la première description de CCR que nous ayons pu recenser date de 1991, ce qui est relativement récent, dans un article de P. Fua [Fua 91]. Dans ce même article, il propose aussi de rejeter tous les appariements isolés, ce qui réduit le nombre de fausses mises en correspondance. Ensuite, il procède à un lissage dans les

directions parallèles aux contours (et non dans les directions perpendiculaires, cela pour éviter de rendre les contours flous). Les contours sont les zones de l'image présentant un fort gradient de niveau de gris. La carte de disparité obtenue étant floue (trop lissée), il effectue une seconde étape de lissage, en prenant cette fois en compte simultanément les informations de gradient de niveau de gris, et les gradients de disparité. On évite alors de lisser les zones trop pentues, et les ruptures de disparité sont bien conservées.

Il faut noter que le lissage est une sorte de moyennage, car il ne fait intervenir que les informations de disparité déjà calculées : aucune mesure de corrélation n'est re-calculée, et aucun nouvel appariement n'est effectué. La validité du lissage vis-à-vis de la vraie structure 3D de la scène n'est donc pas assurée, car on n'utilise plus les informations des images après la première phase de corrélation croisée.

Citons enfin l'algorithme PMF, inventé par S.B. Pollard, J.E.W. Mayhew et J.P. Frisby à l'université de Sheffield, pour sa grande ressemblance avec CCR. Il est décrit dans la thèse de S.B. Pollard [Pol 85a], ou plus succinctement dans [Pol 85b], donc bien antérieurement aux premières descriptions de CCR. Le coût d'appariement c_{ij} entre les points p_i dans l'image 1 et q_j dans l'image 2 est cette fois la *somme* des mesures de ressemblance entre tous les points d'un voisinage de p_i et tous les points d'un voisinage de q_j. On constitue ainsi un tableau à deux dimensions du coût d'appariement entre chaque point p_i et chaque point q_j, qui prend en compte un certain support local. Nous supposerons pour la suite que les points de l'image 1 indexent les lignes de ce tableau, et les points de l'image 2, les colonnes.

Le meilleur élément $c_{i_0 j_0}$ du tableau signale un appariement entre le point p_{i_0} dans l'image 1 et le point q_{j_0} dans l'image 2. Ensuite, la ligne i_0 et la colonne j_0 ne sont plus considérées, et on recherche le meilleur élément restant, et ainsi de suite. Cela diffère de CCR, car dans CCR la ligne i_0 et la colonne j_0 sont toujours considérées. Les deux algorithmes peuvent donner des résultats différents, comme le montre la figure 3.11.

	q_1	q_2	q_3	q_4
p_1	0.7	0.5	0.1	0.7
p_2	0.7	0.8	0.2	0.8

FIG. 3.11: *Tableau de coûts d'appariement entre 2 points de l'image 1 et 4 points de l'image 2. Les algorithmes* CCR *et* PMF *donnent des résultats différents (voir texte).*

Sur l'exemple de la figure 3.11, l'algorithme CCR fonctionne de la façon suivante : sur la ligne p_1, le meilleur score est 0.1, dans la colonne q_3. Dans cette colonne, le meilleur score est 0.1, sur la ligne p_1. La vérification croisée fonctionne, et il y a donc un appariement (p_1, q_3). Sur la ligne p_2, le meilleur score est 0.2, dans la colonne q_3. Dans cette colonne, le meilleur score est 0.1, sur la ligne p_1. La vérification croisée échoue, et il n'y a pas d'appariement pour p_2.

Pour l'algorithme PMF en revanche, après le choix d'apparier p_1 et q_3 (sur le même principe), la ligne p_1 et la colonne q_3 ne sont plus considérées. On cherche donc le meilleur score *restant* pour la ligne p_2, et celui-ci est 0.7, pour la colonne q_1.

En résumé, PMF fournit la liste d'appariements $\{(p_1, q_3), (p_2, q_1)\}$, alors que CCR fournit

la liste $\{(p_1, q_3)\}$. Il est clair que PMF est moins robuste que CCR, car il peut fournir des appariements peu probables, comme (p_2, q_1), qui a un score très mauvais de 0.7. Il nous semble que CCR est plus correct : p_2 ressemble beaucoup à q_3, mais comme ce dernier a déjà été apparié, il est normal (contrainte d'unicité) de dire que p_2 n'a pas de correspondant.

La nouveauté principale de PMF est en fait d'intégrer une contrainte de gradient de disparité, en plus de la contrainte épipolaire et de la contrainte d'unicité, qui permet de mieux désambiguïser les appariements. Cette contrainte empêche de considérer les appariements qui conduiraient à la perception de surfaces trop pentues ; elle est issue de constatations expérimentales sur les performances du système visuel humain. PMF a été repris ensuite, par exemple par M.A. O'Neill ([O'N 92], University College of London), qui l'utilise sur des contours extraits des images. Les régions délimitées par les contours appariés sont ensuite traitées à leur tour, pour produire un appariement dense. L'article ne présente que les cartes de disparité obtenues, et ne fournit pas d'évaluation claire.

3.2.3.4 Programmation dynamique

Les algorithmes de programmation dynamique sont apparus il y a une quinzaine d'années, et appliquent cette technique de recherche opérationnelle à l'appariement de deux listes de points ordonnées. Le principe est une recherche de chemin optimal dans un graphe.

Pour apparier les listes $(p_1, ..., p_n)$ et $(q_1, ..., q_m)$, on examine successivement la ressemblance de chacun des couples (p_i, q_j), et leur corrélation c_{ij}. Dans le graphe de la programmation dynamique, ceci correspond au nœud (i, j). À partir de ce nœud, trois transitions vers d'autres nœuds sont possibles, qui correspondent à trois décisions :

1. (p_i, q_j) forme un appariement. On examinera par la suite la validité du couple (p_{i+1}, q_{j+1}). Cette transition a un coût k^1_{ij}, et mène au nœud $(i+1, j+1)$.

2. Les points ne peuvent pas être appariés. Le point q_j est occulté dans l'image 1, et on examinera par la suite la validité du couple (p_i, q_{j+1}). Cette transition a un coût k^2_{ij}, et mène au nœud $(i, j+1)$.

3. Les points ne peuvent pas être appariés. Le point p_i est occulté dans l'image 2, et on examinera par la suite la validité du couple (p_{i+1}, q_j). Cette transition a un coût k^3_{ij}, et mène au nœud $(i+1, j)$.

Un tableau 2D peut stocker les coûts de ces $3(n-1)(m-1)$ transitions, et l'algorithme de programmation dynamique permet de trouver en un temps $\mathcal{O}(nm)$ le chemin optimal menant du nœud $(1, 1)$ au nœud (n, m), le coût total du chemin étant égal à la somme des coûts élémentaires des transitions le composant. La contrainte d'unicité est implicitement respectée, ainsi que la contrainte d'ordre : si p_i et q_j sont appariés, alors p_{i+a} ne pourra être apparié qu'avec un q_{j+b}, avec $a > 0$ et $b > 0$. Dans ce tableau, les transitions verticales et horizontales correspondent à des pixels occultés dans l'image 1 ou l'image 2, et les transitions diagonales à des appariements. En observant le chemin optimal, on repère immédiatement les zones occultées dans les deux images.

Nous appelons cet algorithme DP3 : «Dynamic Programming 3-Transitions».

La bonne marche du système dépend du réglage des coûts k^1, k^2, k^3. Dans toutes les implémentations recensées, $k^1_{ij} = c_{ij}$, et $k^2_{ij} = k^3_{ij} = \kappa$, et le vrai problème est le réglage du coût d'occultation κ.

Ce coût d'occultation fixe la pénalité appliquée au chemin pour chaque couple non-apparié. Il est délicat à régler, car il peut changer entièrement le comportement de l'algorithme : si κ est très faible, seuls les points très fiables sont appariés ; si κ est fort, l'appariement est plus dense, mais les occultations ne sont pas détectées. Selon les valeurs de κ, les appariements obtenus peuvent être sans rapport, car l'ensemble des appariements n'est pas croissant (au sens de l'inclusion) avec κ. Il impose de plus un comportement binaire de l'algorithme : un couple de mesure $c_{ij} = \kappa - \varepsilon$ est déclaré «complètement» apparié, car une transition de ce type est plus avantageuse qu'une transition d'occultation de coût κ, alors qu'un couple de mesure $c_{ij} = \kappa + \varepsilon$ ne l'est «pas du tout». Bien que ce comportement *local* soit tempéré par la recherche d'un chemin *globalement* optimal, des mesures progressives seraient peut-être plus adaptées (des probabilités d'appariement, par exemple).

La contrainte d'ordre étant imposée, il faut pouvoir définir un ordre logique sur les listes $(p_1, ..., p_n)$ et $(q_1, ..., q_m)$. Aussi, la programmation dynamique n'est utilisée que pour des points appartenant à des lignes épipolaires conjuguées, et elle est particulièrement adaptée à l'appariement dense de points (elle peut aussi s'utiliser sur des contours). Sur les lignes épipolaires conjuguées, la notion d'ordre a un sens, et correspond au fait qu'il ne peut pas y avoir de «croisement» dans les appariements. C'est un inconvénient pour de nombreuses scènes, où les croisements ne sont pas rares : présence d'arbres, ou de poteaux, au premier plan de la scène (voir par exemple la figure 3.3).

La mesure de ressemblance employée est souvent une simple différence d'intensités (corrélation SAD 1×1). Le fait que le chemin optimal soit nécessairement continu impose une cohérence semi-globale, et suffit à régulariser les appariements de manière satisfaisante. La cohérence n'est pas globale, c.-à-d. elle ne s'applique pas à toute l'image, mais à chaque couple d'épipolaires conjuguées, de façon indépendante les uns des autres.

Nous citons rapidement quelques références sur le sujet, qui apportent parfois des réponses aux problèmes évoqués.

Historiquement, la première référence sur l'utilisation de la programmation dynamique pour l'appariement d'un couple d'images semble être l'article de H.H. Baker et T.O. Binford à Stanford, dans IJCAI'81, sur la base de ce qui se faisait déjà en reconnaissance de signal de parole.

Dans [Bak 81], les auteurs extraient des segments des images, les apparient par programmation dynamique puis apparient les pixels entre les segments. La résolution est hiérarchique *coarse-to-fine*, ce qui limite la complexité et régularise la solution. De plus, une contrainte de continuité figurale est utilisée pour l'appariement de segments : les segments appariés doivent présenter la même topologie. La fonction de ressemblance utilisée pour les segments prend en compte leur contraste et leur orientation, et traite des demi-segments : en séparant les deux «faces» des segments, les segments correspondant à des frontières d'occultation sont correctement traités (on apparie seulement les faces droites ou les faces gauches des deux segments, voir figure 3.12).

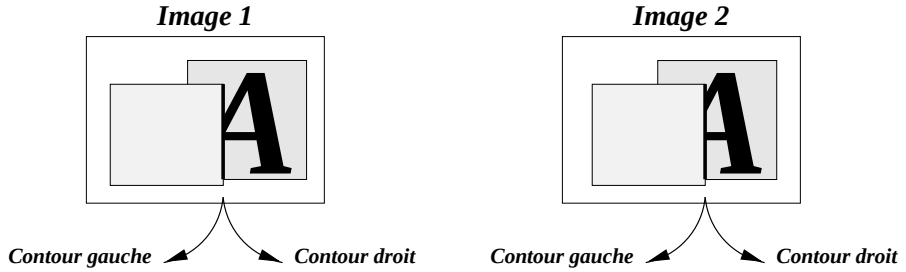


FIG. 3.12: *Le contour gauche de l'image 1 est apparié au contour gauche de l'image 2, mais les contours droits ne sont pas appariés (il s'agit d'un contour occultant).*

Les images utilisées sont des images aériennes, où la contrainte d'ordre est toujours respectée. Elles sont entièrement étalonnées, ce qui permet de calculer la géométrie épipolaire (aujourd'hui, on la calculerait directement à partir des images).

La méthode est un peu plus détaillée dans [Bak 82], mais on ne parvient toujours pas à savoir si les occultations de pixels (entre les contours) sont traitées. Ce n'est probablement pas le cas, car l'appariement de segments est supposé avoir éliminé toutes les causes d'occultation ; l'algorithme utilisé serait alors plus proche de DP3NO que de DP3 (l'algorithme DP3NO sera décrit en 3.4.4.3, et correspond au fait que les occultations sont considérées comme des appariements multiples : plusieurs pixels de l'image 1 sont appariés à un seul pixel de l'image 2, ou inversement). Ces travaux sont repris par G.V.S. Raju, de l'Ohio University (en collaboration avec T.O. Binford) dans [Raj 87], en améliorant la complexité algorithmique.

Y. Ohta, de l'Université de Tsukuba et T. Kanade, à CMU, proposent dans [Oht 85] un algorithme imposant une cohérence globale sur l'image. Outre la cohérence intra-ligne obtenue par DP3, ils prennent en compte des contraintes inter-lignes, qui sont en fait des contraintes de continuité figurale sur des segments. Le principe est d'empiler tous les tableaux 2D de programmation dynamique pour tous les couples d'épipolaires. Ceci est possible sans redondance, car les images sont rectifiées. On applique alors un algorithme de programmation dynamique dans ce cube 3D, qui trouve le chemin optimal pour la correspondance des pixels et simultanément pour la correspondance des segments. Certaines contraintes topologiques doivent êtres respectées pour les segments extraits (par exemple ils ne doivent pas se croiser), et l'algorithme d'extraction de contours et de chaînage doit être adapté en conséquence. Les résultats sont meilleurs que ceux de H.H. Baker, mais la cohérence inter-lignes, bien que renforcée, n'est toujours pas strictement respectée. À notre sens, des résultats équivalents pourraient être obtenus avec un simple DP3 utilisant des masques de corrélation plus grands.

D. Geiger, chez Siemens à Princeton, établit dans [Gei 92] une formulation probabiliste des coûts de transition. En posant que la probabilité d'appariement du couple (p_i, q_j) est de la forme $e^{c_{ij}}$, et que la surface de disparité est localement lisse, alors il obtient une formulation mathématique du coût d'occultation :

$$\kappa(\delta) = \varepsilon|\delta| + \frac{\mu}{N}\sqrt{|\delta|} \tag{3.29}$$

δ est la largeur de l'occultation, donc pour nous $\delta = 1$. La variable N est le nombre de points à apparier, et le problème reste que ε et μ sont des paramètres «à déterminer». De fait, même avec cette formulation, la valeur de κ doit être déterminée empiriquement. La mesure de ressemblance utilisée est une corrélation 3×3, ce qui permet de prendre en compte les informations des lignes supérieure et inférieure, et d'imposer une certaine régularité. Les résultats sont relativement bons.

I.J. Cox au NEC Research Institute [Cox 92] ajoute des contraintes de cohérence à la fonction de coût, en prenant en compte les appariements de la ligne épipolaire située juste au-dessus de la ligne courante. Cela n'est pas symétrique, mais l'auteur a constaté qu'un algorithme en deux passes, où l'on peut comparer la ligne courante à la fois à la ligne supérieure et à la ligne inférieure, n'améliore pas notablement la solution. Le calcul est très rapide, car basé sur de simples différences d'intensités de pixels (masque de corrélation 1×1). C'est d'ailleurs sans doute une des causes de la faible cohérence observée dans les résultats de l'algorithme de programmation dynamique standard. Le coût κ est dérivé d'une formule mathématique prenant en compte le champ de vue de la caméra, la probabilité d'occultation, et la covariance attendue sur les erreurs de mesure. En pratique, ces deux derniers paramètres semblent fixés arbitrairement.

L'algorithme est étendu dans [Cox 96] au cas N-oculaire. La recherche de correspondance a toujours lieu dans un cadre binoculaire, mais la fonction de coût intègre aussi les intensités des pixels dans les $N-2$ autres images : pour chaque hypothèse d'appariement testée dans les 2 images de référence, il suffit de reconstruire le point 3D et de le reprojeter dans les $N-2$ autres images.

C. Baillard à l'IGN et H. Maître à l'ENST appliquent la programmation dynamique sur des segments extraits d'images aériennes, puis sur les pixels situés entre les segments [Bai 96]. Des contraintes de cohérence et des corrections après l'étape d'appariement de segments permettent d'améliorer les résultats. On note que la valeur de κ (fixée ici arbitrairement) est déterminante pour de bons résultats, et que cette valeur correspond en fait à la corrélation maximale acceptable pour un couple apparié. La cohérence inter-lignes est assurée par l'utilisation de mesures de corrélation 7×7 ou 9×9.

B. Serra (Aérospatiale) et M. Berthod (INRIA) réalisent un appariement sous-pixellique de contours par programmation dynamique continue [Ser 95]. Ils calculent d'abord analytiquement l'appariement optimal entre deux segments. Les contours pouvant être approximés par des chaînes de micro-segments, on peut alors calculer l'appariement de deux contours comme l'appariement globalement optimal des segments les composant. Comme les extrémités de ces micro-segments peuvent être situées sur des pixels non-entiers (extrémités sous-pixelliques), il est impossible d'utiliser la programmation dynamique classique pour trouver l'appariement optimal. B. Serra utilise une programmation dynamique *continue* : les transitions optimales ne sont pas de direction fixe, par exemple du nœud (i, j) au nœud $(i+1, j+1)$, mais de direction variable, par exemple du nœud (i, j) au nœud $(i + di, j + dj)$, où $di \leq 1$ et $dj \leq 1$ sont les valeurs conduisant à une transition de coût

minimal. L'algorithme est garanti de fonctionner dans certaines conditions, en particulier sur la fonction de coût, et ces aspects sont détaillés dans [Ser 94] (certaines heuristiques simplificatrices doivent être utilisées).

Les résultats sont très supérieurs à la programmation dynamique classique (discrète), si on l'appliquait de la même façon au même problème, c.-à-d. par l'appariement de micro-segments (mais ce n'est pas la façon habituelle de procéder). Les appariements obtenus sont, par construction, de précision sous-pixellique.

Détermination des seuils

Comme nous l'avons vu, l'une des difficultés dans la mise en œuvre de la programmation dynamique est le réglage des différents coûts et seuils, et essentiellement du coût d'occultation κ. Nous avons tenté nous-mêmes d'établir une formulation de ce coût en fonction d'hypothèses *a priori*, dans la lignée des travaux de I.J. Cox et D. Geiger. Si l'on considère un couple (p_i, q_j), de coût de ressemblance $c_{ij} = c_0$, alors la probabilité que l'événement A : «p_i et q_j sont appariés» soit observé est donné par l'équation 3.30 (formule de Bayes).

$$P(A|c_{ij} = c_0) = \frac{P(c_{ij} = c_0|A)\ P(A)}{P(c_{ij} = c_0)} \tag{3.30}$$

Les différents termes peuvent être déterminés de manière empirique. $P(A)$ est la probabilité que deux points soient appariés *a priori*; cette valeur est fonction du nombre de couples observés, et du taux d'appariement attendu. $P(c_{ij} = c_0)$ est la probabilité que le coût de ressemblance d'un couple soit égal à c_0 *a priori*; cela dépend de la distribution de la fonction de ressemblance parmi les couples observés, et cette distribution doit être estimée en premier lieu. Enfin, $P(c_{ij} = c_0|A)$ est la probabilité que le coût de ressemblance d'un couple apparié soit égal à c_0. Là aussi, une première passe est nécessaire afin de déterminer des appariements, puis d'évaluer la distribution de la fonction de ressemblance pour ces couples appariés.

D'après notre expérience, une telle formulation apporte peu : certaines valeurs doivent toujours être fixées arbitrairement (le taux d'occultation), et une première passe d'appariement est nécessaire, ce qui biaise les résultats. De plus, l'intérêt de cette formulation serait de pouvoir déterminer la limite κ pour les algorithmes de programmation dynamique : dans le cas de la formulation probabiliste, on fixerait le seuil à 0.5. Cependant, ceci ne fonctionne pas, car même pour des couples très ressemblants, la probabilité qu'ils soient appariés est très inférieure à 0.5 : il suffit que la probabilité $P(c_{ij} = c_0)$ soit assez forte (ce qui signifie que plusieurs candidats sont possibles). Dans ce cas, l'algorithme de programmation dynamique favorisera systématiquement les transitions d'occultation, car elles seront toujours beaucoup plus probables que les transitions d'appariement.

Le réglage des paramètres des algorithmes de programmation dynamique est donc une question délicate. La leçon que nous retirons des travaux de D. Geiger et I.J. Cox, et de nos propres expériences sur la formulation bayesienne de la probabilité d'appariement, est qu'il est inutile d'établir une formule mathématique, car il faut de toute façon fixer les valeurs

de certaines variables par des connaissances *a priori* sur la scène ou sur les fonctions de ressemblance utilisées. Il nous semble aussi simple de conserver la formulation initiale de la programmation dynamique, et de fixer directement la valeur de κ.

3.2.3.5 Appariement dans un espace de paramètres

C. Tomasi et R. Manduchi, à Stanford, présentent dans [Tom 96a] une méthode d'appariement originale, dite sans recherche : «Stereo Without Search». Il s'agit de calculer un appariement dense en mettant en correspondance des courbes implicites qui représentent les caractéristiques de deux épipolaires conjuguées.

Par exemple, si on n'utilise que les deux caractéristiques «intensité» et «gradient», alors on peut tracer une courbe 2D pour chaque ligne épipolaire, représentant l'intensité de chaque pixel en fonction de son gradient. La courbe présente des boucles (ce n'est pas une fonction explicite), et son abscisse curviligne est l'abscisse du point dans le segment épipolaire. On apparie ensuite les deux telles courbes sur des critères de distance.

Comme dans toutes les autres méthodes, les problèmes restent la définition de cette distance, et la gestion des occultations. Les occultations modifient en effet profondément la forme de la courbe 2D, et il faut que l'algorithme soit robuste à ces écarts.

On ne peut pas dire que cette méthode soit réellement «sans recherche», car en fait, la recherche est déplacée dans une autre dimension. L'article ne contient pas de tests permettant de se faire une opinion. L'algorithme est plus détaillé dans [Tom 96b], mais sans tests non plus.

3.2.3.6 Approches énergétiques

Dans les approches énergétiques, on tente d'optimiser *globalement* la mise en correspondance. Chaque configuration de mise en correspondance représente une certaine énergie, qu'on essaie de minimiser.

Une implémentation presque directe de cette méthode est proposée par M.H. Ouali, à l'École des Mines de Paris [Oua 96]. Il réalise un appariement dense par recuit simulé, où la configuration du système est l'ensemble des appariements. L'énergie de la configuration est la somme des valeurs de corrélation des points + un terme de lissage + une contrainte sur le nombre d'occultations (points non appariés). Le terme de lissage est «déconnecté» sur les arêtes, pour ne pas lisser les ruptures de disparité. La contrainte sur le nombre d'occultations sert à éviter de converger vers une solution où aucun point ne serait apparié (l'énergie serait optimale, car nulle). Enfin, les contraintes d'unicité et épipolaire sont assurées par la forme des configurations autorisées. Le principe du recuit simulé est le suivant :

1. initialiser la configuration aléatoirement, et la température T ;
2. chercher un équilibre thermique à T :
   (a) effectuer un changement aléatoire dans la configuration (c.-à-d. modifier un appariement) ;

(b) calculer la différence ΔE d'énergie :

- □ si l'énergie diminue, accepter le changement ;
- □ si elle augmente, l'accepter avec une probabilité $e^{-\frac{\Delta E}{T}}$;

(c) répéter (a) et (b) jusqu'à l'équilibre ;

3. diminuer T ;

4. répéter les étapes 2 et 3 jusqu'au minimum d'énergie.

L'implémentation proposée n'est testée que sur des images de taille 60×40. Utiliser de si petites images rend impossible l'évaluation des résultats, et pose la question du temps de convergence pour des images de taille plus réaliste. Enfin, on ne connaît pas les pondérations des différents termes d'énergie, ni leur sensibilité. Il n'est pas garanti que l'algorithme converge à la même vitesse, ni vers le même minimum local, si l'on change ces pondérations.

L. Robert et R. Deriche à l'INRIA posent le problème de la même façon [Rob 96]. Soit la fonction Z, qui donne la profondeur $Z(p)$ de chaque point p ; on cherche alors Z qui minimise l'énergie $E(Z)$, comprenant un terme d'énergie $M_{12}(Z)$ et un terme de régularisation $S(Z)$. Plus formellement :

$$E(Z) = M_{12}(Z) + \lambda S(Z) \tag{3.31}$$

$M_{12}(Z)$ mesure la corrélation globale des deux images :

$$M_{12}(Z) = \iint \|I_1(p) - I_2(f(p, Z(p)))\|^2 \, dp \tag{3.32}$$

La fonction $I_k(p)$ représente l'intensité du point p dans l'image k, mais peut aussi être plus évoluée, et multidimensionnelle, intégrant par exemple des informations sur les gradients ou les contours.

Dans cette formulation, intégrer l'information de plus de deux images est facile, car les points sont tous repérés par leur profondeur Z dans un repère commun, et l'énergie ne dépend que de la fonction Z. On peut simplement sommer les corrélations binoculaires en un terme $M(Z)$:

$$M(Z) = \sum_{i=2..n} M_{1i} \tag{3.33}$$

$S(Z)$ est le terme régularisateur. Il est délicat à choisir, car il faut qu'il préserve les discontinuités. Aussi L. Robert et R. Deriche choisissent-ils un terme de filtrage anisotropique, qui lisse dans les directions parallèles aux contours (lignes du gradient ∇Z), mais non dans les directions transversales. C'est le rôle de la fonction Φ dans l'équation suivante :

$$S(Z) = \iint \Phi(|\nabla_p Z|)\, dp \tag{3.34}$$

Sur ces critères, les auteurs établissent une liste de fonctions Φ utilisables. Ainsi, dans la figure 3.13, bien que les trois fonctions se ressemblent, seules les deux dernières satisfont les critères mentionnés et garantissent un bon fonctionnement de l'algorithme. L'explication mathématique de ce phénomène constitue un apport essentiel de cet article.

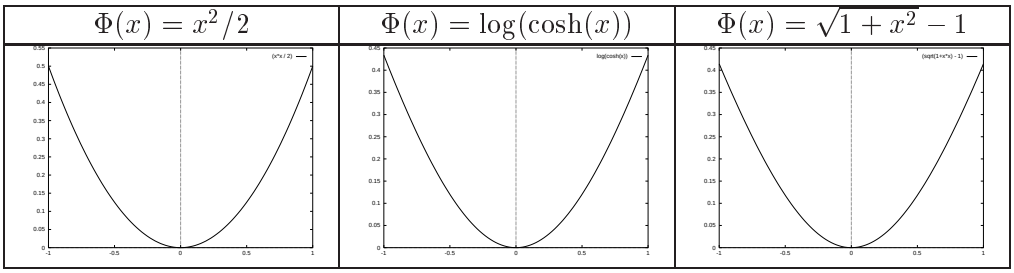


FIG. 3.13: *De ces trois fonctions semblables, seules les deux dernières permettent de réaliser le filtrage anisotropique.*

Les résultats semblent très bons, mais ne sont évalués que visuellement, et sur un seul couple d'images synthétiques. L'algorithme n'est pas utilisable tel quel dans notre cas, car il nécessite un étalonnage pour le calcul des profondeurs Z. L'influence de λ sur la qualité des résultats n'est pas explicitée.

3.2.3.7 Recherche dans un espace 3D

Certains algorithmes d'appariement sont parfois nommés «sans correspondance», car ils ne reposent pas sur l'établissement préalable de correspondances de points dans les images. Ils effectuent une recherche dans l'espace 3D, et n'en déduisent des informations 2D qu'*a posteriori*.

À l'université du Massachusetts à Amherst, Y-Q. Cheng et R.T. Collins [Che 94] mettent en correspondance les points $(p_1, ..., p_n)$ dans l'image 1 avec les points $(q_1, ..., q_m)$ dans l'image 2. À partir de deux points quelconques p_i et q_j, ils reconstruisent le point 3D par triangulation, et le reprojettent dans les images 1 et 2 en p'_i et q'_j. Le critère d'erreur E_{ij} est l'erreur de reprojection $d(p_i, p'_i) + d(p_j, p'_j)$, c.-à-d. $-E_{ij}$ mesure l'attraction entre les points p_i et p_j, comme le ferait une mesure de corrélation. Un algorithme CCR appliqué à la matrice E permet alors de déterminer un ensemble d'appariements sous-optimal.

Il est étonnant que la mesure E_{ij} ne prenne absolument pas en compte le contenu des images : niveaux de gris, et ressemblance des points. Le système fonctionne malgré tout, car n et m sont suffisamment petits (de l'ordre d'une centaine de points) et les caméras si précisément étalonnées, qu'il est très improbable que E_{ij} soit faible «par hasard» (il est très probable qu'une mesure E_{ij} faible corresponde bien à un couple (p_i, q_j) apparié). Notons enfin que dans le cas binoculaire, cela est équivalent à tester si le couple (p_i, q_j)

respecte la géométrie épipolaire : on ne peut pas désambiguïser les appariements qui se trouveraient le long d'une même droite épipolaire.

Dans un article postérieur, R.T. Collins [Col 96] reprend le même principe. Un étalonnage précis est donc nécessaire, et le système ne fonctionne que pour un nombre prédéterminé de points à apparier. Un plan quadrillé d'équation $Z = z_i$ balaie l'espace 3D. On compte le nombre de lignes de vue passant par les points des images et coupant chaque cellule du plan quadrillé. Les cellules 3D recueillant le plus grand nombre de votes sont susceptibles de correspondre à un point 3D détecté (voir figure 3.14).

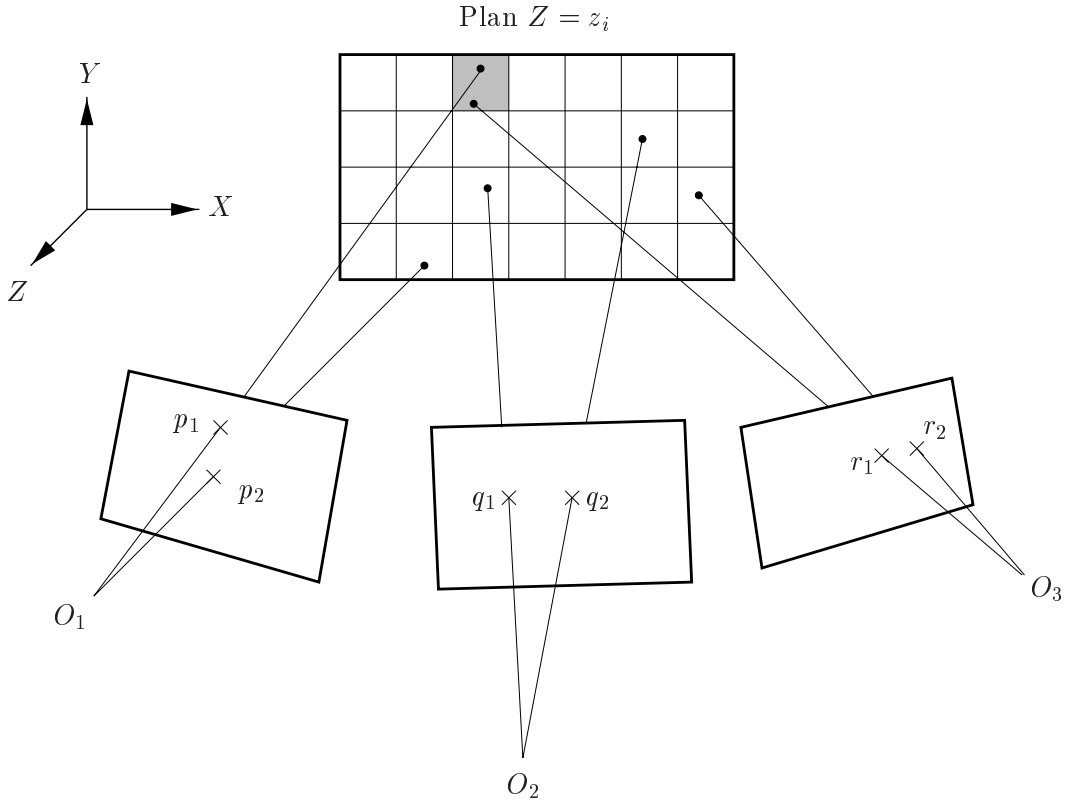


FIG. 3.14: *Pour cette valeur de z_i, seule la cellule grisée recueille un nombre suffisant de votes. En conséquence, seul un appariement (p_1, r_1) est établi à ce stade (le plan $Z = z_i$ est ensuite translaté le long de l'axe des Z, à d'autres valeurs de z_i).*

Le réglage du seuil de détection est problématique : avec un seuil élevé, on détecte peu de points 3D, et ils sont fiables ; avec un seuil plus faible, on détecte plus de points, mais ils sont moins fiables. Comme précédemment, la méthode revient en fait à grouper en appariements les points cohérents vis-à-vis de la géométrie multi-oculaire.

L'approche de S.M. Seitz (Université du Wisconsin à Madison) est exactement similaire, à ceci près qu'elle utilise les informations des images. Dans [Sei 97], il déplace un plan 3D de plus en plus loin des caméras et reprojette chaque cellule (ou voxel) du plan dans toutes les images. Si la couleur des projections des voxels est constante (modulo des

considérations de robustesse), alors un point 3D est détecté, et on lui attribue la couleur de ses projections. Sinon, il n'y a pas de point 3D dans ce voxel.

Cette différence fondamentale permet de faire de l'appariement dense, car l'ambiguïté est beaucoup moins forte, même pour n et m élevés. La méthode revient alors à tester des mesures de corrélations 1×1 dans un cadre multi-oculaire (respect des contraintes multi-linéaires). Les résultats sur quelques images-tests sont très bons visuellement, mais il n'y a pas d'évaluation quantitative.

Les occultations sont prises en compte de la façon suivante : toutes les images sont doublées d'un plan de bits indiquant si chacun des pixels est occulté (1) ou non-occulté (0). Initialement, toutes ces matrices sont à 0. Lorsqu'on vérifie que la couleur des projections est constante, on ne prend en compte que des pixels non-occultés. À chaque fois ensuite qu'un point 3D est détecté, alors les pixels de ses projections dans les images sont marqués comme occultés par ce voxel. Grâce à l'ordre du parcours 3D (le plan 3D s'éloigne des caméras), il est garanti que tous les pixels marqués à 0 ne sont pas occultés par des voxels précédemment calculés.

Ce principe de recherche dans l'espace 3D est aussi apparu sous une forme moins élaborée dans [Oku 93], où M. Okutomi calcule la valeur de vraisemblance qu'un point p de l'image 1 ait une profondeur z. Cette valeur est la valeur de la corrélation entre le point p dans l'image 1, et ses projections dans les autres images. Ces projections peuvent être déterminées grâce à z, car les caméras sont toutes étalonnées.

Formulation énergétique de la recherche 3D

Dans [Fua 94b], P. Fua au SRI propose une formulation énergétique de la mise en correspondance par recherche dans un espace 3D.

Il retrouve la structure de la scène et les paramètres extrinsèques des caméras en minimisant une fonction d'énergie comprenant toutes ces variables : arêtes du maillage représentant la surface observée, les $6(N-1)$ paramètres des N caméras, et les contraintes stéréo (c.-à-d. les scores de corrélation). La fonction d'énergie n'étant pas convexe, une simple descente de gradient est impossible, et on ajoute comme d'habitude un terme de régularisation qui convexifie la fonction, et dont on réduit progressivement l'influence pour garantir la convergence. Ce terme sert non seulement à forcer la convergence, mais aussi à régulariser (lisser) la solution, et à limiter l'influence du bruit. Les résultats montrent une certaine robustesse à un étalonnage approximatif, ou à des changements de structure de la scène : une erreur jusqu'à 10 pixels dans l'étalonnage initial ne conduit qu'à des erreurs de 1 pixel dans les calculs finaux. On peut ainsi effectuer un suivi d'objets déformables, ou une mise à jour itérative sur une séquence d'images. Inconvénient : la méthode ne fonctionne que sur une seule surface (connexe), et qui doit probablement être assez lisse.

Dans [Fua 94c], P. Fua aborde le problème de la pondération des différents termes dans la fonction d'énergie. Dans ses expériences, elles sont fixées une fois pour toutes par l'utilisateur pour chaque contexte d'utilisation, mais fonctionnent ensuite pour toutes les images qui entrent dans ce contexte. L'auteur a remarqué une certaine robustesse du résultat à un changement de ces paramètres ; le plus important semble être en fait de normaliser les différents termes de la fonction d'énergie. Ce qu'il entend par «contexte d'utilisation»

est par exemple : «utilisation combinée du *shape from shading* et des silhouettes 2D », ou encore : «images aériennes». C'est donc un indicateur du type de données à traiter. On ne retrouve toujours qu'une seule surface, lisse.

Il essaye d'étendre le système à des surfaces multiples dans [Fua 94a, Fua 95b, Fua 95a], où il génère des particules orientées (petits morceaux de plans de contour elliptique). Ces particules sont représentées par des couples (*point*, *orientation*), et elles sont ensuite reliées sur des critères de co-courbure, une courbure maximale étant imposée arbitrairement. On rejette aussi toutes celles qui n'ont pas suffisamment de voisins (robustesse). Suit une étape d'affinage, où on projette dans les images les particules 3D calculées, et on optimise une corrélation robuste sur les niveaux de gris dans les images entre les ellipses 2D projetées, quand elles sont visibles (il y a un test d'occultation). On minimise enfin une fonction d'énergie sommant le terme de corrélation, et un terme de résistance. Comme on peut déterminer, sur le critère de courbure, quelles sont les particules appartenant à la même surface, on obtient en fin de compte un ensemble de particules orientées étiquetées (mais on n'obtient toujours pas une ou des surface(s) connexe(s)).

3.3 Comment évaluer ?

Nous avons vu que les méthodes d'appariement proposées dans la littérature étaient très rarement évaluées quantitativement. L'évaluation se limite très souvent à une inspection visuelle des cartes de disparité, sur lesquelles il est impossible d'obtenir des mesures chiffrées.

Dans le cas des scènes planes cependant, il est possible de chiffrer les erreurs d'appariement, par calcul d'une homographie. Dans le cas général (scènes quelconques), une forme d'évaluation peut être menée par calcul de la géométrie épipolaire.

3.3.1 Scènes planes

Si la scène est plane, alors ses projections dans les images de référence sont liées par une homographie plane $H_{1,2\ (3\times3)}$. On peut donc évaluer une liste de N appariements binoculaires $\mathcal{L} = ((x_1, y_1)_i, (x_2, y_2)_i), i = 1..N$, en procédant comme suit.

1. Effectuer un calcul aussi précis et robuste que possible de $H_{1,2}$. Ce calcul est généralement mené par tirages aléatoires de points de $\mathcal{L}$, après normalisation des coordonnées de ces points.

2. Calculer les erreurs en distance d des points à leurs correspondants idéaux selon $H_{1,2}$:

$$d_i = \frac{\delta(H_{1,2}(x_1, y_1, 1)_i^T, (x_2, y_2, 1)_i^T) + \delta(H_{2,1}(x_2, y_2, 1)_i^T, (x_1, y_1, 1)_i^T)}{2} \tag{3.35}$$

$\delta(p_1, p_2)$ est la distance euclidienne dans le plan entre les points p_1 et p_2 ; on a bien sûr $H_{2,1} = {H_{1,2}}^{-1}$.

On étudie ensuite la distribution des d_i, qui donne l'erreur de reprojection pour chaque appariement i de $\mathcal{L}$. Cette méthode est biaisée, car elle se sert des appariements trouvés pour établir un critère permettant justement de mesurer leur précision. Néanmoins on peut l'utiliser sans problème s'il est «suffisamment probable» qu'une proportion significative des appariements trouvés sont parfaitement corrects et précis. P. Brand [Bra 95] par exemple calcule $H_{1,2}$ à partir de cibles circulaires correctement et très précisément appariées dans les images, et vérifiées manuellement ; les *autres* appariements sont ensuite testés vis-à-vis de cette homographie.

Une difficulté expérimentale est d'assurer que la scène observée soit parfaitement plane, ce qui est nécessaire si on veut mesurer des appariements de grande précision. On peut obtenir une scène texturée presque parfaitement plane par projection optique d'une texture (p. ex. diapositive) sur une surface polie : une plaque de verre épaisse, ou mieux, un socle de marbre.

3.3.2 Scènes quelconques

Si la scène est quelconque, le seul lien géométrique existant entre les deux images est la géométrie épipolaire, décrite par la matrice fondamentale $F_{1,2\ (3\times3)}$. La procédure est identique.

1. Effectuer un calcul aussi précis et robuste que possible de $F_{1,2}$.
2. Calculer les erreurs en distance d des points à leurs droites épipolaires idéales selon $F_{1,2}$:

$$d_i = \frac{\delta(F_{1,2}(x_1, y_1, 1)_i^T, (x_2, y_2, 1)_i^T) + \delta(F_{2,1}(x_2, y_2, 1)_i^T, (x_1, y_1, 1)_i^T)}{2} \tag{3.36}$$

$\delta(d_1, p_2)$ est cette fois la distance euclidienne dans le plan entre la droite d_1 et le point p_2, et $F_{2,1} = {F_{1,2}}^T$.

L'estimation est encore plus biaisée, si l'on peut dire, puisque le calcul de la matrice fondamentale $F_{1,2}$ est encore plus sensible à la précision initiale des appariements de $\mathcal{L}$.

3.3.3 Autres méthodes

Nous pouvons remarquer que, à part celles s'appuyant sur des cibles précises et parfaitement connues, ces méthodes n'évaluent pas seulement le processus d'appariement, mais évaluent aussi le processus de calcul de $H_{1,2}$ ou $F_{1,2}$. Le résultat est un mélange indiscernable des performances de ces deux algorithmes.

D'autres méthodes consistent à reconstruire tous les couples appariés dans l'espace 3D euclidien, et à vérifier par exemple que certains angles sont bien droits. Mais :

- on évalue cette fois simultanément l'appariement, l'étalonnage, et la qualité de la reconstruction 3D ;
- il n'est pas certain que les angles soient réellement droits sur les objets observés ;

- les critères d'évaluation sont considérablement réduits (la qualité d'appariement de centaines de points est réduite à la mesure d'un seul angle).

Nous sommes donc réservés sur l'utilisation de telles méthodes d'évaluation. Nous montrerons d'ailleurs qu'elles sont facilement mises en défaut par l'expérience.

3.3.4 Images synthétiques

Une solution au problème de l'évaluation des appariements est d'utiliser des images synthétiques. Ce procédé est d'ailleurs déjà utilisé, avec des images de stéréogrammes aléatoires représentant 3 plans empilés, vus de face, à des disparités connues. Bien que ces disparités soient connues, les auteurs en font rarement un usage numérique, et se contentent d'observer la carte de disparité en des endroits stratégiques (les ruptures de disparité par exemple).

Nous proposons de généraliser cette approche, et d'utiliser des images synthétiques pour la validation des algorithmes d'appariement. Nous pourrons de cette façon modéliser des surfaces non planes, objet de distorsions perspectives, ce qui n'est pas le cas avec les stéréogrammes aléatoires classiques, et tester la robustesse des différentes mesures de ressemblance à de telles déformations.

On fait souvent l'objection que les images synthétiques ne représentent pas une scène réaliste, et qu'il est improbable qu'on observe un tel signal dans les conditions réelles de fonctionnement de l'algorithme. Nous répondons à cela que les méthodes de synthèse d'images tendent justement à produire des images les plus réalistes possibles ; ainsi, les méthodes de synthèse par ray-tracing, ou par calcul de radiosité, s'attachent à calculer précisément les phénomènes physiques intervenant dans la formation des images : trajet des rayons lumineux, échanges d'énergie, interaction avec les matériaux selon leurs propriétés physiques. Nous ne modéliserons pas les effets de bruit de caméra, de distorsion optique, ou de profondeur de champ, bien que tout ceci puisse être simulé par des ray-tracers conventionnels.

La scène étant entièrement définie par son modèle synthétique, nous pourrons calculer des appariements théoriques avec une précision aussi grande que souhaité. De cette façon, nous évaluerons précisément la qualité des algorithmes d'appariement, et non la qualité d'une chaîne de traitements complète.

La scène que nous modélisons est celle-ci : deux splines superposées, à l'intérieur d'une boîte fermée par 6 plans infinis. Fermer la scène évite de devoir traiter le cas particulier des rayons lumineux qui sortent de la scène, et qui n'ont aucune intersection avec des objets définis. Les splines sont texturées par une image aléatoire montrée en figure 3.15, ce qui est certes un cas favorable, mais nous permet de sérier les problèmes. Aussi, nous calculerons deux autres séquences de la même scène, avec cette fois des textures réelles. Nous effectuerons aussi des tests sur des images entièrement réelles (géométrie et texture), prises à l'aide d'appareils photographiques ou de caméras vidéo. Les splines sont superposées pour créer des occultations franches, comme il s'en produit dans les scènes réelles. Les surfaces synthétiques sont parfaitement lisses, car les ray-tracers ne peuvent simuler des textures 3D que par une perturbation artificielle des normales des surfaces, et non par modification de la surface des objets.

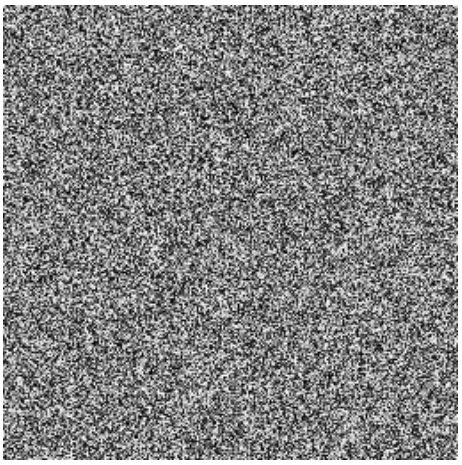


FIG. 3.15: *Texture aléatoire appliquée aux surfaces synthétiques.*

Nous avons synthétisé 6 images de cette sorte, notées `im0` à `im5`, de taille 256×256. Elles sont espacées de 5° par rotation autour d'un axe vertical situé au centre de la scène, selon la figure 3.16. Le but est de tester la robustesse du processus de synthèse de nouvelles vues, à partir d'images de plus en plus éloignées, donc de plus en plus dissemblables (ce qui conduira à des problèmes d'appariement).

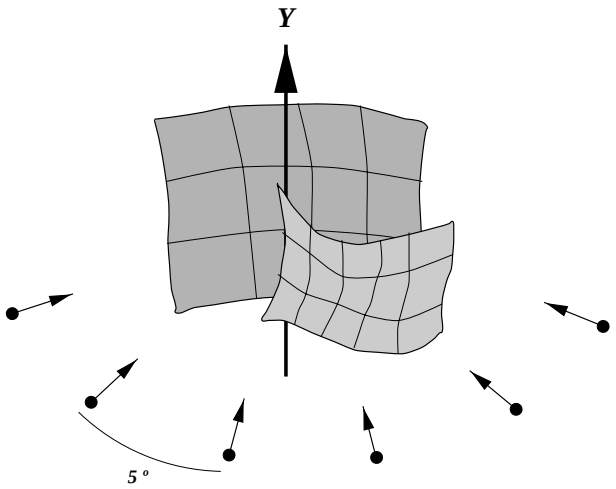


FIG. 3.16: *Disposition des six images synthétiques.*

Les images générées sont montrées en figure 3.17.

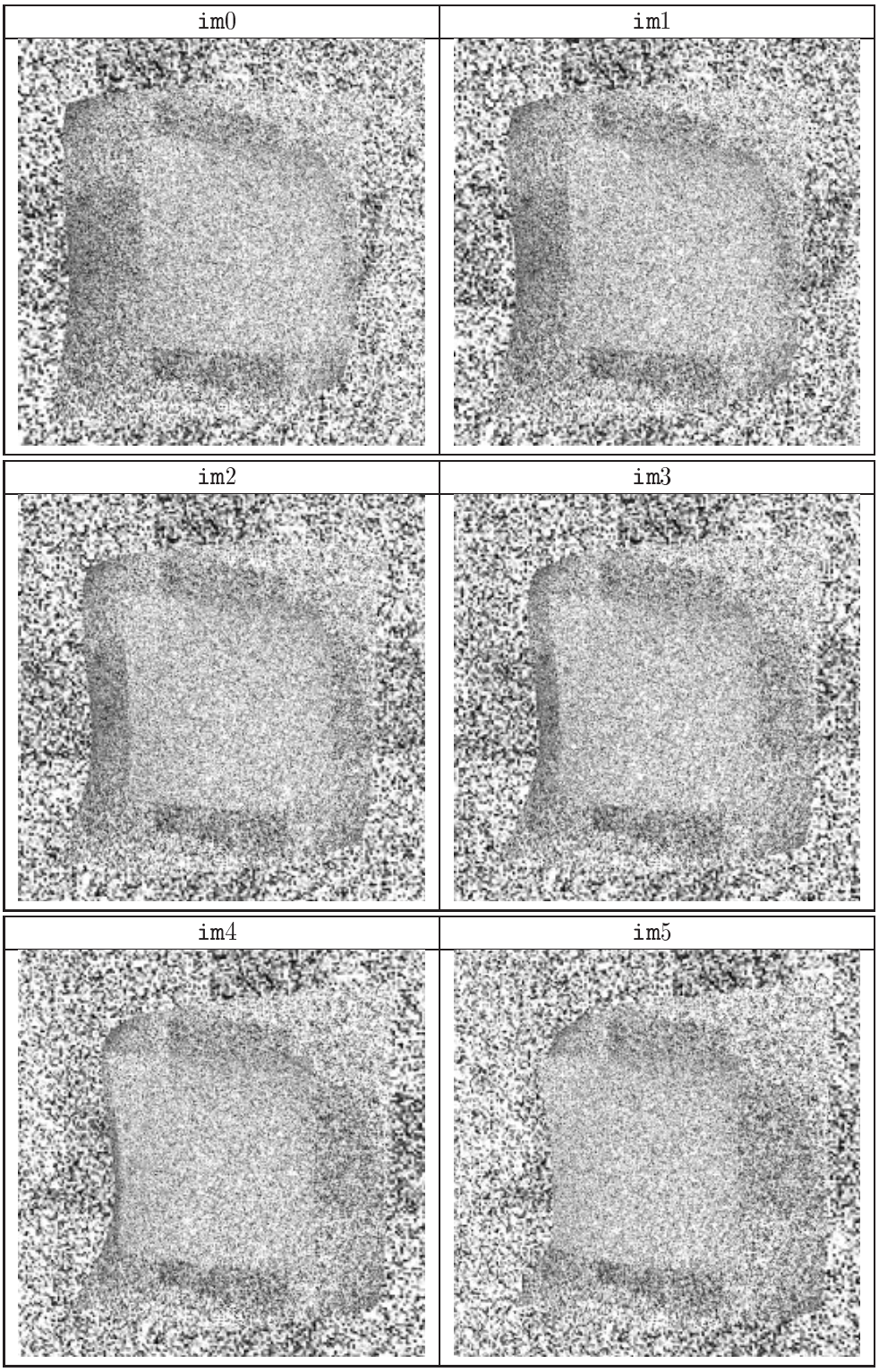


FIG. 3.17: *Les 6 images synthétiques, texture aléatoire.*

Pour les créer, nous avons utilisé le ray-tracer freeware POV–Ray, avec quelques modifications. Les modifications apportées nous permettent de récupérer des résultats intermédiaires du ray-tracing, afin de pouvoir calculer les appariements théoriques idéaux ; il a aussi été nécessaire de modifier le calcul du rayon initial, de façon à ce que la couleur du pixel (i, j) soit bien la couleur du *centre* du pixel (i, j), et non de son coin supérieur gauche, comme c'est le cas par défaut. Nous supposerons en effet dans toute la suite que la valeur du pixel à la ligne i et colonne j de l'image est l'intensité du point mathématique $(i + 0.5, j + 0.5)$ dans le repère de l'image débutant dans le coin supérieur gauche (voir figure 3.18).

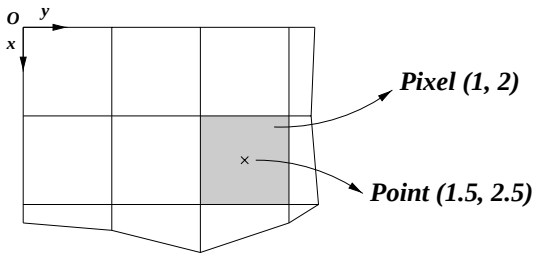


FIG. 3.18: *Le système de coordonnées de* POV–*Ray doit être décalé de* $(+0.5, +0.5)$ *pixels.*

Les images synthétiques nous permettront aussi d'évaluer toute la chaîne de traitement, puisque nous pourrons comparer les images synthétisées par notre procédé, aux images théoriques synthétisées par le ray-tracer. Lors du calcul, elles sont anti-crénelées (*anti-aliased*) pour fournir des images plus proches de la réalité.

Comme des images aussi parfaites pourraient conduire à des comportements singuliers non représentatifs de la réalité (mesures de corrélation strictement nulles, par exemple), nous avons bruité cette séquence à différentes intensités : nous perturbons aléatoirement le niveau de gris m de chaque pixel par une loi gaussienne centrée en m et d'écart-type σ, pour des valeurs de σ égales à 1, 2, 5, 10. La figure 3.19 montre l'image `im0` bruitée.

Enfin, nous présentons en figures 3.21 et 3.22 les deux séquences (géométriquement équivalentes) où la texture mathématique a été remplacée par une texture réelle. Ces textures réelles sont issues de photos ou de vidéos (figure 3.20), et reproduisent donc les approximations de mesure des capteurs utilisés. De plus, elles représentent de façon réaliste la distribution des intensités qu'on doit s'attendre à trouver dans des scènes d'extérieur (texture répétitive : herbe, et grands aplats : ciel) ou d'intérieur (objets polyédriques peu texturés).

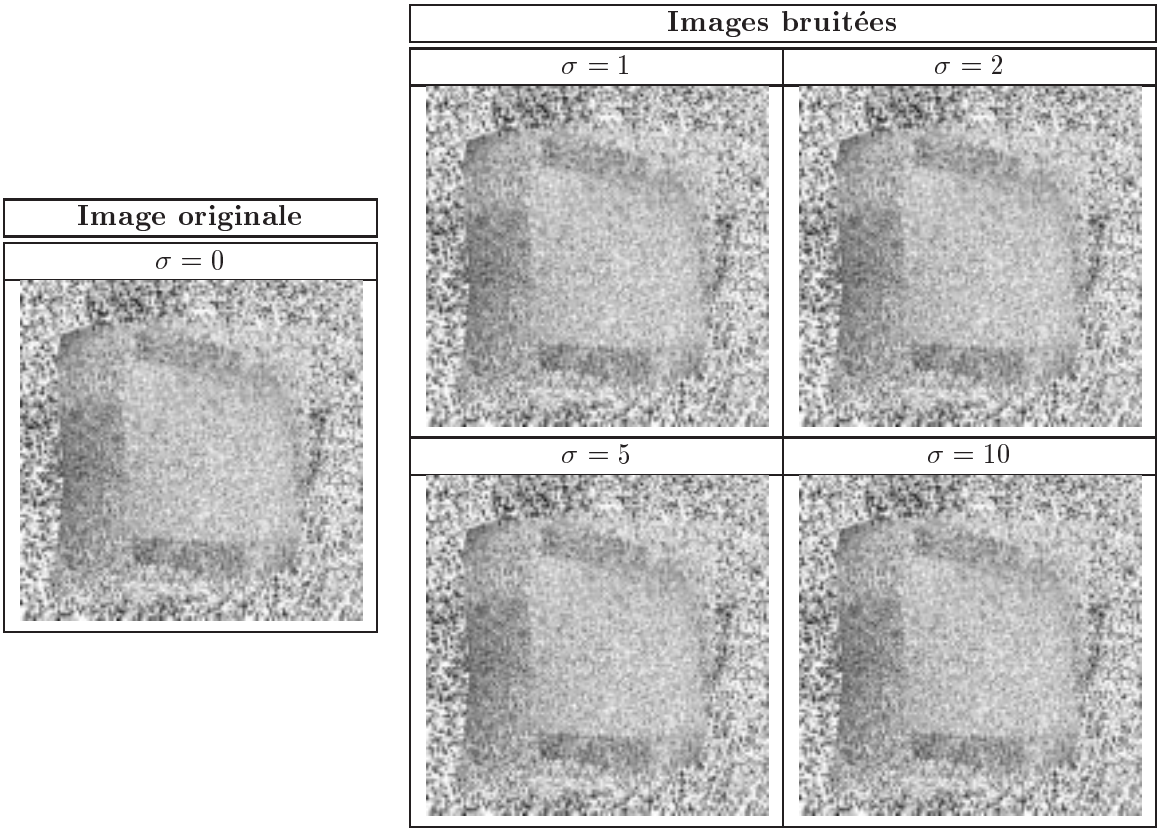


FIG. 3.19: *Image* `im0` *bruitée à différentes intensités.*

Texture d'extérieur Texture d'intérieur

FIG. 3.20: *Les deux textures réelles appliquées aux surfaces synthétiques.*

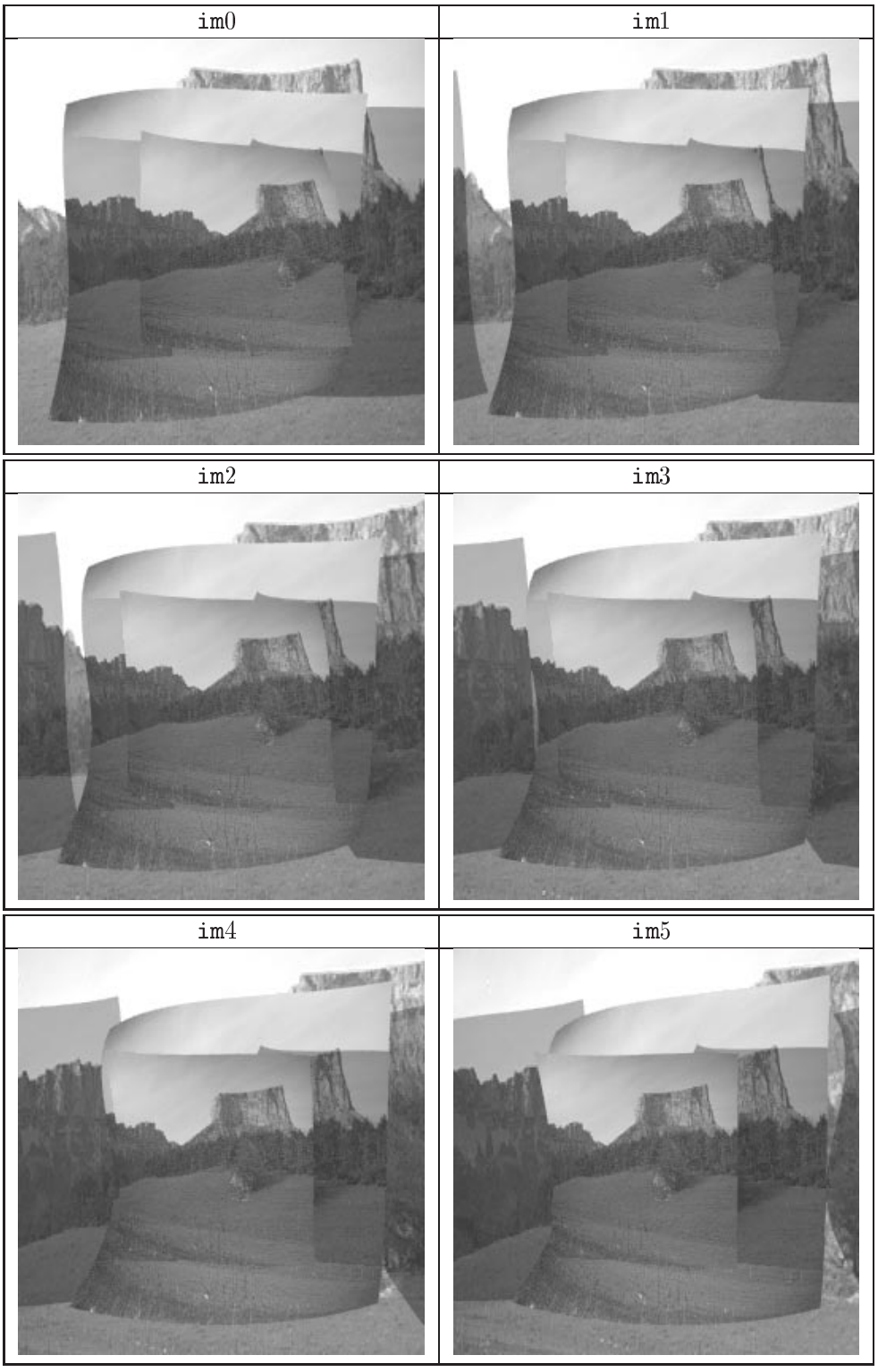


FIG. 3.21: *Les 6 images synthétiques, texture d'extérieur.*

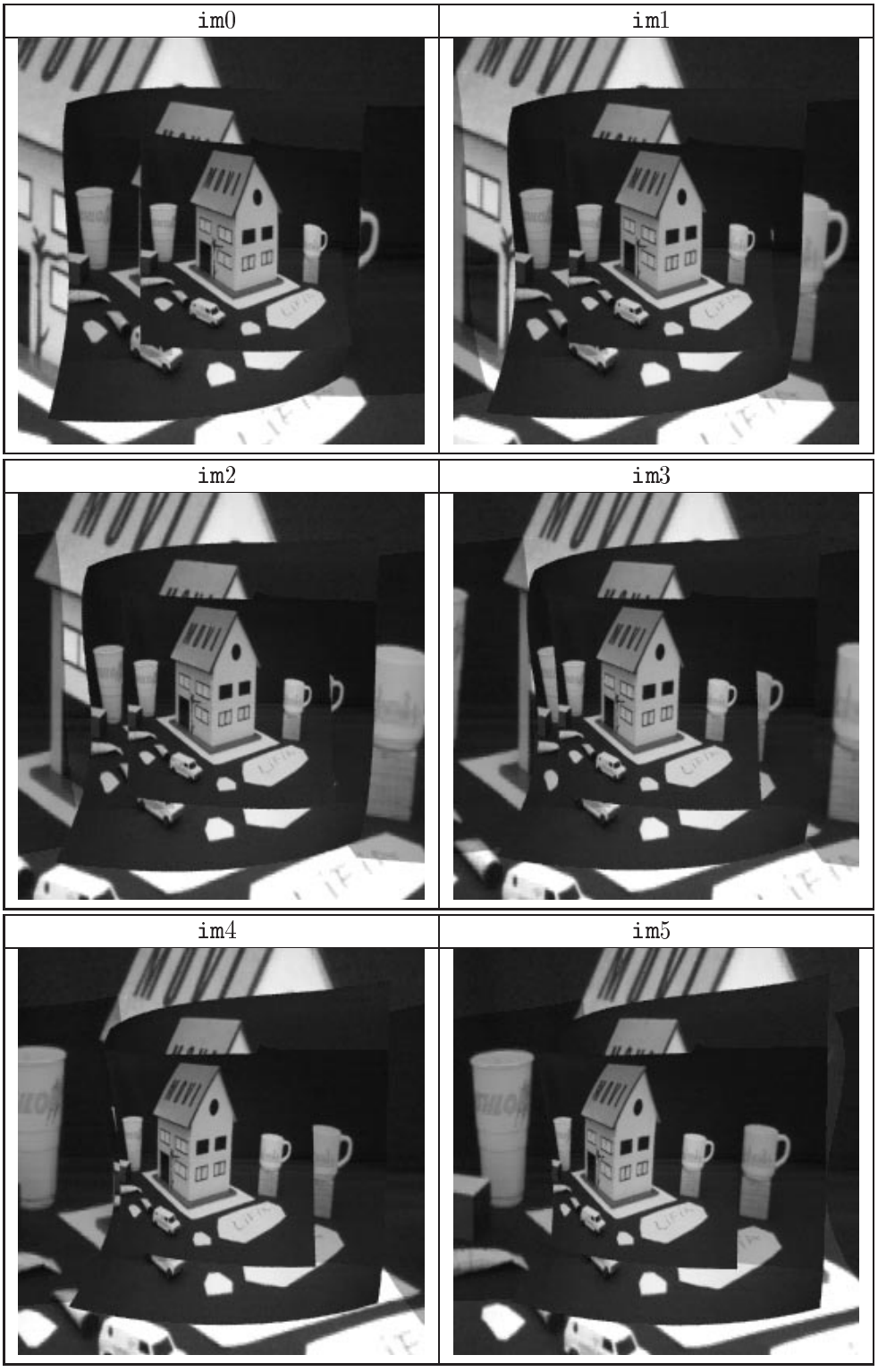


FIG. 3.22: *Les 6 images synthétiques, texture d'intérieur.*

3.4 Nos méthodes d'appariement

Dans la lignée des remarques précédentes concernant la difficulté de détecter des structures évoluées (comme des segments) de façon cohérente dans plusieurs images, nous avons décidé de n'utiliser que des pixels comme structures de base. Nous exposons donc ici nos méthodes d'appariement de pixels entre 2 ou N images.

3.4.1 Appariement dense / appariement épars

Le choix de la méthode de synthèse se répercute sur les techniques d'appariement. Nous avons vu en effet que deux grandes classes de synthèse étaient envisageables : à partir d'un nuage de points appariés puis reconstruits, ou à partir de structures plus évoluées, en général des triangles texturés.

- Synthétiser les nouvelles scènes comme des projections d'un nuage de points 3D implique d'avoir réalisé un appariement *dense*, pour pouvoir reconstruire un grand nombre de points. Notons qu'un nuage de points n'a aucune structure géométrique, et en particulier, il n'y a pas de notion de connexité. Aussi, des pixels connexes dans les images de référence, une fois appariés et reconstruits, pourront être disjoints dans l'image synthétisée : il suffit qu'ils se projettent à plus de 1 pixel de distance. L'image synthétique pourra donc comporter des trous (zones non renseignées), plus ou moins importants selon la distance du nouveau point de vue aux images de référence. On pourrait aussi tenter de reconstruire une surface mathématique continue passant par tous les points 3D reconstruits. Cette tâche est très complexe ; elle a été explorée au SRI, et nous en avons parlé au chapitre précédent.

  Un grand avantage de cette modélisation est que les points mal placés seront noyés dans l'image (s'ils restent en proportion raisonnable). Un simple filtre pourrait les éliminer, et aussi boucher les trous. La méthode est donc relativement robuste aux erreurs d'appariement.

- Synthétiser les nouvelles scènes à partir de modèles en triangles 3D texturés impose seulement de connaître les positions 3D des sommets des triangles. Un appariement *épars* de points judicieusement choisis peut donc être suffisant. Ces points pourraient par exemple être les sommets d'une scène polyédrique. Un appariement dense peut également convenir, sous réserve de diminuer la complexité du maillage généré.

  Plus complexe à construire, le modèle est aussi intrinsèquement plus compact qu'un modèle en nuage de points [1]. Il peut être visualisé par de très nombreux *viewers*, mettant à profit des accélérations matérielles spécifiques pour l'affichage de triangles texturés [2] ; pour notre part, nous construirons un modèle VRML, qui peut être visualisé avec VRWeb par exemple.

1. Cela est faux si l'on utilise des modèles multiples de maillage, ou qu'on applique des textures dynamiques aux triangles. Voir le chapitre 4.
2. *Idem.*

Enfin, le modèle ne présentera pas de trous, même sous des angles extrêmes de visualisation, puisque le maillage triangulaire restera connexe. En revanche, un seul sommet mal placé (donc un seul mauvais appariement) peut conduire à des déformations du maillage telles que le modèle sera visuellement inacceptable.

Les deux types de représentation sont aussi peu manipulables l'un que l'autre. Le nuage de points permet des incrustations faciles, mais interdit de manipuler des surfaces ou des objets. Le maillage triangulaire autorise la manipulation et la déformation de certaines surfaces (les *patches* triangulaires), mais pas des objets de la scène, puisqu'ils ne sont pas perçus comme des éléments isolés.

Le tableau 3.3 résume les caractéristiques de ces deux approches.

Type de modèle	**Nuage de points**	**Maillage triangulaire**
Appariement	Dense	Dense ou épars
Compacité du modèle	−	+
Simplicité de création	+	−
Simplicité de manipulation	−	−
Simplicité de visualisation	+	+
Rapidité de visualisation	−	+
Robustesse	+	−

TAB. 3.3: *Caractéristiques comparées des deux classes de modélisation de la scène.*

À ce stade, l'absence d'avantage clair en faveur d'une modélisation ou de l'autre nous oblige à les étudier simultanément. Se pose alors la question de savoir si nous devons réaliser un appariement dense ou un appariement épars des vues de référence.

- L'appariement dense permet une régularisation simultanée (intrinsèque), ou *a posteriori*, du champ de disparité calculé.
- Les algorithmes d'appariement épars ne sont pas nécessairement plus rapides, et ne permettent pas aussi simplement de résoudre les ambiguïtés d'appariement (bien qu'il existe quelques algorithmes de relaxation sur les disparités d'un voisinage de chaque pixel). Un appariement épars peut aussi être calculé comme un appariement dense réduit : la phase d'appariement dense renforce la cohérence globale, et l'on choisit ensuite seulement quelques points du champ de disparité calculé.

En conclusion, nous nous attacherons surtout aux algorithmes d'appariement dense. Pour ces algorithmes, nous verrons qu'il est néanmoins souhaitable de disposer d'un appariement épars en première passe.

3.4.2 Schéma des opérations

Obtenir des appariements dans N images, sans autres données, est une tâche trop complexe, car elle revient à chercher un optimum dans un très grand espace (l'espace de tous les appariements N-oculaires possibles).

Nous nous limitons pour l'instant au cas binoculaire. Dans ce cas, l'espace de recherche à balayer pour trouver le correspondant d'un point de l'image 1 est l'ensemble des pixels de l'image 2, ou l'ensemble des pixels d'une zone de l'image 2 (espace 2D). La connaissance de la géométrie épipolaire diminue la dimension de l'espace de recherche à une droite de l'image 2, ou à un segment de l'image 2 (espace 1D).

Or, pour calculer la géométrie épipolaire, il nous faut déjà une première estimation du champ de disparité, donc une première passe d'appariement. Cependant, ces appariements ne doivent pas nécessairement être denses, ni tous précis, ni tous justes. En effet, nous utiliserons des algorithmes relativement robustes au manque de précision, et robustes aux faux appariements. De plus, des passes successives peuvent permettre d'affiner la précision des appariements itérativement.

Pour simplifier la première passe d'appariements, nous ne traiterons dans cette phase que des points facilement reconnaissables d'une image à l'autre, des *points d'intérêt*. De nombreux travaux ont porté sur la détection de tels points (points de contour, points de contraste), et suivant les conclusions de la thèse de C. Schmid [Sch 96b], qui recense les détecteurs existants, nous utiliserons un détecteur de Harris, modifié pour plus de répétabilité. Ce détecteur calcule la courbure du gradient local en chaque pixel de l'image, et garde les points de courbure maximale. Un seuil permet de modifier sa sensibilité.

Une fois les appariements denses calculés, nous obtiendrons une carte de disparité bruitée, que nous pourrons améliorer et rendre plus cohérente par une étape de régularisation. Enfin, les appariements obtenus devront être éventuellement affinés pour atteindre une précision sous-pixellique. Le déroulement des opérations sera donc le suivant :

phase 1 : extraction de points d'intérêt dans les images de référence ;

phase 2 : appariement de ces points (donc épars) ;

phase 3 : calcul robuste de la géométrie épipolaire, et rejet des appariements aberrants. Éventuellement, affinage préalable des appariements de la phase 2 ;

phase 4 : appariement dense contraint par la géométrie épipolaire ;

phase 5 : régularisation des appariements denses ;

phase 6 : affinage des appariements denses.

Outre la contrainte épipolaire, d'autres contraintes géométriques sont possibles, portant sur plus de deux images, et nous en parlerons en section 3.4.8.2.

3.4.3 Choix d'un algorithme

Tentons de synthétiser les points forts et les points faibles des algorithmes d'appariement que nous avons exposés.

- L'algorithme WTA n'est pas symétrique, et surtout, il n'est pas robuste. Il favorise une image, et attribue un correspondant à chaque pixel de cette image, même s'il se trouve dans une zone occultée.

- Les algorithmes énergétiques nécessitent de toute façon une initialisation, et ne peuvent donc pas être utilisés en première passe. Leur convergence est problématique, car elle peut être lente, et pour certains, elle n'est pas garantie.

- L'algorithme DP3 ne fonctionne que si la contrainte d'ordre est respectée dans les images, et demande le réglage d'un seuil assez sensible.

- L'algorithme CCR est symétrique et robuste, mais n'impose aucune cohérence locale. De fait, les zones uniformes des images ne sont pas appariées, car elles ne sont pas suffisamment remarquables pour résister à l'étape de vérification croisée. L'appariement obtenu constitue cependant une bonne initialisation pour une étape de régularisation ultérieure. Enfin, CCR peut être utilisé pour des appariements denses ou épars.

3.4.4 Proposition de nouveaux algorithmes

Nous proposons de nouveaux algorithmes ci-dessous ; ils concernent essentiellement des modifications de la programmation dynamique.

3.4.4.1 Recherche exhaustive

Comme nous l'avons vu, une recherche exhaustive de tous les appariements possibles est impossible en pratique. Ainsi, apparier 12 points dans l'image avec 12 points dans l'image 2 amènerait à explorer un arbre de plus de 50 milliards de feuilles.

Nous avons cependant implémenté un tel algorithme, en utilisant des procédures de coupure permettant de réduire considérablement la taille de l'arbre. Nous appelons cet algorithme ES : Exhaustive Search. En utilisant les procédures de coupure, on parvient à traiter jusqu'à 12 points dans chaque image, ce qui est inutilisable dans notre contexte, mais pourrait être envisagé dans d'autres applications.

3.4.4.2 Vérification croisée avec seuil

Lors du déroulement de l'algorithme CCR, on pourrait imposer que les appariements retenus aient une corrélation inférieure à un certain seuil. Cet algorithme est appelé CCT : Cross Check Threshold. Ceci limiterait le nombre de faux appariements.

Cependant, comme pour les autres algorithmes avec seuil (par exemple DP3), celui-ci est très sensible et délicat à choisir. Il peut être déterminé empiriquement, ou bien après une première passe d'appariement : pour tous les appariements obtenus en première passe, on accumule les scores de corrélation. Si on suppose qu'ils suivent une distribution laplacienne, la valeur médiane de ces scores fournit une estimation robuste de l'écart-type σ de cette distribution, selon l'équation 3.12, p. 34. On peut ensuite fixer le seuil de rejet à, par exemple, 2.5σ.

Nous testerons cette méthode.

3.4.4.3 Modifications de DP3

Comme nous l'avons vu, la programmation dynamique sous sa forme DP3 est très sensible au bon réglage du coût κ. De fait, il est curieux de constater que dans DP3, les appariements multiples sont traités comme des occultations : si, du fait de la distorsion perspective, 2 pixels consécutifs de l'image 1 correspondent à 1 seul pixel de l'image 2, l'algorithme DP3 imposera au moins une occultation sur le chemin optimal. Cette occultation sera une transition de coût κ (assez élevé), alors que ce devrait être une transition correspondant à un appariement, de coût égal au score de corrélation (donc faible). Cela rend impossible un réglage cohérent de κ, qui correspond soit à une occultation, soit à un appariement multiple.

Nous proposons un nouvel algorithme, où les transitions sont toutes des transitions d'appariement : sur chaque transition d'occultation, κ est remplacé par le score de corrélation, et la transition correspond à un appariement multiple. Dans cet algorithme, chaque nœud (i, j) mène vers 3 autres nœuds, *via* les 3 types de transition suivants :

1. (p_i, q_j) forme un appariement. On examinera par la suite la validité du couple (p_{i+1}, q_{j+1}). Cette transition a un coût k^1_{ij}, et mène au nœud $(i+1, j+1)$.

2. (p_i, q_j) forme un appariement. On examinera par la suite la validité du couple (p_i, q_{j+1}). Cette transition a un coût k^4_{ij}, et mène au nœud $(i, j+1)$.

3. (p_i, q_j) forme un appariement. On examinera par la suite la validité du couple (p_{i+1}, q_j). Cette transition a un coût k^5_{ij}, et mène au nœud $(i+1, j)$.

On prend $k^1_{ij} = k^4_{ij} = k^5_{ij} = c_{ij}$. Dans ce cas, la contrainte d'unicité n'est plus respectée, puisqu'un seul pixel peut avoir plusieurs correspondants. De plus, les occultations ne peuvent pas être détectées, et l'algorithme ne peut être utilisé que sur des images sans occultation prononcée. Cet algorithme est nommé DP3NO : «Dynamic Programming 3-Transitions No-Occlusion».

Une autre solution est de combiner les deux approches : ce nouvel algorithme comporte 5 transitions, et nous l'appelons DP5 : «Dynamic Programming 5-Transitions». Ici, chaque nœud (i, j) mène vers 5 autres nœuds, *via* les 5 types de transition suivants :

1. (p_i, q_j) forme un appariement. On examinera par la suite la validité du couple (p_{i+1}, q_{j+1}). Cette transition a un coût k^1_{ij}, et mène au nœud $(i+1, j+1)$.

2. Les points ne peuvent pas être appariés. Le point q_j est occulté dans l'image 1, et on examinera par la suite la validité du couple (p_i, q_{j+1}). Cette transition a un coût k^2_{ij}, et mène au nœud $(i, j+1)$.

3. Les points ne peuvent pas être appariés. Le point p_i est occulté dans l'image 2, et on examinera par la suite la validité du couple (p_{i+1}, q_j). Cette transition a un coût k^3_{ij}, et mène au nœud $(i+1, j)$.

4. (p_i, q_j) forme un appariement. On examinera par la suite la validité du couple (p_i, q_{j+1}). Cette transition a un coût k^4_{ij}, et mène au nœud $(i, j+1)$.

5. (p_i, q_j) forme un appariement. On examinera par la suite la validité du couple (p_{i+1}, q_j). Cette transition a un coût k^5_{ij}, et mène au nœud $(i+1, j)$.

Les trois premières transitions sont celles de DP3. Les transitions de type 4 et 5 autorisent les appariements multiples. Comme dans DP3NO, nous fixons $k^1_{ij} = k^4_{ij} = k^5_{ij} = c_{ij}$, et comme dans DP3, $k^2_{ij} = k^3_{ij} = \kappa$.

Constatons enfin un comportement indésirable de l'algorithme classique DP3 : si parmi les points (p_1, p_2) de l'image 1 et (q_1, q_2, q_3) de l'image 2, les appariements sont (p_1, q_1) et (p_2, q_3), alors l'algorithme passe nécessairement par une transition d'occultation de coût κ, signalant que le point q_2 n'a pas de correspondant. Or, l'algorithme DP3 respectant les contraintes d'ordre et d'unicité, et sachant que les appariements (p_1, q_1) et (p_2, q_3) sont réalisés, alors le point q_2 est *nécessairement* non-apparié, et il n'y a pas lieu d'ajouter un coût κ au coût du chemin global.

Nous proposons donc un nouvel algorithme, où κ est fixé à 0. Le chemin optimal est alors de coût nul, et ne comporte que des occultations. Aussi, nous limitons le nombre de sauts d'occultation autorisés, que nous mémorisons dans les états du graphe de la programmation dynamique. Les nœuds sont désormais notés (i, j, niv), et correspondent à l'état : «les listes $(p_1, ..., p_i)$ et $(q_1, .., q_j)$ ont été appariées, et ceci avec niv sauts d'occultation». Les trois transitions de DP3 sont modifiées en conséquence. À partir du nœud (i, j, niv) :

1. (p_i, q_j) forme un appariement. On examinera par la suite la validité du couple (p_{i+1}, q_{j+1}). Cette transition a un coût c_{ij} (mesure de ressemblance entre p_i et q_j), et mène au nœud $(i+1, j+1, niv)$.

2. (p_i, q_j) ne forme pas un appariement. On examinera par la suite la validité du couple (p_i, q_{j+1}). Cette transition a un coût $\kappa = 0$, et mène au nœud $(i, j+1, niv+1)$.

3. (p_i, q_j) ne forme pas un appariement. On examinera par la suite la validité du couple (p_{i+1}, q_j). Cette transition a un coût $\kappa = 0$, et mène au nœud $(i+1, j, niv+1)$.

Le graphe est maintenant contenu dans un tableau 3D (un cube), et le chemin optimal est le chemin de coût minimal parvenant à l'extrémité (n, m, niv) de l'un des niveaux niv du cube. Le problème est de fixer le nombre maximal de sauts d'occultation autorisés (la profondeur du cube), et nous retrouvons des questions similaires au réglage des coûts d'occultation. Ce nombre maximal sera fixé par une probabilité d'occultation p_{occ} fixée *a priori*, ou après une première estimation des appariements. Cet algorithme est appelé DP3JC : «Dynamic Programming 3-Transitions Jumps Count».

Nous testerons les algorithmes DP3, DP3NO, DP5 et DP3JC sur nos images, et évaluerons leur sensibilité aux différents seuils ou paramètres. Les algorithmes d'appariement sont résumés en tableau 3.4. Ce tableau est repris à la fin de ce chapitre, en page 141, ainsi que les autres abréviations utilisées.

Algorithme	Description	Paramétrage
WTA	Winner Takes All. Meilleur score.	Inutile
ES	Exhaustive Search. Recherche exhaustive de tous les appariements.	Inutile
CCR	Cross Check Raw. Meilleur score + vérification croisée.	Inutile
CCT	Cross Check Threshold. CCR avec score maximal égal à s.	s
DP3	Dynamic Programming 3-Transitions. Programmation dynamique classique, coût d'occultation κ.	κ
DP3NO	Dynamic Programming 3-Transitions No Occlusion. Programmation dynamique sans occultation.	Inutile
DP5	Dynamic Programming 5-Transitions. Programmation dynamique sans contrainte d'unicité, coût d'occultation κ.	κ
DP3JC	Dynamic Programming 3-Transitions Jumps Count. Programmation dynamique avec comptage d'occultations, coût d'occultation nul.	p_{occ}

TAB. 3.4: *Récapitulatif des algorithmes d'appariement.*

3.4.5 Choix d'une mesure

Parmi les mesures classiques, SAD et SSD seront les plus rapides, mais seront sensibles à des changements d'illumination. Leurs versions centrées : ZSAD et ZSSD corrigent ce problème, en soustrayant aux pixels de chaque masque la moyenne de ses niveaux de gris. Comme nous l'avons vu, cette correction est trop brutale, et peut mener à de faux appariements.

Les invariants seront sans doute trop lents pour un appariement dense, et il n'en existe pas de version robuste aux occultations partielles. Cependant, contrairement aux mesures de corrélation, ils peuvent apparier des points ayant subi une forte rotation, et même un changement de taille (avec une version multi-échelles des invariants). Aussi, un appariement par invariants pourrait être effectué en première passe, sur des points épars. Cela permettrait d'estimer la rotation globale des images et le facteur d'échelle. Sur la base de cette estimation, on pourrait redresser les images, pour les amener à une rotation relative nulle et à un facteur d'échelle égal à 1. Les corrélations classiques seraient ensuite pleinement utilisables. Les images que nous traitons ne comportent pas de rotation, et nous n'appliquons pas cette technique.

3.4.6 Proposition de nouvelles mesures

Nous sommes plutôt favorables à l'emploi de mesures très simples comme SAD, et sur la base de cette mesure, nous en construisons une version pondérée WSAD, une version robuste RSAD, et une mesure partiellement robuste PRSAD. Rappelons que SAD est définie par l'équation 3.37, où $p_1 = (u_1, v_1)$ est le point dans l'image 1 de signal I_1, et $p_2 = (u_2, v_2)$

le point dans l'image 2, de signal I_2.

$$\textsc{Sad}(p_1,p_2)=\sum_{du=-n}^{du=+n}\sum_{dv=-n}^{dv=+n}|I_2(u_2+du,v_2+dv)-I_1(u_1+du,v_1+dv)| \tag{3.37}$$

La mesure WSAD («Weighted SAD ») est la mesure SAD, où chaque pixel est pondéré par l'inverse du carré de la distance qui le sépare du centre de la fenêtre. Le pixel central garde un poids 1. Sur un masque 5×5, les coefficients de pondération sont ceux de la figure 3.23.

$$\begin{array}{|ccccc|}\hline 1/8&1/5&1/4&1/5&1/8\\ 1/5&1/2&1/1&1/2&1/5\\ 1/4&1/1&1/1&1/1&1/4\\ 1/5&1/2&1/1&1/2&1/5\\ 1/8&1/5&1/4&1/5&1/8\\ \hline\end{array} \quad=\quad \frac{1}{40}\begin{array}{|ccccc|}\hline 5&8&10&8&5\\ 8&20&40&20&8\\ 10&40&40&40&10\\ 8&20&40&20&8\\ 5&8&10&8&5\\ \hline\end{array}$$

FIG. 3.23: *Pondération des différents pixels pour* WSAD, *sur un masque* 5×5.

Plus formellement, la mesure WSAD est définie par l'équation 3.38.

$$\begin{cases}\omega(0,0)=1\\ (du,dv)\neq(0,0)\Rightarrow\omega(du,dv)=\frac{1}{du^2+dv^2}\\ \textsc{Wsad}(p_1,p_2)=\sum_{du=-n}^{du=+n}\sum_{dv=-n}^{dv=+n}\omega(du,dv)|I_2(u_2+du,v_2+dv)-I_1(u_1+du,v_1+dv)|\end{cases} \tag{3.38}$$

Nous espérons donner à cette mesure une certaine robustesse, car elle est plus localisée que SAD : elle accorde plus d'importance à son 4-voisinage qu'aux autres pixels du masque.

La mesure RSAD («Robust SAD ») est dérivée directement de la mesure SAD de façon à la rendre robuste, comme RSSD est dérivé de SSD. Nous supposons maintenant que la loi est laplacienne (et non plus gaussienne), ce qui est cohérent avec le fait que nous mesurons des différences absolues d'intensité, et non plus des différences au carré. L'écart-type de la loi est estimé de façon robuste à partir de sa médiane, et comme déjà noté, nous rejetons tous les pixels qui présentent une erreur de plus de 4.39 fois l'écart-type. L'équation 3.39 décrit entièrement la mesure RSAD.

$$\begin{cases}\sigma=1.442695041\,\mathrm{med}\,(|I_2(u_2+du,v_2+dv)-I_1(u_1+du,v_1+dv)|)_{(du,dv)\in[-n;+n]^2}\\ |I_2(u_2+du,v_2+dv)-I_1(u_1+du,v_1+dv)|>4.39\sigma\Rightarrow\omega(I_1,I_2,du,dv)=0\\ |I_2(u_2+du,v_2+dv)-I_1(u_1+du,v_1+dv)|\leq 4.39\sigma\Rightarrow\omega(I_1,I_2,du,dv)=1\\ \textsc{Rsad}(p_1,p_2)=\sum_{du=-n}^{du=+n}\sum_{dv=-n}^{dv=+n}\omega(I_1,I_2,du,dv)|I_2(u_2+du,v_2+dv)-I_1(u_1+du,v_1+dv)|\end{cases} \tag{3.39}$$

La mesure PRSAD enfin, est un prototype de mesure «partiellement robuste» : Partially Robust SAD. Comme nous l'avons déjà signalé, les mesures robustes sont trop sélectives :

elles rejettent dans le calcul de corrélation tous les pixels qui ne conviennent pas. Les pixels sont rejetés sur la base d'un critère calculé à partir de leurs valeurs mêmes, et cette estimation est trop locale, et biaisée : si les masques sont très ressemblants, l'écart-type estimé σ est petit, et on rejette un grand nombre de pixels ; mais si les masques sont très différents, σ est grand, et on peut conserver des pixels qui n'appartiennent pas au modèle. Ceci peut être détecté, car les occultations ont une forme cohérente. Aussi, nous allons contraindre les pixels rejetés à suivre une disposition compacte, correspondant plus probablement à une occultation réelle (voir discussion en 3.2.2.2, p. 35).

Pour cela, notre mesure PRSAD est construite comme un vecteur à 2 composantes : la première est la mesure de corrélation RSAD, et la seconde est une mesure de non-compacité (dispersion) du masque de pondération ω (équation 3.40). La mesure de dispersion est décrite par les équations 3.41 et 3.42. Une bonne ressemblance correspond à une mesure de corrélation et à une dispersion les plus faibles possibles, donc ce vecteur doit être aussi proche de $(0, 0)$ que possible.

$$\text{PRSAD}(p_1, p_2) = v_{1,2} = \begin{pmatrix} \text{RSAD}(p_1, p_2) \\ \text{disp}(\omega(I_1, I_2)) \end{pmatrix} \tag{3.40}$$

$$\text{disp}(\omega) = \sum_{(u,v)\in[-n;n]^2} \text{dispelem}(\omega(u, v)) \tag{3.41}$$

$$\text{dispelem}(\omega(u, v)) = \sum_{(du,dv)\in[-1;+1]^2} |\omega(u + du, v + dv) - \omega(u, v)| \tag{3.42}$$

La mesure de dispersion est en quelque sorte une somme des modules des gradients sur le masque ω. La figure 3.24 montre un exemple de masque dispersé, et la figure 3.25 un masque compact.

$$\omega = \begin{array}{|c|c|c|c|c|}\hline 0&1&0&1&0\\\hline 1&0&1&0&1\\\hline 0&1&0&1&0\\\hline 1&0&1&0&1\\\hline 0&1&0&1&0\\\hline\end{array} \Longrightarrow \text{dispelem} = \begin{array}{|c|c|c|c|c|}\hline 2&3&3&3&2\\\hline 3&4&4&4&3\\\hline 3&4&4&4&3\\\hline 3&4&4&4&3\\\hline 2&3&3&3&2\\\hline\end{array} \Longrightarrow \text{disp} = 80$$

FIG. 3.24: *Exemple de mesure de dispersion — 1 : un masque dispersé.*

$$\omega = \begin{array}{|c|c|c|c|c|}\hline 0&0&1&1&1\\\hline 0&0&1&1&1\\\hline 0&0&1&1&1\\\hline 0&0&1&1&1\\\hline 0&0&1&1&1\\\hline\end{array} \Longrightarrow \text{dispelem} = \begin{array}{|c|c|c|c|c|}\hline 0&2&2&0&0\\\hline 0&3&3&0&0\\\hline 0&3&3&0&0\\\hline 0&3&3&0&0\\\hline 0&2&2&0&0\\\hline\end{array} \Longrightarrow \text{disp} = 26$$

FIG. 3.25: *Exemple de mesure de dispersion — 2 : un masque compact.*

Un bon appariement se caractérise par une mesure PRSAD proche de $(0, 0)$. Nous avons donc besoin d'une mesure de distance entre vecteurs, et nous utiliserons pour cela une distance de Mahalanobis (équation 3.43) ; cette méthode est usuelle pour mesurer la distance entre des vecteurs dont les composantes ne sont pas de même nature. Pour cela, nous devons déterminer la matrice de covariance $\Lambda_{2\times 2}$.

$$d(p_1, p_2) = \sqrt{(v_2 - v_1)^T \Lambda^{-1} (v_2 - v_1)} \tag{3.43}$$

Dans notre cas, Λ sera déterminé à partir des masques en cours d'examen : pour apparier les points $(p_1, ..., p_n)$ de l'image 1 avec les points $(q_1, ..., q_m)$ de l'image 2, nous calculons toutes les valeurs de $v_{i,j} = \text{PRSAD}(p_i, q_j), (i, j) \in [1..n] \times [1..m]$; puis nous déterminons la matrice de covariance Λ des vecteurs $v_{i,j}, (i, j) \in [1..n] \times [1..m]$. Dans une seconde étape, nous pouvons calculer chaque distance $d(v_{i,j}, 0)$, et ceci constitue le score de corrélation entre le point p_i et le point q_j. Cette façon d'estimer la covariance des vecteurs directement sur les vecteurs à mesurer est un peu biaisée, et peut poser des problèmes si ces vecteurs ne sont pas suffisamment représentatifs, comme nous le verrons lors des expériences.

Notons que la procédure est identique pour INVAR : pour cette dernière, $\Lambda_{9\times 9}$ est déterminée à partir d'invariants mesurés sur les points $(p_1, ..., p_n)$ et $(q_1, ..., q_m)$. La mesure de ressemblance entre les points p_i et q_j est cette fois la distance de Mahalanobis $d(v_i, v_j)$ entre les vecteurs d'invariants v_i de p_i et v_j de q_j.

Le tableau 3.5 résume les mesures de ressemblance entre pixels que nous utiliserons, et leurs notations. Ce tableau est repris à la fin de ce chapitre, en page 141, ainsi que les autres abréviations utilisées.

Mesure	Description
SAD	Somme des différences absolues des niveaux de gris.
ZSAD	Version centrée de SAD.
SSD	Somme des différences au carré des niveaux de gris.
ZSSD	Version centrée de SSD.
RSSD	Version robuste de SSD.
RZSSD	Version robuste de ZSSD.
WSAD	SAD, avec pondération décroissante en fonction de la distance au centre.
RSAD	Version robuste de SAD.
PRSAD	Version partiellement robuste de SAD.
INVAR	Invariants (comme C. Schmid dans [Sch 96b]).

TAB. 3.5: *Récapitulatif des mesures de ressemblance.*

3.4.7 Aspects algorithmiques

Toutes les mesures de corrélation sont en $\mathcal{O}(n^2)$, où n est la taille du masque de calcul, c.-à-d. la longueur en pixels d'un côté de la fenêtre (carrée). Le facteur multiplicatif varie dans une amplitude de 1 à 5 selon les mesures, et le tableau 3.6 donne les temps de calcul, mesurés sur une UltraSPARC 1 à 200 MHz.

Mesure	3×3	5×5	7×7	9×9	11×11	13×13	15×15	17×17	19×19	Progression
SAD	794913	526593	353232	246792	181028	135355	105843	84531	69589	
ZSAD	372995	231642	149903	102543	74405	54025	43516	33967	28313	
SSD	566251	367647	243784	170999	124254	93197	73314	58720	47733	
ZSSD	365364	230840	149903	103359	75245	55991	43917	35088	28066	
RSSD	220507	102533	56689	36075	24444	17637	13408	10455	8354	
RZSSD	147798	69881	38417	24498	16689	12101	9132	7062	5770	
WSAD	783699	524109	352858	246914	180473	136500	106406	85690	69300	
RSAD	233863	109004	60423	38536	26233	18997	14329	11251	9033	

TAB. 3.6: *Vitesse comparée des mesures de corrélation évaluées (en nombre de mesures par seconde), et progression selon la taille du masque.*

La complexité est donc bien quadratique. Néanmoins, une mesure comme SAD se code en quelques instructions d'assembleur, et peut se calculer en un temps moins que quadratique. Nous observons d'ailleurs que la dégradation de la vitesse de calcul de SAD en fonction de la taille du masque est moins importante que pour les autres mesures.

D'autre part, il est possible de calculer très rapidement les corrélations de points voisins dans l'image, par report des calculs partiels. Cette technique développée dans l'équipe d'O. Faugeras à l'INRIA est décrite dans [Fau 93b]. Elle suppose une certaine régularité des images (étape préliminaire de rectification). Nous n'avons pas appliqué ces techniques.

Enfin, les mesures PRSAD et INVAR n'apparaissent pas dans ce tableau, car elles nécessitent plusieurs passes : une passe pour estimer la matrice de covariance Λ (respectivement 2×2 ou 9×9), et une passe pour effectivement mesurer les distances. Leur temps de calcul apparaîtra dans les temps globaux d'appariement, lors des expérimentations.

3.4.8 Calcul de la géométrie épipolaire

Le calcul de la géométrie épipolaire à partir seulement d'appariements dans les images a longtemps été problématique : les résultats étaient extrêmement sensibles à la précision des appariements, et encore fallait-il s'assurer que tous les appariements fussent rigoureusement corrects. De nombreuses études ont été menées sur cette question, notamment par Q.T. Luong dans sa thèse [Luo 92], par Z. Zhang pour les aspects de robustesse [Zha 94], et par R. Hartley pour les questions de précision et de conditionnement numérique [Har 95].

L'algorithme que nous utilisons pour le calcul de la matrice fondamentale combine ces avancées. Nous décrivons ci-dessous l'algorithme linéaire proposé par R. Hartley, pour le calcul de $F_{1,2}$ entre les images 1 et 2, connaissant au moins 8 appariements (p_i, q_i) entre les images («calcul linéaire avec obligation de rang 2»).

1. Résoudre les équations $q_i^T F p_i = 0$.
2. Forcer F à être de rang 2, par une décomposition SVD : $F = UDV^T$. D est une matrice diagonale contenant les valeurs propres λ_1, λ_2, λ_3, dont on annule la plus faible, puis on recompose $F_{1,2}$: $F_{1,2} = UD'V^T$.

Cet algorithme garantit que la matrice fondamentale calculée est bien de rang 2. Néanmoins, si les coordonnées des points sont des coordonnées en pixels dans les images, elles varient dans une gamme trop importante (typiquement de 0 à 500), et la résolution numérique est mal conditionnée : les coefficients du système sont dans une amplitude de 0 à 500^2. Aussi, R. Hartley propose de normaliser les coordonnées, et il a montré que dans ce cas, les performances de l'algorithme linéaire sont équivalentes aux meilleurs algorithmes non-linéaires. Le nouvel algorithme est le suivant.

1. Normaliser les appariements (p_i, q_i) de façon à les placer dans un cercle de centre 0 et de rayon 1.
2. Procéder au calcul linéaire de $F_{1,2}$ décrit précédemment.
3. Dé-normaliser $F_{1,2}$.

On autorise enfin que dans les appariements donnés, une proportion τ soit fausse (*outliers*). Nous appliquons alors une technique de moindres carrés médians, classique en statistiques robustes.

1. Choisir 8 appariements au hasard.
2. Sur ces 8 appariements, procéder à un calcul de $F_{1,2}$ avec normalisation.
3. Calculer l'erreur médiane à τ.
4. Répéter N fois les étapes 1 à 3, en conservant la matrice correspondant à l'erreur médiane minimale. De cette erreur médiane, on peut estimer de façon robuste l'écart-type σ de la distribution des erreurs, en supposant que celles-ci suivent une loi gaussienne (même principe que pour la mesure RZSSD).
5. Pour la meilleure matrice, établir la liste de tous les appariements respectant cette contrainte épipolaire à 2.5σ près.
6. Effectuer un calcul final avec tous ces appariements.

Il existe une relation entre le nombre de tirages N à effectuer, le taux τ d'*outliers* attendu, et la probabilité souhaitée de trouver la bonne matrice. Si on souhaite atteindre une probabilité de 99.9 % de trouver la matrice fondamentale correcte, alors :

$$P(\text{au moins un tirage parmi les } N \text{ est correct}) = 0.999 \tag{3.44}$$
$$1 - P(\text{les } N \text{ tirages donnent tous des matrices incorrectes}) = 0.999 \tag{3.45}$$
$$1 - P(\text{un tirage donne une matrice incorrecte})^N = 0.999 \tag{3.46}$$
$$1 - P(\text{un tirage contient au moins un faux appariement parmi les 8})^N = 0.999 \tag{3.47}$$
$$1 - (1 - P(\text{les 8 appariements du tirage sont corrects}))^N = 0.999 \tag{3.48}$$
$$1 - (1 - (1 - \tau)^8)^N = 0.999 \tag{3.49}$$

Pour $\tau = 0.5$ et dans les conditions précédentes, on trouve $N = 1765$ tirages à effectuer.

Notons que la méthode de moindres carrés médians n'est plus garantie de fonctionner pour $\tau > 0.5$. En effet, si on suppose que plus de la moitié des appariements sont faux, il est possible qu'une proportion de ces appariements soient cohérents entre eux par hasard. L'algorithme trouvera une matrice cohérente avec une certaine proportion des appariements, mais sans aucune garantie que ce soit la bonne matrice.

Enfin, les appariements rejetés sont ceux qui ne respectent pas la géométrie épipolaire. Il se peut donc que nous conservions pour ce calcul des appariements faux, respectant par hasard la contrainte épipolaire. En conclusion, nous ne pourrons pas supprimer toutes les ambiguïtés à cette étape, et il faudra pouvoir travailler sur des matrices imprécises aux étapes suivantes.

3.4.8.1 Distance entre matrices fondamentales

L'utilisation d'images synthétiques parfaitement étalonnées nous permettra de comparer les matrices fondamentales calculées aux matrices fondamentales théoriques. Pour cela, nous avons besoin d'une mesure de distance entre matrices fondamentales. Celle que nous proposons s'appuie sur le calcul des distances des points à leurs épipolaires dans les images. Elle mesure la distance entre la matrice théorique ${F_{1,2}}^{th}$ et la matrice estimée ${F_{1,2}}^{es}$.

1. Tirer un point p au hasard dans l'image 1.
2. Calculer sa ligne épipolaire estimée $D_2^{es} = {F_{1,2}}^{es} p$.
3. Tirer un point q au hasard dans l'image 2, et sur D_2^{es}. Nous avons maintenant un couple aléatoire (p, q) respectant ${F_{1,2}}^{es}$.
4. Calculer la distance d_2 de q à son épipolaire théorique $D_2^{th} = {F_{1,2}}^{th} p$, et la distance d_1 de p à son épipolaire théorique $D_1^{th} = {F_{2,1}}^{th} q$.
5. Effectuer N fois les étapes 1 à 4, en accumulant les valeurs de $d = (d_1 + d_2)/2$.

Les valeurs que nous donnerons seront les valeurs moyenne, médiane et maximale de d, pour $N = 10000$ tirages (les images sont de taille 256×256, et au-delà de 10000 tirages, la distribution de d ne change plus). Il faut remarquer que cette mesure ne prend en compte que des points dans les images : on n'obtient pas une mesure de distance universelle entre deux matrices fondamentales, mais plutôt une mesure de la distance entre matrices *pour un couple d'images donné*. Cependant, ceci est conforme à l'usage que nous allons en faire, puisque nous n'utiliserons la matrice fondamentale que pour des points situés à l'intérieur des images.

De façon indépendante, nous avons retrouvé une mesure de distance similaire dans les travaux de Z. Zhang.

3.4.8.2 Autres contraintes géométriques

La contrainte épipolaire sert à contraindre la phase de mise en correspondance dense, et nous permet de rejeter les faux appariements. Nous pourrions aussi utiliser des contraintes multi-linéaires d'ordre supérieur (comme le tenseur trilinéaire), afin de rendre les appariements encore moins ambigus. Car contrairement à la géométrie épipolaire, il est impossible que des appariements faux entre 3 images respectent par hasard la contrainte trilinéaire. Le tenseur est de plus aisément calculé (bien qu'un peu plus instable) par les mêmes techniques normalisées et robustes.

Cependant, un algorithme d'appariement dense trinoculaire serait complexe à mettre en œuvre, si on veut qu'il reste symétrique, et il serait plus coûteux que deux appariements binoculaires, car il faudrait employer des mesures de corrélation trinoculaires, éventuellement robustes. Ceci est une voie intéressante qu'il faudrait explorer, mais nous préférons simplement fusionner des appariements binoculaires, et reporter ces questions à l'étape de reconstruction 3D, où nous pourrons facilement tester la cohérence d'appariements multi-oculaires (et non plus seulement trinoculaires).

3.4.9 Régularisation

Les algorithmes d'appariement dense ne calculent généralement pas un appariement rigoureusement dense, et ne renvoient pas une valeur de disparité pour tous les pixels où le calcul est possible (pixels non-occultés). Dans ce cas, la carte de disparité finale comportera des zones non renseignées, ce qui mènera à des trous dans l'image synthétisée.

L'étape de régularisation doit remplir les zones manquantes par une information de disparité, et diverses méthodes ont été proposées pour régulariser les cartes de disparité. Elles sont généralement incluses dans les algorithmes d'appariement, dont nous avons déjà parlé.

- Une étape de filtrage simple (filtre moyen) ne peut pas convenir, car elle aurait pour effet de lisser la carte, et les frontières d'occultation deviendraient mal définies.

- Un filtre médian pourrait convenir, car il préserve les contours. Cependant, il tend à étaler les appariements isolés en des amas de faux appariements.

- On peut aussi appliquer un filtrage anisotropique, qui nécessite beaucoup d'itérations et dont nous avons déjà parlé en 3.2.3.6.

- Il est enfin possible d'approximer les points connus par un modèle de surface 3D continu. Le modèle mathématique donne alors une valeur de disparité interpolée pour les points manquants. À cette étape, ce n'est pas possible pour nous, car nous n'avons pas d'information d'étalonnage.

Certaines de ces méthodes ne s'appuient que sur la carte de disparité (filtre médian), et ne font pas intervenir les informations présentes dans les images. Le filtrage anisotropique, lui, fait intervenir les gradients des images, afin de déterminer la direction de lissage. L'approximation par un modèle 3D fait parfois intervenir la seule carte de disparité, ou intègre dans d'autres cas des informations de contour des images (travaux de P. Fua).

Nous proposons de notre côté un autre algorithme, s'appuyant sur la carte de disparité et les images. Le principe est le suivant.

1. Pour un pixel p de la carte de disparité, calculer la disparité médiane d dans un voisinage 9×9. La carte de disparité est dans le repère de la première image, donc p est un point de l'image 1.

2. Calculer l'erreur médiane en disparité par rapport à d, dans ce voisinage.

3. En supposant que ces erreurs suivent une loi normale, on peut estimer l'écart-type σ de leur distribution, ce qui indique la pente moyenne locale de la surface (σ, calculé à partir de la médiane des écarts par rapport à la disparité médiane, est une estimation robuste du gradient de disparité).

4. Rechercher dans la seconde image un appariement q pour p. La zone de recherche est donnée par $d \pm \sigma$.

5. Dans les points q de cette zone, garder le meilleur appariement (p, q) sur un critère de corrélation SAD 5×5.

6. Répéter les étapes 1 à 5 pour tous les pixels (renseignés ou non) de la carte de disparité.

Si on suit l'algorithme tel quel, il s'apparente à un WTA (étape 5), et fournira un appariement pour tous les points de l'image. Aussi, nous rajoutons comme critère à l'étape 5, que le meilleur appariement doit avoir un coût inférieur à la médiane des coûts des appariements de ses voisins : dans le voisinage de p, on calcule la distribution des coûts des appariements entre les voisins de p et leurs correspondants établis ; la valeur médiane de cette distribution donne une idée du coût de corrélation localement «acceptable», en deçà duquel un couple (p, q) peut être déclaré apparié.

Cet algorithme sera évalué en 3.5.1.5.

3.4.10 Affinage d'appariements

Les appariements obtenus, éventuellement régularisés, sont positionnés au mieux au pixel près, car tous nos algorithmes d'appariement travaillent sur des pixels entiers.

Nous pourrons avoir besoin d'une plus grande précision, et nous verrons qu'une étape d'affinage est nécessaire dès l'étape de calcul de la géométrie épipolaire, préalable à l'appariement dense. Nous proposons quatre méthodes d'affinage. À partir d'un couple apparié (p, q), elles renvoient un couple apparié (p, q'), où la position de p dans l'image 1 reste fixe, et la position de q dans l'image 2 a été affinée en q'.

3.4.10.1 Affinage itératif

À cet algorithme, que nous appelons AffSi, nous devons fournir (dx, dy), déplacement maximal autorisé pour le point q à chaque itération. La zone centrée sur q et de taille $(2dx, 2dy)$ est découpée en 121 points q_i répartis sur une grille 11×11. Une corrélation SAD 11×11 est évaluée entre p et chacun des points q_i, et le meilleur point q_{i_0} est conservé. L'algorithme est ensuite appelé récursivement pour affiner l'appariement (p, q_{i_0}), avec un déplacement maximal autorisé $(dx/2, dy/2)$ (figure 3.26). Les corrélations sur des pixels non-entiers sont calculées de façon usuelle, en prenant comme valeur d'intensité de chaque pixel une interpolation bilinéaire de ses 4 plus proches voisins entiers. La récursion s'arrête lorsque $dx < 0.01$ et $dy < 0.01$.

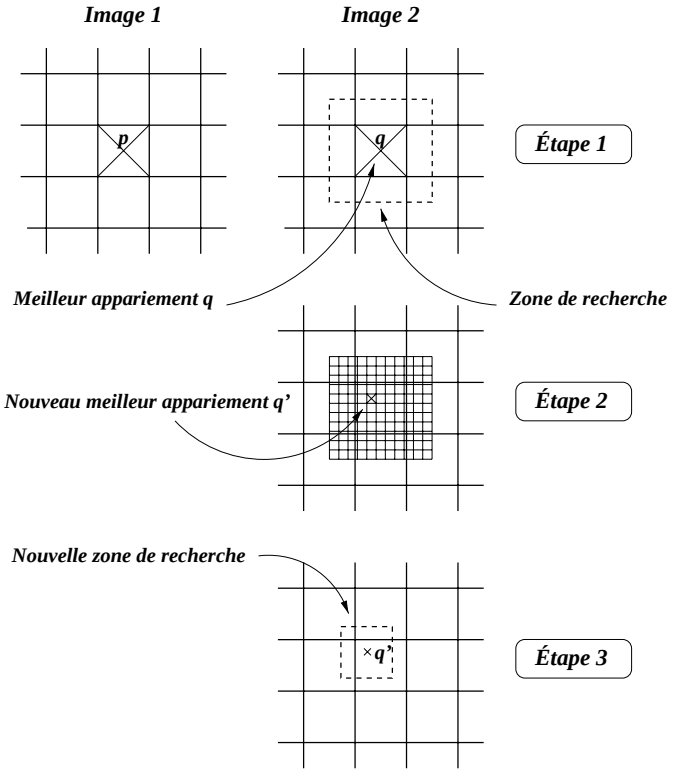


FIG. 3.26: *Principe d'une itération de l'affinage* AFFSI.

Tester les corrélations sur une grille 11×11, donc assez fine, pour ne descendre ensuite en précision que d'un facteur 2, a pour but d'éviter autant que possible de converger vers un minimum local de la fonction de corrélation.

On peut de plus obtenir des déplacements supérieurs à l'amplitude de (dx, dy). En effet, si à l'étape 2, le point q' se trouve dans un angle de la grille, alors à l'étape 3, la nouvelle zone de recherche débordera la zone initiale. Ainsi, pour $(dx, dy) = (0.5, 0.5)$, on pourra obtenir au maximum des déplacements de $0.5 + 0.25 + 0.125 = 0.975$ pixel. Pour $(dx, dy) = (1, 1)$, le déplacement maximal pourra être de $1 + 0.5 + 0.25 + 0.125 = 1.975$ pixels.

3.4.10.2 Affinage non-itératif — 1

La méthode d'affinage non-itératif de Z.D. Lan à 4 voisins décrite en 3.2.2.3 est nommée AFFZD4. Nous l'utiliserons avec une fenêtre de taille 11×11 pour l'estimation des paramètres.

3.4.10.3 Affinage non-itératif — 2

Une autre méthode non-itérative est proposée dans [Chr 98] (à paraître). Nommée AFFSC1, elle se base sur l'hypothèse que le point q représente une approximation d'un

minimum local de la fonction de corrélation. Cette fonction $z(x, y)$ peut être approximée pour sa composante sur l'axe des abscisses par une équation de parabole (éq. 3.50), ainsi que sa composante sur l'axe des ordonnées (éq. 3.51).

$$z(0, y) = g(y) = dy^2 + ey + f \tag{3.50}$$
$$z(x, 0) = f(x) = ax^2 + bx + c \tag{3.51}$$

À partir des cinq mesures de corrélation des couples formés de p dans l'image 1, et de q et de ses 4 voisins dans l'image 2 : $(p, q + (-1, 0))$, $(p, q + (0, -1))$, (p, q), $(p, q + (0, 1))$, et $(p, q + (1, 0))$, on calcule a, b, c, d et e de façon linéaire, puis le minimum x_0 de $f(x)$, et le minimum y_0 de $g(y)$. On pose alors $q' = (x_0, y_0)$, c.-à-d. :

$$\begin{cases} c_1 &= \text{SAD}(p, q + (-1, 0)) \\ c_2 &= \text{SAD}(p, q + (0, -1)) \\ c_3 &= \text{SAD}(p, q + (0, 0)) \\ c_4 &= \text{SAD}(p, q + (0, 1)) \\ c_5 &= \text{SAD}(p, q + (1, 0)) \\ a &= c_1 + c_5 - 2c_3 \\ b &= c_5 - c_1 \\ d &= c_2 + c_4 - 2c_3 \\ e &= c_4 - c_2 \\ q' &= q + (-b/(2a), -e/(2d)) \end{cases} \tag{3.52}$$

Les solutions où le nouveau minimum local q' est à plus de 1 pixel de distance sont rejetées, ainsi que les cas où $a = 0$ ou $b = 0$. Dans ces situations, on prend $q' = q$ (pas de déplacement).

3.4.10.4 Affinage non-itératif — 3

Dans cette méthode AffSc2, on suppose que la valeur de corrélation z décrit un paraboloïde (éq. 3.53), que l'on estime à partir des valeurs de corrélation de p avec q et ses 8 voisins.

$$z = ax^2 + by^2 + cxy + dx + ey + f \tag{3.53}$$

La valeur de la fonction z est en effet connue en ces 9 points, et on peut estimer ses 6 paramètres a, b, c, d, e et f aux moindres carrés (équation 3.56).

$$A = \begin{pmatrix} 1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 0 & 0 & -1 & 0 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 \\ 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \qquad X = \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} \tag{3.54}$$

$$B = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{pmatrix} = \begin{pmatrix} \text{SAD}(p, q + (-1, -1)) \\ \text{SAD}(p, q + (-1, 0)) \\ \text{SAD}(p, q + (-1, 1)) \\ \text{SAD}(p, q + (0, -1)) \\ \text{SAD}(p, q + (0, 0)) \\ \text{SAD}(p, q + (0, 1)) \\ \text{SAD}(p, q + (1, -1)) \\ \text{SAD}(p, q + (1, 0)) \\ \text{SAD}(p, q + (1, 1)) \end{pmatrix} \tag{3.55}$$

(3.56)

Il faut résoudre $AX = B$, soit $X = (A^T A)^{-1} A^T B$. La résolution est très rapide, car $M = (A^T A)^{-1} A^T$ peut être précalculée, et les inconnues a, b, c, d, e et f sont obtenues par une simple multiplication matricielle (équations 3.57 et 3.58).

$$M = \frac{1}{36} \begin{pmatrix} 6 & 6 & 6 & -12 & -12 & -12 & 6 & 6 & 6 \\ 6 & -12 & 6 & 6 & -12 & 6 & 6 & -12 & 6 \\ 9 & 0 & -9 & 0 & 0 & 0 & -9 & 0 & 9 \\ -6 & -6 & -6 & 0 & 0 & 0 & 6 & 6 & 6 \\ -6 & 0 & 6 & -6 & 0 & 6 & -6 & 0 & 6 \\ -4 & 8 & -4 & 8 & 20 & 8 & -4 & 8 & -4 \end{pmatrix} \tag{3.57}$$

$$X = MB \tag{3.58}$$

Le minimum local est atteint en un point où la dérivée en x et la dérivée en y s'annulent, et on trouve :

$$q' = q + (2bd - ce, 2ae - cd)/(c^2 - 4ab) \tag{3.59}$$

Comme pour AffSc1, si le déplacement obtenu est supérieur à 1 pixel, ou si $c^2 - 4ab = 0$, alors les hypothèses sont mal respectées, et nous prenons $q' = q$.

3.4.10.5 Récapitulatif

Le tableau 3.7 résume les caractéristiques des quatre algorithmes d'affinage d'appariements. Ce tableau est repris à la fin de ce chapitre, en page 142.

Nom	Nature	Description
AffSi	itératif	Plusieurs mesures SAD dans un voisinage de plus en plus restreint.
AffZd4	non-itératif	Développement limité du signal d'intensité des images.
AffSc1	non-itératif	Approximation de la fonction de coût de corrélation par deux paraboles.
AffSc2	non-itératif	Approximation de la fonction de coût de corrélation par un paraboloïde.

TAB. 3.7: *Récapitulatif des algorithmes d'affinage.*

Les méthodes AffSc1 et AffSc2 s'appuient sur une hypothèse de minimum local : l'appariement initial (p, q) doit correspondre à un minimum local de la fonction de corrélation pour p parmi les 8 voisins de q. Cette hypothèse ne peut pas être garantie par les algorithmes d'appariement que nous utilisons, mais elle est globalement respectée, et nous évaluerons les résultats dans ces conditions.

3.5 Évaluation

Dans cette partie, nous évaluons les méthodes d'appariement évoquées, et choisissons les plus adaptées. Les tests portent sur nos images de synthèse avec textures mathématiques ou textures réelles (en 3.5.1, 3.5.2 et 3.5.3) et des images réelles (3.5.4), en suivant les 6 étapes de l'appariement dense décrites en 3.4.2.

3.5.1 Évaluation sur images de synthèse — 1

Comme déjà évoqué, les images de synthèse nous permettent d'établir des comparaisons chiffrées, par comparaison des données obtenues aux données théoriques. Les données théoriques sont disponibles, car nous connaissons exactement le modèle 3D de la scène, et les paramètres de projection (coefficients des matrices de projection). Dans toute cette partie, nous traitons les images de synthèse avec texture mathématique, avec différents niveaux de bruit.

3.5.1.1 Phase 1

Nous extrayons les points d'intérêt avec un détecteur de Harris modifié. Ce filtre repère les pixels dont le gradient des niveaux de gris présente une forte courbure. Il détecte donc des points isolés, ou des extrémités, ou des coins.

Ce détecteur est mal localisé, c.-à-d. il ne renvoie pas le point situé juste sur le coin, mais un point situé à, au mieux, 1 pixel de distance. Cependant, il a une bonne répétabilité : le biais observé est reproductible, et se situe toujours du même côté du coin. On repère donc la projection du même point physique dans toutes les images (quel que soit l'angle de vue), et c'est ce qui importe pour la bonne marche des algorithmes de stéréovision.

Dans nos 6 images de synthèse nous avons détecté le nombre de points suivant noté en tableau 3.8.

Image	**Nombre de points**
im0	509
im1	513
im2	530
im3	524
im4	525
im5	526

TAB. 3.8: *Nombre de points d'intérêt détectés dans les 6 images synthétiques.*

Les textures des images de synthèse ont été générées de façon à faciliter la détection, et c'est pourquoi sur ces images, nous obtenons facilement un grand nombre de points d'intérêt, bien répartis sur l'image (voir figure 3.27 pour un exemple). Ceci peut se produire de façon naturelle pour des scènes d'extérieur réelles, qui sont généralement très texturées : végétation, reliefs du sol, arbres et objets. C'est bien sûr un cas très favorable par rapport à des scènes d'intérieur classiques, qui présentent souvent de grands aplats de couleur uniforme : murs, sol, meubles, plafond. Les points ne sont alors détectés qu'à des ruptures de profondeur : coins des meubles, angles des murs, des fenêtres, car ces points sont généralement bien contrastés.

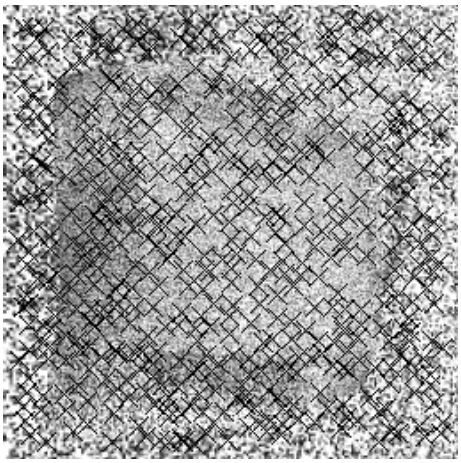


FIG. 3.27: *Les 509 points d'intérêt détectés dans im0 (croix noires).*

Le but dans cette partie n'est de toute façon pas de modéliser la «pire réalité», mais d'évaluer l'enchaînement algorithmique sur des données vérifiables, et de séparer les problèmes. Nous verrons lors de l'évaluation sur images à textures réelles quels problèmes peut soulever une éventuelle mauvaise répartition des points d'intérêt.

3.5.1.2 Phase 2

Dans cette étape, nous apparions les points d'intérêt détectés, à l'aide d'un algorithme CCR d'appariement. L'algorithme CCR a été choisi pour toutes les raisons évoquées précédemment : robustesse, symétrie, absence d'ordre dans les données en entrée. Nous utilisons une mesure SAD de taille 15×15, ce qui sera justifié *a posteriori* lors de l'évaluation des algorithmes d'appariement dense, en 3.5.1.4. Nous pouvons prendre un masque de taille relativement importante (15×15), car nous n'effectuons que de l'ordre de 10^6 mesures de corrélation. Nous n'imposons pas de limite de disparité.

Pour les 5 couples d'images (`im0`, `im1`) à (`im0`, `im5`), le nombre de points appariés est donné en tableau 3.9.

Couple	**Nombre d'appariements**
(`im0`, `im1`)	156
(`im0`, `im2`)	125
(`im0`, `im3`)	106
(`im0`, `im4`)	74
(`im0`, `im5`)	70

TAB. 3.9: *Nombre de points d'intérêt appariés dans les 5 couples.*

3.5.1.3 Phase 3

Nous calculons maintenant la géométrie épipolaire de chaque couple d'images, sur la base des appariements obtenus en phase 2. L'algorithme de calcul est celui décrit en 3.4.8 : calcul normalisé et robuste, basé sur SVD. Le tableau 3.10 résume les résultats.

- L'erreur moyenne est la distance moyenne des points conservés pour le calcul à leur droite épipolaire, mesurée en pixels (les points conservés sont ceux qui ont été trouvés conformes à la géométrie épipolaire).

- l'erreur médiane est la distance médiane des points *initiaux* à leur droite épipolaire. Cette valeur est donc liée à l'erreur maximale des points conservés pour le calcul (et non à leur erreur médiane).

Couple	Nombre d'appariements en entrée	Nombre d'appariements conservés	Erreur finale moyenne	Erreur finale médiane
(im0, im1)	156	132	0.391025	0.488084
(im0, im2)	125	98	0.337167	0.417800
(im0, im3)	106	66	0.314718	0.585288
(im0, im4)	74	44	0.637562	1.341120
(im0, im5)	70	45	0.382604	1.022550

TAB. 3.10: *Résultats du calcul robuste de géométrie épipolaire.*

Les erreurs moyennes et médianes laissent penser que les trois premières matrices fondamentales obtenues sont assez précises. Pour en être certain, nous pouvons comparer les matrices obtenues aux vraies matrices, que nous connaissons pour tous les couples d'images synthétiques. En utilisant la distance définie en 3.4.8, nous obtenons les résultats présentés en tableau 3.11.

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	0.581991	0.447368	3.545080
$F_{0,2}$	0.219233	0.186790	0.999508
$F_{0,3}$	0.290123	0.276557	0.950314
$F_{0,4}$	0.764849	0.556875	3.672280
$F_{0,5}$	0.529804	0.409399	3.001620

TAB. 3.11: *Comparaison des matrices calculées aux matrices théoriques.*

D'après le tableau 3.11, l'erreur moyenne et l'erreur médiane sont toujours inférieures à 1 pixel, mais l'erreur maximale n'est inférieure à 1 pixel que pour les couples (im0, im2) et (im0, im3).

Cela montre que les résultats du tableau 3.10 ne sont pas nécessairement une indication de la qualité des résultats. Dans ce tableau, le couple (im0, im1) conduisait à un bon calcul, en termes d'erreur médiane et d'erreur moyenne. Cependant, le tableau 3.11 montre que la matrice $F_{0,1}$ est parmi les deux plus mauvaises, et présente une erreur maximale de 3.5 pixels. La distribution des erreurs de la matrice $F_{0,1}$ calculée par rapport à la matrice $F_{0,1}$ réelle est donnée en figure 3.28

Les erreurs médianes et moyennes ne doivent donc pas être interprétées de façon trop optimiste, et un appariement dense basé sur le respect strict de la géométrie épipolaire ne pourra pas être correct sur toute l'image. Nous reviendrons sur ce point plus loin.

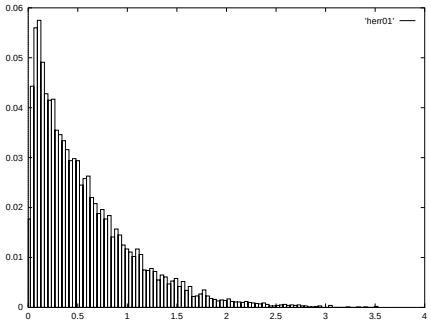


FIG. 3.28: *Distribution des erreurs de la matrice $F_{0,1}$ calculée, en pixels.*

Pour tenter d'améliorer ces résultats, nous effectuons un affinage des appariements originaux, obtenus en sortie de la phase 2. La méthode employée est AFFSI 11×11, avec une amplitude $(dx, dy) = (0.5, 0.5)$. Nous calculons ensuite les matrices fondamentales comme précédemment ; les résultats sont présentés en tableaux 3.12 et 3.13.

Couple	**Nombre d'appariements en entrée**	**Nombre d'appariements conservés**	**Erreur finale moyenne**	**Erreur finale médiane**
(im0, im1)	156	128	0.089676	0.135555
(im0, im2)	125	95	0.106240	0.141463
(im0, im3)	106	65	0.228901	0.395962
(im0, im4)	74	42	0.189002	0.563886
(im0, im5)	70	44	0.190206	0.312832

TAB. 3.12: *Résultats du calcul robuste de géométrie épipolaire, sur des appariements affinés.*

Matrice fondamentale	**Erreur moyenne**	**Erreur médiane**	**Erreur maximale**
$F_{0,1}$	0.223500	0.208757	0.694974
$F_{0,2}$	0.121809	0.115804	0.383600
$F_{0,3}$	1.808370	1.720290	5.183190
$F_{0,4}$	0.455319	0.429282	1.504480
$F_{0,5}$	0.181927	0.146914	0.726565

TAB. 3.13: *Comparaison des matrices calculées aux matrices théoriques, sur des appariements affinés.*

Les résultats sont donc très fortement améliorés par l'étape de précision itérative, sauf pour $F_{0,3}$, qui correspond à une séparation de 15° entre les images, et où l'on aboutit «par malchance» à une mauvaise configuration (erreur médiane de 1.7 pixels). Seule une autre matrice présente une erreur maximale supérieure à 1 pixel : $F_{0,4}$, déjà mauvaise au

départ, et qui représente une séparation de 20 ° . Si nous procédions à un affinage AffSi avec $(dx, dy) = (1, 1)$ au lieu de $(0.5, 0.5)$, alors les qualités des matrices calculées seraient modifiées selon le tableau 3.14.

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	0.131375	0.109849	0.621448
$F_{0,2}$	0.103004	0.084604	0.544506
$F_{0,3}$	0.054882	0.055286	0.149241
$F_{0,4}$	0.281090	0.259904	0.998700
$F_{0,5}$	0.151822	0.117107	0.687994

Tab. 3.14: *Comparaison des matrices calculées aux matrices théoriques, sur des appariements affinés avec une amplitude* $(dx, dy) = (1, 1)$.

La matrice $F_{0,3}$ est améliorée d'un facteur 50, alors qu'elle avait été dégradée par le précédent affinage. En revanche, la matrice $F_{0,2}$ est légèrement moins précise après affinage $(dx, dy) = (1, 1)$, qu'après affinage $(dx, dy) = (0.5, 0.5)$. Malgré toutes les précautions prises, le calcul de la géométrie épipolaire peut donc, dans certains cas (dépendants de la position des appariements initiaux, et de leur répartition), se révéler relativement instable. En supposant ne pas disposer de ce type d'information, nous poursuivrons nos calculs sur la base d'un affinage avec $(dx, dy) = (0.5, 0.5)$.

Enfin, nous testons tout le processus : extraction de points d'intérêt, appariement, affinage, et calcul de la géométrie épipolaire pour les images bruitées. Les résultats sont reportés dans le tableau 3.15. Rappelons que pour bruiter les images originales, nous perturbons aléatoirement le niveau de gris m de chaque pixel par une loi gaussienne centrée en m et d'écart-type σ, pour différentes valeurs de σ.

Même sur les images bruitées, les erreurs moyennes et médianes restent toujours inférieures à 1 pixel, et très souvent inférieures à 0.5 pixel. Enfin, l'erreur maximale sur $F_{0,1}$ est toujours inférieure à 1 pixel, sauf pour un bruit de $\sigma = 10$ niveaux de gris.

Les matrices fondamentales calculées par ces techniques sont donc utilisables dans une large gamme : pour les images les plus écartées (25 °), les erreurs moyenne et médiane restent raisonnables, autour de 0.5 pixel. Notons que les points d'intérêt sont bien répartis, et nous établirons d'autres résultats sur les images synthétiques à textures réelles.

Ce bon fonctionnement dans le cas général ne doit pas empêcher de rester prudent, car nous avons aussi rencontré de très mauvais résultats : erreurs de plusieurs pixels en moyenne et en médiane pour les images (`im0`, `im3`) et un bruit $\sigma = 0$ (tableau 3.13). Cela constitue un cas rare, mais susceptible de survenir. La source des problèmes se situe en amont, lors de la phase de calcul des appariements. Leur précision et leur correction restent déterminantes pour la qualité du calcul de géométrie épipolaire, malgré l'utilisation de techniques robustes et de normalisation.

$\sigma = 1$

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	0.142096	0.132337	0.486521
$F_{0,2}$	0.056309	0.042361	0.336239
$F_{0,3}$	0.150394	0.141390	0.456391
$F_{0,4}$	0.332485	0.318283	1.051960
$F_{0,5}$	0.277438	0.213381	1.256960

$\sigma = 2$

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	0.176740	0.135878	0.991496
$F_{0,2}$	0.287991	0.235065	1.378390
$F_{0,3}$	0.559737	0.534011	1.652230
$F_{0,4}$	0.623187	0.594677	1.831230
$F_{0,5}$	0.182025	0.141027	0.812231

$\sigma = 5$

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	0.202212	0.166752	0.971653
$F_{0,2}$	0.327671	0.307181	1.043880
$F_{0,3}$	0.171162	0.143686	0.860316
$F_{0,4}$	0.467631	0.427387	1.718590
$F_{0,5}$	0.250131	0.185094	1.141230

$\sigma = 10$

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	0.204990	0.146932	1.252150
$F_{0,2}$	0.300081	0.207226	1.819220
$F_{0,3}$	0.421629	0.372187	1.638480
$F_{0,4}$	0.199520	0.194170	0.535890
$F_{0,5}$	0.429641	0.349853	1.856940

TAB. 3.15: *Comparaison des matrices calculées aux matrices théoriques, sur des appariements affinés, à partir d'images bruitées.*

3.5.1.4 Phase 4

Dans cette phase, nous calculons un appariement dense binoculaire entre deux images de la séquence, sous la contrainte de la géométrie épipolaire, calculée à l'étape précédente.

Comparaison des mesures

Nous devons tester 10 mesures de ressemblance : SAD, ZSAD, SSD, ZSSD, RSSD, RZSSD, WSAD, RSAD, PRSAD, INVAR. Pour les 9 premières (mesures de corrélation), nous devons tester plusieurs tailles de masque, au moins de 1×1 à 15×15, soient 8 tailles. Enfin, nous voulons tester la robustesse des mesures aux déformations perspectives, c.-à-d. la qualité d'appariement pour des images de plus en plus écartées : nous testerons les couples (`im0`, `im1`) à (`im0`, `im5`), soient 5 couples, avec des bruitages croissants d'écart-type 0, 1, 2, 5 et 10 niveaux d'intensité. Dans un premier temps, l'algorithme utilisé sera CCR.

Il est impossible de tester exhaustivement ces (9 mesures $\times$ 8 tailles de masque + la mesure INVAR) $\times$ 5 couples d'images $\times$ 5 niveaux de bruit = 1825 configurations. Nous allons donc sérier les tests.

Nous comparons tout d'abord les mesures, avec 9 mesures de corrélation toutes de même taille 5×5 ou 15×15, et la mesure INVAR (qui travaille sur une zone 31×31). Pour chacune de ces mesures, nous calculons un appariement dense par l'algorithme CCR entre les images `im0` et `im1` (bruitées avec un écart-type σ), sous contrainte épipolaire donnée par la matrice $F_{0,1}$, calculée à l'étape 3 dans les mêmes conditions de bruitage. Après examen des images, nous avons fixé une limite de disparité de 25 pixels. Les résultats donnés sont les suivants.

- Proportion τ d'appariements trouvés par rapport au nombre maximal d'appariements qu'il était possible de trouver. Par exemple, si l'on tente d'apparier 5 points de l'image 1 avec 300 points de l'image 2, et qu'on trouve 4 appariements, cette proportion est de 80 %.

- Proportion d'appariements respectivement à moins de 0.05, 0.10, 0.50, 1.00 et 2.00 pixels de leur position exacte théorique. Le point dans l'image 1 est supposé être exact, ce qui permet de calculer la position du point 3D dans la scène synthétique (le modèle de la scène est connu ainsi que la matrice de projection dans la première image) ; la distance mentionnée est celle séparant la projection de ce point 3D dans la seconde image, de la position trouvée dans le calcul d'appariement. Les proportions sont données *par rapport au nombre total d'appariement trouvés* (et non par rapport au nombre maximal d'appariements qu'il était possible de trouver).

- Temps de calcul, avec pour base 100 le calcul le plus rapide, dans chaque tableau. Ceci permet de mesurer le temps nécessaire à l'appariement, y compris pour les mesures PRSAD et INVAR, qui nécessitent plusieurs passes, et que nous n'avions pas pu quantifier jusqu'ici.

Les résultats sont résumés dans le tableau 3.16 pour les mesures 5×5, et 3.17 pour les mesures 15×15.

Couple (im0, im1), $\sigma = 0$.

Mesure 5×5	τ	**Erreur** (proportion des appariements trouvés) < 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**Temps** CPU
SAD	62.80	1.13	4.10	62.03	86.34	87.94	100
ZSAD	59.91	1.17	4.20	62.45	84.58	85.70	110
SSD	62.44	1.12	4.06	62.23	87.19	88.71	137
ZSSD	59.56	1.16	4.20	63.06	85.46	86.48	157
RSSD	56.70	1.17	4.00	52.14	69.00	71.64	198
RZSSD	53.87	1.15	4.11	50.28	64.48	66.44	222
WSAD	58.08	1.19	4.25	59.33	79.87	81.60	176
RSAD	61.45	1.14	4.11	60.02	82.27	84.09	157
PRSAD	58.74	0.80	3.14	55.87	82.62	84.73	745
INVAR	37.73	0.32	1.23	21.37	47.95	66.98	1314

TAB. 3.16: *Comparaison des mesures de corrélation* 5×5. *Tous les chiffres sont des pourcentages.*

Couple (im0, im1), $\sigma = 0$.

Mesure 15×15	τ	**Erreur** (proportion des appariements trouvés) < 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**Temps** CPU
SAD	75.52	0.97	3.38	55.66	91.87	95.28	100
ZSAD	74.48	0.98	3.43	56.33	92.55	95.64	122
SSD	74.46	0.97	3.38	55.68	92.26	95.56	255
ZSSD	73.76	0.97	3.41	56.27	92.75	95.73	284
RSSD	75.00	0.97	3.41	56.02	90.96	94.28	499
RZSSD	70.82	1.05	3.65	58.26	92.70	95.36	577
WSAD	67.81	1.11	3.86	60.66	89.84	91.99	408
RSAD	75.71	0.97	3.40	55.78	92.07	95.47	464
PRSAD	68.76	0.55	2.10	41.04	86.35	93.34	2231
INVAR	37.73	0.32	1.23	21.37	47.95	66.98	668

TAB. 3.17: *Comparaison des mesures de corrélation* 15×15. *Tous les chiffres sont des pourcentages.*

Les taux d'appariement τ sont faibles, de l'ordre de 60 % pour les mesures 5×5. La figure 3.29 montre la carte de disparité obtenue pour le couple (im0, im1) apparié par une mesure SAD 5×5. Une zone claire correspond à une disparité maximale, et une zone sombre à une disparité nulle. Les zones non-appariées sont représentées en blanc. La carte de disparité est donnée dans le repère de la première image du couple (im0 pour le couple (im0, im1)).

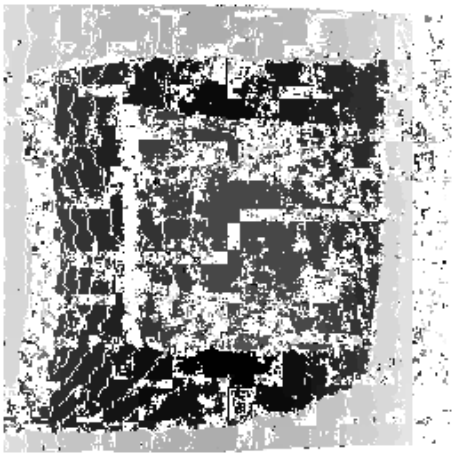


FIG. 3.29: *La carte de disparité obtenue pour le couple (*im0*,* im1*), apparié par une mesure* SAD 5×5.

Les zones non-appariées correspondent soit à des pixels non couverts par les lignes épipolaires successives (problèmes de discrétisation), soit à des pixels difficiles à apparier car trop ambigus, soit enfin à des pixels non conformes à la géométrie épipolaire. En effet, la géométrie épipolaire $F_{0,1}$ est assez précise (erreur en moyenne de 0.22 pixel, au maximum de 0.69 pixel), mais les points des épipolaires conjuguées sont approximés par une droite de Bresenham pour couvrir des pixels entiers, donc à 0.5 pixel près. Les erreurs s'accumulent, et l'algorithme tente d'apparier des points qui ne peuvent pas se correspondre (la texture étant fine, des points déplacés de 1 pixel se ressemblent peu).

Nous avons mesuré ces erreurs. Pour chaque pixel p de l'image `im0`, nous traçons son épipolaire conjuguée d dans `im1`. Puis nous calculons l'erreur maximale de p par rapport à tous les points de d dans une limite de disparité de 25 pixels, vis-à-vis de la géométrie épipolaire théorique. Pour tous les pixels p de `im0`, la plus grande valeur de cette erreur a été estimée à 1.84 pixels, beaucoup plus que les 0.69 pixels théoriques. Nous montrons en figure 3.30 la distribution de ces erreurs sur l'image `im0`, qui explique la distribution compacte et régulière des zones non-appariées.

Ceci montre qu'avec l'algorithme CCR, qui effectue des vérifications croisées, une géométrie épipolaire même assez précise peut mener à des résultats peu denses. La solution usuelle est de prendre des points non pas situés strictement sur les épipolaires, mais aussi les points proches (à ± 1 pixel près). C'est la solution que nous prendrons pour les expérimentations finales. Néanmoins, cela représente en moyenne 3 fois plus de points à comparer dans les images, et pour des raisons de temps de calcul, nous poursuivons les tests avec l'algorithme original: nous ne testons que les pixels situés strictement sur les lignes épipolaires, donc τ restera assez faible. Cela n'a pas d'importance, puisque nous ne faisons qu'établir des comparaisons d'appariements effectués tous dans les mêmes conditions.

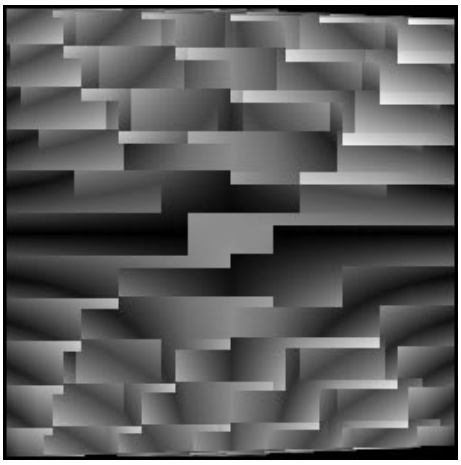


FIG. 3.30: *Répartition des erreurs maximales de géométrie épipolaire dues à la discrétisation de Bresenham. Les zones noires correspondent à une erreur nulle, et les zones blanches à une erreur de* 1.84 *pixels.*

Reprenons l'examen des tableaux 3.16 et 3.17. Nous constatons d'abord que, bien que toutes les mesures soient au mieux au pixel près, on trouve une proportion importante d'appariements corrects à 0.5 pixel près.

Sur le seul critère de précision d'appariement, les meilleures mesures sont les classiques SAD, ZSAD, SSD et ZSSD. Les mesures 5×5 RSAD, WSAD et PRSAD sont encore relativement bonnes, et les mesures INVAR, RSSD et RZSSD arrivent en queue de peloton. En revanche, toutes les mesures 15×15 sont bonnes ou très bonnes (sauf INVAR, qui bien sûr ne change pas). L'explication que nous fournissons est la suivante.

- Lorsque les masques sont relativement petits, ils ont toutes les chances de se correspondre pixel à pixel, car il sont situés sur des épipolaires conjuguées. Les erreurs sont rares, et elles suivent une distribution laplacienne. Ceci correspond à l'hypothèse de RSAD et PRSAD, mais pas RSSD ni RZSSD, ce qui explique les différences de résultat.

- Lorsque les masques sont plus grands, les pixels pris en compte pour la corrélation sont plus aléatoirement distribués, et l'hypothèse gaussienne devient correcte. Plus généralement, toutes les mesures sont bonnes, car elles intègrent un très grand nombre de pixels (225), ce qui les rend toutes robustes.

Cette hypothèse a été vérifiée expérimentalement sur le couple (`im0`, `im1`). Pour chaque point p de l'image `im0` et chaque point q de l'image `im1` situé dans une limite de disparité de 25 pixels, nous calculons l'erreur pixel par pixel d'une fenêtre centrée autour de p par rapport aux pixels d'une fenêtre centrée sur q, et nous traçons la distribution de ces erreurs. La figure 3.31 montre que pour un masque de petite taille, l'hypothèse laplacienne est la plus correcte, alors que pour un masque de grande taille, l'hypothèse gaussienne est bien adaptée. Sur cette figure, les paramètres des courbes laplaciennes et gaussiennes ont été déterminés par calcul de la médiane de la distribution, comme expliqué en 3.2.2.2.

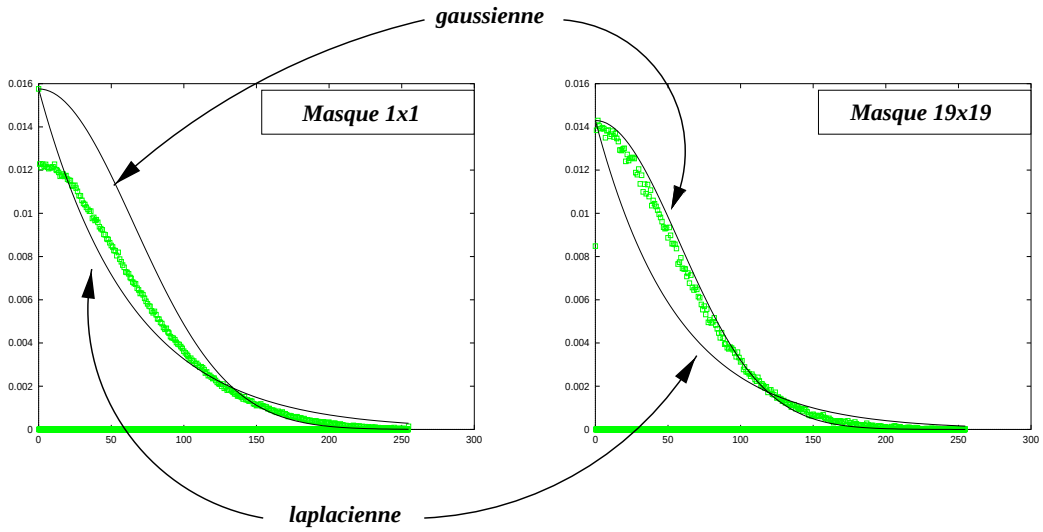


FIG. 3.31: *Distribution des erreurs pixel à pixel des fenêtres de corrélation entre les images* $im0$ *et* $im1$ *(voir texte).*

Les mesures robustes ne semblent donc pas présenter un net avantage : pour les petites tailles de masque, elles sont au mieux aussi bonnes que les mesures classiques, et pour les grandes tailles, elles sont inutiles. Par construction, elles ne sont vraiment adaptées qu'à proximité des zones d'occultation, et les chiffres des tableaux précédents ne permettent pas de voir précisément l'influence de la robustesse, car nous testons en même temps tous les pixels de l'image. Aussi, nous dressons ci-dessous les mêmes tableaux, mais spécifiquement sur les pixels d'un contour d'occultation. La zone occultée et le contour d'occultation ont été établis manuellement, et ils sont décrits en figure 3.32. Les résultats pour les pixels de ce contour sont donnés dans les tableaux 3.18 et 3.19.

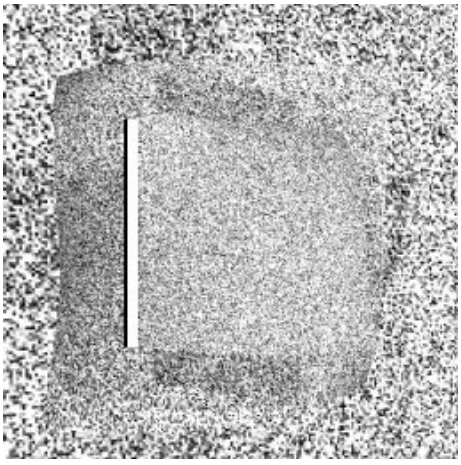


FIG. 3.32: *La zone occultée de l'image* $im0$*, en blanc, n'est pas visible dans l'image* $im1$*. La frontière d'occultation est donnée par le segment noir, à gauche de cette zone.*

Couple (im0, im1), $\sigma = 0$.

Mesure 5×5	**Erreur** < 0.50	< 1.00	< 2.00
SAD	56.76	78.38	78.38
ZSAD	47.37	71.05	73.68
SSD	62.16	81.08	81.08
ZSSD	57.58	75.76	75.76
RSSD	38.30	46.81	48.94
RZSSD	25.00	40.91	43.18
WSAD	45.65	63.04	63.04
RSAD	51.35	67.57	67.57
PRSAD	56.76	78.38	78.38
INVAR	0.00	11.76	41.18

TAB. 3.18: *Comparaison des mesures de corrélation* 5×5 *sur la frontière d'occultation.*

Couple (im0, im1), $\sigma = 0$.

Mesure 15×15	**Erreur** < 0.50	< 1.00	< 2.00
SAD	72.41	96.55	96.55
ZSAD	75.86	100.00	100.00
SSD	72.41	93.10	93.10
ZSSD	68.42	94.74	94.74
RSSD	62.79	93.02	93.02
RZSSD	80.00	97.50	97.50
WSAD	71.15	94.23	94.23
RSAD	72.41	96.55	96.55
PRSAD	41.67	95.83	95.83
INVAR	0.00	11.76	41.18

TAB. 3.19: *Comparaison des mesures de corrélation* 15×15 *sur la frontière d'occultation.*

Les résultats des tableaux 3.18 et 3.19 ne montrent aucune supériorité des mesures robustes, même sur la frontière d'occultation. Seule PRSAD 5×5 fournit des résultats honorables. Nous testons enfin toutes les mesures sur des images de texture aléatoire vues de face, représentant deux carrés empilés (figure 3.33). Ces images respectent l'hypothèse translationnelle, qui veut que les images ne présentent pas de déformation perspective.

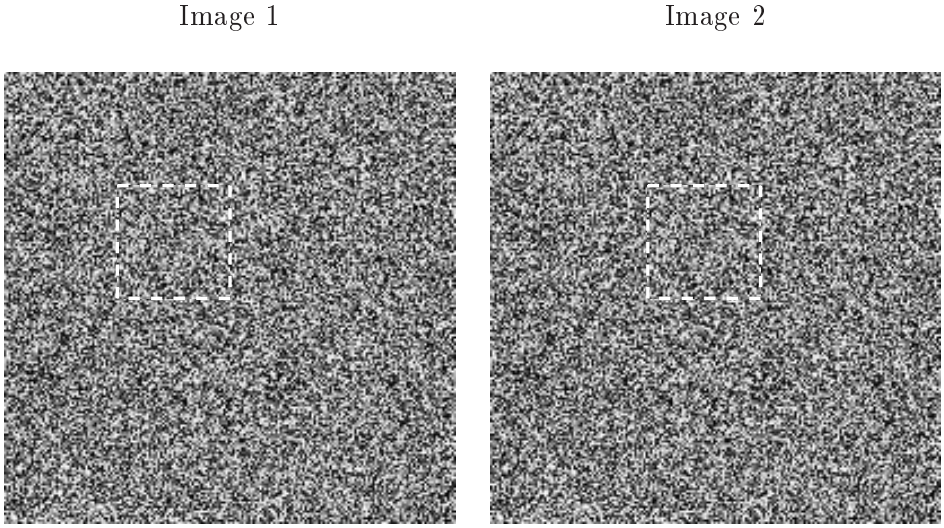


FIG. 3.33: *Deux images synthétiques sous hypothèse translationnelle. La zone translatée est contenue dans le rectangle pointillé blanc.*

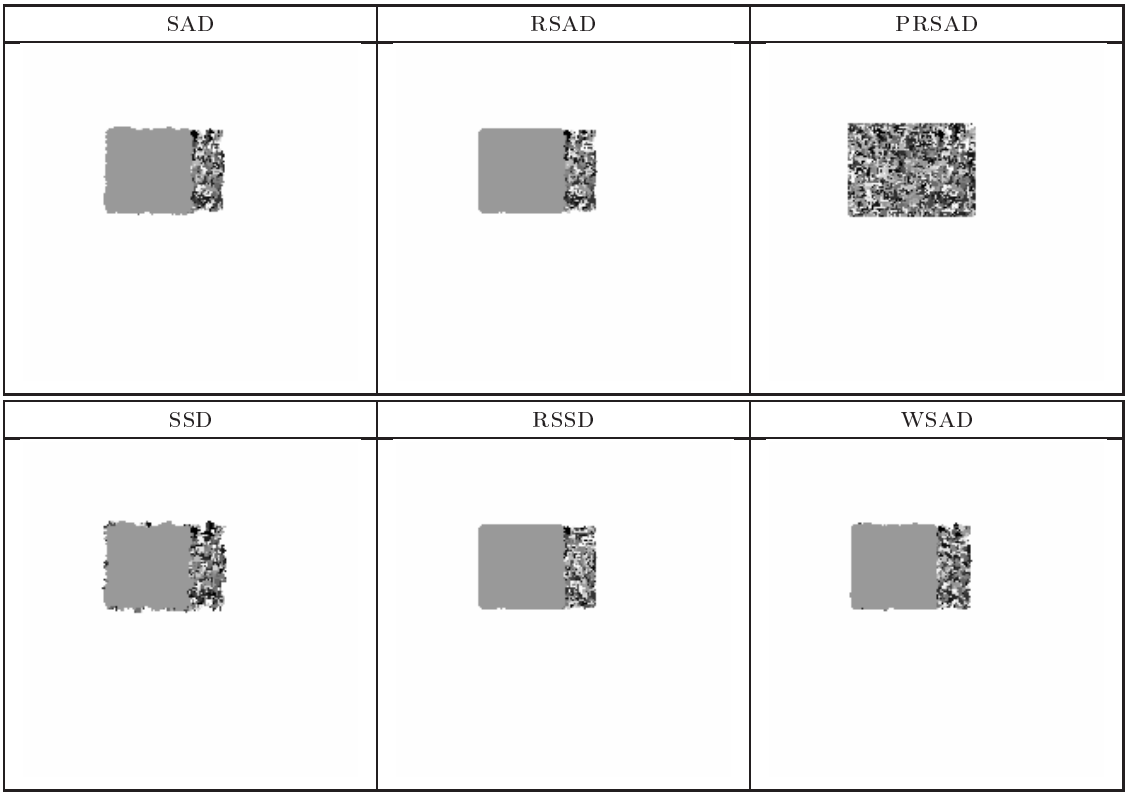


FIG. 3.34: *Cartes de disparité obtenues pour le couple de la figure 3.33.*

Nous montrons en figure 3.34 les cartes de disparité obtenues pour ce couple d'images,

pour plusieurs mesures 5×5, avec un algorithme d'appariement WTA. Nous voyons que les mesures de corrélation robustes donnent de très bons résultats sur les frontières d'occultation, même avec l'algorithme d'appariement WTA (qui est moins robuste que CCR). Sur ces images, la zone découverte présente une disparité aléatoire, car elle n'est pas visible dans l'autre image.

Les versions robustes RSAD et RSSD fonctionnent bien mieux que leurs contreparties SAD et SSD. Même la mesure WSAD inclut une part de robustesse, en accordant un poids plus fort aux pixels proches du centre de la fenêtre qu'aux pixels périphériques.

Cela confirme que les mesures robustes que nous utilisons ne sont vraiment adaptées qu'au cas translationnel: dans le cas non-translationnel, les zones comparées sont trop différentes, et les pixels rejetés du calcul robuste ne sont pas contraints à suivre une cohérence spatiale. La mesure n'a plus grande signification dans ce cas. PRSAD fonctionne mieux, car par construction, les pixels rejetés respectent une forme de compacité, ce qui correspond à de réelles occultations. Cependant, nous voyons sur la figure 3.34 que PRSAD ne fonctionne pas: ceci est dû au fait que la matrice de covariance Λ est calculée sur les images mêmes que nous sommes en train de traiter. Or, ces deux images synthétiques (planes) sont rigoureusement identiques (la translation du carré mise à part), et les mesures relevées sont en très grande majorité strictement nulles, ce qui n'est pas représentatif.

Nous en tirons les conclusions suivantes:

- les mesures robustes ne doivent être utilisées que dans le cas translationnel; si ce n'est pas le cas, il faut envisager des pré-traitements (redressement local des images), ou explorer les mesures partiellement robustes comme PRSAD;
- les mesures robustes ne doivent être utilisées que sur les frontières d'occultation, et si l'on ne connaît pas ces frontières à l'avance, il est inutile d'employer de telles mesures;
- un masque plus grand suffit à rendre une mesure classique plus robuste;
- nous n'utiliserons pas les mesures robustes pour la suite de l'appariement dense.

Revenons à l'image globale. Pour des masques 15×15, les mesures SAD et SSD sont bien adaptées, car pour ces tailles de masque élevées, leur avantage en vitesse s'accroît encore. Cela justifie *a posteriori* le choix de SAD pour l'appariement épars de points d'intérêt (phase 2). L'emploi d'un masque plus grand augmente fortement le taux τ d'appariement, mais améliore peu la précision: quand les images sont très texturées, les corrélations même de taille 5×5 sont très peu ambiguës, et augmenter la taille du masque n'apporte rien.

Aussi, nous reprenons ces tests sur la séquence bruitée, afin de déterminer l'influence de la taille du masque. Pour les séquences bruitées à $\sigma = 1, 2, 5, 10$, nous calculons des appariements denses pour le couple (`im0`, `im1`), pour toutes les mesures de ressemblance, et des tailles de masque de 5×5 ou 15×15. Les résultats sont en tableaux 3.20 et 3.21.

Couple (im0, im1), $\sigma = 1$

Mesure 5×5	τ	**Erreur** < 0.50	< 1.00	< 2.00
SAD	58.07	54.33	81.34	83.53
ZSAD	55.33	54.89	79.25	80.91
SSD	58.03	54.15	82.12	84.29
ZSSD	55.06	55.16	79.86	81.43
WSAD	54.50	51.45	73.76	76.16
INVAR	37.56	17.70	45.29	66.27

Couple (im0, im1), $\sigma = 2$

Mesure 5×5	τ	**Erreur** < 0.50	< 1.00	< 2.00
SAD	60.36	56.90	83.12	85.00
ZSAD	57.48	57.56	81.31	82.69
SSD	60.12	57.12	84.13	85.98
ZSSD	57.29	58.05	82.08	83.32
WSAD	56.45	54.14	76.00	78.18
INVAR	38.13	20.13	47.58	67.25

Couple (im0, im1), $\sigma = 5$

Mesure 5×5	τ	**Erreur** < 0.50	< 1.00	< 2.00
SAD	61.53	59.60	84.68	86.41
ZSAD	58.59	60.10	82.73	84.04
SSD	61.52	59.78	85.67	87.35
ZSSD	58.54	60.47	83.54	84.68
WSAD	56.82	56.66	77.43	79.38
INVAR	37.73	19.77	46.87	67.00

Couple (im0, im1), $\sigma = 10$

Mesure 5×5	τ	**Erreur** < 0.50	< 1.00	< 2.00
SAD	58.86	56.68	82.51	84.28
ZSAD	55.94	57.19	80.21	81.57
SSD	58.94	57.05	83.82	85.39
ZSSD	55.77	57.95	81.47	82.63
WSAD	54.75	52.82	73.98	76.09
INVAR	36.27	17.59	44.70	63.65

TAB. 3.20: *Comparaison des mesures de corrélation* 5×5*, pour différentes valeurs de* σ*. Tous les chiffres sont des pourcentages.*

Couple (im0, im1), $\sigma = 1$

Mesure 15×15	τ	**Erreur** < 0.50	< 1.00	< 2.00
SAD	66.77	45.82	89.82	94.65
ZSAD	65.57	46.63	90.68	95.04
SSD	65.79	45.91	90.24	94.94
ZSSD	64.82	46.67	90.93	95.10
WSAD	58.02	52.33	85.90	88.59
INVAR	37.56	17.70	45.29	66.27

Couple (im0, im1), $\sigma = 2$

Mesure 15×15	τ	**Erreur** < 0.50	< 1.00	< 2.00
SAD	67.55	50.21	90.80	94.69
ZSAD	66.47	50.95	91.48	95.02
SSD	66.55	50.36	91.34	95.04
ZSSD	65.69	51.13	91.91	95.21
WSAD	59.61	55.73	87.28	89.69
INVAR	38.13	20.13	47.58	67.25

Couple (im0, im1), $\sigma = 5$

Mesure 15×15	τ	**Erreur** < 0.50	< 1.00	< 2.00
SAD	68.87	53.37	91.43	94.88
ZSAD	67.76	54.22	92.17	95.23
SSD	67.67	53.79	92.04	95.31
ZSSD	66.92	54.43	92.49	95.33
WSAD	60.88	58.48	88.81	90.91
INVAR	37.73	19.77	46.87	67.00

Couple (im0, im1), $\sigma = 10$

Mesure 15×15	τ	**Erreur** < 0.50	< 1.00	< 2.00
SAD	68.72	51.70	91.48	94.35
ZSAD	67.52	52.51	92.22	94.83
SSD	67.76	51.72	91.81	94.68
ZSSD	66.93	52.42	92.41	95.03
WSAD	59.06	56.57	87.73	89.60
INVAR	36.27	17.59	44.70	63.65

TAB. 3.21: *Comparaison des mesures de corrélation* 15×15*, pour différentes valeurs de* σ*. Tous les chiffres sont des pourcentages.*

Les tableaux 3.20 et 3.21 montrent une influence quasiment nulle du bruit sur la qualité des appariements. Ceci est dû à la nature très texturée de nos images : même avec un bruit de $\sigma = 10$ niveaux de gris, les masques restent très discriminants, surtout sur des fenêtres de taille élevée. Nous verrons en effet ci-dessous que le bruit n'a une influence que pour les tailles 1×1 et 3×3, mais qu'un masque 5×5 suffit à compenser ses effets.

Enfin, les invariants donnent de médiocres résultats. Ils travaillent pourtant sur une zone de taille 31×31, presque 40 fois plus grand que celui des mesures de corrélation 5×5, qui devrait être peu ambigu et très robuste. Mais ils ne sont pas adaptés à ces images très texturées, car le calcul des dérivées du signal de l'image devient difficile à mener à bien (instable). Ils se révéleraient meilleurs sur une texture moins fine, plus géométrique. Et surtout, le temps de calcul est rédhibitoire.

En conclusion provisoire, la mesure SAD semble un compromis intéressant : c'est une mesure très rapide, assez précise, qui résiste relativement bien au bruit (avec les réserves que nous avons formulées), et qui donne un grand nombre d'appariements. Nous évitons les mesures robustes et les invariants pour des raisons de coût de calcul.

Nous évitons aussi les mesures centrées, qui peuvent à tort apparier des zones très différentes : pour ZSAD, une fenêtre uniformément blanche est égale à une fenêtre uniformément noire. Ceci peut conduire à des erreurs, qui n'apparaissent pas dans nos tests, car nous n'évaluons pas l'effet des changements de luminance.

En revanche, si les images présentent de forts changements de luminance, rappelons que nous suggérons de normaliser globalement les intensités des pixels, par la formule 3.5, p. 29. De plus, si la zone de recherche est suffisamment réduite pour ne mener à aucune confusion, une mesure centrée peut se révéler profitable, en particulier pour pallier des différences locales d'intensité dues à des comportements non lambertiens (reflets). En relation avec ceci, W.K. Pratt note dans [Pra 78] qu'une normalisation locale du signal par rapport à sa moyenne (sous la forme d'un filtre lambertien) mène à des pics plus nets du signal de corrélation. La *position* des pics ne change pas, mais leur caractère plus marqué pourrait faciliter la tâche de certains algorithmes d'appariement.

Nous ne nous situons pas dans cette hypothèse, et nous choisissons donc la mesure SAD. Nous nous attachons maintenant à déterminer l'influence de la taille du masque. En théorie, un masque plus grand mène à une détection plus dense et plus correcte (plus robuste), mais moins précise (moins bien localisée). Les résultats sont donnés en tableaux 3.22 et 3.23, pour les couples (`im0`, `im1`) et (`im0`, `im3`) respectivement.

Couple (im0, im1), $\sigma = 1$

Mesure SAD	τ	**Erreur** < 0.50	< 1.00	< 2.00
1×1	62.83	3.87	7.21	11.47
3×3	49.13	40.50	55.51	58.37
5×5	58.07	54.33	81.34	83.53
7×7	64.20	54.19	87.81	90.22
9×9	67.67	51.83	89.78	92.59
11×11	69.93	49.62	90.37	93.82
13×13	71.71	47.46	90.24	94.46
15×15	72.55	45.82	89.82	94.65

Couple (im0, im1), $\sigma = 2$

Mesure SAD	τ	**Erreur** < 0.50	< 1.00	< 2.00
1×1	63.00	3.58	7.05	11.10
3×3	50.14	43.13	58.08	60.83
5×5	60.36	56.90	83.12	85.00
7×7	66.05	56.44	88.65	90.81
9×9	69.34	54.50	90.43	92.99
11×11	71.28	52.94	91.10	94.05
13×13	72.68	51.51	91.21	94.66
15×15	73.37	50.21	90.80	94.69

Couple (im0, im1), $\sigma = 5$

Mesure SAD	τ	**Erreur** < 0.50	< 1.00	< 2.00
1×1	62.50	3.04	6.29	10.30
3×3	50.09	44.10	58.45	60.93
5×5	61.53	59.60	84.68	86.41
7×7	68.20	60.15	90.60	92.40
9×9	71.54	58.39	92.27	94.38
11×11	73.46	56.66	92.39	94.91
13×13	74.24	55.13	92.13	95.02
15×15	74.68	53.37	91.43	94.88

Couple (im0, im1), $\sigma = 10$

Mesure SAD	τ	**Erreur** < 0.50	< 1.00	< 2.00
1×1	62.41	2.55	5.79	9.87
3×3	48.64	38.97	53.49	56.31
5×5	58.86	56.68	82.51	84.28
7×7	65.86	57.87	89.57	91.34
9×9	69.83	56.55	91.66	93.54
11×11	72.22	54.72	91.97	94.37
13×13	73.55	53.18	91.70	94.37
15×15	74.53	51.70	91.48	94.35

TAB. 3.22: *Influence des tailles de masque pour la mesure* SAD, *pour différentes valeurs de* σ, *sur le couple (*im0, im1*). Tous les chiffres sont des pourcentages.*

Couple (im0, im3), $\sigma = 1$

Mesure SAD	τ	Erreur < 0.50	Erreur < 1.00	Erreur < 2.00
1×1	61.91	1.86	3.97	6.84
3×3	42.70	20.59	28.60	31.09
5×5	44.77	37.85	55.19	57.53
7×7	47.65	40.85	65.06	68.29
9×9	48.74	39.87	68.46	73.04
11×11	49.64	37.65	68.32	75.15
13×13	50.29	35.27	67.01	76.39
15×15	50.10	33.62	66.21	78.06

Couple (im0, im3), $\sigma = 2$

Mesure SAD	τ	Erreur < 0.50	Erreur < 1.00	Erreur < 2.00
1×1	61.51	1.73	3.79	6.92
3×3	42.76	20.70	28.60	31.38
5×5	44.55	37.59	54.05	56.73
7×7	47.24	40.86	64.13	67.49
9×9	48.31	40.39	68.25	73.03
11×11	49.09	37.72	67.96	74.68
13×13	49.65	35.22	66.28	76.04
15×15	49.98	33.39	65.14	77.40

Couple (im0, im3), $\sigma = 5$

Mesure SAD	τ	Erreur < 0.50	Erreur < 1.00	Erreur < 2.00
1×1	61.60	1.54	3.71	6.61
3×3	42.43	20.43	28.33	31.08
5×5	44.80	38.78	55.75	57.94
7×7	48.02	42.19	66.00	68.87
9×9	49.21	41.52	69.60	73.68
11×11	50.02	39.11	69.42	75.63
13×13	50.58	36.68	68.06	77.09
15×15	50.56	35.28	67.08	78.35

Couple (im0, im3), $\sigma = 10$

Mesure SAD	τ	Erreur < 0.50	Erreur < 1.00	Erreur < 2.00
1×1	61.41	1.52	3.60	6.53
3×3	41.48	18.50	25.65	28.28
5×5	44.02	37.15	52.66	55.00
7×7	46.78	42.76	65.15	67.88
9×9	48.76	42.45	69.51	73.38
11×11	50.10	40.38	70.25	76.02
13×13	50.54	38.22	69.03	77.29
15×15	50.59	36.02	67.88	78.62

TAB. 3.23: *Influence des tailles de masque pour la mesure* SAD, *pour différentes valeurs de σ, sur le couple (im0, im3). Tous les chiffres sont des pourcentages.*

Conclusions des tableaux 3.22 et 3.23 :

- comme on pouvait s'y attendre, le bruit a bien une influence négative sur la qualité des appariements, qui peut être compensée en augmentant la taille de la fenêtre de corrélation ;

- cette influence disparaît dès la taille 5×5, ce qui explique que nous ne l'ayons pas détectée lors des expérimentations précédentes ;

- de même, une distorsion perspective forte peut être rattrapée par une grande taille de masque (les images 0 et 3 sont assez éloignées, voir figure 3.17, p.59);

- en pratique, dans nos conditions d'expérimentation, une mesure SAD 5×5 est suffisante. Sur ces images, cette taille de masque suffit à capturer des informations de texture suffisamment discriminantes, et présente pour nous un bon compromis vitesse/performance.

Comparaison des algorithmes

Nous testons maintenant les différents algorithmes d'appariement sur le couple (`im0`, `im1`), pour une mesure SAD 5×5. Certains de ces algorithmes nécessitent le réglage d'un coût d'occultation ; celui-ci peut être estimé par les appariements obtenus par l'algorithme CCR que nous avons employé jusqu'à maintenant.

Calculons la distribution des valeurs de corrélation SAD 5×5 pour tous les couples appariés par CCR. Nous obtenons la distribution montrée en figure 3.35. Sur la base de cette distribution, nous imposons que tout couple ayant une mesure de corrélation supérieure à 0.2 ne pourra pas être apparié, et ce seuil sera utilisé dans CCT. Le coût κ d'une transition d'occultation est donc fixé à 0.1 dans les algorithmes de programmation dynamique, car il faut parcourir deux transitions d'occultation pour éviter une transition d'appariement.

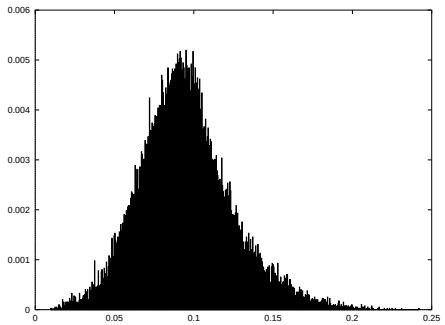
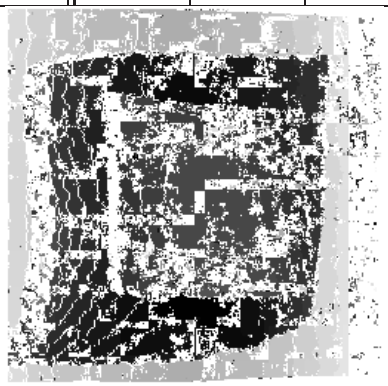


FIG. 3.35: *Distribution des mesures de corrélation des couples appariés.*

Le réglage du seuil d'appariement est un compromis entre la densité et la fiabilité attendues. Le tableau 3.24 rappelle les résultats obtenus par CCR, et le tableau 3.25 compare les résultats obtenus avec CCT à différents seuils s.

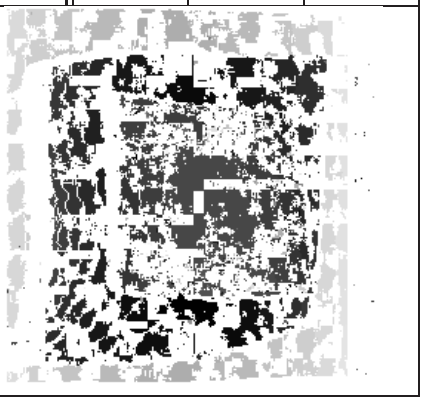
τ	**Erreur**		
	< 0.50	< 1.00	< 2.00
62.80	62.03	86.34	87.94



TAB. 3.24: *Résultats de* CCR.

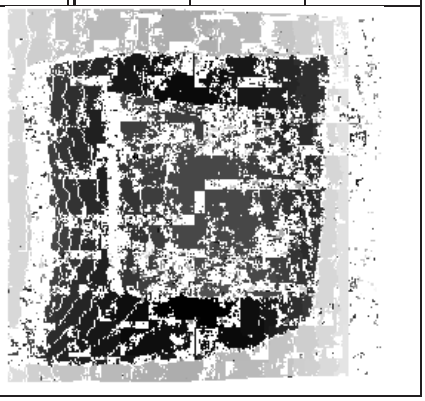
$s = 0.10$

τ	**Erreur**		
	< 0.50	< 1.00	< 2.00
38.15	80.85	93.78	94.40



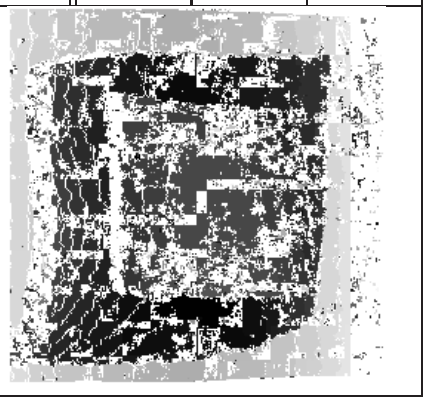
$s = 0.15$

τ	**Erreur**		
	< 0.50	< 1.00	< 2.00
59.93	64.75	88.34	89.64



$s = 0.20$

τ	**Erreur**		
	< 0.50	< 1.00	< 2.00
62.72	62.11	86.42	88.01

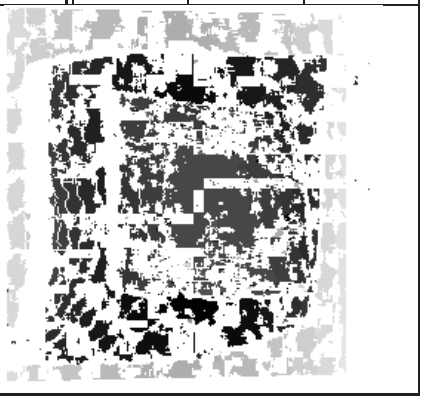


TAB. 3.25: *Influence de s sur* CCT.

Le tableau 3.26 montre la grande sensibilité de l'algorithme DP3 au réglage du coût d'occultation. Un coût un peu trop faible favorise trop les occultations, et un coût à peine plus élevé lisse trop le résultat. On remarque aussi que la minimisation globale de la programmation dynamique entraîne un lissage le long des droites épipolaires, et permet des appariements même sur les zones de l'image où les erreurs de discrétisation se sont accumulées (zones non appariées de CCR, voir discussion précédente sur la figure 3.30, p. 93).

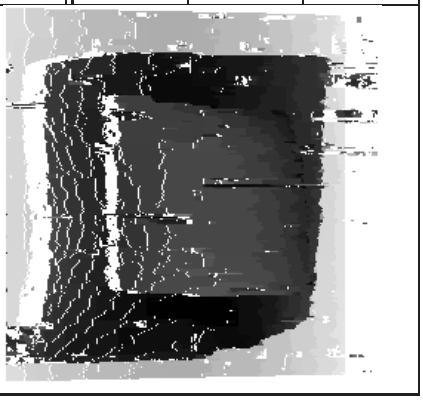
$\kappa = 0.05$

τ	Erreur		
	< 0.50	< 1.00	< 2.00
38.02	82.18	96.09	96.73



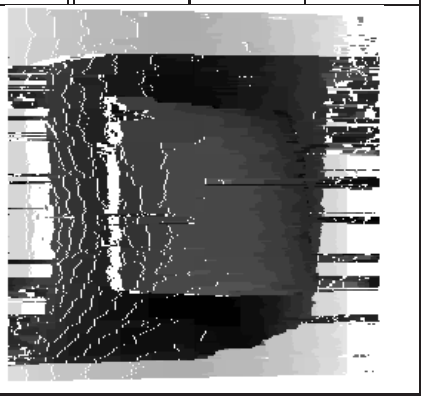
$\kappa = 0.10$

τ	Erreur		
	< 0.50	< 1.00	< 2.00
79.59	54.87	89.82	94.19



$\kappa = 0.15$

τ	Erreur		
	< 0.50	< 1.00	< 2.00
84.80	50.16	82.83	87.59

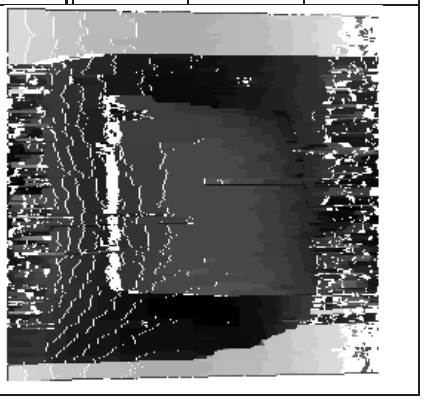


TAB. 3.26: *Influence de κ sur* DP3.

Le tableau 3.27 établit la sensibilité de DP3JC au réglage de la proportion p_{occ} d'occultations attendues. Sur la base des résultats de l'algorithme CCR, nous réglons cette proportion à 10 %, c.-à-d. nous autorisons un nombre de transitions d'occultation égal à 20 % du nombre total de transitions. Le nombre de sauts autorisés est donc $0.2(N_0 + N_1)$, où N_0 (resp. N_1) est le nombre de points sur la ligne épipolaire dans `im0` (resp. `im1`).

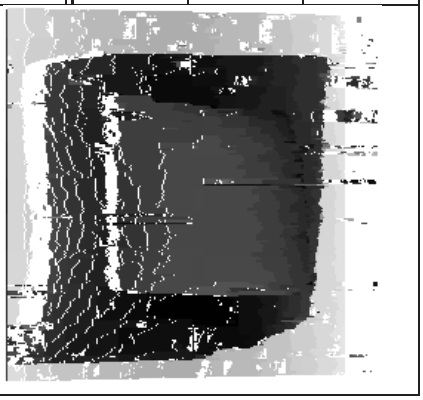
$p_{occ} = 0.05$

τ	Erreur		
	< 0.50	< 1.00	< 2.00
88.25	43.27	73.14	77.58



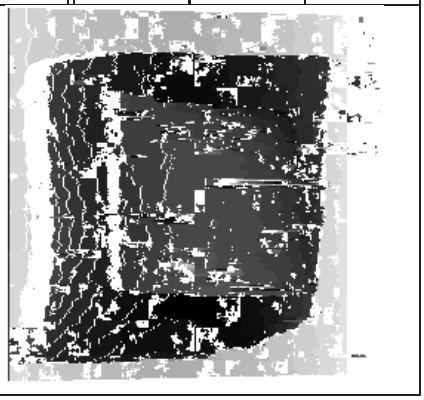
$p_{occ} = 0.10$

τ	Erreur		
	< 0.50	< 1.00	< 2.00
78.60	55.24	89.46	93.96



$p_{occ} = 0.15$

τ	Erreur		
	< 0.50	< 1.00	< 2.00
69.05	60.72	89.99	93.24



TAB. 3.27: *Influence de p_{occ} sur* DP3JC.

Enfin, le tableau 3.28 synthétise les résultats, et donne le nombre d'appariements trou-

vés dans la zone occultée (ce nombre doit donc être le plus faible possible).

Algorithme	τ	**Erreur**			**Nombre**	**Temps**	**Paramétrage**
		< 0.50	< 1.00	< 2.00	**d'app.**	**CPU**	
WTA	99.08	41.81	62.62	65.19	768	65	inutile
CCR	62.80	62.03	86.34	87.94	53	67	inutile
CCT	38.15	80.85	93.78	94.40	12	67	$s = 0.10$
CCT	62.72	62.11	86.42	88.01	53	67	$s = 0.20$
DP3NO	97.82	39.77	68.26	72.99	768	70	inutile
DP3	79.59	54.87	89.82	94.19	129	68	$\kappa = 0.10$
DP5	80.20	54.83	89.81	94.17	130	68	$\kappa = 0.10$
DP3JC	78.60	55.24	89.46	93.96	139	655	$p_{occ} = 0.10$

TAB. 3.28: *Comparaison des algorithmes. Les temps de calcul sont en secondes* CPU.

De ces comparaisons, nous tirons les observations suivantes.

- Le résultat le plus mauvais est donné par l'algorithme le plus simple WTA, qui fournit tout de même presque deux tiers d'appariements corrects à moins de 2 pixels. τ est inférieur à 100 % car pour certaines droites épipolaires dans l'image 1, la droite épipolaire conjuguée dans l'image 2 n'est pas visible.
- En termes de performances, DP3NO vient juste après. τ est encore inférieur, car certains couples d'épipolaires ne respectent pas la limite de disparité en leurs extrémités, et l'algorithme ne traite pas ce cas.
- Parmi les trois algorithmes ne nécessitant pas de paramétrage, c.-à-d. WTA, CCR et DP3NO, CCR est de loin le plus performant.
- Parmi les algorithmes nécessitant un paramétrage, les programmations dynamiques sont toutes équivalentes, et largement supérieures en densité et en précision à CCT ($s = 0.20$).
- Comme prévu, CCT ($s = 0.20$) est équivalent à CCR, puisque presque tous les scores de corrélation sont inférieurs à 0.20.
- CCT ($s = 0.10$) est l'algorithme le plus précis et le plus sûr (seulement 12 points occultés appariés à tort), mais le moins dense : il n'apparie que 38.15 % des points, mais nous verrons qu'en tenant compte des problèmes de discrétisation (en n'imposant la contrainte épipolaire qu'à 1 pixel près), il apparie plus de 50 % des points. La régularisation fera monter ce taux à plus de 80 %.
- Le temps de calcul est identique pour toutes les méthodes, sauf DP3JC, qui est 10 fois plus lent (mais ceci dépend du réglage de p_{occ}).

Nous utiliserons l'algorithme CCT de mise en correspondance, avec $s = 0.10$. Cette valeur correspond à la médiane des scores de corrélation pour des couples appariés : nous

n'accepterons donc que les appariements dont le score est dans les 50 % des meilleurs scores calculés sur ces images, pour des couples appariés en première passe par CCR. De plus, par rapport aux algorithmes de programmation dynamique, CCT a l'avantage supplémentaire de ne pas imposer la contrainte d'ordre.

Conclusion

Sur la base de ces tests, nous préconisons l'utilisation d'une mesure simple, de petite taille, donc rapide : SAD 5×5 semble parfaitement adaptée. Enfin, l'algorithme CCT d'appariement est aussi simple et rapide, pour de bons résultats ; son paramétrage s est déterminé par une première passe d'appariements CCR. Ces résultats sont liés à la nature des images, et nous ne les retrouverons pas sur les images synthétiques à textures réelles.

En vue d'obtenir des appariements multi-oculaires pour la suite de notre travail, nous calculons 5 appariements binoculaires, entre les couples successifs (`im0`, `im1`), (`im1`, `im2`), (`im2`, `im3`), (`im3`, `im4`) et (`im4`, `im5`), pour un bruit $\sigma = 0$. En effet, nous n'utilisons plus maintenant que les images non-bruitées, car nous estimons avoir démontré l'influence du bruit sur la qualité de l'appariement, et les remèdes qu'on peut y apporter (augmenter la taille du masque). Nous utilisons cette fois une recherche sur les lignes épipolaires avoisinantes (à ± 1 pixel près, comme décrit précédemment), pour des résultats plus denses.

3.5.1.5 Phase 5

Cette phase concerne la régularisation des appariements denses obtenus en phase 4. Une telle régularisation est nécessaire, car les appariements ne couvrent pas toute l'image : même avec une recherche à ± 1 pixel près, le taux τ d'appariement est de 51.26 %. La figure 3.36 montre les cartes de disparité obtenues pour les 5 couples appariés.

Nous appliquons notre algorithme de régularisation progressivement, à des fins de test, sur le couple (`im0`, `im1`). Après 1, 5, 10, 20 ou 40 itérations, nous obtenons les appariements décrits en tableau 3.29, pour l'image entière, pour le contour d'occultation, et pour la zone occultée.

Au cours de la régularisation, nous voyons que τ croît significativement (de 51 % à 82 %), et que la qualité des appariements augmente aussi : le taux d'appariements précis à 1 pixel passe de 90 % à plus de 96 %. L'amélioration est encore plus grande pour les points du contour d'occultation : on passe de 75 % à plus de 95 % de points qui se projettent à moins de 1 pixel de leur position exacte théorique (pour 20 itérations).

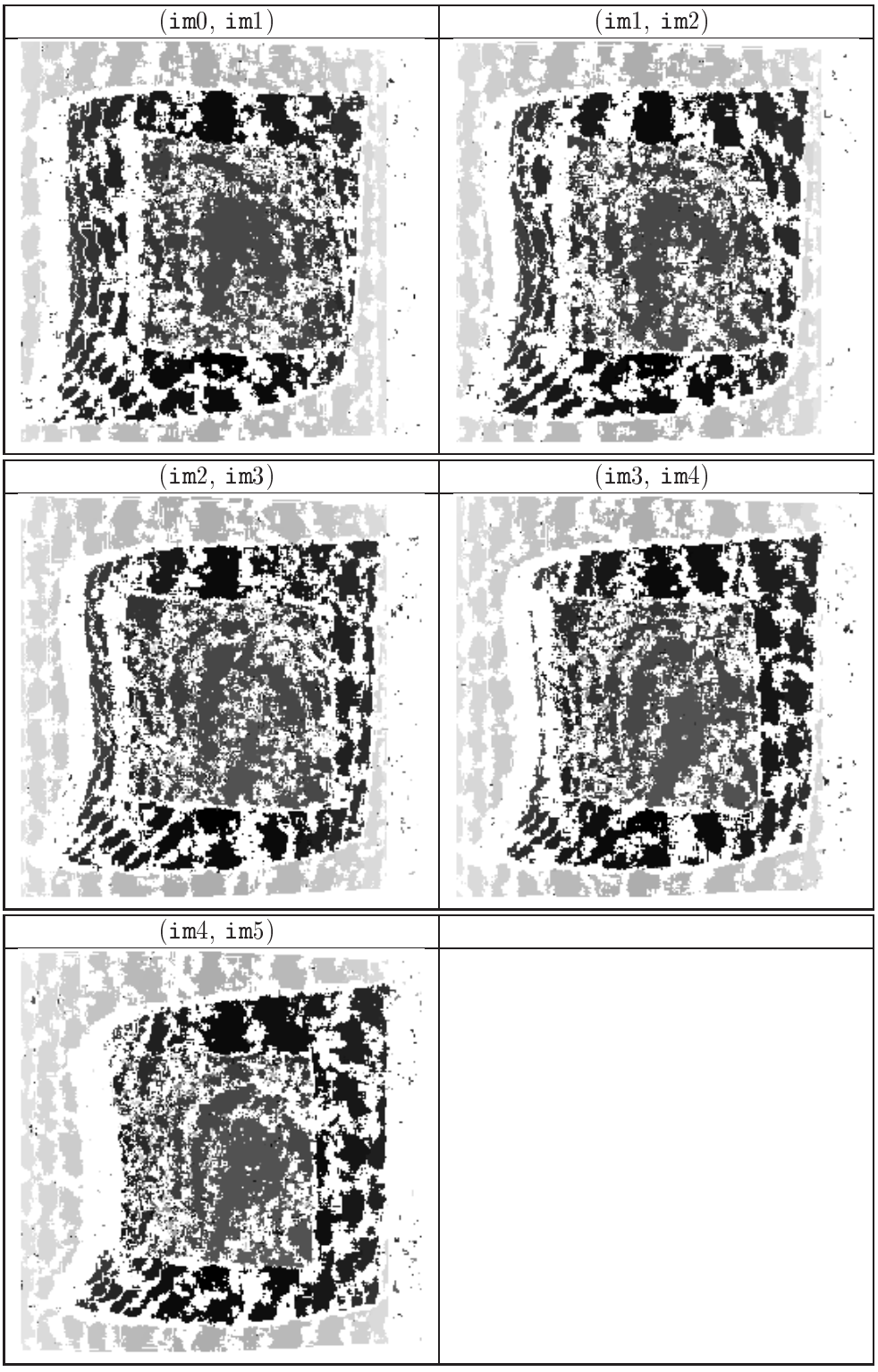


FIG. 3.36: *Les 5 cartes de disparité.*

Image entière

Nombre d'itérations	τ	**Erreur** < 0.50	< 1.00	< 2.00
0	51.26	77.60	90.36	91.06
1	57.73	80.80	94.59	95.05
5	69.65	78.68	96.95	97.22
10	76.66	74.53	96.99	97.32
20	81.08	71.99	96.89	97.34
40	82.57	70.68	96.67	97.34

Contour d'occultation

Nombre d'itérations	**Erreur** < 0.50	< 1.00	< 2.00
0	50.00	75.00	75.00
1	50.00	75.00	75.00
5	50.00	75.00	75.00
10	78.26	91.30	91.30
20	88.89	95.56	95.56
40	72.92	91.67	95.83

Zone occultée

Nombre d'itérations	**Nombre d'appariements**
0	27
1	27
5	27
10	27
20	49
40	82

TAB. 3.29: *Étapes de régularisation : progression de τ, et des points appariés à tort.*

En revanche, la précision globale des appariements est dégradée (nombre d'appariements à moins de 0.5 pixel : de 77 % à 70 %). De plus, la zone occultée, qui comportait déjà 27 pixels appariés (à tort), est progressivement comblée par l'algorithme de régularisation. Après 40 itérations, la zone occultée compte 82 pixels appariés, soit le triple. Pour cette raison, nous jugeons que 20 passes de l'algorithme seront suffisantes pour ces images, et nous appliquons ces 20 passes aux 5 listes d'appariements obtenues en phase 4.

À titre d'illustration, nous donnons en figure 3.37 les cartes de disparité successivement obtenues, où nous voyons que les zones non renseignées sont progressivement remplies. Dans notre implémentation, une passe dure approximativement 20 secondes pour ces images 256×256.

Enfin, nous testons l'algorithme de régularisation sur les appariements que nous avions obtenus par l'algorithme DP3 avec différentes valeurs de κ. Les cartes de disparité sont montrées en figure 3.38.

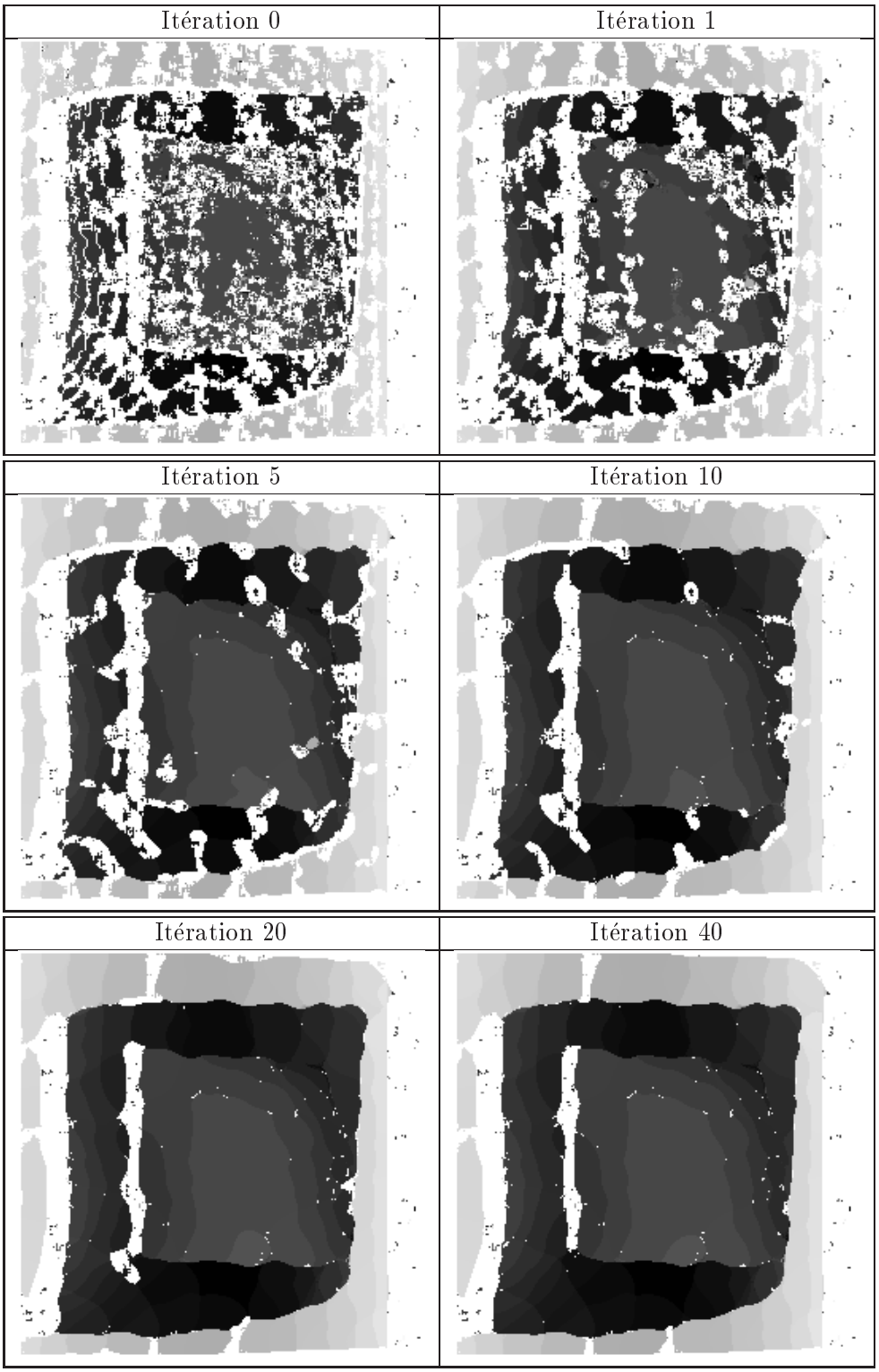


FIG. 3.37: *Les cartes de disparité du couple (*im0*,* im1*), à différents stades de régularisation.*

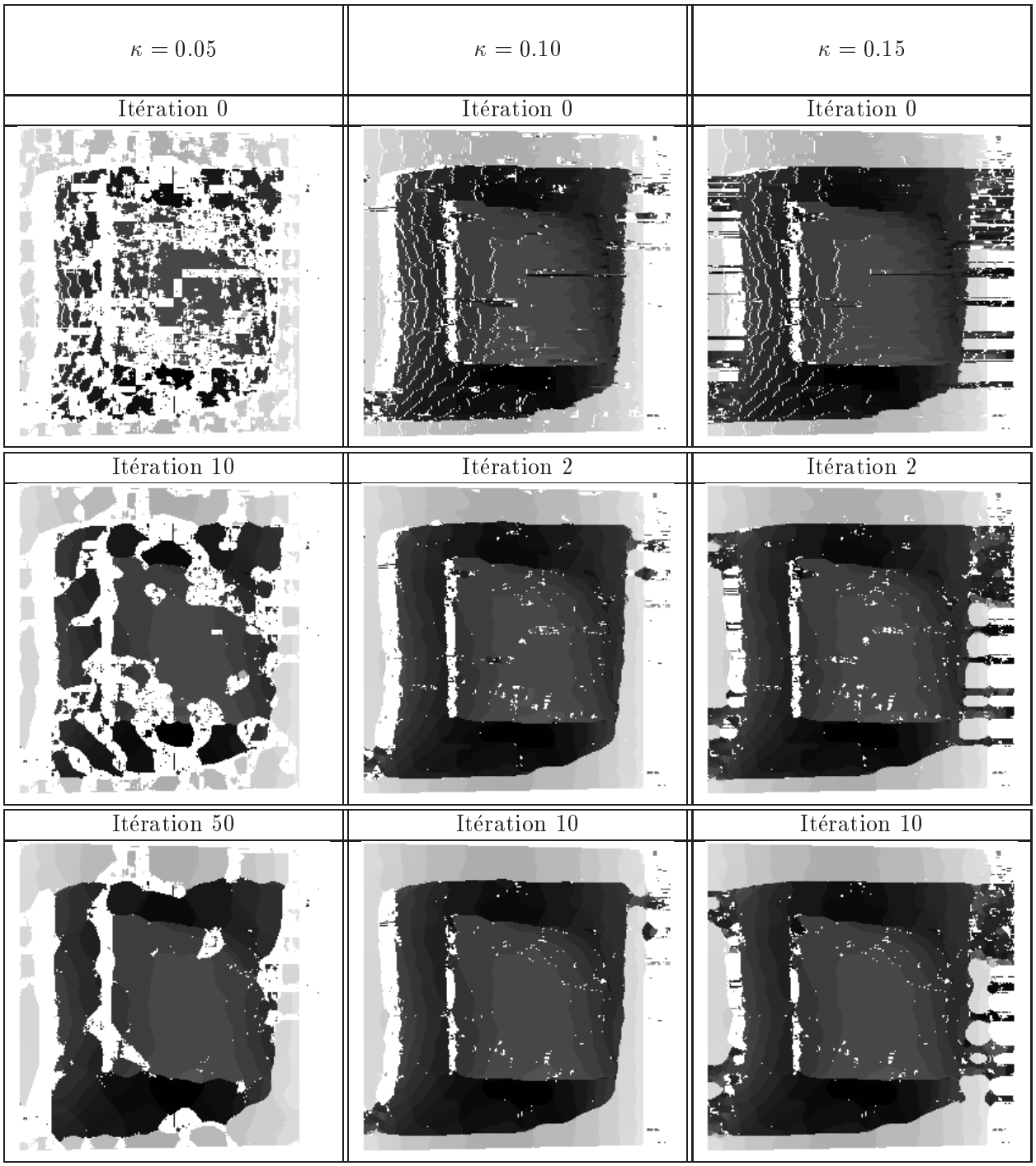


FIG. 3.38: *Influence de la régularisation sur des appariements issus d'un algorithme* DP3.

Cette figure montre qu'une assez bonne initialisation est nécessaire pour que la régularisation converge vite et bien. L'exactitude de l'initialisation conditionne l'exactitude de

la régularisation, et sa densité conditionne essentiellement la vitesse de convergence.

3.5.1.6 Phase 6

Nous disposons de 5 listes d'appariements binoculaires, entre les 5 couples d'images successifs ; nous allons dans cette étape affiner les appariements. Les méthodes d'affinage utilisables sont : AffSi, AffZd4, AffSc1, et AffSc2.

Nous fusionnons d'abord les 5 listes d'appariements binoculaires, afin de constituer une seule liste L d'appariements multi-oculaires pour l'affinage. Ceci pourrait être réalisé par un simple parcours des listes, avec comparaison des points. Dans cette liste, un appariement 6-oculaire est noté comme la liste des projections du point 3D dans les 6 images. Deux exemples d'appariement 6-oculaire sont donnés dans le tableau 3.30. Lorsqu'un point n'est pas visible dans une image (occulté, ou non détecté), ses coordonnées sont remplacées par une suite de tirets : «----------».

Point 3D n. 1	
Proj. dans im0	(123, 127)
Proj. dans im1	(203, 206)
Proj. dans im2	----------
Proj. dans im3	(48, 321)
Proj. dans im4	----------
Proj. dans im5	(222, 6)

Point 3D n. 2	
Proj. dans im0	----------
Proj. dans im1	(203, 206)
Proj. dans im2	(64, 13)
Proj. dans im3	----------
Proj. dans im4	----------
Proj. dans im5	(222, 6)

Tab. 3.30: *Deux exemples d'appariements 6-oculaires.*

Ces deux appariements correspondent au même point 3D, car ils peuvent être fusionnés en un seul (tableau 3.31) :

Proj. dans im0	(123, 127)
Proj. dans im1	(203, 206)
Proj. dans im2	(64, 13)
Proj. dans im3	(48, 321)
Proj. dans im4	----------
Proj. dans im5	(222, 6)

Tab. 3.31: *Fusion des deux appariements du tableau 3.30.*

L'algorithme est en $\mathcal{O}(n^2)$, n étant le nombre total d'appariements à fusionner. Dans notre cas, $n \simeq 150\,000$ conduit à plus de 2.10^{10} tests à effectuer, car nous avons 5 couples d'images où 30 000 points sont appariés. L'implémentation par parcours de liste n'est donc pas applicable, et il est plus efficace de stocker les points appariés dans un arbre quaternaire (*quadtree*), pour un accès rapide. L'algorithme est alors en $\mathcal{O}(n \log_4(n))$, et nous l'appliquons aux 5 listes d'appariements binoculaires denses obtenus à la phase 5,

pour obtenir la liste unique L. Nous obtenons en fin de calcul près de 100 000 appariements multi-oculaires dans L.

Pour évaluer et comparer la précision des affinages, nous devons disposer d'une mesure de qualité. Pour ces appariements multi-oculaires, nous procédons de la même façon que pour les appariements binoculaires : pour un appariement (tableau de 6 points 2D partiellement renseigné), nous choisissons le premier point défini. POV–Ray nous donne le point 3D P correspondant à ce point, nous reprojetons P dans toutes les autres images *via* les matrices de projection (connues), et nous cumulons les erreurs par rapport aux appariements trouvés. Nous donnons ensuite la distribution simplifiée de ces valeurs, dans le tableau 3.32, et les histogrammes en figure 3.39.

Appariements régularisés	**Erreur** < 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**Temps CPU**	**Remarques**
L	0.63	2.42	47.57	84.37	96.47	—	App. initiaux
L + AFFSI	2.92	10.40	69.06	90.97	96.89	7867.26	$(dx, dy) = (0.5, 0.5)$
L + AFFSI	3.39	12.12	74.18	92.32	96.84	11462.90	$(dx, dy) = (1.0, 1.0)$
L + AFFZD4	2.26	8.36	63.76	88.35	96.79	189.68	
L + AFFSC1	1.69	6.46	53.34	81.55	96.03	146.22	
L + AFFSC2	1.36	5.17	47.12	78.56	95.72	277.71	

TAB. 3.32: *Précisions comparées avant et après affinage.*

Le tableau 3.32, et surtout la figure 3.39, montrent que les affinages agissent essentiellement sur la distribution des erreurs à moins de 1 pixel, et très peu sur les erreurs plus grossières. L'affinage itératif AFFSI est le plus performant dans tous les cas, mais il est handicapé par sa lenteur : plusieurs heures CPU de calcul d'une UltraSPARC pour le traitement des 100 000 appariements. La méthode AFFSC1 donne d'assez bons résultats, mais AFFZD4 est bien le meilleur compromis. Enfin, la méthode AFFSC2 donne les moins bons résultats, et dégrade parfois les appariements initiaux. Cela est dû au fait que les scores d'appariement locaux ne correspondent pas à un paraboloïde : l'hypothèse de minimum local n'est pas respectée (voir 3.4.10).

Nous entamerons donc les procédures de reconstruction avec la liste L d'appariements 6-oculaires, affinée par la méthode non-itérative AFFZD4.

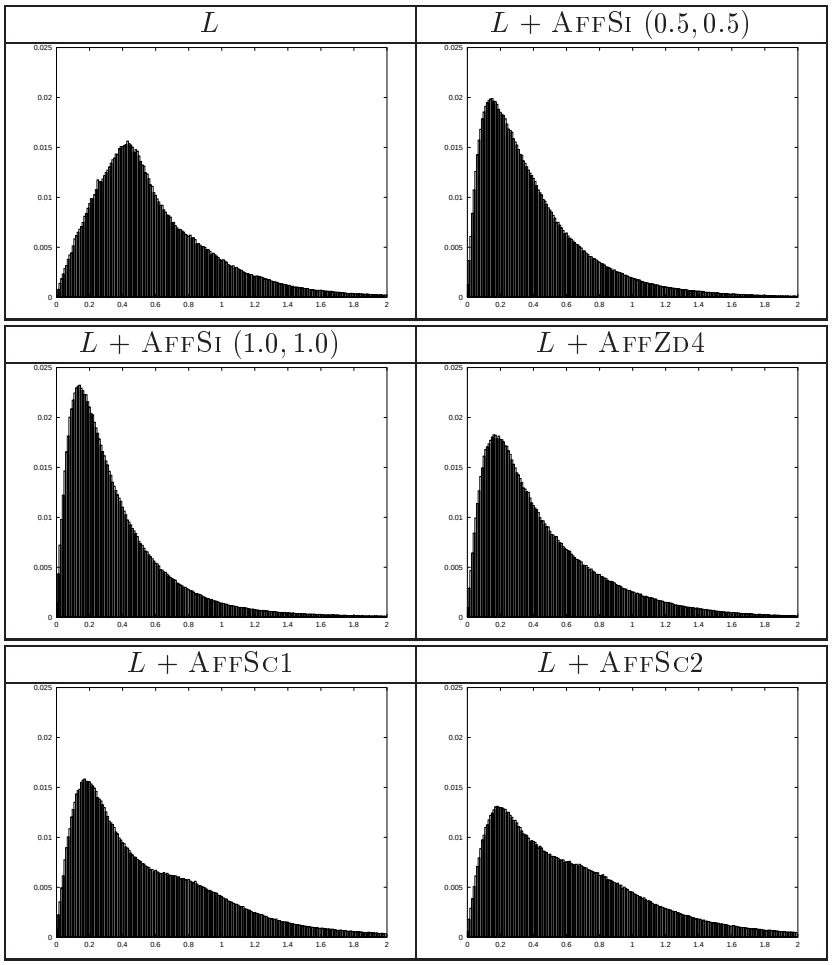


FIG. 3.39: *Histogrammes comparés de la précision en localisation des appariements, avant et après affinage.*

3.5.2 Évaluation sur images de synthèse — 2

Nous faisons maintenant subir les mêmes tests aux images synthétiques à textures réelles. Nous commençons par la texture d'extérieur, et enchaînons les phases exactement de la même façon. Nous verrons que la présence de texture a une influence considérable, dès les toutes premières étapes.

3.5.2.1 Phase 1

Nous extrayons les points d'intérêt comme précédemment. Si nous gardons le même seuil de détection, nous obtenons un nombre de points décrit dans le tableau 3.33.

Image	Nombre de points
im0	85
im1	112
im2	112
im3	148
im4	113
im5	122

TAB. 3.33: *Nombre de points d'intérêt détectés dans les 6 images synthétiques, seuil standard.*

Ce nombre de points d'intérêt peut sembler suffisant, mais les points sont mal répartis (voir figure 3.40). La texture de l'herbe est en effet trop fine et trop peu contrastée ; elle semble uniforme, et on ne peut pas y détecter de point d'intérêt. Les points d'intérêt sont presque tous détectés à la lisière du ciel, à cause du fort contraste. Cette distribution mènera à des erreurs dans l'estimation de la géométrie épipolaire de plus de 1 pixel en moyenne, même sur les couples d'images les plus proches. L'erreur maximale sera même de presque 7 pixels sur le couple (im0, im1), par le hasard d'une mauvaise configuration. Nous abaissons donc le seuil de détection des points d'intérêt d'un facteur 100, pour obtenir finalement les points listés en tableau 3.34. La distribution de ces points dans l'image est aussi montrée en figure 3.40.

Image	Nombre de points
im0	297
im1	341
im2	344
im3	375
im4	341
im5	359

TAB. 3.34: *Nombre de points d'intérêt détectés dans les 6 images synthétiques, seuil abaissé.*

Seuil standard (85 points) | Seuil abaissé (297 points)

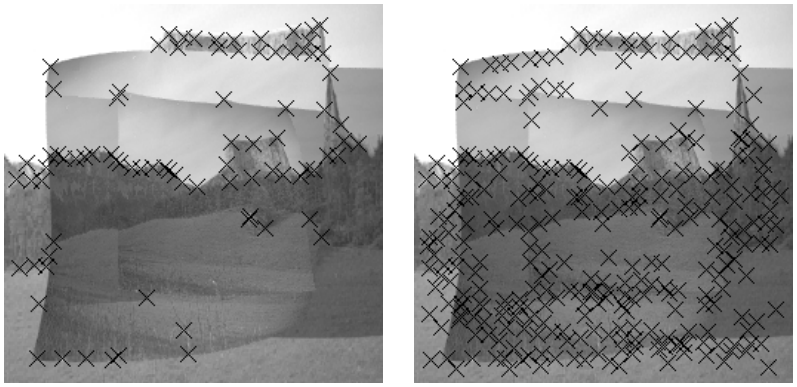


FIG. 3.40: *Les points d'intérêt détectés dans* `im0` *(croix noires).*

3.5.2.2 Phase 2

Suivant le même processus, nous apparions les points d'intérêt par un algorithme CCR d'appariement, et une mesure de corrélation SAD 15×15, sans limite de disparité.

Pour les 5 couples d'images (`im0`, `im1`) à (`im0`, `im5`), le nombre de points appariés est donné en table 3.35.

Couple	**Nombre d'appariements**
(`im0`, `im1`)	178
(`im0`, `im2`)	151
(`im0`, `im3`)	141
(`im0`, `im4`)	121
(`im0`, `im5`)	103

TAB. 3.35: *Nombre de points d'intérêt appariés dans les 5 couples.*

3.5.2.3 Phase 3

Les appariements sont affinés de la même façon que précédemment, et le même algorithme de calcul de la géométrie épipolaire donne les résultats des tableaux 3.36 et 3.37.

Couple	Nombre d'appariements en entrée	Nombre d'appariements conservés	Erreur finale moyenne	Erreur finale médiane
(im0, im1)	178	134	0.162864	0.216676
(im0, im2)	151	101	0.246110	0.389146
(im0, im3)	141	100	0.325032	0.504616
(im0, im4)	121	84	0.838693	1.146480
(im0, im5)	103	63	0.416875	0.756835

TAB. 3.36: *Résultats du calcul robuste de géométrie épipolaire.*

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	0.771554	0.717888	2.332950
$F_{0,2}$	1.317670	1.015820	7.435700
$F_{0,3}$	0.468944	0.351615	2.492980
$F_{0,4}$	6.645890	5.544250	29.376600
$F_{0,5}$	0.612647	0.584660	2.163750

TAB. 3.37: *Comparaison des matrices calculées aux matrices théoriques.*

Nous aboutissons à des erreurs plus importantes qu'avec les images à texture aléatoire, seulement exploitables si nous appliquons le principe du balayage épipolaire à ± 1 ligne près. En effet, l'erreur moyenne sur la géométrie épipolaire est de 0.77 pixels, et les droites discrètes sont calculées à 0.5 pixel près, soit une erreur cumulée de plus de 1 pixel.

Nous tentons donc maintenant d'affiner les appariements, toujours avec AFFSI 11×11, mais sur une amplitude de $(dx, dy) = (1.0, 1.0)$ au lieu de $(0.5, 0.5)$. Les nouveaux résultats sont donnés en tableaux 3.38 et 3.39.

Couple	Nombre d'appariements en entrée	Nombre d'appariements conservés	Erreur finale moyenne	Erreur finale médiane
(im0, im1)	178	137	0.144917	0.172472
(im0, im2)	151	108	0.227141	0.360395
(im0, im3)	141	94	0.253454	0.465598
(im0, im4)	121	79	0.553163	1.070410
(im0, im5)	103	66	0.389008	0.793369

TAB. 3.38: *Résultats du calcul robuste de géométrie épipolaire.*

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	0.263910	0.200306	1.550670
$F_{0,2}$	0.279091	0.259234	0.903362
$F_{0,3}$	1.275870	0.875551	7.873180
$F_{0,4}$	0.333927	0.270907	1.668730
$F_{0,5}$	1.301410	1.217400	4.393080

TAB. 3.39: *Comparaison des matrices calculées aux matrices théoriques.*

Les résultats sont plus facilement exploitables, puisque l'erreur est en moyenne de 0.26 pixel pour le couple le plus rapproché (`im0`, `im1`). Ceci confirme la grande instabilité du calcul de la matrice fondamentale, malgré l'utilisation de techniques robustes, et de la normalisation.

Constatons une fois de plus que les erreurs (moyenne ou médiane) du calcul robuste de matrice fondamentale ne sont en aucun cas une indication de la qualité de la matrice. En effet, pour le couple (`im0`, `im4`), l'erreur médiane de calcul est semblable avant et après modification de l'affinage (tableau 3.38 par rapport au tableau 3.36), alors que l'erreur sur le résultat change de façon spectaculaire : de 6.6 pixels en moyenne, à 0.3 pixel (tableau 3.39 par rapport au tableau 3.37).

3.5.2.4 Phase 4

Nous testons l'appariement dense sur le couple (`im0`, `im1`), avec les mesures classiques SAD, ZSAD, SSD, ZSSD, WSAD et INVAR et un algorithme CCR d'appariement. Le tableau 3.40 présente les résultats pour les mesures 5×5, et le tableau 3.41 pour les mesures 15×15.

Mesure 5×5	τ	Erreur (proportion des appariements trouvés) < 0.05	< 0.10	< 0.50	< 1.00	< 2.00	Temps CPU
SAD	64.28	0.94	3.31	46.27	72.51	78.92	100
ZSAD	55.68	1.02	3.62	44.51	65.45	69.89	111
SSD	60.41	0.96	3.34	43.59	69.33	76.40	134
ZSSD	52.56	1.08	3.72	43.24	63.12	67.67	148
WSAD	60.63	1.00	3.50	44.51	69.77	76.30	180
INVAR	45.90	0.72	2.36	22.12	47.39	68.12	1359

TAB. 3.40: *Comparaison des mesures de corrélation* 5×5.

Mesure 15×15	τ	**Erreur** (proportion des appariements trouvés) < 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**Temps** CPU
SAD	71.29	0.87	2.92	44.47	75.66	82.34	100
ZSAD	67.89	0.88	2.94	43.22	72.53	79.07	120
SSD	68.73	0.84	2.85	40.79	70.79	78.09	247
ZSSD	66.80	0.87	2.91	40.82	69.29	76.20	280
WSAD	65.92	0.98	3.37	48.11	76.72	83.34	427
INVAR	45.90	0.72	2.36	22.12	47.39	68.12	696

TAB. 3.41: *Comparaison des mesures de corrélation* 15×15.

Les conclusions sont sans surprise :

- les mesures non-centrées sont meilleures que les mesures centrées ;
- plus le masque est grand, meilleur est l'appariement (quantité et précision) ;

La mesure INVAR est mieux adaptée à ces images où des contours francs existent : les valeurs des dérivées sont plus significatives que dans le cas de textures très fines ou aléatoires. La performance de cette mesure est équivalente à ZSSD 5×5. Nous confirmons ainsi que la mesure INVAR ne doit être utilisée qu'en première passe d'appariement, afin d'orienter correctement les images ; une fois la rotation globale annulée, les corrélations classiques doivent être utilisées.

Nous retenons encore la mesure SAD, et nous testons l'influence de sa taille dans le tableau 3.42.

Mesure SAD	τ	**Erreur** < 0.50	< 1.00	< 2.00
1×1	85.18	8.40	15.63	22.61
3×3	58.69	39.15	61.35	68.34
5×5	64.28	46.27	72.51	78.92
7×7	68.00	48.56	76.05	82.16
9×9	70.03	47.95	76.71	82.92
11×11	70.93	46.73	76.67	83.00
13×13	71.30	45.60	76.49	82.93
15×15	71.29	44.47	75.66	82.34

TAB. 3.42: *Comparaison des tailles de masque pour la mesure* SAD.

Ici encore, la mesure SAD 5×5 semble convenir, sur des critères à la fois de performance, et de rapidité. Après cette phase d'appariement sur les images (`im0`, `im1`), réalisé avec l'algorithme CCR, nous pouvons calculer les scores de corrélation des couples appariés. Leur distribution est montrée en figure 3.41.

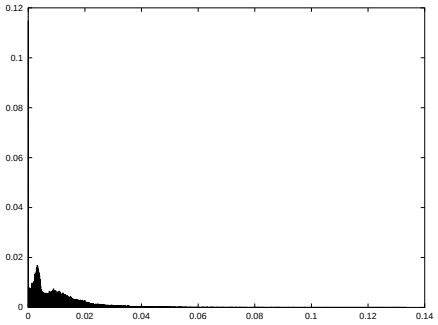


FIG. 3.41: *Distribution des mesures de corrélation des couples appariés. Il existe un pic très important en 0, confondu avec l'axe des ordonnées, et montant à une valeur de* 0.12 *(invisible).*

Nous voyons que cet histogramme présente un pic très important en 0, et ceci est dû à la texture des images : tous les masques appartenant au ciel sont de couleur uniforme, et leur corrélation est presque nulle. Aussi, déduire de cette distribution un seuil s d'appariement pour une seconde passe CCT n'est pas très significatif : si nous fixons s à la valeur médiane, nous n'apparierons que des points du ciel ; si nous augmentons s, nous ne sommes plus certains de ne garder que de bons appariements.

Pour ces images, nous décidons de réaliser une seule passe d'appariement dense par un algorithme CCR, sur les couples successifs (`im0`, `im1`), (`im1`, `im2`), (`im2`, `im3`), (`im3`, `im4`) et (`im4`, `im5`), et avec contrainte épipolaire à ± 1 pixel près.

3.5.2.5 Phase 5

Le taux d'appariement obtenu en fin de phase 4 est maintenant de l'ordre de 75 %. Nous devons à nouveau effectuer une régularisation, mais moins de passes seront nécessaires. Les cartes de disparité sont en figure 3.42.

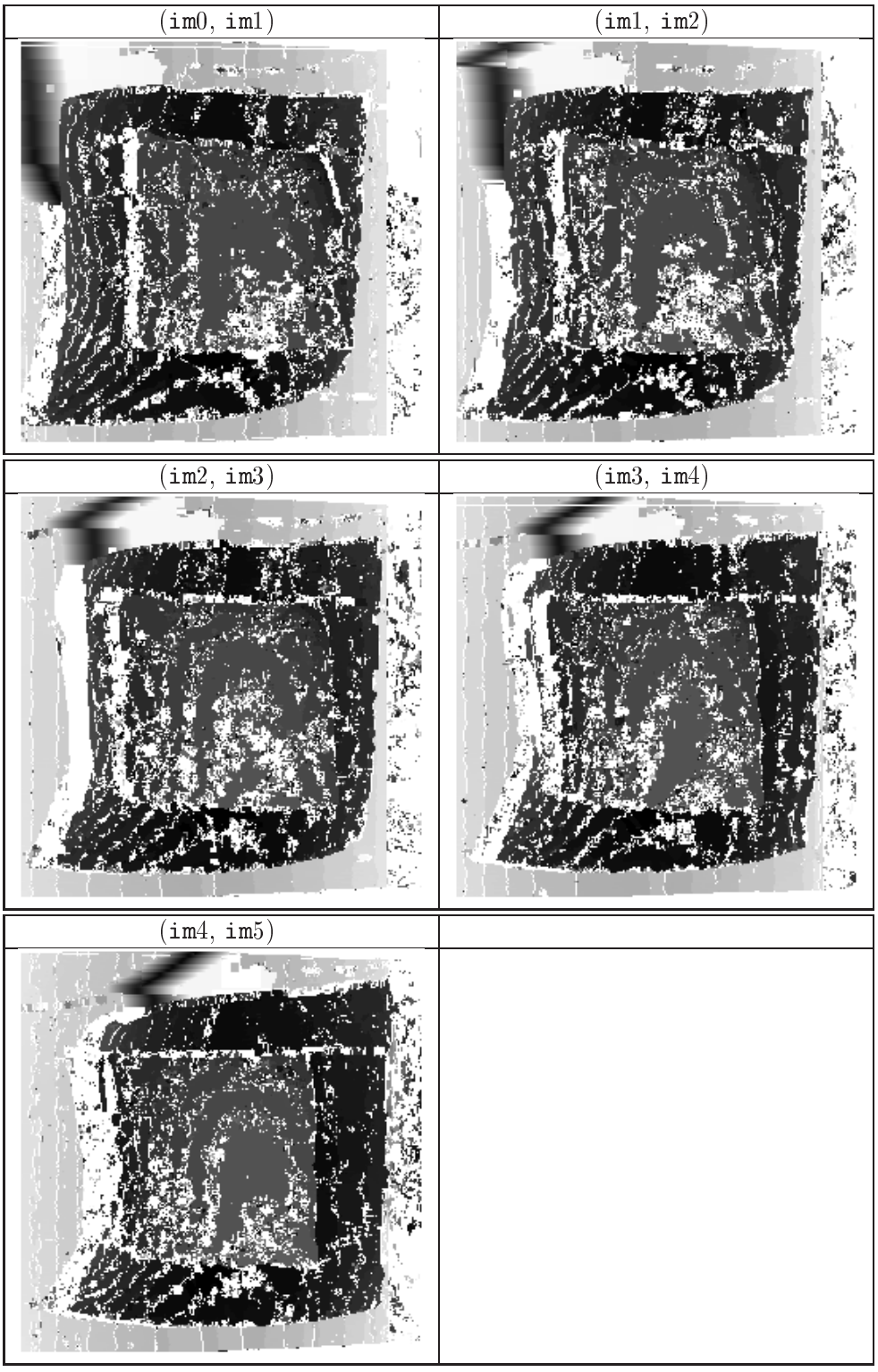


FIG. 3.42: *Les 5 cartes de disparité.*

Les effets des itérations successives de régularisation sont décrits en tableau 3.43.

Image entière

Nombre d'itérations	τ	**Erreur** < 0.50	< 1.00	< 2.00
0	75.53	52.38	74.13	79.58
1	84.57	58.42	80.56	84.37
2	86.23	60.31	82.63	85.33
3	87.09	61.15	83.56	85.70
4	87.64	61.49	84.11	85.88
5	88.08	61.68	84.33	85.86

Contour d'occultation

Nombre d'itérations	**Erreur** < 0.50	< 1.00	< 2.00
0	20.00	26.67	26.67
1	35.29	41.18	44.12
2	45.95	51.35	56.76
3	67.50	82.50	82.50
4	73.08	90.38	90.38
5	75.44	94.74	94.74

Zone occultée

Nombre d'itérations	**Nombre d'appariements**
0	158
1	163
2	186
3	213
4	260
5	298

TAB. 3.43: *Étapes de régularisation : progression de τ, et des points appariés à tort.*

La figure 3.43 montre les cartes de disparité successivement obtenues pour le couple (`im0`, `im1`). En 5 passes, le taux d'appariement τ passe de 75 % à 80 %. Trois passes mènent à une bonne régularisation du contour d'appariement, mais à une dégradation significative de la définition de la zone occultée : le nombre d'appariements à tort passe dans cette zone de 158 à 213, soient 35 % d'augmentation. Aussi, nous décidons de n'effectuer que 2 passes de régularisation, sur les 5 listes d'appariements.

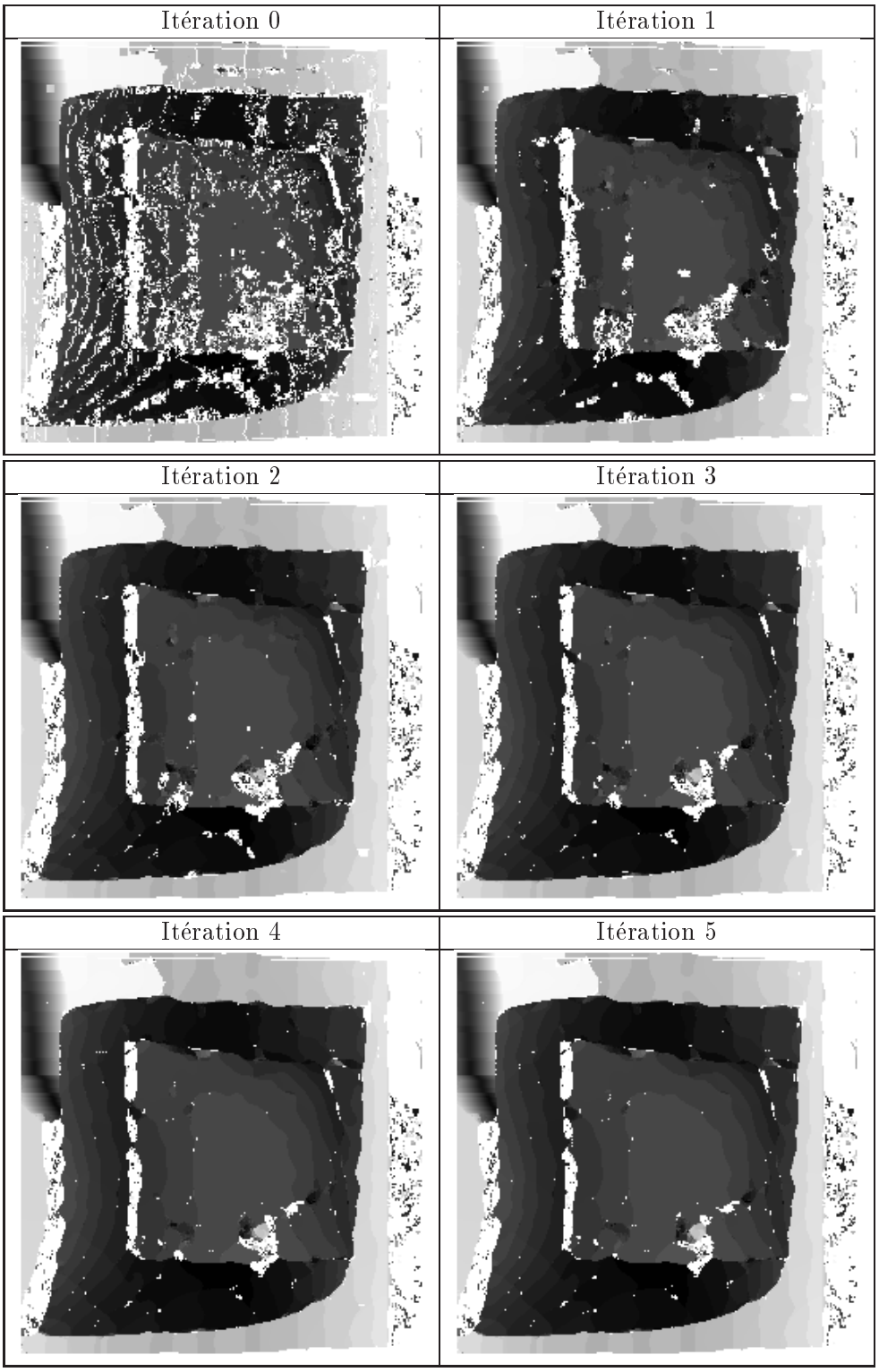


FIG. 3.43: *Les cartes de disparité du couple (*im0*,* im1*), à différents stades de régularisation.*

3.5.2.6 Phase 6

Nous fusionnons et affinons les appariements obtenus. Les résultats sont présentés en tableau 3.44 et en figure 3.44.

Appariements régularisés	**Erreur** < 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**Temps CPU**	**Remarques**
L	0.63	2.40	43.75	77.42	87.84	—	App. initiaux
L + AffSi	1.49	5.21	48.17	76.28	87.96	8251.48	$(dx, dy) = (0.5, 0.5)$
L + AffSi	1.46	5.06	45.19	72.77	87.09	12428.40	$(dx, dy) = (1.0, 1.0)$
L + AffZd4	0.93	3.53	44.94	76.20	87.73	209.51	
L + AffSc1	0.92	3.43	42.79	73.80	87.56	156.14	
L + AffSc2	0.74	2.82	38.86	71.59	87.31	297.68	

Tab. 3.44: *Précisions comparées avant et après affinage.*

Le gain est moins flagrant qu'avec la texture aléatoire : si les variations locales du signal-image ne sont pas assez fortes, les méthodes d'affinage, qui sont basées sur une recherche locale de minimum, ne sont pas très bien conditionnées. On constate tout de même une amélioration de l'erreur médiane, qui passe de 0.4 pixel pour les appariements initiaux, à 0.2 pixel pour les appariements affinés. Les performances comparées des algorithmes sont les mêmes que précédemment : AffSi est le meilleur (surtout en déplacement $(1.0, 1.0)$), suivi de AffZd4 (presque équivalent), puis AffSc1, puis AffSc2 (qui dégrade les résultats initiaux, pour les mêmes raisons).

L'algorithme AffZd4 sera donc appliqué à ces appariements, pour la suite de notre processus.

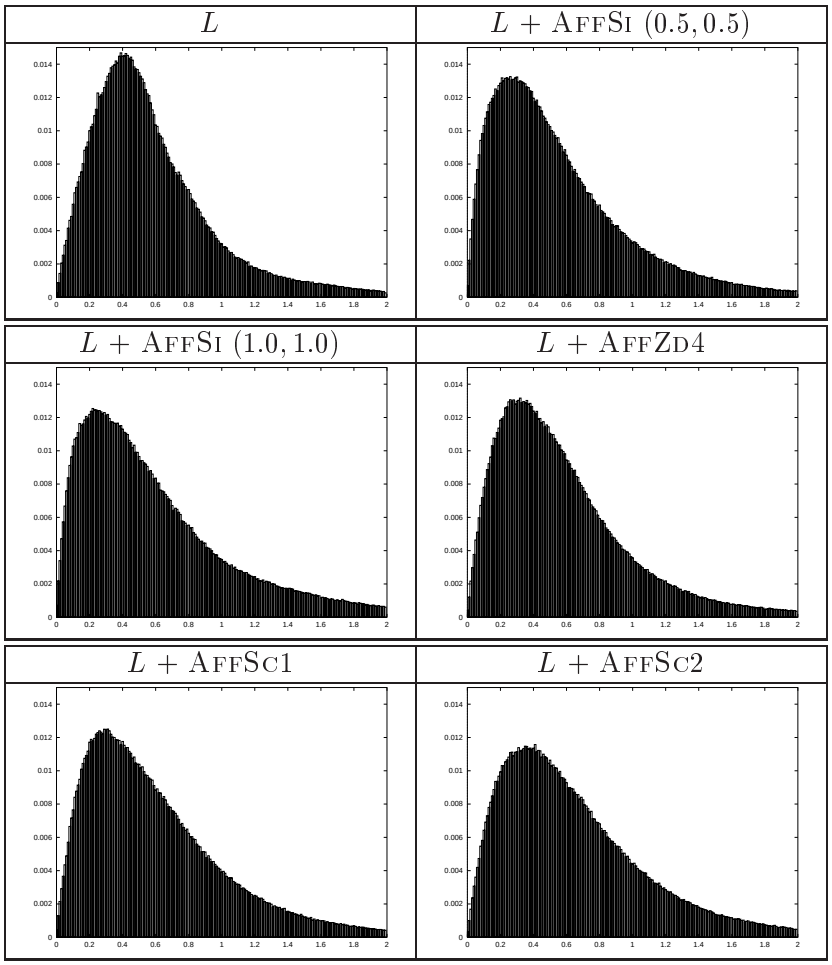


FIG. 3.44: *Histogrammes comparés de la précision en localisation des appariements, avant et après affinage.*

3.5.3 Évaluation sur images de synthèse — 3

Nous traitons ici la scène synthétique, avec texture d'intérieur.

3.5.3.1 Phase 1

Avec un seuil de détection standard, nous obtenons de l'ordre de 150 points d'intérêt dans chaque image. Cela semble suffisant, d'autant plus qu'ils sont relativement bien répartis. Il serait de toute façon très difficile d'obtenir plus de points d'intérêt, étant donnée l'absence de texture des images. Le tableau 3.45 et la figure 3.45 indiquent le nombre et la répartition des points détectés.

Image	Nombre de points
im0	167
im1	169
im2	157
im3	152
im4	143
im5	131

TAB. 3.45: *Nombre de points d'intérêt détectés dans les 6 images synthétiques, seuil standard.*

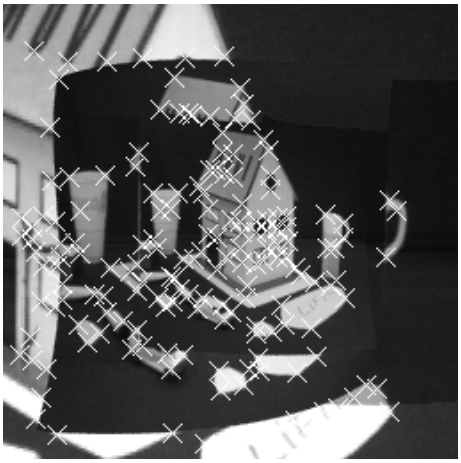


FIG. 3.45: *Les 167 points d'intérêt détectés dans* im0 *(croix blanches).*

3.5.3.2 Phase 2

Nous apparions les points comme précédemment. Le nombre de points appariés est donné dans le tableau 3.46.

Couple	Nombre d'appariements
(im0, im1)	131
(im0, im2)	116
(im0, im3)	100
(im0, im4)	95
(im0, im5)	84

TAB. 3.46: *Nombre de points d'intérêt appariés dans les 5 couples.*

3.5.3.3 Phase 3

Après affinage, le calcul de la géométrie épipolaire donne les résultats des tableaux 3.47 et 3.48. Comme pour la texture précédente, l'amplitude de l'affinage doit être fixée à $(dx, dy) = (1.0, 1.0)$ pour obtenir des résultats corrects.

Couple	Nombre d'appariements en entrée	Nombre d'appariements conservés	Erreur finale moyenne	Erreur finale médiane
(im0, im1)	131	99	0.056531	0.062850
(im0, im2)	116	87	0.105911	0.136348
(im0, im3)	100	67	0.086744	0.136132
(im0, im4)	95	63	0.127917	0.180506
(im0, im5)	84	59	0.154718	0.210121

TAB. 3.47: *Résultats du calcul robuste de géométrie épipolaire.*

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$F_{0,1}$	1.121410	0.919688	5.649130
$F_{0,2}$	1.156520	1.032880	4.604430
$F_{0,3}$	1.982020	1.391440	12.176200
$F_{0,4}$	0.687581	0.535718	4.267610
$F_{0,5}$	1.274900	1.067680	5.936510

TAB. 3.48: *Comparaison des matrices calculées aux matrices théoriques.*

Les erreurs sont beaucoup plus fortes que dans le cas de la texture d'extérieur, puisque nous aboutissons à une erreur moyenne de plus de 1 pixel pour le couple (im0, im1).

Nous continuons nos expérimentations sur ces données incorrectes, simulant le déroulement d'un processus «aveugle» menant à la synthèse de nouvelles vues. Nous utiliserons le balayage épipolaire à ± 1 ligne près.

3.5.3.4 Phase 4

Nous testons de nouveau l'appariement dense sur le couple (im0, im1), avec les mesures classiques SAD, ZSAD, SSD, ZSSD, WSAD et INVAR et un algorithme CCR d'appariement. Grâce au parcours épipolaire à ± 1 ligne, les taux d'appariements seront artificiellement améliorés. Le tableau 3.49 présente les résultats pour les mesures 5×5, et le tableau 3.50 pour les mesures 15×15.

Mesure 5×5	τ	**Erreur** (proportion des appariements trouvés) < 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**Temps CPU**
SAD	73.50	1.03	3.88	51.52	74.92	84.00	100
ZSAD	70.55	1.01	3.79	45.07	65.68	74.85	110
SSD	71.19	1.01	3.76	45.97	69.54	80.66	135
ZSSD	67.85	1.03	3.83	43.41	63.65	73.14	147
WSAD	71.40	1.05	3.95	47.76	71.06	80.93	175
INVAR	63.10	0.75	2.85	28.95	49.61	72.72	1002

TAB. 3.49: *Comparaison des mesures de corrélation* 5×5.

Mesure 15×15	τ	**Erreur** (proportion des appariements trouvés) < 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**Temps CPU**
SAD	67.65	0.95	3.52	56.04	81.17	87.94	100
ZSAD	66.81	0.94	3.44	51.86	76.74	84.80	118
SSD	66.19	0.95	3.49	50.54	75.91	84.87	251
ZSSD	66.36	0.95	3.47	49.64	74.45	83.18	280
WSAD	66.75	1.02	3.80	55.77	80.68	88.65	413
INVAR	63.10	0.75	2.85	28.95	49.61	72.72	541

TAB. 3.50: *Comparaison des mesures de corrélation* 15×15.

Comme précédemment, et pour les mêmes raisons, nous choisissons d'utiliser la mesure SAD. Les conclusions sur les autres mesures ne changent pas.

Le tableau 3.51 établit l'influence de la taille du masque.

Mesure SAD	τ	**Erreur** < 0.50	< 1.00	< 2.00
1×1	92.58	7.84	14.79	22.70
3×3	72.67	39.94	61.09	71.99
5×5	73.49	51.51	74.94	84.01
7×7	74.81	56.58	80.43	88.15
9×9	74.98	57.88	82.35	89.39
11×11	74.65	57.91	82.74	89.31
13×13	74.13	57.10	82.21	88.68
15×15	73.39	55.99	81.17	87.95

TAB. 3.51: *Comparaison des tailles de masque pour la mesure* SAD.

Encore une fois, nous calculerons donc un appariement dense des couples (`im0`, `im1`), (`im1`, `im2`), (`im2`, `im3`), (`im3`, `im4`) et (`im4`, `im5`) par un algorithme CCR, opérant sur une mesure SAD 5×5, avec contrainte épipolaire à ± 1 pixel près. L'algorithme CCR est choisi pour les même raisons que dans le cas précédent : les images contenant de grands aplats de

couleur, de très nombreuses mesures de corrélation sont presque nulles, et il est impossible de fixer un seuil s pour CCT à partir de la simple observation de cette distribution.

3.5.3.5 Phase 5

Pour cette texture, le taux d'appariement obtenu en fin de phase 4 est de moins de 73 %. Les cartes de disparité sont présentées en figure 3.46.

Les effets des itérations successives de régularisation sont décrits en tableau 3.52, et en figure 3.47. Comme précédemment, afin d'établir un compromis entre le taux d'appariement et le nombre de points appariés à tort dans la zone occultée, nous décidons de n'effectuer que 2 passes de régularisation, sur les 5 listes d'appariements.

Image entière

Nombre d'itérations	τ	**Erreur**		
		< 0.50	< 1.00	< 2.00
0	72.92	50.55	73.54	82.73
1	82.44	56.39	80.79	87.98
2	84.06	58.70	83.34	89.44
3	84.81	59.62	84.58	89.83
4	85.28	60.10	85.14	89.98
5	85.55	60.45	85.61	90.07

Contour d'occultation

Nombre d'itérations	**Erreur**		
	< 0.50	< 1.00	< 2.00
0	36.51	36.51	47.62
1	53.42	53.42	57.53
2	56.25	58.75	61.25
3	52.69	54.84	58.06
4	49.53	58.88	61.68
5	48.62	61.47	66.97

Zone occultée

Nombre d'itérations	**Nombre d'appariements**
0	217
1	233
2	251
3	276
4	304
5	318

TAB. 3.52: *Étapes de régularisation : progression de τ, et des points appariés à tort.*

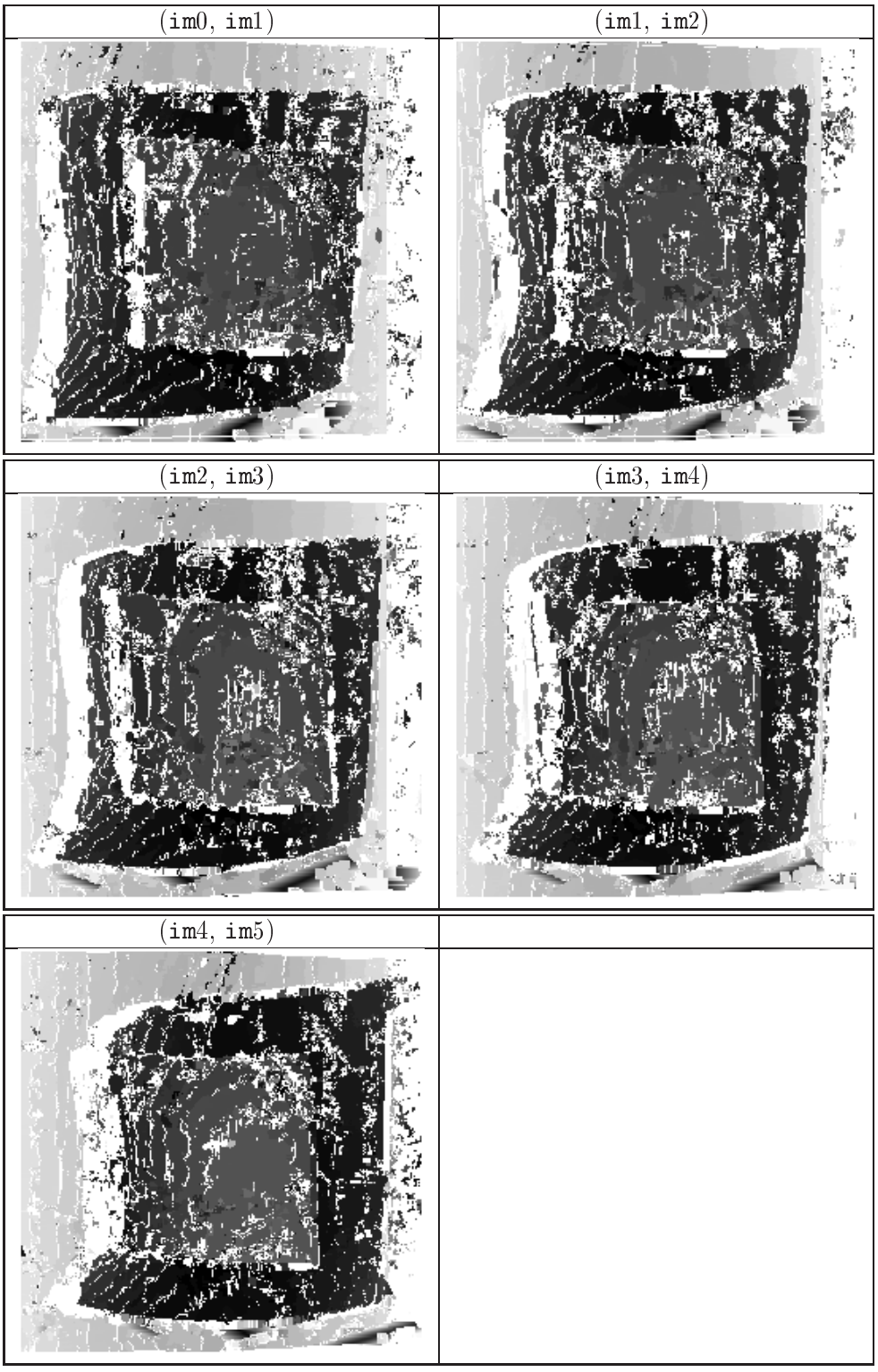


FIG. 3.46: *Les 5 cartes de disparité.*

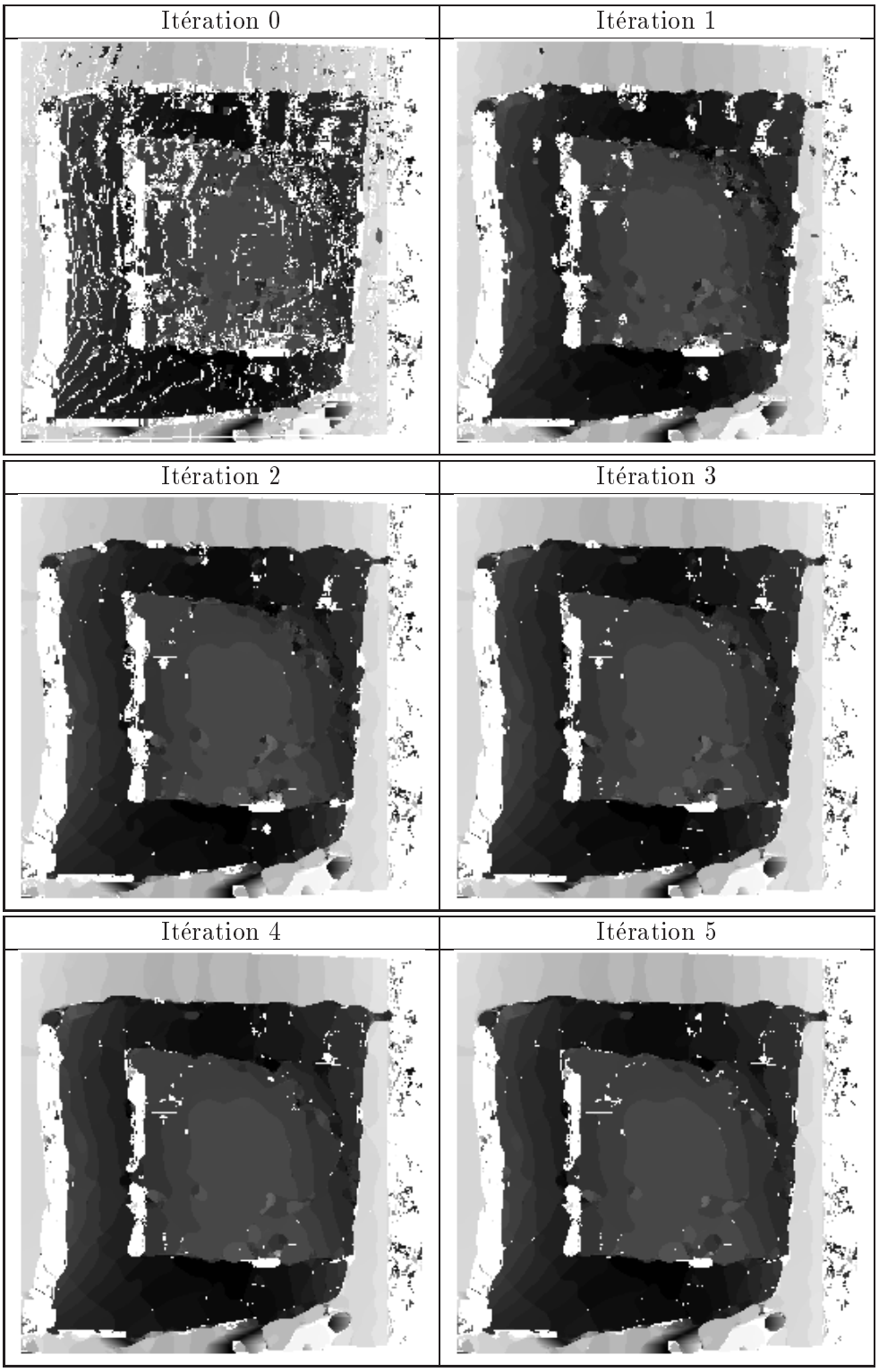


FIG. 3.47: *Les cartes de disparité du couple (*im0*,* im1*), à différents stades de régularisation.*

3.5.3.6 Phase 6

Nous fusionnons et affinons les appariements obtenus. Les résultats sont présentés en tableau 3.53 et en figure 3.48.

Appariements régularisés	Erreur					Temps CPU	Remarques
	< 0.05	< 0.10	< 0.50	< 1.00	< 2.00		
L	0.62	2.40	42.54	75.55	88.36	—	App. initiaux
L + AffSi	1.48	5.16	47.61	75.80	88.36	8145.36	$(dx, dy) = (0.5, 0.5)$
L + AffSi	1.59	5.45	47.11	74.49	87.83	12014.30	$(dx, dy) = (1.0, 1.0)$
L + AffZd4	1.38	4.85	46.94	75.63	88.31	200.31	
L + AffSc1	0.83	3.20	40.04	71.65	87.93	149.79	
L + AffSc2	0.70	2.67	37.02	69.59	87.60	287.14	

TAB. 3.53: *Précisions comparées avant et après affinage.*

Les courbes sont semblables à celles des textures d'extérieur, et nous en tirons les mêmes conclusions, en choisissant d'appliquer AffZd4 à notre liste d'appariements multi-oculaires.

3.5.4 Évaluation sur images réelles

3.5.4.1 Séquence 1

Nous testons la chaîne d'appariement sur la séquence montrée en figure 3.49. Cette séquence, fournie par Carnegie Mellon University, compte 4 images et leurs matrices de projection (4 images étalonnées).

Comme nous disposons des matrices de projection, nous n'effectuons pas le calcul de la géométrie épipolaire à partir des images, mais directement à partir des matrices. En effet, connaissant p. ex. les matrices M_1 et M_2 de projection dans les images `im1` et `im2`, nous pouvons calculer la matrice fondamentale $F_{1,2}$ décrivant la géométrie épipolaire entre ces deux images par l'équation 3.60 [Fau 92].

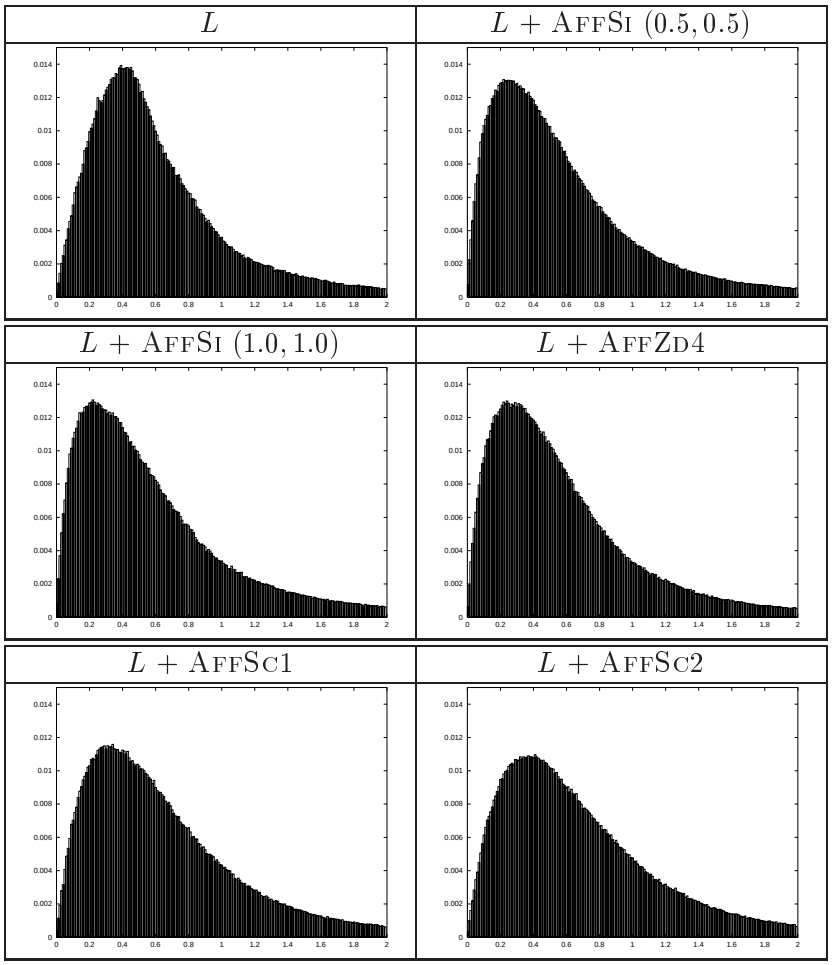


FIG. 3.48: *Histogrammes comparés de la précision en localisation des appariements, avant et après affinage.*

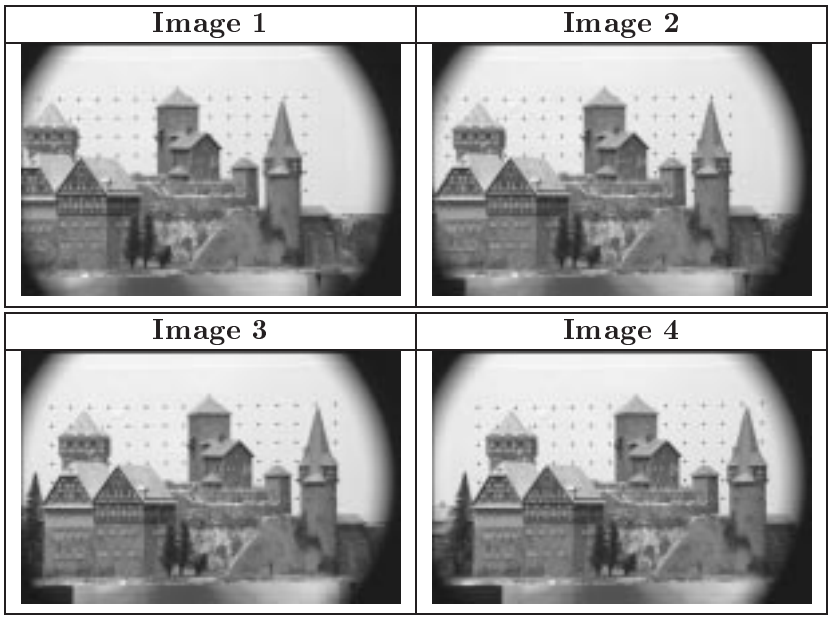


FIG. 3.49: *Séquence étalonnée de* CMU.

$$\left\{\begin{array}{lcl} M_1 & = & \left(\boxed{M'_1} \,\middle\|\, \boxed{v_1}\right) \\ M_2 & = & \left(\boxed{M'_2} \,\middle\|\, \boxed{v_2}\right) \\ F_{1,2} & = & [v_2 - M'_2 M_1'^{-1} v_1]_\times M'_2 M_1'^{-1} \end{array}\right. \tag{3.60}$$

Nous calculons donc les matrices fondamentales $F_{1,2}$, $F_{2,3}$, et $F_{3,4}$, puis un appariement dense des points des couples (`im1`, `im2`), (`im2`, `im3`) et (`im3`, `im4`). Les cartes de disparité obtenues sont montrées en figure 3.50 ; l'algorithme employé est un CCR, avec une mesure SAD 11×11, et une limite de disparité de 40 pixels.

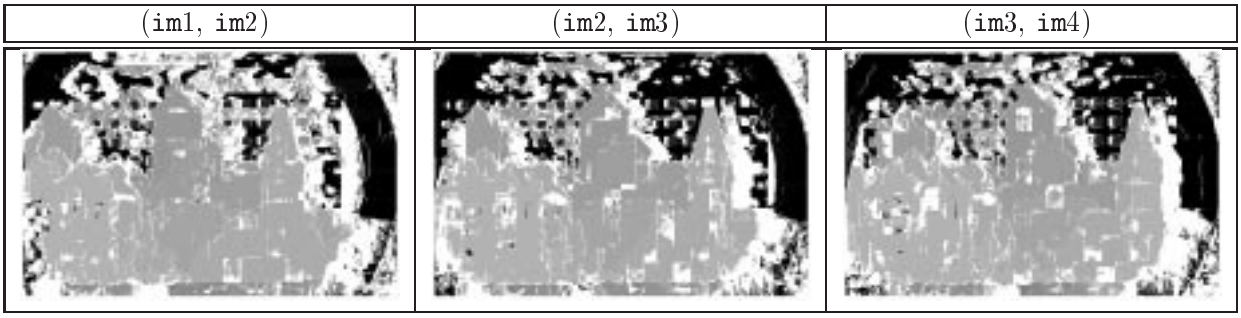


FIG. 3.50: *Cartes de disparité obtenues pour la séquence de la figure 3.49.*

Ces appariements sont régularisés en une passe et fusionnés, pour obtenir les appariements représentés par les cartes de disparité en figure 3.51. Ils sont enfin affinés par AFFZD4, pour donner les cartes de disparité de la figure 3.52.

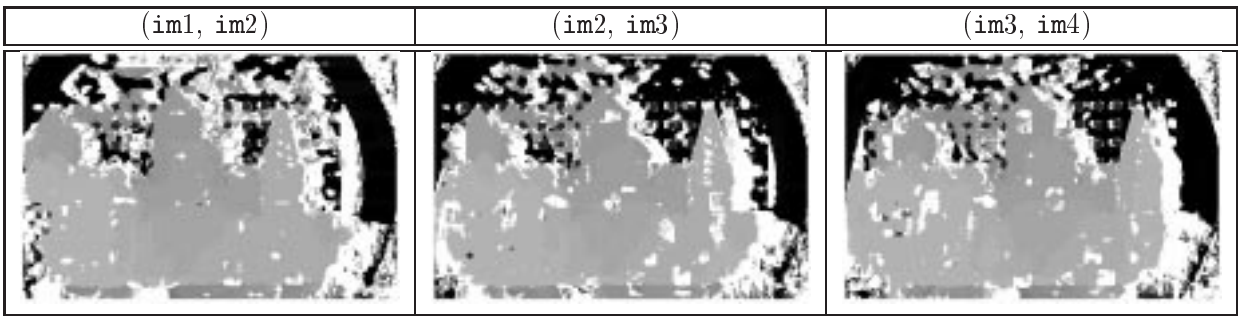


FIG. 3.51: *Cartes de disparité obtenues pour la séquence de la figure 3.49, après régularisation et fusion.*

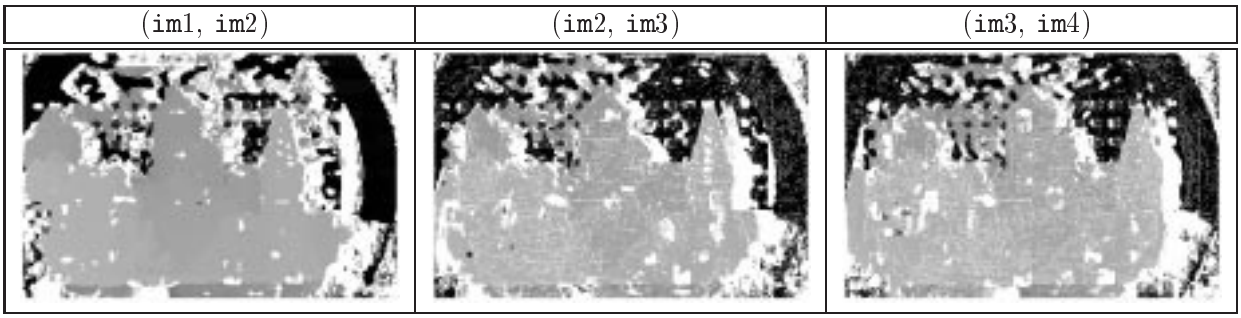


FIG. 3.52: *Cartes de disparité obtenues pour la séquence de la figure 3.49, après régularisation, fusion, et affinage.*

Nous faisons plusieurs remarques sur ces résultats.

□ Les cartes de disparité semblent assez uniformes. De fait, avant l'étape d'affinage, les algorithmes ne fonctionnent qu'au pixel près, et on ne perçoit que 5 ou 6 profon-

deurs différentes. L'étape d'affinage est donc indispensable, et son effet est montré clairement sur un agrandissement de la carte de disparité (figure 3.53).

□ Les appariements sont assez difficiles à obtenir, car la texture est absente (zones grises uniformes), ou trop répétitives (fenêtres de la maquette). C'est pourquoi nous avons pris un masque de taille 11×11.

□ Le fond est impossible à apparier correctement, car il est composé de points noirs identiques, sur un fond uniformément blanc.

□ Après affinage, les cartes de disparité des couples (`im2`, `im3`) et (`im3`, `im4`) deviennent moins denses. Cela est dû au fait que les coordonnées des projections dans les images 2, 3 et 4 ne sont plus entières. Les positions des points sont arrondies à l'entier le plus proche, donc tous les pixels de la carte de disparité ne sont pas renseignés. En revanche, les positions des points dans l'image 1 ne sont pas modifiées par l'affinage, ce qui explique que la carte de disparité du couple (`im1`, `im2`) reste dense.

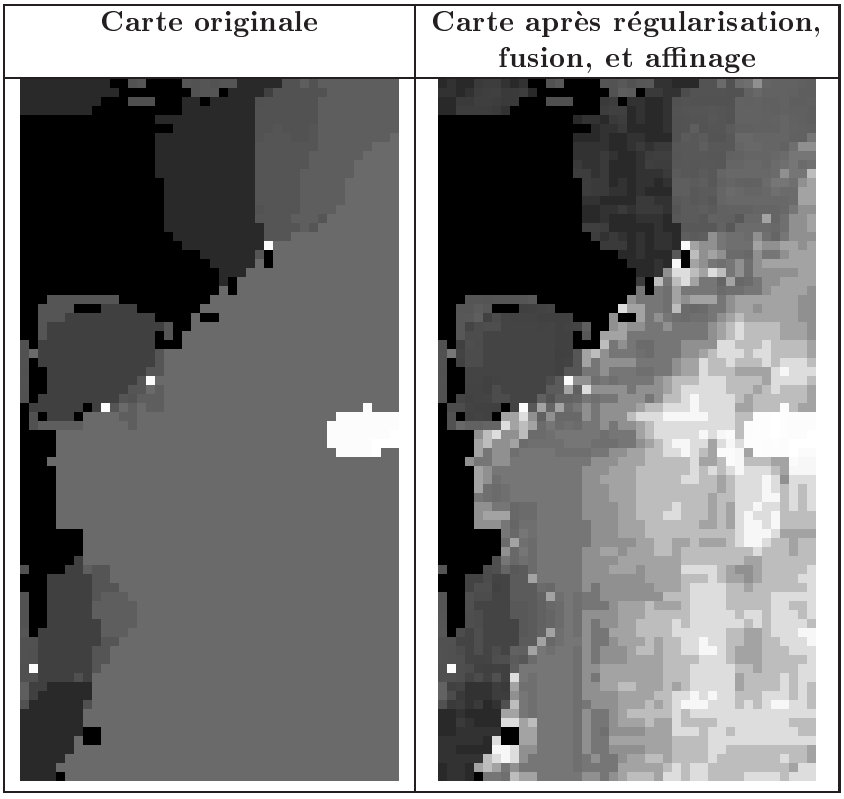


FIG. 3.53: *Agrandissement de la zone centre-haut de la carte de disparité : la plus grande tour carrée, dans le fond de la scène. Les intensités ont subi une égalisation d'histogramme.*

3.5.4.2 Séquence 2

Nous testons maintenant la chaîne d'appariement sur la séquence montrée en figure 3.54. Cette séquence de deux images a été prise par B. Boufama, dans notre équipe ; contrairement à la précédente, elle n'est pas étalonnée.

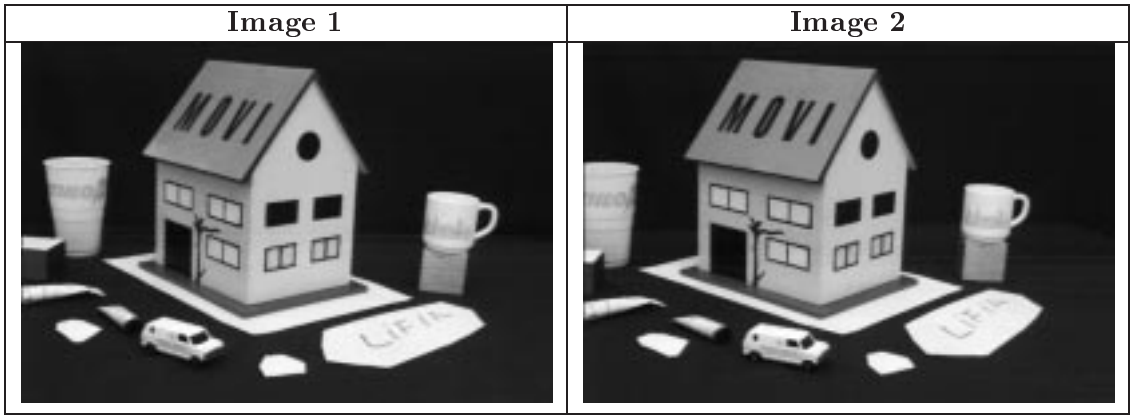


FIG. 3.54: *Séquence non-étalonnée de deux images : la maison* MOVI.

Nous ne disposons pas des matrices de projection, et la géométrie épipolaire doit être déterminée à partir des images. Les points d'intérêts utilisés sont montrés en figure 3.55.

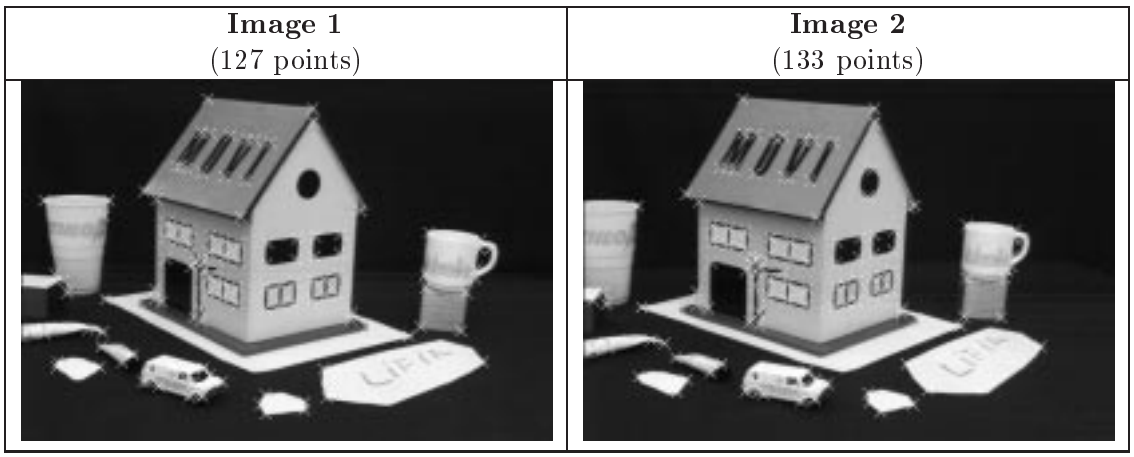


FIG. 3.55: *Points de Harris détectés dans les deux images.*

Une liste de 89 appariements est établie entre ces points à l'aide d'un algorithme CCR utilisant une mesure SAD 11×11. La position des points dans l'image 2 est ensuite améliorée par un affinage AFFSI avec $(dx, dy) = (1.0, 1.0)$. On calcule la matrice fondamentale $F_{1,2}$ à partir de ces 89 appariements affinés, sans autre vérification.

Vient ensuite l'étape d'appariement dense. L'algorithme CCR semble inadapté, car les images sont très peu texturées. De plus, les images ne présentent pas d'occultation, car le

fond noir uniforme peut être considéré comme une surface attenante aux objets. L'algorithme DP3NO de programmation dynamique peut donc être employé, et nous l'utilisons avec une mesure SAD 11×11, en fixant une limite de disparité à 40 pixels. La carte de disparité obtenue est présentée en figure 3.56, à comparer avec la carte obtenue par l'algorithme CCR sur les mêmes images.

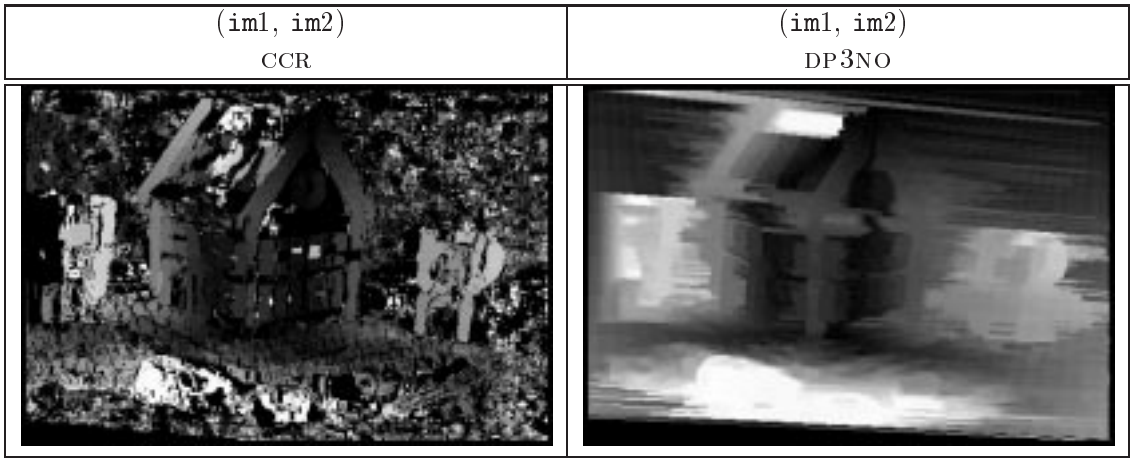


FIG. 3.56: *Cartes de disparité obtenues pour la séquence de la figure 3.54, avec les algorithmes* CCR *ou* DP3NO.

L'algorithme CCR donne peu d'appariements, car la vérification croisée échoue sur toutes les zones uniformes : les façades de la maison, et le fond noir. Seuls les points proches des zones contrastées sont correctement mis en correspondance (essentiellement les contours). Il est impossible de régulariser une telle carte, car les faux appariements sont trop nombreux sur le fond noir, et la texture des images reste trop faible pour un calcul correct.

L'algorithme DP3NO impose une minimisation globale le long de chaque épipolaire, et les zones de fort contraste suffisent le plus souvent à faire correctement converger la solution. Les autres points prennent automatiquement des valeurs de disparité intermédiaires. Cela ne fonctionne pas toujours, et le haut du toit, par exemple, ainsi que la gouttière, sont visiblement mal appariés. Nous utiliserons néanmoins ces données, et nous verrons quels problèmes cela soulève lors de l'étape de transfert.

Les appariements obtenus sont régularisés en une passe afin de limiter l'effet d'empilement de lignes, caractéristique des algorithmes de programmation dynamique, puis ils sont affinés par AFFZD4. La carte de disparité finale est montrée en figure 3.57.

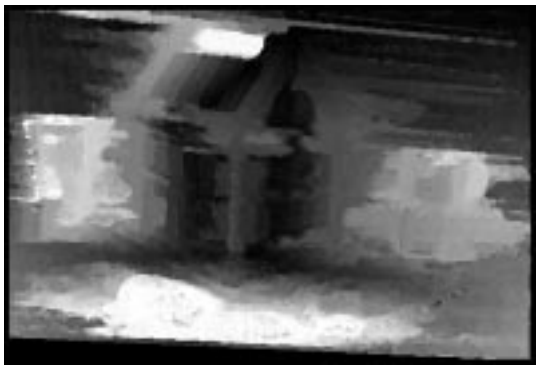


FIG. 3.57: *Carte de disparité finale pour la séquence de la figure 3.54, après régularisation et affinage.*

3.6 Conclusion

Dans ce chapitre, nous avons calculé et évalué des appariements denses, en comparant plusieurs mesures de ressemblance, plusieurs algorithmes d'optimisation globale, ainsi que des étapes de post-traitement telles que la régularisation, ou l'affinage. L'utilisation d'images synthétiques nous a permis d'établir des comparaisons chiffrées, et même si les images employées ne sont pas représentatives de toutes les scènes que nous rencontrerons, elles permettent néanmoins de percevoir où se situent les difficultés, et quels remèdes y apporter. Ces images représentaient des splines, parfois vues de côté, et permettaient de tester la validité des algorithmes classiques, tels que la mise en correspondance par corrélation, dans des cas non favorables : occultations, déformations perspectives importantes. Une telle évaluation n'avait pas été menée auparavant.

Nous pensons que ce procédé devrait être étendu, et il serait souhaitable d'utiliser d'autres types de scènes synthétiques, plus élaborées. Cela nécessite la définition de protocoles expérimentaux encore plus lourds et complexes.

Des expériences que nous avons menées, nous tirons les enseignements suivants.

- Le calcul de la géométrie épipolaire liant deux images est fiable dans la majorité des cas. Il nécessite tout de même une vérification manuelle, qui consiste par exemple à s'assurer que les appariements utilisés sont correctement répartis dans le volume de la scène.

- Les estimations de qualité issues du calcul de la géométrie épipolaire doivent être interprétés avec prudence. Les chiffres pris en compte (p. ex. l'erreur médiane) s'appuient sur les données initiales. Or, celles-ci peuvent correspondre par hasard très précisément à une matrice fondamentale, qui n'est pas la bonne. Il est bien sûr impossible d'évaluer précisément la qualité de la géométrie épipolaire calculée si on ne dispose d'aucune autre source d'information (comme un étalonnage).

- L'appariement par vecteur d'invariants ne convient qu'à des images où le signal peut être dérivé de plusieurs ordres, et de façon stable. Nous préconisons d'utili-

ser les invariants sur des points d'intérêt uniquement, et d'utiliser cette première phase d'appariement pour estimer de façon robuste la transformation globale liant les deux images : rotation, translation, facteur d'échelle. Ensuite, nous pouvons redresser les images, et procéder à l'appariement des autres points à l'aide de mesures de corrélation classiques.

- Les mesures de corrélation centrées sont trop permissives, et nous suggérons plutôt de préalablement normaliser *globalement* les niveaux de gris des images (équation 3.5, p. 29), puis utiliser des mesures non-centrées. Cela n'a pas été testé, car nous n'avons traité que des images présentant les mêmes conditions d'illumination.

- La mesure de corrélation SAD fonctionne au moins aussi bien que toutes les autres mesures, et a l'avantage de la simplicité, donc de la rapidité. Nous l'utiliserons systématiquement.

- Les effets des occultations et des déformations perspectives peuvent être limités par l'emploi de masques de corrélation plus grands, tout en conservant des mesures classiques de corrélation. Les mesures robustes sont inutilisables sans information *a priori*, et les mesures partiellement robustes, telle PRSAD, sont encore immatures.

- De même, l'algorithme d'appariement CCR présente un bon compromis : il est aussi rapide que les autres algorithmes ; il est plus fiable (étape de vérification croisée), mais moins dense. Surtout, il ne nécessite pas de paramétrage délicat, et ne fait pas d'hypothèse *a priori* sur la nature des images (pas de contrainte d'ordre).

- De toute façon, la densité de l'algorithme d'appariement importe peu, car l'étape de régularisation comble les zones non renseignées, et densifie le résultat, en l'améliorant. En revanche, le nombre de passes de régularisation à effectuer doit toujours être déterminé par une appréciation subjective de l'utilisateur.

- L'utilisation d'algorithmes d'affinage est indispensable (cas du château de CMU, par exemple). Parmi ceux proposés, AFFZD4 est très rapide et efficace, bien que les images soient assez éloignées de l'hypothèse translationnelle. L'algorithme itératif AFFSI est meilleur, mais 50 fois plus lent.

Les procédés évalués ont été appliqués aux trois séquences d'images synthétiques, ainsi qu'à deux séquences d'images réelles. Les appariements obtenus seront traités dans le chapitre suivant pour générer de nouvelles vues synthétiques de ces scènes, et évaluer leur qualité.

Abréviations utilisées dans ce chapitre

Algorithme	Description	Paramétrage
WTA	Winner Takes All. Meilleur score.	Inutile
ES	Exhaustive Search. Recherche exhaustive de tous les appariements.	Inutile
CCR	Cross Check Raw. Meilleur score + vérification croisée.	Inutile
CCT	Cross Check Threshold. CCR avec score maximal égal à s.	s
DP3	Dynamic Programming 3-Transitions. Programmation dynamique classique, coût d'occultation κ.	κ
DP3NO	Dynamic Programming 3-Transitions No Occlusion. Programmation dynamique sans occultation.	Inutile
DP5	Dynamic Programming 5-Transitions. Programmation dynamique sans contrainte d'unicité, coût d'occultation κ.	κ
DP3JC	Dynamic Programming 3-Transitions Jumps Count. Programmation dynamique avec comptage d'occultations, coût d'occultation nul.	p_{occ}

TAB. 3.54: *Algorithmes d'appariement.*

Mesure	Description
SAD	Somme des différences absolues des niveaux de gris.
ZSAD	Version centrée de SAD.
SSD	Somme des différences au carré des niveaux de gris.
ZSSD	Version centrée de SSD.
RSSD	Version robuste de SSD.
RZSSD	Version robuste de ZSSD.
WSAD	SAD, avec pondération décroissante en fonction de la distance au centre.
RSAD	Version robuste de SAD.
PRSAD	Version partiellement robuste de SAD.
INVAR	Invariants (comme C. Schmid dans [Sch 96b]).

TAB. 3.55: *Mesures de ressemblance.*

Nom	Nature	Description
AffSi	itératif	Plusieurs mesures SAD dans un voisinage de plus en plus restreint.
AffZd4	non-itératif	Développement limité du signal d'intensité des images.
AffSc1	non-itératif	Approximation de la fonction de coût de corrélation par deux paraboles.
AffSc2	non-itératif	Approximation de la fonction de coût de corrélation par un paraboloïde.

TAB. 3.56: *Algorithmes d'affinage.*

Chapitre 4

Transfert

4.1 Introduction

Nous avons calculé au chapitre précédent des appariements denses multi-oculaires ; certains sont faux, d'autres sont seulement imprécis. Nous tentons dans ce chapitre de procéder au transfert de ces données, et de synthétiser de nouvelles vues. Comme précédemment, les résultats seront évalués quantitativement. Nous présenterons des critères permettant de réaliser cette évaluation en 4.2. Nous reviendrons ensuite sur les questions d'étalonnage en 4.3, en précisant ce qu'il est possible de réaliser avec ou sans ces informations, et la façon dont nous réaliserons le cas échéant la reconstruction 3D, le maillage triangulaire, et le calcul de texture. Enfin, nous mènerons des tests à la fois sur des images synthétiques et sur des images réelles en 4.6, 4.7, 4.8 et 4.9.

4.2 Critère de qualité

Évaluer le transfert revient à tester la qualité des images produites. Nous ne testerons pas la qualité du modèle 3D éventuellement généré, car nous nous plaçons dans la problématique $\mathcal{P}1$ (synthèse de nouvelles images) et non $\mathcal{P}2$ (calcul d'une représentation tridimensionnelle de la scène).

Nous disposons pour cette évaluation de quelques critères quantitatifs simples :

- l'erreur absolue moyenne EAM ;
- l'erreur quadratique moyenne EQM ;
- le rapport signal sur bruit SNR.

Soient deux images de signal I_1 (image originale) et I_2 (image synthétisée, de qualité à quantifier). Les trois critères sont donnés par les équations 4.1, 4.2 et 4.4 ci-dessous. Le critère SNR est exprimé en dB.

$$\text{EAM} = \frac{\sum_{x=1}^{x=N} \sum_{y=1}^{y=M} |I_2(x,y) - I_1(x,y)|}{N\,M} \tag{4.1}$$

$$\text{EQM} = \frac{\sum_{x=1}^{x=N} \sum_{y=1}^{y=M} (I_2(x,y) - I_1(x,y))^2}{N\,M} \tag{4.2}$$

$$\text{SNR} = 10 \log_{10} \frac{E(I_1(x,y)^2)}{E((I_2(x,y) - I_1(x,y))^2)} \tag{4.3}$$

$$= 10 \log_{10} \frac{\sum_{x=1}^{x=N} \sum_{y=1}^{y=M} I_1^2(x,y)}{\sum_{x=1}^{x=N} \sum_{y=1}^{y=M} (I_2(x,y) - I_1(x,y))^2} \tag{4.4}$$

Nous choisissons la mesure EAM pour sa simplicité et sa rapidité.

Notons que d'une façon générale, il existe peu de critères numériques pour évaluer la qualité d'images générées par un ordinateur. Dans les travaux de compression d'images fixes ou de compression de vidéo (avec perte), les tests les plus couramment employés font appel à des observateurs humains. En répondant à des questionnaires, ils notent la qualité des images observées sur une échelle subjective. Une mesure objective est aussi possible si la question est simplement : «pouvez-vous percevoir une différence entre l'image calculée et l'image originale ?». En effet, il suffit dans ce cas de présenter à l'observateur sur un écran les images originale et calculée en alternance, en parfaite superposition. S'il ne détecte aucun changement, on peut dire que les images sont perçues exactement de la même façon (par *cet* observateur). Mais ce cas est assez rare, et on utilise le plus souvent des notations qualitatives. Le travail de dépouillement doit prendre en compte des critères tels que l'ordre d'examen des images, ou d'autres facteurs extérieurs, et mener à une mesure chiffrée de la qualité des images.

Cela représente un travail colossal, dont les résultats ne sont pas répétables. Aussi, certains auteurs ont proposé des critères quantitatifs prenant en compte des données psychovisuelles. D. Barba dans [Bar 97] étudie les fonctions biologiques de la vision humaine, et les inclut dans un «critère objectif de qualité subjective» d'image. Les résultats montrent que la fonction obtenue est très bien corrélée avec les résultats classiques obtenus par sondage d'observateurs humains.

Le critère EAM que nous avons choisi ne présente pas les avantages que devrait avoir une mesure psychovisuelle.

- Si dans les images 1 et 2, seuls les contours sont très différents (contours déplacés de 10 pixels par exemple), alors la différence entre ces images sera faible, car toutes les zones uniformes entre les contours donneront une somme partielle faible. Elles paraissent cependant assez dissemblables.

□ Si les images 1 et 2 sont les images d'un bruit gaussien, alors la différence est très forte, bien que les images paraissent très semblables.

Aussi, nous commenterons les résultats de la mesure EAM selon la nature des images.

4.3 Étalonnage

Les images que nous traitons peuvent correspondre à plusieurs sortes d'information : des informations propres aux images (leur contenu : informations photométriques sur la scène observée), ou des informations relatives à leurs conditions de prise de vue (position et orientation des caméras, et/ou paramètres optiques des caméras). Selon les informations dont nous disposons, nous pouvons réaliser différentes sortes d'étalonnage.

► Si nous disposons d'au moins deux images de la scène, prises d'au moins deux points de vues distincts, et que la scène observée est non-plane, alors on peut établir un étalonnage faible.

Dans le cas binoculaire, l'étalonnage faible est équivalent au calcul de la géométrie épipolaire. Il permet d'établir une reconstruction projective. En effet, connaissant $F_{1,2}$ entre les images 1 et 2, on peut calculer les matrices de projection M_1 et M_2 dans les deux images (voir ci-dessous, en 4.5.1). On peut ensuite reconstruire tous les points par triangulation, et ils sont obtenus dans une base projective arbitraire.

Dans le cas de N caméras, on peut également réaliser un étalonnage faible, par exemple par des méthodes de factorisation, qui permettent de trouver les matrices de projection M_1, ..., M_N dans les N images, connaissant uniquement des appariements multi-oculaires. De telles méthodes ont été développées par P. Sturm et B. Triggs dans notre laboratoire, et sont décrites dans [Stu 96]. L'inconvénient est que tous les points doivent être vus dans toutes les images, sinon la méthode devient moins directe : il faut reprojeter les points dans les images d'où ils sont absents. Enfin, tous les points doivent être corrects, ce que nous ne pouvons pas garantir. Une autre méthode utilise les matrices fondamentales et les tenseurs trilinéaires entre les images [Tri 97b]. On peut donc s'affranchir de la présence ou de l'absence de points dans certaines images, car les matrices fondamentales et les tenseurs sont calculés avec des informations seulement bi- ou trinoculaires.

Enfin, si nous disposons des paramètres intrinsèques des caméras, nous pouvons établir un étalonnage fort, et amener la reconstruction projective à une reconstruction euclidienne. L'étalonnage fort consiste à calculer tous les paramètres de projection, à savoir la position et l'orientation de la caméra, et ses paramètres optiques : distance focale, taille des pixels, angle des lignes par rapport aux colonnes de pixels du capteur CCD. On peut alors réaliser une reconstruction euclidienne, dans un repère orthonormé arbitraire, à un facteur d'échelle global près.

Si nous ne connaissons pas les paramètres intrinsèques, nous pouvons aussi utiliser certains informations connues dans les images *a priori*, telles que des longueurs de segments, ou des angles entre segments. Dans sa thèse, B. Boufama parvient ainsi à étalonner fortement une scène urbaine, en désignant manuellement quels sont les segments orthogonaux ou parallèles dans l'espace 3D, et leurs longueurs respectives [Bou 94].

► Si nous disposons d'au moins trois images de la scène, nous pouvons réaliser un étalonnage fort sans information sur les paramètres intrinsèques. Cependant, les caméras ne doivent pas décrire l'un des mouvements suivants : mouvement orbital, rotation autour d'axes parallèles et translation arbitraire, mouvements planaires, translations pures, rotations pures. Dans ces cas, l'étalonnage et la reconstruction 3D sont ambigus, voire impossibles. Ceci est décrit et expliqué par P. Sturm dans [Stu 97]. Les images prises naturellement sont souvent dans l'une de ces configurations critiques, et il est impossible de mener à bien l'auto-étalonnage. La technique émerge et des algorithmes apparaissent (notamment B. Triggs dans [Tri 97a]), mais ils sont encore difficilement applicables à des cas réels, où les données sont bruitées, absentes, ou fausses dans une certaine proportion.

Enfin, si nous disposons des caméras, il suffit de les placer devant une mire 3D, et de procéder à un étalonnage (fort) classique. R.Y. Tsai décrit de nombreuses procédures d'étalonnage dans [Tsa 86], et apporte également de nouvelles techniques très précises.

4.3.1 Nécessité d'un étalonnage fort

Pour le transfert, un étalonnage fort n'est pas nécessaire en théorie : à partir des appariements, on peut réaliser un étalonnage faible, donc calculer une reconstruction projective, qu'on projette sur le plan de la caméra virtuelle pour obtenir une nouvelle image. Mais il est dans ce cas difficile de spécifier la position de la caméra virtuelle, car ni les angles, ni les longueurs, ne sont définis ; S. Laveau propose dans [Lav 94b] de la spécifier par la position de 5 points dans la nouvelle image. Dans ces conditions, il n'existe bien sûr pas de méthode pour placer ces points de façon à être certain d'obtenir une image réaliste, et non projectivement déformée. Le principe est cependant valable si l'image à synthétiser est connue d'avance, ce qui est le cas en compression vidéo : la position des 5 points est alors connue. On pourrait même dans ce cas calculer le tenseur liant 3 vues, et le transfert serait applicable sans reconstruction explicite (et sans étalonnage fort, figure 4.1).

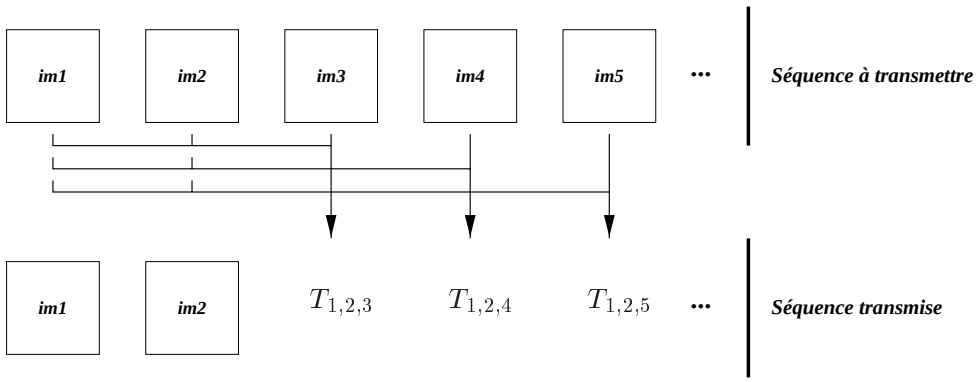


FIG. 4.1: *Principe de la compression vidéo sans étalonnage fort. Pour transmettre une séquence, il suffit de transmettre deux images, et un tenseur trilinéaire pour chaque image suivante.*

Aucun étalonnage n'est non plus nécessaire pour la construction de mosaïques, dont

nous avons déjà parlé, et que nous expérimenterons nous-mêmes ci-dessous, en 4.4.

4.4 Mosaïques

Nous avons supposé jusqu'ici que les caméras formaient des configurations stéréo. Or ce n'est pas toujours le cas : si les centres des caméras sont confondus (ou dans le cas d'une caméra en rotation autour de son centre), alors il est impossible d'obtenir une quelconque information de relief : la reconstruction de points 3D à partir d'appariements mène toujours à des lignes 3D confondues.

On peut cependant réaliser le transfert en reprojetant cette ligne 3D sur le plan-image d'une nouvelle caméra, de même centre optique que les précédentes, comme déjà décrit en figure 2.5, p. 21. Si nous le faisons pour chaque pixel des images de référence, nous obtenons une nouvelle image, vue du même point, mais sous un autre angle : l'image est pivotée. Il est d'ailleurs inutile de procéder à une reconstruction explicite de lignes 3D, car dans cette configuration, les coordonnées des points se correspondant dans les images sont liés par une homographie plane, représentable par une matrice 3×3 inversible [Moh 93]. À partir d'un seul point p dans une image i, on peut donc calculer sa position q dans une autre image j par l'équation 4.5.

$$q = H_{i,j}\ p \tag{4.5}$$

Des appariements denses ne sont plus nécessaires. Il suffit d'appariements épars pour calculer les homographies entre les couples d'images, et de nouvelles images arbitraires de même centre optique peuvent être synthétisées. On peut par exemple aligner toutes les images dans le même repère, et composer une mosaïque par recollage. Nous testons cette technique dans les sections suivantes.

4.4.1 Tests sur images synthétiques

Nous effectuerons nos tests sur une nouvelle série d'images synthétiques. Contrairement aux précédentes, elles représenteront des images en correspondance homographique, c.-à-d. prises du même point de vue. La disposition de ces images est donnée en figure 4.2. Les vues sont espacées de 5 ° par rotation autour d'un axe vertical passant par le centre optique de la caméra.

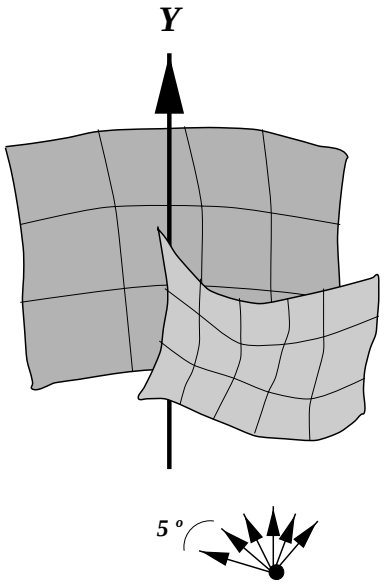


FIG. 4.2: *Disposition des six images synthétiques.*

Les images sont texturées par la texture aléatoire ou les deux textures naturelles, pour produire les images visibles en figures 4.3, 4.4 et 4.5. Elles sont générées par POV–Ray, modifié pour tenir compte du décalage systématique de $(0.5, 0.5)$ pixel (voir 3.3.4, p. 57).

4.4.1.1 Procédure

Dans un premier temps, nous allons tester la procédure permettant de calculer l'homographie entre deux images. Elle comprend trois étapes :

phase 1 : extraction de points d'intérêt dans les images de référence ;

phase 2 : appariement de ces points (épars) et affinage ;

phase 3 : calcul robuste de l'homographie liant les deux images.

Dans un second temps, nous calculerons les homographies liant les images `im0` à `im4` à l'image `im2`, et en dernier lieu, nous créerons une mosaïque composée des images `im0` à `im4`, dans le repère de `im2`.

4.4.1.2 Calcul des homographies

L'homographie est décrite par une matrice 3×3 inversible, soient 8 coefficients indépendants. Elle peut être estimée à partir de 4 appariements, qui fournissent 4 équations du type 4.5, soient 8 équations scalaires. Pour estimer une homographie à partir d'appariements, nous procédons à un calcul robuste et normalisé comme pour la matrice fondamentale (voir 3.4.8), que nous ne décrivons pas ici de nouveau.

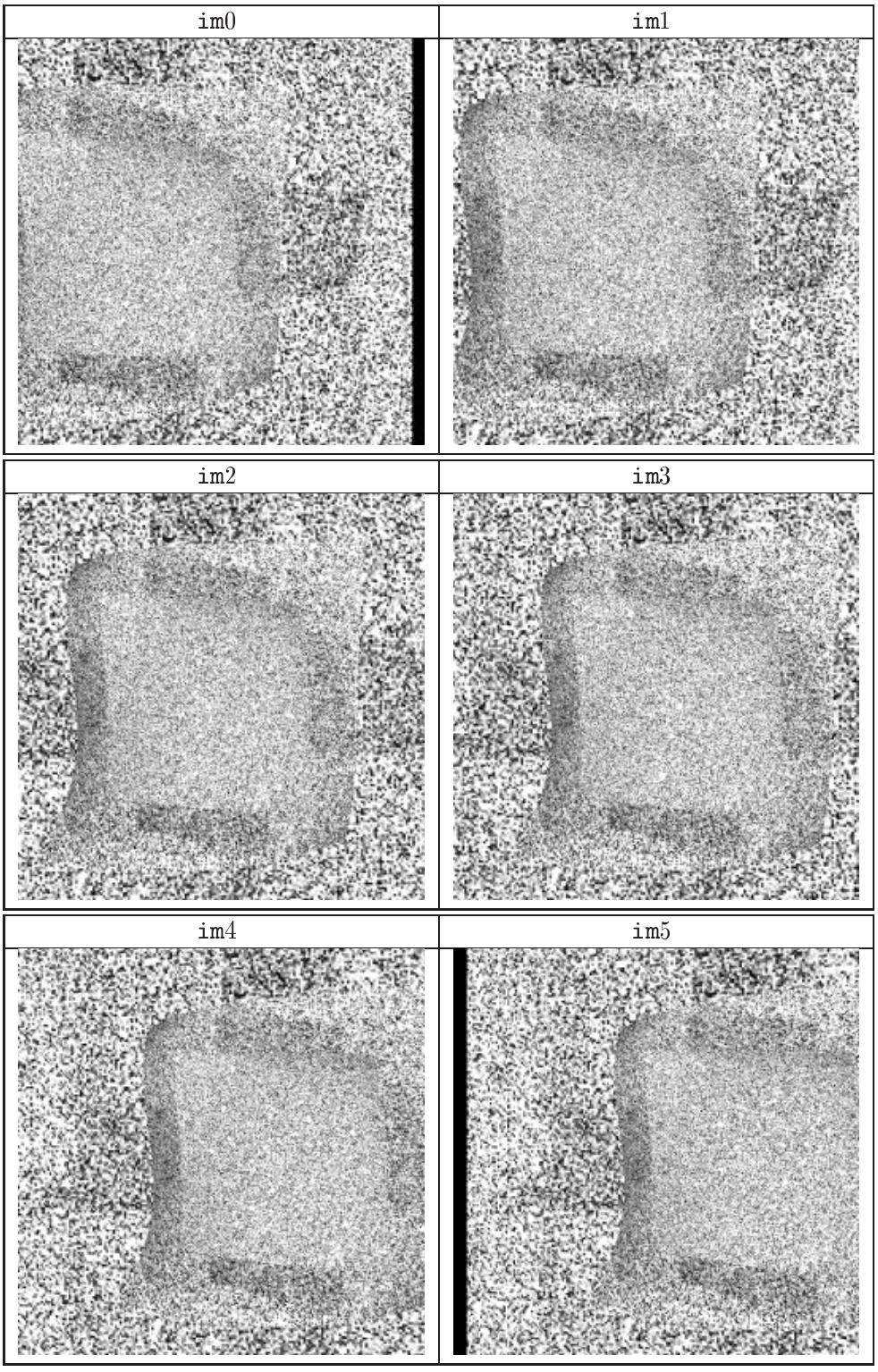


FIG. 4.3: *Les 6 images synthétiques, texture aléatoire.*

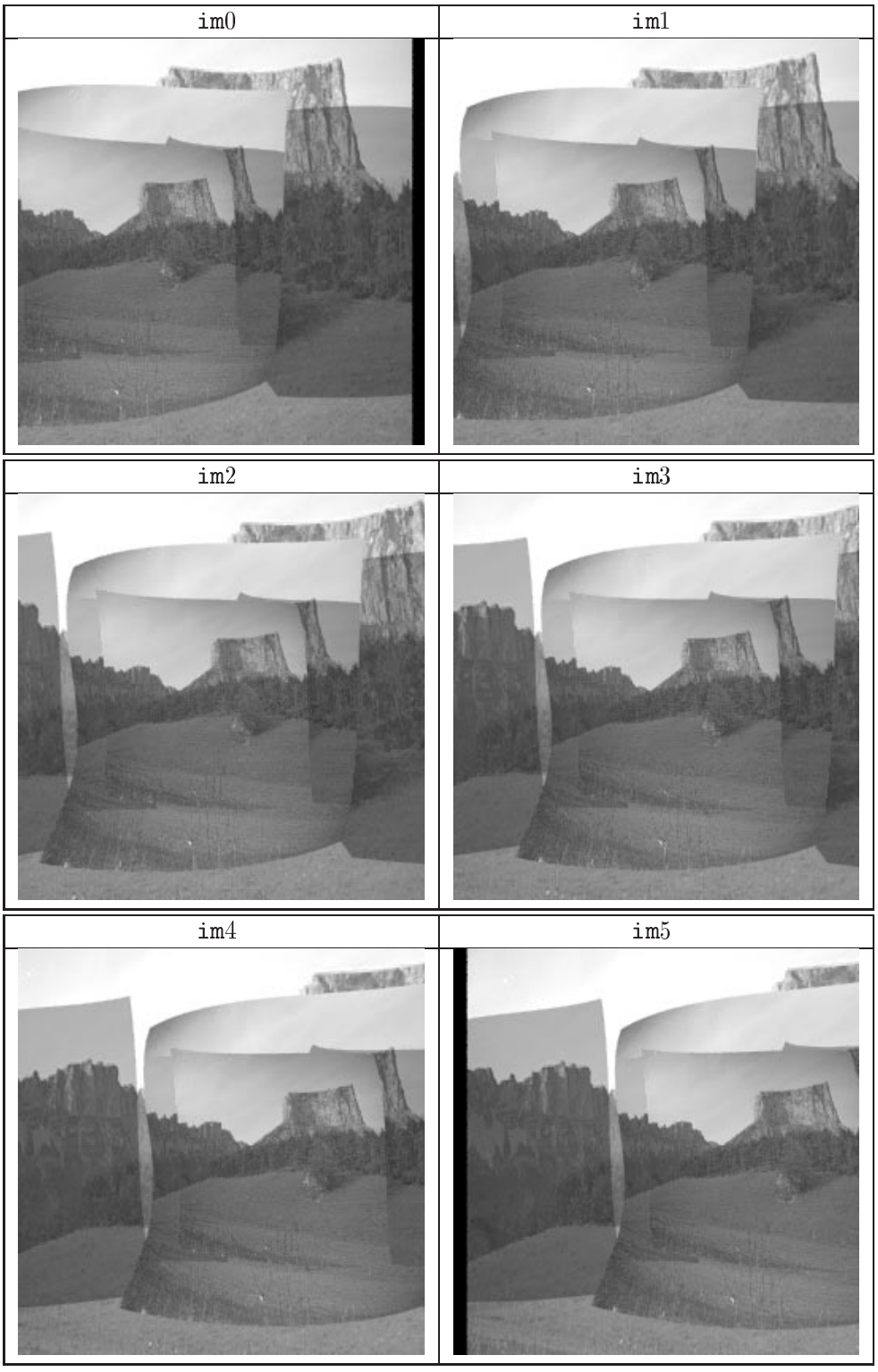


FIG. 4.4: *Les 6 images synthétiques, texture d'extérieur.*

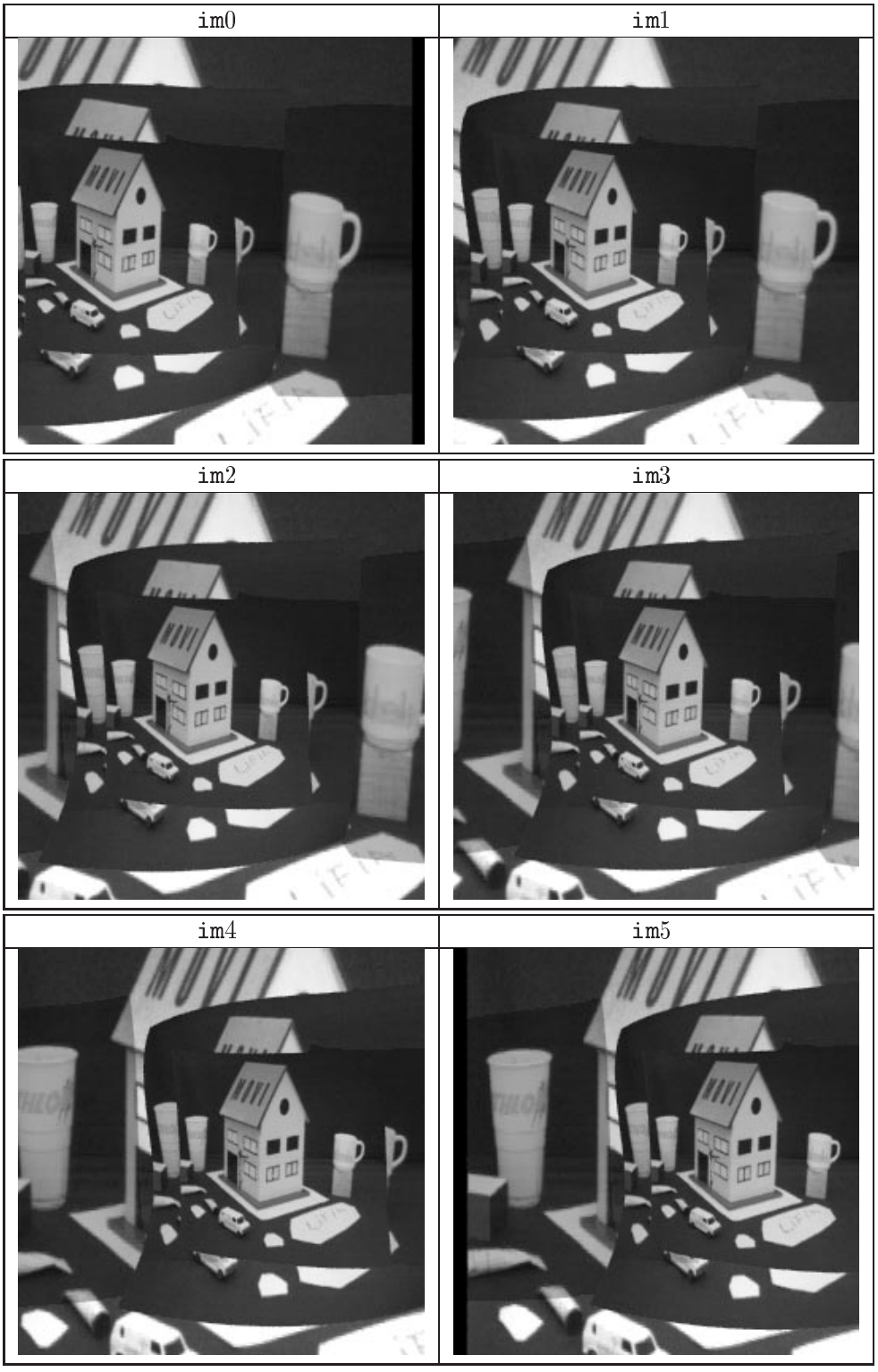


FIG. 4.5: *Les 6 images synthétiques, texture d'intérieur.*

Le nombre N de tirages aléatoires nécessaires pour un taux attendu de faux appariements τ, et une probabilité de 99.9 % de trouver l'homographie correcte, est cette fois donné par :

$$1 - (1 - (1 - \tau)^4)^N = 0.999 \tag{4.6}$$

Pour $\tau = 0.5$, on doit effectuer $N = 107$ tirages.

Nous comparerons l'homographie obtenue ${H_{1,2}}^{es}$ à l'homographie exacte théorique ${H_{1,2}}^{th}$, parfaitement connue, selon une méthode similaire à celle employée pour les matrices fondamentales (description en 3.4.8.1, p. 77) :

1. Tirer un point p_1 au hasard dans l'image 1.
2. Calculer son correspondant estimé $q_1^{es} = {H_{1,2}}^{es} p_1$ dans l'image 2.
3. Tirer un point q_2 au hasard dans l'image 2.
4. Calculer son correspondant estimé $p_2^{es} = {H_{2,1}}^{es} q_2$ dans l'image 1 (on a bien sûr $H_{2,1} = {H_{1,2}}^{-1}$).
5. Calculer la distance d_2 de q_1^{es} à sa position théorique $q_1^{th} = {H_{1,2}}^{th} p_1$, et la distance d_1 de p_2^{es} à sa position théorique $p_2^{th} = {H_{2,1}}^{th} q_2$.
6. Effectuer de nombreuses fois les étapes 1 à 5, en accumulant les valeurs de $d = (d_1 + d_2)/2$. Ici aussi, 10000 tirages suffisent en pratique, sur des images de cette taille.

4.4.1.3 Phase 1

Les points d'intérêt sont extraits dans les images par le détecteur de Harris amélioré. Le tableau 4.1 donne le nombre de points détectés dans les 3 séries de 6 images.

Image	Texture aléatoire	Texture d'extérieur	Texture d'intérieur
im0	508	246	137
im1	504	277	154
im2	517	269	156
im3	493	259	160
im4	518	241	156
im5	525	199	162

TAB. 4.1: *Nombre de points d'intérêt détectés dans les 3 séries de 6 images synthétiques.*

Comme précédemment, le seuil de détection a dû être divisé par 10 pour les images à texture d'extérieur, car sinon, les points d'intérêt seraient trop rares et trop mal répartis pour mener à un calcul correct des homographies.

4.4.1.4 Phase 2

Nous apparions les points de `im0` avec les points des images suivantes par un algorithme CCR utilisant une mesure SAD 15×15. Les appariements obtenus sont ensuite affinés par une méthode AFFSI, avec $(dx, dy) = (1.0, 1.0)$, opérant sur une mesure SAD de taille 11×11. Le nombre d'appariements (binoculaires) obtenus entre les couples d'images est donné par le tableau 4.2.

Couple	Texture aléatoire	Texture d'extérieur	Texture d'intérieur
(im0, im1)	203	166	125
(im0, im2)	168	150	119
(im0, im3)	125	128	113
(im0, im4)	90	119	112
(im0, im5)	90	89	104

TAB. 4.2: *Nombre de points d'intérêt appariés dans les 5 couples des 3 séries.*

4.4.1.5 Phase 3

À partir de ces appariements affinés, nous calculons les homographies entre l'image `im0` et les images suivantes. Comme pour les matrices fondamentales, le but est de tester la robustesse du processus sur des images de plus en plus éloignées. Les erreurs de calcul sont données en tableaux 4.3, 4.4 et 4.5, et les erreurs par rapport aux homographies exactes théoriques sont données dans les tableaux 4.6, 4.7 et 4.8. Comme précédemment, l'erreur médiane est la distance médiane de tous les points initiaux à leur position théorique, alors que l'erreur moyenne est la distance moyenne des seuls points conservés pour le calcul final à leur position théorique.

Couple	Nombre d'appariements en entrée	Nombre d'appariements conservés	Erreur finale moyenne	Erreur finale médiane
(im0, im1)	203	179	0.080441	0.098404
(im0, im2)	168	144	0.101890	0.128060
(im0, im3)	125	103	0.121865	0.183542
(im0, im4)	90	70	0.175762	0.254117
(im0, im5)	90	63	0.171384	0.234393

TAB. 4.3: *Résultats du calcul robuste de l'homographie, sur les images à texture aléatoire.*

Couple	Nombre d'appariements en entrée	Nombre d'appariements conservés	Erreur finale moyenne	Erreur finale médiane
(im0, im1)	166	143	0.211359	0.240630
(im0, im2)	150	127	0.224730	0.318640
(im0, im3)	128	100	0.348104	0.496068
(im0, im4)	119	81	0.338453	0.578182
(im0, im5)	89	53	0.327122	0.572578

TAB. 4.4: *Résultats du calcul robuste de l'homographie, sur les images à texture d'extérieur.*

Couple	Nombre d'appariements en entrée	Nombre d'appariements conservés	Erreur finale moyenne	Erreur finale médiane
(im0, im1)	125	114	0.112883	0.123743
(im0, im2)	119	110	0.151011	0.173414
(im0, im3)	113	103	0.194901	0.228639
(im0, im4)	112	93	0.219340	0.242126
(im0, im5)	104	90	0.233571	0.286305

TAB. 4.5: *Résultats du calcul robuste de l'homographie, sur les images à texture d'intérieur.*

Homographie	Erreur moyenne	Erreur médiane	Erreur maximale
$H_{0,1}$	0.012209	0.009157	0.054788
$H_{0,2}$	0.034914	0.028512	0.131627
$H_{0,3}$	0.049532	0.039305	0.222207
$H_{0,4}$	0.089090	0.073970	0.349209
$H_{0,5}$	0.155947	0.127838	0.615712

TAB. 4.6: *Comparaison des homographies calculées aux homographies théoriques, sur les images à texture aléatoire.*

Homographie	Erreur moyenne	Erreur médiane	Erreur maximale
$H_{0,1}$	0.098921	0.091059	0.277510
$H_{0,2}$	0.102274	0.076982	0.390112
$H_{0,3}$	0.126190	0.100442	0.538741
$H_{0,4}$	0.313132	0.272097	0.936658
$H_{0,5}$	0.234972	0.188568	0.874943

TAB. 4.7: *Comparaison des homographies calculées aux homographies théoriques, sur les images à texture d'extérieur.*

Matrice fondamentale	Erreur moyenne	Erreur médiane	Erreur maximale
$H_{0,1}$	0.039508	0.028740	0.208772
$H_{0,2}$	0.067839	0.055963	0.252807
$H_{0,3}$	0.141954	0.122389	0.520306
$H_{0,4}$	0.251646	0.203425	0.801982
$H_{0,5}$	0.164168	0.134768	0.645014

TAB. 4.8: *Comparaison des homographies calculées aux homographies théoriques, sur les images à texture d'intérieur.*

Le calcul des homographies apparaît donc comme bien moins sensible que le calcul des matrices fondamentales, puisque l'erreur maximale, pour tous les couples de toutes les séries, reste inférieure à 1 pixel. Cela est dû au nombre réduit de points qu'il est nécessaire d'utiliser pour effectuer un calcul d'homographie : 4 appariements, au lieu de 8 dans le cas de la géométrie épipolaire.

4.4.1.6 Phase 4

Nous calculons ici les homographies $H_{0,1}$, $H_{1,2}$, $H_{2,3}$ et $H_{3,4}$. Le calcul est précis à 0.1 pixel en moyenne, et ces 4 homographies sont suffisantes pour transférer les images `im0`, `im1`, `im3` et `im4` dans le repère de `im2`. Si nous appliquons le calcul direct, en calculant la position q dans l'image 2 de chaque pixel p de l'image 0 par l'équation 4.7, alors nous obtenons des trous dans l'image transférée, car toutes les positions q ne sont pas renseignées. Aussi, nous utilisons une technique classique, où le transfert est réalisé en partant de l'image à obtenir : pour transférer l'image 0 dans le repère de l'image 2, nous calculons pour chaque pixel q de l'image 2 quel est le point correspondant p dans l'image 0. Nous appliquons donc l'homographie inverse, comme décrit par l'équation 4.8.

$$q = H_{0,1} H_{1,2}\, p \tag{4.7}$$

$$p = (H_{0,1} H_{1,2})^{-1}\, q \tag{4.8}$$

Les coordonnés du point p ne sont pas entières, et nous calculons son intensité par une interpolation bilinéaire de ses 4 voisins entiers. La figure 4.6 montre les images transférées, et leur composition en une mosaïque unique dans le repère de `im2`.

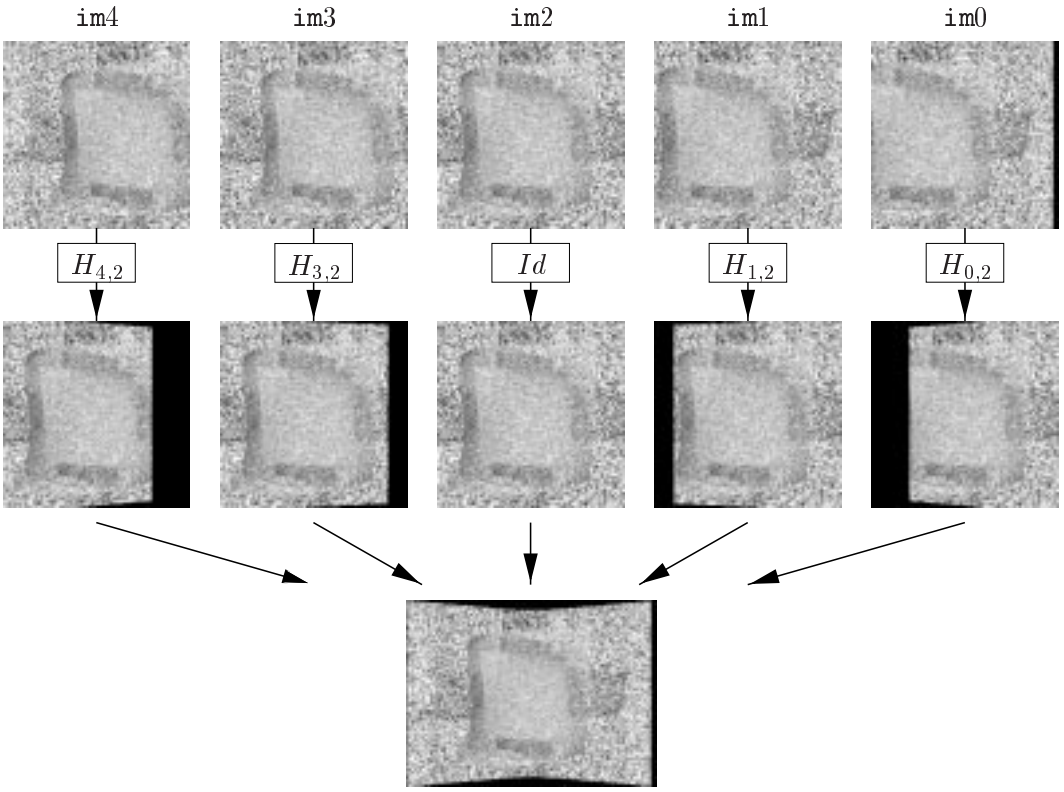


FIG. 4.6: *Transfert et composition d'une mosaïque.*

Pour les deux autres séries d'images, nous calculons aussi les homographies nécessaires, et procédons au transfert et à la composition de la mosaïque. Nous obtenons aussi avec POV–Ray les images que nous devrions théoriquement obtenir. Les mosaïques calculées par nos soins, et les mosaïques théoriques calculées par POV–Ray, sont présentées en figure 4.7. Les images de référence sont de taille 256×256, et les 3 mosaïques obtenues sont toutes de taille 284×379. Ces images rectangulaires ne sont bien sûr pas entièrement renseignées (zones noires en haut et en bas des mosaïques).

Nous donnons l'erreur absolue moyenne et médiane en niveaux de gris sur ces images, ainsi que la distribution de ces erreurs, en figure 4.8. Les erreurs ne sont calculées qu'aux points renseignés des deux images.

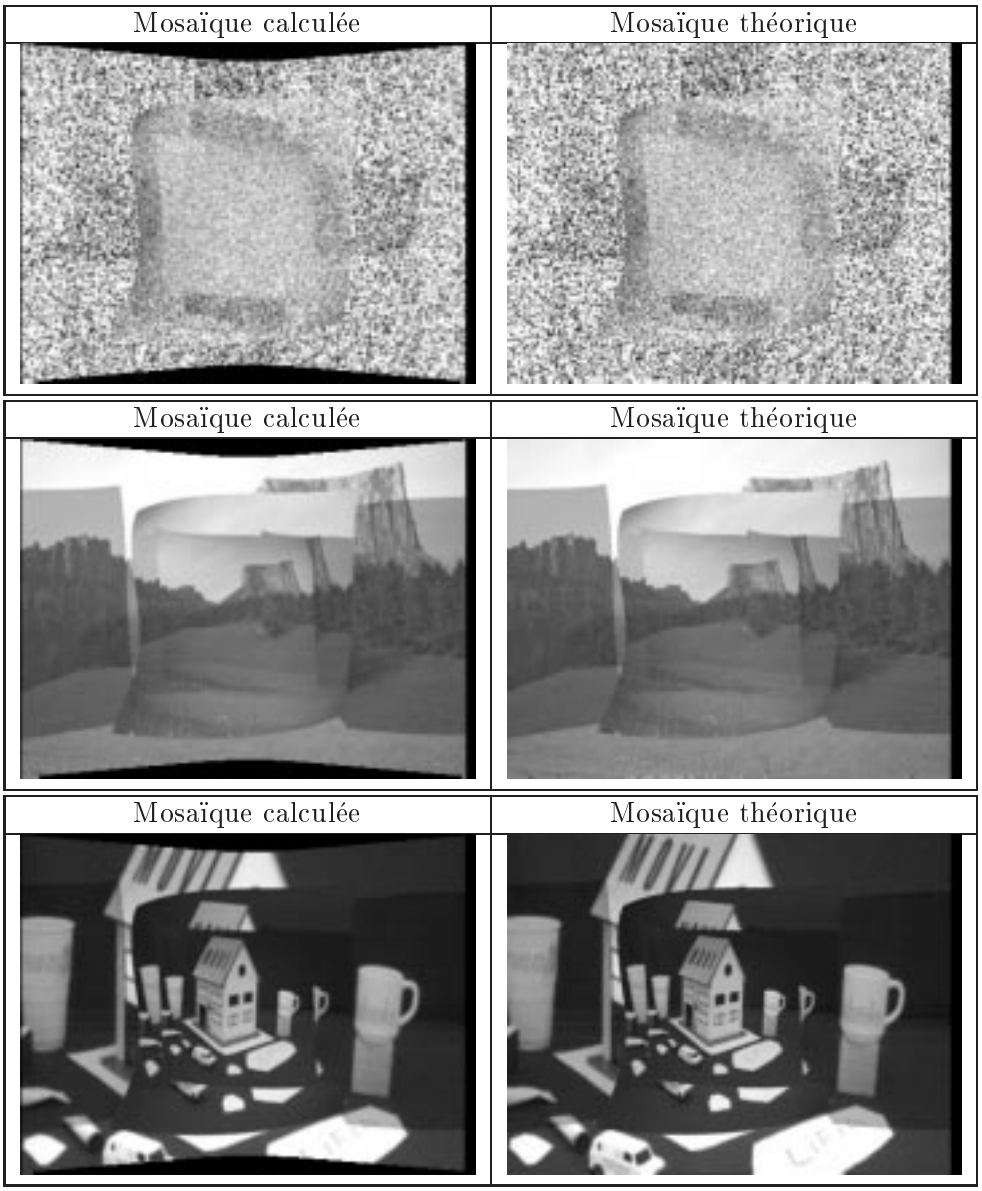


FIG. 4.7: *Comparaison visuelle des mosaïques calculées par transfert aux mosaïques théoriques obtenues avec* POV–*Ray.*

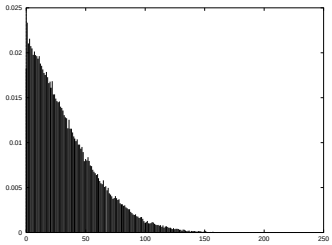
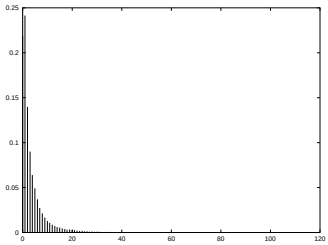
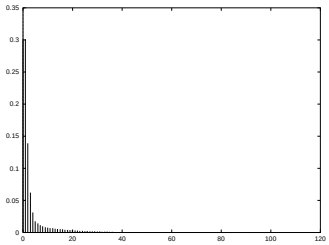
Texture aléatoire	Texture d'extérieur	Texture d'intérieur
Erreur absolue moyenne = 33	Erreur absolue moyenne = 4	Erreur absolue moyenne = 4
Erreur absolue médiane = 27	Erreur absolue médiane = 2	Erreur absolue médiane = 1
		

FIG. 4.8: *Comparaison numérique des mosaïques calculées par transfert, aux mosaïques théoriques obtenues avec* POV–*Ray : distribution des erreurs absolues de niveaux de gris, pixel à pixel.*

Les homographies étant calculées très précisément, il est normal que les images synthétiques soient presque indiscernables des images réelles. Nous constatons cependant un léger effet de flou, qui provient de la méthode de composition : pour chaque pixel de la mosaïque synthétisée contribuent cinq pixels des images de référence. Chacune de ces cinq intensités étant obtenue par combinaison linéaire de quatre pixels voisins, il peut arriver que les contours francs soient légèrement adoucis, et ceci est particulièrement sensible sur l'image à texture aléatoire : interpoler des valeurs de niveaux de gris fait perdre à l'image sa texture caractéristique (qui est discontinue en chaque pixel), et lisse le résultat. C'est pourquoi nous obtenons une EAM de 33 niveaux de gris, alors que les images sont très semblables à l'œil. Ceci corrobore notre remarque sur le fait que le critère EAM était peu significatif pour des images de bruit gaussien.

4.4.2 Tests sur images réelles

Pour effectuer des tests sur images réelles, nous avons utilisé des photographies du bâtiment de l'INRIA à Montbonnot, prises sans précaution particulière : l'appareil était tenu à main levé, les rotations étaient approximatives et leurs axes ne passaient pas nécessairement par le centre optique de l'appareil. Les photographies ont ensuite été transférées sur photo-CD, et utilisées à une résolution de 342×512. Les images originales sont montrées figure 4.9.

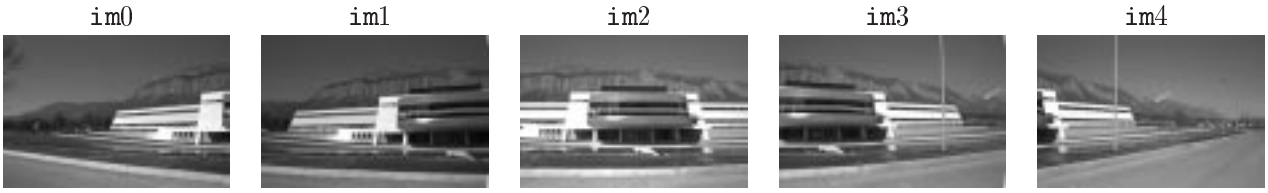


FIG. 4.9: *Cinq images réelles, approximativement de même centre optique.*

Sur ces cinq images, nous avons extrait des points d'intérêt, puis nous avons apparié

ces points, affiné les appariements, et calculé les 4 homographies liant les images (toujours par un calcul normalisé et robuste). À l'aide de ces homographies, nous avons transféré toutes les images dans le repère de l'image centrale, et composé une mosaïque selon la méthode déjà expliquée. La mosaïque construite est de taille 636×1926, et se trouve en figure 4.10.

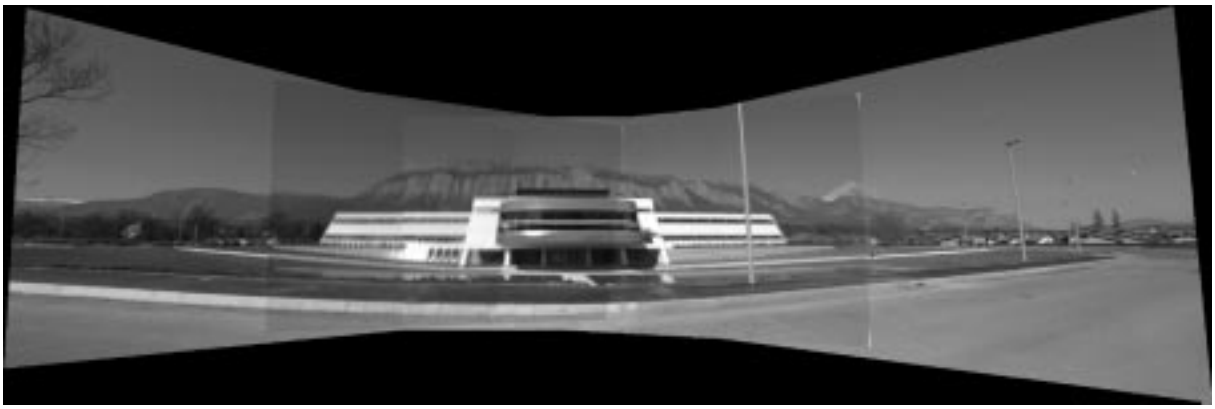


FIG. 4.10: *Mosaïque synthétisée à partir des images de la figure 4.9.*

Nous ne pouvons pas effectuer de test quantitatif ici, car nous ne disposons pas des données exactes théoriques, c.-à-d. une vraie photographie panoramique couvrant les 150° de la scène. Sur des critères visuels, nous pouvons cependant constater une bonne superposition des images, sauf pour la partie située dans le deuxième quart gauche de l'image, où le bâtiment semble dédoublé. Cela provient de l'image `im1`, qui n'est pas parfaitement en correspondance homographique avec les autres images de la séquence. De plus, le trottoir n'est pas rectiligne, et cela provient des images de référence : l'appareil photo ne réalise pas une projection perspective pure, et le trottoir est déjà courbé dans les images, ce qui est particulièrement visible dans l'image de référence `im2`.

Les différences d'intensité des images de référence apparaissent clairement, ainsi que les coins blancs situés en haut à droite et en bas à droite des images `im1` et `im3`, dus à un mauvais développement des photographies. En égalisant les intensités des images, et en ne prenant pas en compte les coins blancs, nous obtenons la nouvelle mosaïque en figure 4.11.

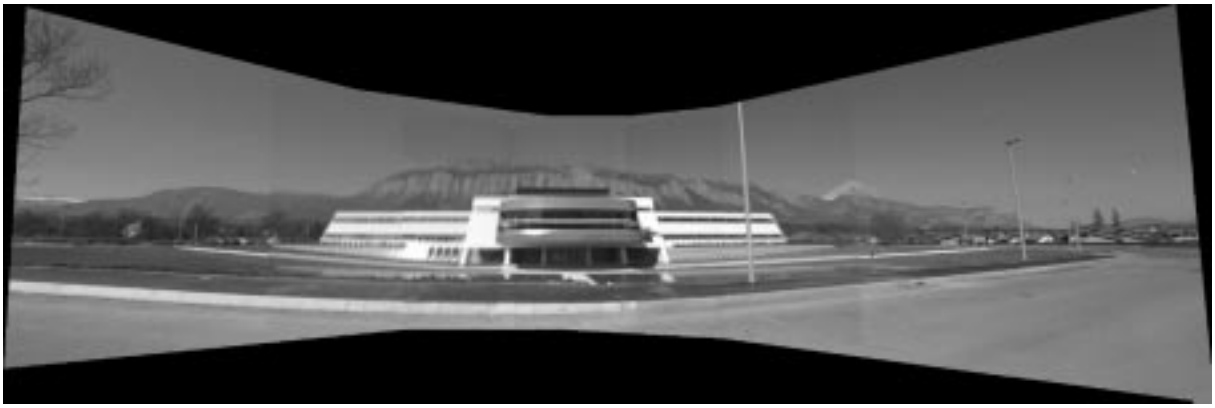


FIG. 4.11: *Mosaïque synthétisée à partir des images de la figure 4.9, avec prétraitement: égalisation des intensités, et extraction des coins blancs des images* `im1` *et* `im3`.

4.4.3 Remarques finales sur les mosaïques

Nous avons vu que la constitution de mosaïques était assez simple et stable pour être utilisée de façon automatique, et que la technique donnait facilement des résultats de très bonne qualité.

Toute forme d'étalonnage est inutile pour de telles méthodes de *recalage*, où les images de référence sont transférées dans le repère de l'une d'entre elles. Si nous voulions les transférer dans un repère totalement arbitraire, il nous faudrait spécifier la position de 4 points manuellement, et nous ne serions pas certains de produire une image géométriquement plausible.

Le problème est exactement similaire à celui de la synthèse 3D à partir d'images stéréoscopiques : l'étalonnage est inutile en théorie, ou pour des opérations de recalage ; mais si l'on veut pouvoir générer des vues arbitraires et géométriquement plausibles[1], tournées d'un angle mesuré en radians autour d'un axe positionné dans un espace 3D mesurable, alors l'étalonnage métrique devient nécessaire. Ce sont les méthodes employées dans QuickTime VR, où l'on dispose d'une connaissance approximative des paramètres optiques des caméras. QuickTime VR inclut aussi des algorithmes astucieux pour accélérer la synthèse [Che 95a] : les images sont pré-fusionnées sur un cylindre-image, ce qui permet un accès rapide lors des mouvements panoramiques de caméra, et fait l'objet d'un brevet [Che 95b] (voir aussi la figure 2.4, p. 20).

Dans la section suivante, nous revenons au cas stéréoscopique pour aborder la construction de modèles tridimensionnels. Le mot «modèle» doit être ici entendu au sens de «représentation 3D », et non au sens CAO de «modèle mathématique paramétré». En effet, nous n'obtiendrons pas une représentation symbolique de la scène en termes d'objets

1. Les images produites sont de toute façon géométriquement *correctes*, c.-à-d. ce sont bien les images qu'observerait une caméra virtuelle. Cependant, elles ne sont pas nécessairement *plausibles*, car la caméra virtuelle peut se situer à une position quelconque, par exemple derrière la surface de la scène observée, et posséder des paramètres optiques arbitraires : focale infinie (caméra affine), ou négative (image renversée), centre optique hors du cadre de l'image, image étirée, et/ou inclinée.

(ce n'est pas notre but), mais plutôt un ensemble de données numériques permettant de représenter certains aspects de la scène.

4.5 Construction d'une représentation 3D

En l'absence d'étalonnage fort, aucune manipulation de la scène n'est possible avec les outils usuels (éditeur de modèles, visualisateur), qui manipulent tous des données 3D «standard» (euclidiennes, et non simplement projectives). On ne peut pas facilement visualiser la scène, ni l'éditer, ni la texturer, ni la mixer avec des scènes 3D existantes.

Pour ces raisons, nous décidons autant que possible d'étalonner fortement nos caméras. Deux alternatives se présentent.

1. Nous disposons déjà des matrices de projection M_i dans les images, car elles sont fournies avec les données. C'est le cas des images synthétiques, et de la séquence du château de CMU.

2. Nous ne disposons pas des matrices de projection. Comme nous l'avons précisé en introduction, nous nous autorisons à avoir une connaissance approximative des paramètres d'étalonnage, résumés dans la matrice $K_{3\times 3}$ des paramètres intrinsèques. Aussi, nous pouvons à partir de deux images calculer la matrice fondamentale $F_{1,2}$, et à l'aide de $F_{1,2}$ et K calculer les matrices de projection M_1 et M_2 dans les deux images, permettant une reconstruction euclidienne. Cela est décrit ci-dessous, en 4.5.1.

   Une autre approche, dont nous avons déjà parlé, serait d'utiliser des connaissances *a priori* sur l'image, pour fixer des contraintes euclidiennes sur les éléments observés. Nous n'aurions alors pas besoin de la connaissance de K (méthode de B. Boufama, [Bou 94].)

4.5.1 Calcul des matrices de projection

Dans le cas binoculaire, connaître la géométrie épipolaire entre deux images permet d'établir une reconstruction *projective* des points appariés. En effet, on peut à partir de la matrice fondamentale $F_{1,2}$ calculer deux matrices de projection M_1 et M_2, compatibles avec cette géométrie épipolaire, par l'équation 4.9. Dans cette équation, e_2 est l'épipôle dans l'image 2 ($e_2 = Ker(F_{1,2})$), $[e_2]_\times$ représente la matrice 3×3 associée au produit vectoriel par e_2, b est un vecteur 3×1 quelconque, et c un scalaire quelconque [Rot 95, Bob 96]. Rappelons qu'inversement, $F_{1,2}$ peut être calculée à partir des matrices M_1 et M_2 (voir équation 3.60, p. 134).

$$\begin{cases} M_1 &= \left(\boxed{Id_{3\times 3}} \,\middle\|\, \boxed{0} \right) \\ \\ M_2 &= \left(\boxed{[e_2]_\times F_{1,2} + e_2 b} \,\middle\|\, \boxed{ce_2} \right) \end{cases} \tag{4.9}$$

Si nous disposons des paramètres intrinsèques de la caméra, nous pouvons aussi, à partir de $F_{1,2}$, calculer deux matrices de projection dans les images M_1 et M_2 compatibles, et permettant cette fois une reconstruction *euclidienne* par triangulation. Si les paramètres intrinsèques sont représentés par une matrice K, on calcule la matrice essentielle E par l'équation 4.10.

$$\begin{cases} K &= \begin{pmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \\ \\ E &= K^T F_{1,2} K \end{cases} \tag{4.10}$$

La matrice essentielle peut être décomposée en une rotation et une translation. Selon la méthode exposée par Q.T. Luong [Luo 92] et en suivant ses propres notations, E est d'abord décomposée en valeurs singulières (SVD) par $E = \Delta\Lambda\Theta^T$. Les matrices Δ et Θ étant des matrices orthogonales, on peut écrire l'équation 4.12.

$$E = (\Delta T_1 \Delta^T).(\Delta R_1 \Theta^T) \tag{4.11}$$

$$= T.R = \begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix} .R \tag{4.12}$$

$\Lambda = T_1 R_1$ doit être une décomposition en une matrice symétrique et une matrice orthogonale. E possédant deux valeurs propres non-nulles et égales, on à $E = \mathrm{diag}(1, 1, 0)$ à un facteur près, donc les matrices T_1 et R_1 de l'équation 4.13 sont des solutions acceptables.

$$T_1 = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad R_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \tag{4.13}$$

R doit être une matrice de rotation, donc $\det(R) = +1$. Si $\det(R) = -1$, on change R_1 en $-R_1$. Enfin, de la translation T et de la rotation R obtenues, nous déduisons M_1 et M_2 par l'équation 4.14.

$$\begin{cases} M_1 &= K(\ Id_{3\times 3} \mid 0\ ) \\ M_2 &= K(\ R \mid t\ ) \end{cases} \tag{4.14}$$

Ces deux matrices nous permettent de reconstruire les appariements par triangulation. Si tous[2] les points 3D obtenus se situent derrière l'une des caméras, alors il faut recommencer les calculs en utilisant cette fois les *transposées* de T_1 et R_1. Il faut de nouveau vérifier que $\det(R) = +1$, et changer le signe de R_1 si nécessaire. Ces différents cas de figure correspondent au fait qu'il est impossible de déterminer si les images se forment en avant du point focal (modélisation habituelle), ou en arrière du point focal, mais renversées.

Dans le cas binoculaire, nous parvenons donc, connaissant la matrice fondamentale et les paramètres intrinsèques, à établir un étalonnage fort du couple de caméras. Nous pouvons en effet déterminer la rotation et la translation relative des deux caméras, et calculer une reconstruction euclidienne de tous les points appariés dans les deux images.

Cette reconstruction est obtenue dans un repère euclidien arbitraire. Aussi, il n'est pas possible d'étendre cette méthode au cas de $N \geq 3$ images. La figure 4.12 montre la difficulté du problème pour une configuration à trois caméras.

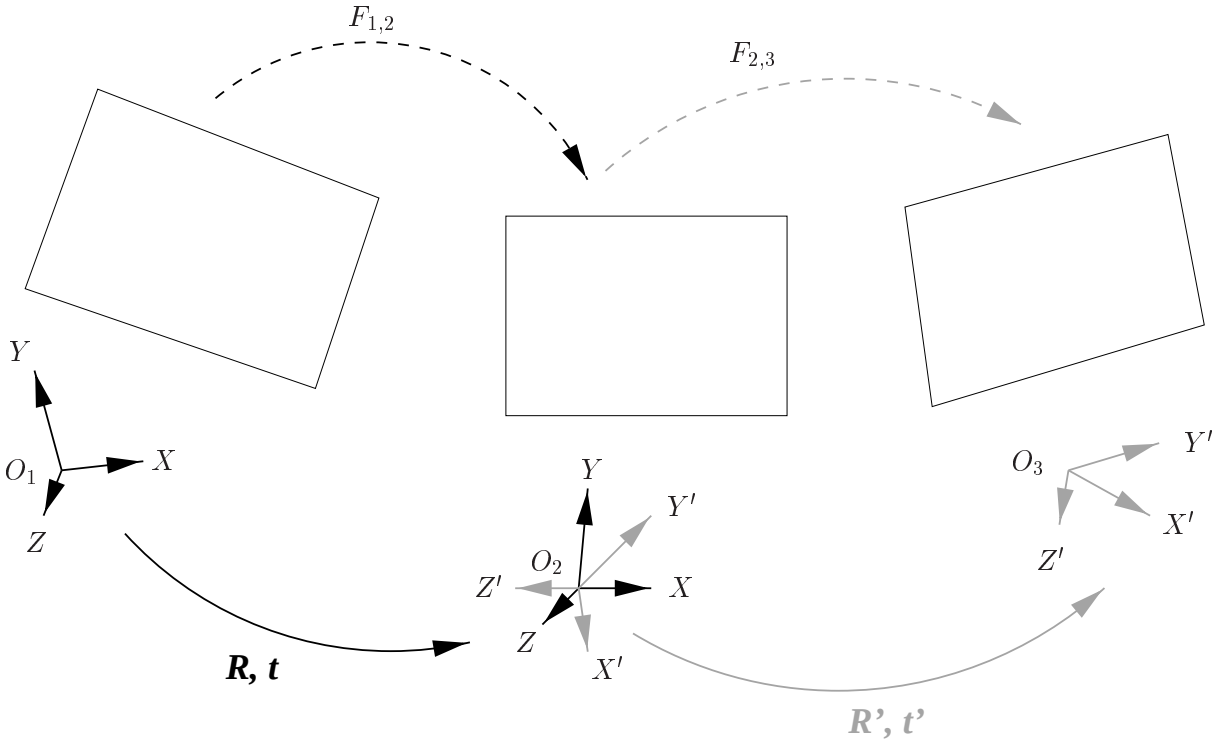


FIG. 4.12: *Lorsque les positions relatives de trois images ne sont définies que par leurs géométries épipolaires (donc sur une base binoculaire), il est impossible d'assurer la cohérence globale des reconstructions* 3D *(voir texte).*

2. Dans notre cas, il existe des appariements incorrects, et nous assurons plutôt que la *majorité* des points 3D obtenus se situent devant les deux caméras.

Comme précédemment, supposons connus et constants les paramètres intrinsèques des caméras. À l'aide de $F_{1,2}$, on peut établir une reconstruction des points appariés entre les images 1 et 2, dans un repère euclidien arbitraire. À l'aide de $F_{2,3}$, on peut aussi établir une reconstruction des points appariés entre les images 2 et 3, dans un *autre* repère euclidien arbitraire. Il est impossible de ramener ces reconstructions dans le même repère. En effet, il subsiste toujours une ambiguïté fondamentale entre la distance de prise de vue, et la taille de la scène observée : si la scène est plus grande, mais vue de plus loin, alors les images formées sur les caméras sont inchangées, et les matrices fondamentales ne varient pas. Il est donc impossible de déterminer le *facteur d'échelle* à partir de cette méthode d'étalonnage. Il ne suffit donc pas de composer les rotations et les translations entre les repères des images 1, 2 et 3 pour obtenir une reconstruction valable dans les trois images, car ces transformations ne sont définies qu'à un facteur d'échelle près, potentiellement différent pour chaque couple d'images.

Un raisonnement simple sur le nombre de degrés de liberté du problème nous amène au même résultat. Une matrice fondamentale détermine 7 paramètres : elle est de taille 3×3, elle est définie à un facteur multiplicatif global près, et elle est de rang 2, ce qui fixe une relation trilinéaire sur ses coefficients. Les trois matrices fondamentales $F_{1,2}$, $F_{2,3}$ et $F_{1,3}$ que l'on peut calculer entre trois images nous fournissent donc 21 coefficients.

Or, les matrices de projection 3×4 dans les trois images déterminent chacune 11 coefficients (elles sont aussi définies à un facteur multiplicatif global près), et la reconstruction projective obtenue serait définie dans l'espace à une homographie 4×4 près (soient 15 coefficients). Ainsi, si l'on considère les matrices de projection, et non plus les matrices fondamentales, la géométrie projective de trois images est définie par $33 - 15 = 18$ coefficients, et non 21.

Les matrices fondamentales existant entre trois images sont donc liées par une condition mathématique. Cela explique pourquoi nous ne pouvons pas retrouver la structure d'une scène tridimensionnelle vue dans trois images en ne considérant que les matrices fondamentales, c.-à-d. en ne considérant les images que deux par deux. C'est pourquoi les tenseurs trilinéaires sont utilisés pour ces configurations ; comme nous l'avons vu, un tenseur trilinéaire contient exactement 18 degrés de liberté, et constitue un ensemble minimal de paramètres pour décrire la géométrie relative de trois images dans leur ensemble. Utiliser un tenseur trilinéaire et les paramètres intrinsèques pourrait donc aider à résoudre l'étalonnage pour trois images, mais le problème ne serait pas résolu pour $N \geq 4$ images. Dans notre application, par simplicité, nous nous limiterons donc au cas binoculaire.

4.5.2 Reconstruction robuste

Nous disposons d'appariements multi-oculaires partiellement définis, parfois imprécis, ou même faux. Nous disposons également des matrices de projection dans les images, et ceci nous permet de reconstruire les points 3D un par un, par triangulation. La méthode classique est de calculer une solution aux moindres carrés sur toutes les équations de projection (figure 4.13).

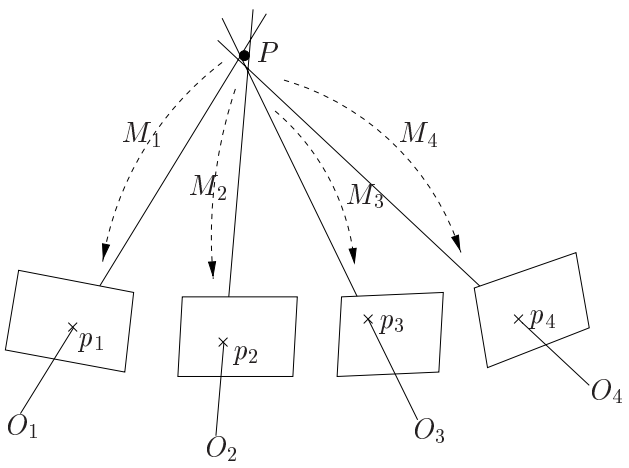


FIG. 4.13: *Reconstruction d'un point* 3D *par triangulation, aux moindres carrés.*

Les équations de projection sont explicitées en équations 4.15 et 4.16, dans le cas de 4 images. L'équation 4.16 est aisément résolue aux moindres carrés, et permet de calculer P en coordonnées homogènes. Ceci permet de traiter le cas de lignes de vue toutes parallèles, et dont le point d'intersection se situe donc à l'infini, soit $P^t = 0$.

$$\begin{cases} p_1 &= M_1.P \\ p_2 &= M_2.P \\ p_3 &= M_3.P \\ p_4 &= M_4.P \end{cases} \tag{4.15}$$

$$\begin{pmatrix} m^1_{31}p^x_1 - m^1_{11} & m^1_{32}p^x_1 - m^1_{12} & m^1_{33}p^x_1 - m^1_{13} & m^1_{34}p^x_1 - m^1_{14} \\ m^1_{31}p^y_1 - m^1_{21} & m^1_{32}p^y_1 - m^1_{22} & m^1_{33}p^y_1 - m^1_{23} & m^1_{34}p^y_1 - m^1_{24} \\ m^2_{31}p^x_2 - m^2_{11} & m^2_{32}p^x_2 - m^2_{12} & m^2_{33}p^x_2 - m^2_{13} & m^2_{34}p^x_2 - m^2_{14} \\ m^2_{31}p^y_2 - m^2_{21} & m^2_{32}p^y_2 - m^2_{22} & m^2_{33}p^y_2 - m^2_{23} & m^2_{34}p^y_2 - m^2_{24} \\ m^3_{31}p^x_3 - m^3_{11} & m^3_{32}p^x_3 - m^3_{12} & m^3_{33}p^x_3 - m^3_{13} & m^3_{34}p^x_3 - m^3_{14} \\ m^3_{31}p^y_3 - m^3_{21} & m^3_{32}p^y_3 - m^3_{22} & m^3_{33}p^y_3 - m^3_{23} & m^3_{34}p^y_3 - m^3_{24} \\ m^4_{31}p^x_4 - m^4_{11} & m^4_{32}p^x_4 - m^4_{12} & m^4_{33}p^x_4 - m^4_{13} & m^4_{34}p^x_4 - m^4_{14} \\ m^4_{31}p^y_4 - m^4_{21} & m^4_{32}p^y_4 - m^4_{22} & m^4_{33}p^y_4 - m^4_{23} & m^4_{34}p^y_4 - m^4_{24} \end{pmatrix} \cdot \begin{pmatrix} P^x \\ P^y \\ P^z \\ P^t \end{pmatrix} = 0_{8\times 1} \tag{4.16}$$

La reconstruction aux moindres carrés a un sens dans un espace euclidien, car elle fournit le point P le plus proche de toutes les lignes de vue simultanément, au sens de la distance euclidienne. Pour une reconstruction projective, une telle méthode n'a pas de sens. On ne peut cependant pas utiliser non plus de méthode purement algébrique, car à cause des imprécisions d'appariement, il n'est pas certain que les lignes de vues soient sécantes. Aussi, pour la reconstruction projective binoculaire, une méthode a été proposée par R. Hartley et P. Sturm dans [Har 94]. Elle consiste à aligner les appariements précisément sur les épipolaires, ce qui garantit que les lignes de vues se coupent. La méthode

est expliquée en figure 4.14. Sur ce schéma, le faisceau d'épipolaires conjuguées D_1, D_2 est paramétré par θ ; $d(\theta) = d(p, D_1(\theta)) + d(q, D_2(\theta))$ est la distance des points p et q à leurs épipolaires respectives ; il suffit donc de calculer $\theta_0 = \operatorname{argmin} d(\theta)$, ce qui fournit le couple d'épipolaires conjuguées le plus proche à la fois de p dans l'image 1 et de q dans l'image 2. Le point p se projette sur $D_1(\theta_0)$ en p', et q sur $D_2(\theta_0)$ en q'. Alors (p', q') est un appariement respectant exactement la géométrie épipolaire.

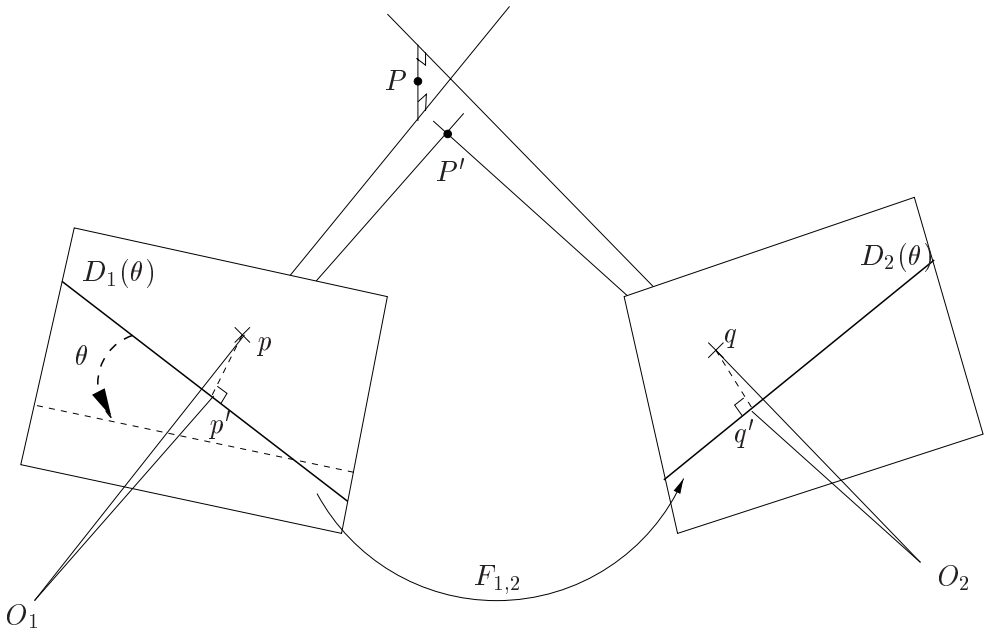


FIG. 4.14: *Méthode d'alignement épipolaire proposée par R. Hartley et P. Sturm [Har 94] (voir texte).*

Cette méthode est rapide : elle rectifie de l'ordre de 1500 appariements par seconde. Elle est numériquement très bien conditionnée, car $d(\theta)$ est un polynôme de degré 6, dont il est aisé de trouver le minimum. De plus, nous verrons lors de nos expérimentations qu'il n'est pas nécessaire d'utiliser une géométrie épipolaire très précise. Malheureusement, le système ne fonctionne que dans le cas binoculaire.

La méthode de reconstruction euclidienne multi-images aux moindres carrés, en revanche, fonctionne bien dans le cas général. Elle intègre toutes les mesures, mais ceci n'est pas forcément utile, car souvent, la plus grande précision est simplement donnée par les positions des appariements extrêmes : dans notre cas, il aurait peut-être été aussi efficace et précis de ne mener qu'une reconstruction à partir des points p_1 et p_4 seulement, car ce sont les plus éloignés.

Enfin, il se peut que cette méthode ne fonctionne pas, à cause de mauvais appariements. Sur la figure 4.15, nous voyons qu'un seul mauvais point dans l'appariement (celui de l'image 3, ici), peut fausser complètement le résultat, car celui-ci est calculé aux moindres carrés, et tente d'obtenir une solution satisfaisant au mieux toutes les contraintes simultanément.

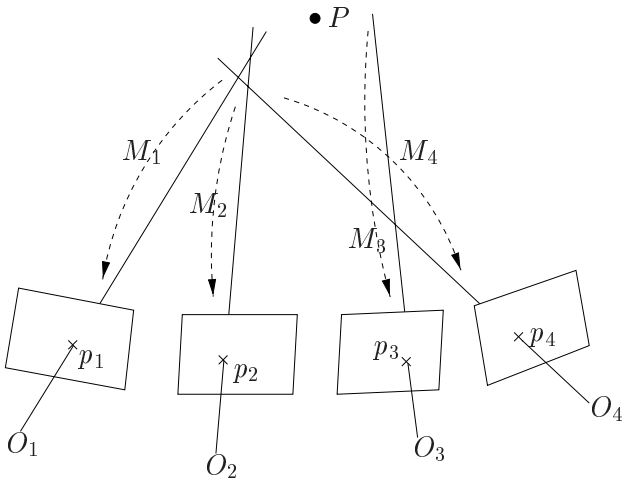


FIG. 4.15: *Reconstruction d'un point* 3D *par triangulation, aux moindres carrés. Un seul point faux dans l'appariement multi-oculaire suffit à fausser le résultat.*

Nous proposons donc une nouvelle méthode de triangulation robuste, fournissant une solution aux moindres carrés médians, par tirages aléatoires. La procédure est classique, nous la donnons ci-dessous, en supposant que le taux de faux points dans chaque appariement multi-oculaire est égal à τ.

1. Choisir dans l'appariement le nombre minimum de points nécessaires pour effectuer la triangulation (donc 2 points).
2. Reconstruire le point P aux moindres carrés.
3. Reprojeter P dans toutes les images, et calculer l'erreur médiane à τ de reprojection dans ces images.
4. Répéter N fois les étapes 1 à 3 avec deux autres points, tirés aléatoirement, et conserver la reconstruction P d'erreur médiane à τ minimale.
5. En supposant que ces erreurs suivent une distribution normale, on peut calculer l'écart-type de cette distribution $\sigma = 1.48026\,\mathrm{med}$.
6. Rejeter tous les points pour lesquels l'erreur de reprojection de P est supérieure à 2.5σ (ce qui indique qu'ils ont 98.76 % de chances de ne pas appartenir au modèle d'erreur).
7. Pour tous les points restants, effectuer une reconstruction multi-oculaire aux moindres carrés, comme précédemment.

Le nombre de tirages est donné par :

$$1 - (1 - (1 - \tau)^2)^N = 0.999 \tag{4.17}$$

Pour $\tau = 0.5$, on doit effectuer $N = 24$ tirages.

Comme nous le verrons lors de l'évaluation, cette procédure fournit une reconstruction plus correcte, plus précise, et permet en plus de rejeter de faux appariements. Notons que les appariements rejetés sont ceux ne respectant pas les contraintes multi-linéaires, ce qui justifie *a posteriori* de ne pas les avoir testées auparavant (par exemple, les contraintes trilinéaires lors de l'appariement).

4.5.3 Construction d'un maillage

Attention, nous employons dans cette partie le terme «triangulation» comme «construction d'un maillage triangulaire», et non plus comme «reconstruction tridimensionnelle par intersection des lignes de vue».

Nous disposons à cette étape d'un nuage de points 3D reconstruits à partir des images de référence. Nous pouvons attribuer à chacun de ces points sa couleur dans l'une des images de référence, et reprojeter ce modèle sur le plan-image d'une caméra virtuelle pour obtenir des images synthétiques. Nous testerons cette méthode.

Comme nous l'avons vu, il peut être plus avantageux de disposer d'un modèle plus élaboré, constitué de facettes triangulaires texturées. Les facettes triangulaires sont en effet des structures garanties être planes, et du matériel spécifique permet de réaliser des synthèses très rapides de tels modèles. De plus, le modèle en facettes est connexe par morceaux, alors que le modèle en points est entièrement discret, avec les problèmes que nous verrons lors de la re-synthèse.

Il est difficile d'obtenir de façon automatique un maillage triangulaire d'une surface définie par des points 3D discrets. Nous avons déjà cité les travaux de P. Fua, mais ils se limitent au calcul d'une seule surface (connexe, et assez lisse). Nous pourrions envisager de réaliser une triangulation de Delaunay en quelques points choisis, par exemple les points d'intérêt de la scène, qui représentent souvent des angles, donc des changements de disparité. Mais nous ne pouvons pas contrôler de façon suffisamment fine la triangulation pour éviter certains phénomènes non désirables, comme le recouvrement de zones occultées (figure 4.16). De tels triangles n'auront pas un aspect réaliste lors de la phase de synthèse, sous d'autres angles.

Triangulation obtenue Triangulation souhaitée

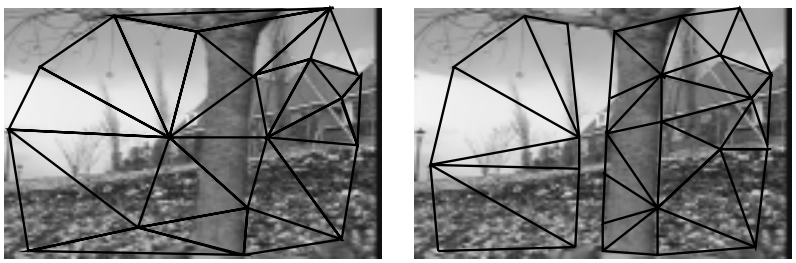


FIG. 4.16: *La construction incontrôlée de triangles peut mener à des phénomènes non désirables, comme le recouvrement des zones occultées.*

Notre algorithme de triangulation repose sur une autre méthode : nous triangulons autant de points que possible, puis nous fusionnons les petits triangles sur des critères de coplanarité. De cette façon, nous sommes certains de ne pas créer de triangle se situant à cheval sur des régions de l'image de profondeurs très différentes. De plus, nous utiliserons une reprojection des points 3D par un Z-buffer, afin de déterminer quels sont les points visibles dans chaque image, pour ne pas créer de triangle dans les zones occultées. Pour créer une triangulation dans l'une des images de référence, l'algorithme est le suivant.

1. Projeter tous les points 3D dans l'image, en conservant seulement ceux qui sont visibles : dans les limites de l'image, et plus proches que tous les autres points se projetant sur le même pixel (Z-buffer). Cela constitue la carte des pixels renseignés (figure 4.17).

2. Pour chaque groupe de 4 pixels de l'image (renseignés ou non), créer un *patch* carré, dont les sommets sont les centres de ces quatre pixels. Ce *patch* est supposé être plan (figure 4.18).

3. Tenter de fusionner récursivement les *patches* par groupe de quatre, sur des critères de coplanarité, explicités plus loin (figure 4.19).

4. Lorsqu'on a regroupé tous les *patches* qu'il était possible de fusionner, déplacer les sommets du maillage vers les pixels renseignés les plus proches (figure 4.20).

5. Découper chacun des *patches* obtenus en deux triangles, le long de leur diagonale (figure 4.21).

Nous obtenons en fin de compte le maillage présenté en figure 4.22. La surface obtenue couvre autant que possible les parties renseignées de l'image, évite les zones non-renseignées, et les sommets des triangles obtenus sont tous des points renseignés.

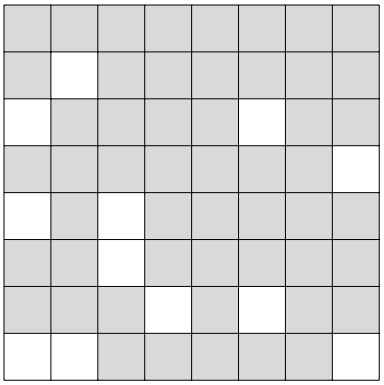


FIG. 4.17: *Maillage, étape 1 : carte des pixels renseignés. Sur cette image* 8×8*, seuls les pixels grisés sont renseignés (c.-à-d. correspondent à la projection d'un point* 3D*).*

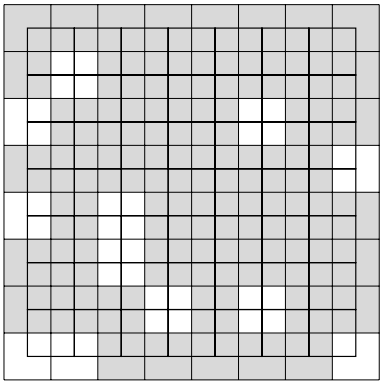


FIG. 4.18: *Maillage, étape 2 : création des patches* 2×2.

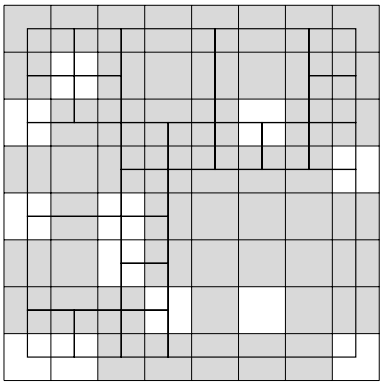


FIG. 4.19: *Maillage, étape 3 : fusion des patches.*

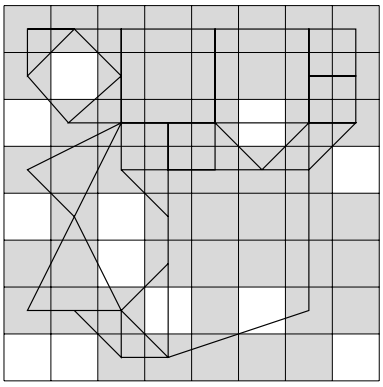


FIG. 4.20: *Maillage, étape 4 : déplacement des sommets des patches vers les pixels renseignés les plus proches.*

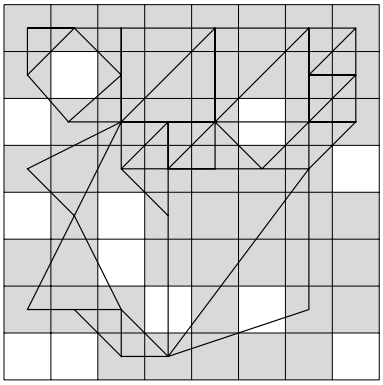


FIG. 4.21: *Maillage, étape 5 : découpage des patches quadrilatéraux en triangles.*

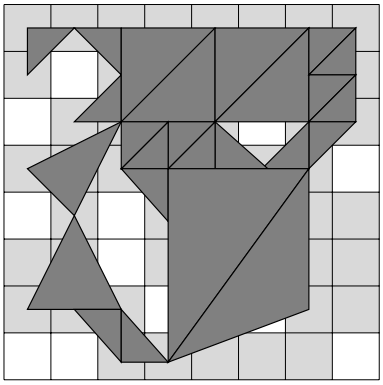


FIG. 4.22: *Résultat final du maillage.*

La structure de données est un arbre quaternaire (*quadtree*), dont les nœuds représentent des zones carrées de l'image. Les critères pour fusionner les 4 fils d'un nœud sont les suivants :

1. Tous les fils doivent être des feuilles.

2. Il faut qu'au moins deux des pixels recouverts par le plan soient renseignés. Autoriser les cellules à être fusionnées même si certains points couverts ne sont pas définis, permet de couvrir les endroits où aucun point ne se projette par suite d'erreurs d'appariement, pourvu qu'ils restent de taille raisonnable.

3. Il faut que les pixels recouverts par les 4 fils correspondent à des points 3D coplanaires. La coplanarité est testée de façon robuste par une méthode usuelle de moindres carrés médians, avec un taux de faux points 3D fixé à 50 % : on effectue plusieurs tirages aléatoires de 4 points 3D, et à chaque fois, on calcule un plan aux moindres carrés (nous utilisons 4 points au lieu de 3, pour éviter de trouver la solution identiquement nulle : $0\ x + 0\ y + 0\ z + 0 = 0$). Parmi les tirages aléatoires,

celui présentant l'erreur médiane minimale est conservé, et le *patch* est déclaré plan si cette erreur est inférieure à un seuil, fixé par l'utilisateur.

Enfin, parmi les triangles obtenus ne sont conservés que ceux répondant aux critères suivants :

- le triangle doit couvrir une surface de 16 pixels au moins (critère de taille) ;
- le triangle doit couvrir une zone où au moins la moitié des pixels sont renseignés (critère de pertinence).

Nous obtenons à chaque fois une triangulation valable sur une image de référence, et dans un certain domaine de validité avoisinant. Le programme de visualisation devrait donc changer de modèle selon le point de vue, affichant le modèle construit sur l'image de référence i si l'observateur se situe à proximité de la position de l'image de référence i. À cause des phénomènes d'occultation et de recouvrement, il serait de toute façon très difficile de réaliser une triangulation valide simultanément dans toutes les images de référence.

Cet algorithme sera testé sur nos images.

4.5.4 Calcul des textures

Supposons que dans la scène, deux triangles couvrent un carré contenant la lettre «H». Ce carré est vu incliné dans l'image de référence, et la distorsion perspective fait que les branches du «H» et les bords du carré ne sont pas parallèles, mais se coupent en un point de fuite (figure 4.23).

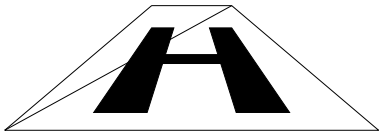


FIG. 4.23: *L'image de référence est un plan incliné, que le maillage triangulaire découpe en deux triangles.*

Si nous synthétisons une nouvelle vue de ce carré, de face, alors nous obtenons l'une des deux images présentées en figure 4.24, selon que le mapping de texture suit une projection affine, ou perspective.

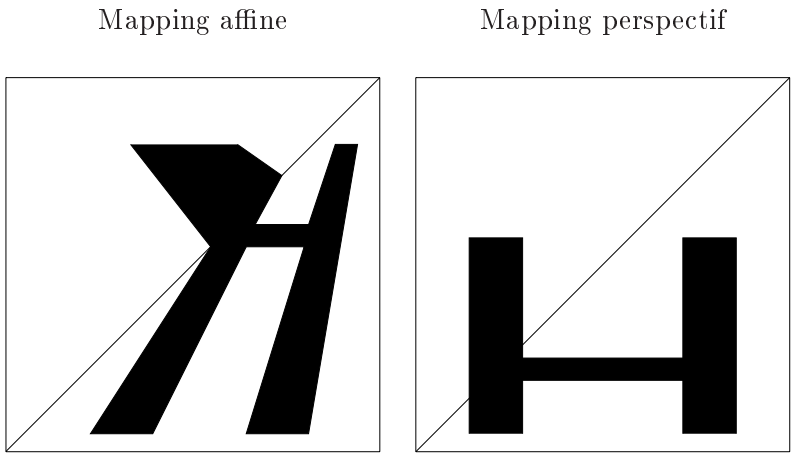


FIG. 4.24: *L'image synthétique obtenue dépend de la transformation de texture utilisée.*

Le mapping affine se réalise de façon directe, par calcul de coordonnées barycentriques, et il est souvent implémenté directement par le matériel : il suffit de spécifier à la carte graphique les positions 3D des trois sommets de chaque triangle, et la texture à appliquer, sous la forme d'un *bitmap*. Pour la plupart des cartes ou des applications graphiques, ce *bitmap* doit être carré, de dimensions $2^n \times 2^n$. C'est en particulier le cas pour quelques éditeurs VRML. Cela permet d'accélérer notablement l'affichage des textures, car le processus d'affichage peut facilement adapter la résolution des textures en fonction de la visibilité des triangles : les triangles les plus éloignés voient leur texture réduite d'un facteur 2^k, sans différence notable de qualité, simplement en n'utilisant que 1 pixel sur k dans le *bitmap* original (technique du *mip-mapping*).

Le mapping perspectif en revanche, demande plus de calculs. La transformation projective à appliquer aux textures ne peut pas être déduite des simples positions des sommets des triangles, car quatre points sont nécessaires pour définir une transformation projective du plan. Notre technique est la suivante : pour chaque triangle 3D de la scène, nous calculons le plan 128×128 couvrant au mieux ce triangle ; puis nous projetons ces 16384 points dans l'image de référence considérée, afin de déterminer leur couleur. Cela nous permet de construire un *bitmap* texturé et redressé, de taille 128×128, que nous appliquerons à notre triangle (cette taille a été choisie pour les raisons évoquées précédemment ; elle pourrait aussi être adaptée à la taille réelle du triangle). La figure 4.25 illustre les différentes étapes du calcul de la texture d'un triangle 3D.

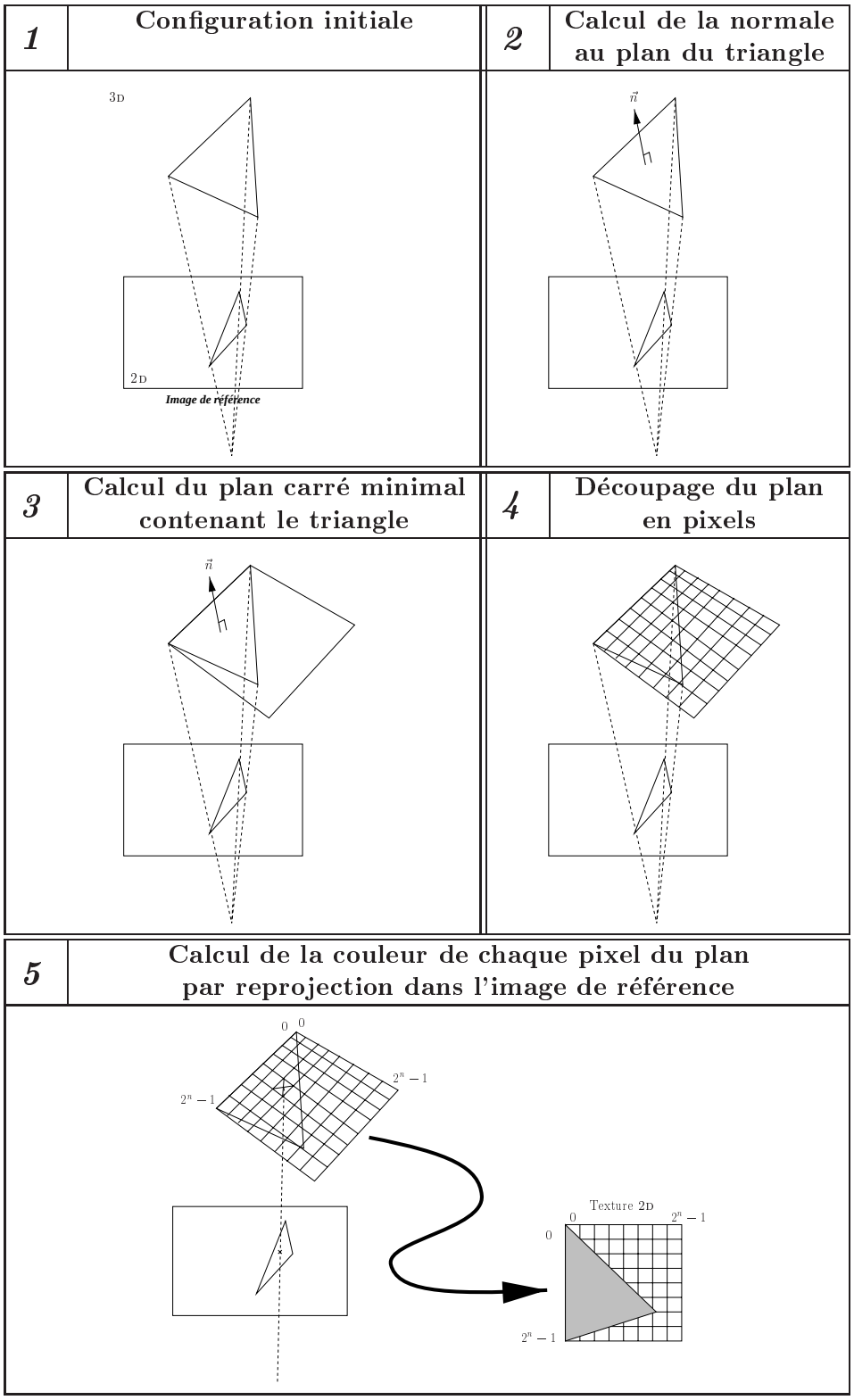


FIG. 4.25: *Étapes de la construction du bitmap décrivant la texture d'un triangle* 3D.

Le calcul d'un *bitmap* de texture 128×128 nécessite 0.2 seconde. Les images obtenues sont ensuite utilisées comme des textures standard pour VRWeb (mapping affine).

4.6 Évaluation sur images de synthèse — 1

L'ordre des opérations sera le suivant :

1. réaliser une reconstruction 3D ;
2. synthétiser des images à partir du nuage de points, évaluer leur qualité, et éventuellement décider de post-traitements ;
3. calculer le maillage triangulaire, et les textures ;
4. synthétiser des vues du modèle en triangles, et évaluer leur qualité.

4.6.1 Reconstruction 3D

Nous utiliserons dans un premier temps les matrices de projection fournies par POV–Ray, pour la reconstruction 3D multi-oculaire robuste.

Cette technique est appliquée aux appariements issus de l'affinage AFFZD4 (chapitre 3). Le tableau 4.9 montre l'erreur de reprojection obtenue après la phase de reconstruction aux moindres carrés, et après la phase de reconstruction aux moindres carrés médians. Le tableau 4.10 rappelle la qualité des appariements initiaux, et donne celle des appariements restants après robustification.

Points 3D	**Nombre**	**Erreur**						**Temps**
		< 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**max**	**CPU**
Reconstruction	91169	13.74	31.26	89.34	96.89	98.00	30.49	12.92
Reconstruction robuste	82942	19.53	37.87	92.39	99.64	99.98	10.40	259.40

TAB. 4.9: *Erreur de reprojection après la phase de reconstruction* 3D*, versions standard ou robuste.*

Appariements	**Nombre**	**Erreur**					**Temps**
		< 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**CPU**
Initiaux	91169	2.26	8.36	63.76	88.35	96.79	—
Après robustification	82942	2.11	8.05	63.97	89.36	98.31	259.40

TAB. 4.10: *Précision initiale de l'appariement, et précision des appariements restants, après la phase de reconstruction robuste.*

Concernant la reconstruction 3D (tableau 4.9), le gain est appréciable surtout pour les hautes précisions, puisque le taux de points ayant une erreur de reprojection inférieure à 0.05 pixel passe de 13.74% à 19.53%, et de 31.26% à 37.87% pour une erreur inférieure à

0.10 pixel. L'erreur maximale de reprojection passe de 30.49 à 10.40 pixels, et le temps de calcul reste raisonnable. Enfin, 9 % des points sont rejetés lors du calcul robuste. Les appariements, quant à eux (tableau 4.10), ne sont que très légèrement améliorés.

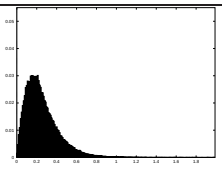
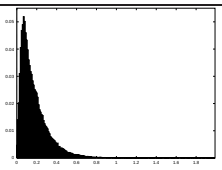
Dans le cas binoculaire, on ne peut bien sûr pas effectuer de reconstruction robuste telle que décrite ci-dessus. On peut cependant effectuer une étape d'ajustement épipolaire, décrite en 4.5.2, p. 166. L'évaluation que nous effectuons est la suivante :

1. dans les appariements initiaux (affinés, multi-oculaires), ne conserver que la liste L des appariements binoculaires entre les images `im0` et `im1` ;

2. calculer un ajustement épipolaire L' des appariements de L, à partir de la matrice fondamentale $F_{0,1}$ calculée à partir des images ;

3. évaluer L et L' vis-à-vis des matrices de projection réelles (celles calculées par POV–Ray lors de la création des images de référence).

Évaluer les reconstructions 3D obtenues ne nous apporterait rien, car l'évaluation des reconstructions 3D repose sur le calcul des erreurs de reprojection. Or, puisque les épipolaires s'intersectent, les erreurs de reprojection des points 3D reconstruits sont toutes strictement nulles. En revanche, évaluer les erreurs de reprojection des *appariements* des listes L ou L' a bien un sens, car un couple peut respecter parfaitement la géométrie épipolaire tout en étant mal apparié. Le tableau 4.11 donne les résultats de cette évaluation.

Appariements	**Erreur**					**Histogramme**
	< 0.05	< 0.10	< 0.50	< 1.00	< 2.00	
L	3.88	14.10	89.21	97.21	97.58	
L'	10.79	34.59	94.14	97.43	97.64	

TAB. 4.11: *Précision des appariements binoculaires entre* `im0` *et* `im1`*, avant et après ajustement épipolaire.*

Dans le cas binoculaire, l'étape d'ajustement épipolaire améliore donc considérablement la qualité des appariements, *même si la matrice fondamentale utilisée n'est pas exacte*. Dans ces tests en effet, nous avons réalisé l'ajustement épipolaire sur la base de la matrice fondamentale calculée à partir des images, alors que l'évaluation est menée par rapport aux matrices de projection réelles, calculées par POV–Ray.

4.6.2 Synthèse d'images à partir de points

Nous réalisons ici une synthèse de nouvelles images à partir des points 3D reconstruits à l'étape précédente :

- dans le cas de $N \geq 3$ images, après reconstruction robuste aux moindres carrés médians ;
- dans le cas binoculaire, après ajustement épipolaire, puis reconstruction aux moindres carrés.

4.6.2.1 Calcul de la couleur des points

Les points obtenus sont affectés d'une couleur, calculée comme la moyenne de la couleur des pixels correspondant aux appariements dans chaque image. Sur la figure 4.26, la couleur du point P est la moyenne des couleurs des pixels p_1, p_2, p_4 et p_6 (P ayant été reconstruit aux moindres carrés à partir des positions des ces 4 points).

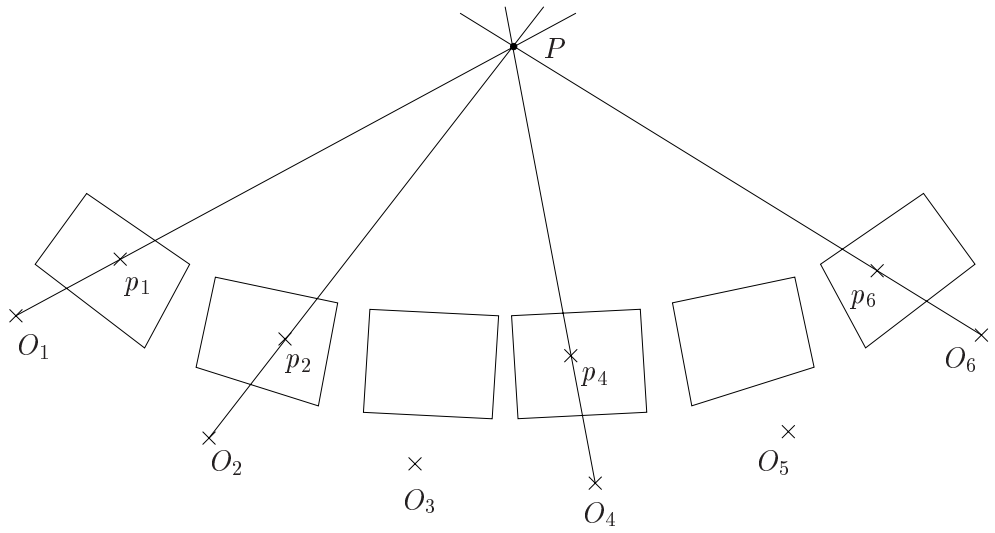


FIG. 4.26: *Principe du mixage de couleurs : P prend la couleur moyenne des pixels p_1, p_2, p_4 et p_6.*

Prendre la valeur moyenne de ces intensités revient à effectuer une interpolation, p. ex. sur les ruptures de disparité. En effet, si les 4 pixels appariés sont au centre de masques tels que décrits en figure 4.27 : 3 pixels blancs et 1 gris, alors il est logique que la couleur du point 3D P soit la moyenne (gris clair) de ces quatre couleurs, car P apparaît tantôt noir, tantôt blanc à la caméra : on minimise l'erreur moyenne. Comme la théorie le prévoit, c'est en effet le phénomène physiquement observé lors de la prise de vues : chaque élément de l'image représente une *moyenne* des intensités lumineuses observées sur une certaine surface. Pour cette raison, les contours contrastés d'une image ne sont jamais francs, ils sont légèrement flous. Cela est dû à la constitution des systèmes optiques (phénomènes de

diffraction), et le problème est encore amplifié par l'utilisation d'un capteur CCD, d'une résolution limitée. C'est d'ailleurs une composante importante du processus de perception, puisqu'en synthèse d'images, les contours doivent être anti-crénelés pour obtenir un effet plus réaliste. L'anti-crénelage (*anti-aliasing*) consiste à moyenner les intensités des pixels proches des contours contrastés de l'image, selon une pondération adéquate.

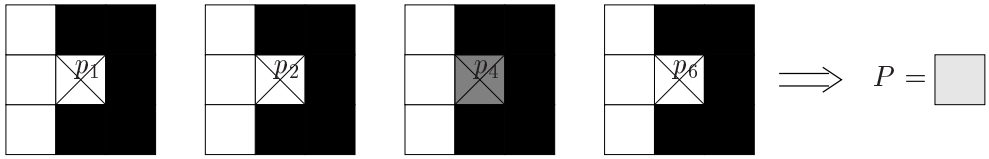


FIG. 4.27: *L'intensité du point* 3D *est calculée comme la moyenne des intensités de ses projections dans les images.*

4.6.2.2 Calcul de l'image projetée

Les points 3D sont projetés *via* une matrice de projection 3×4 sur le plan de la caméra virtuelle. Les points se projettent en des pixels non-entiers, et nous avons deux options :

1. arrondir au pixel le plus proche, et lui affecter la couleur du point 3D ;
2. déborder sur les 4 pixels les plus proches, et pondérer les contributions selon la surface recouverte.

La seconde méthode est illustrée sur la figure 4.28 : si un point 3D se projette en p, alors les 4 pixels les plus proches sont affectés, selon la surface recouverte par un carré 1×1 centré en p.

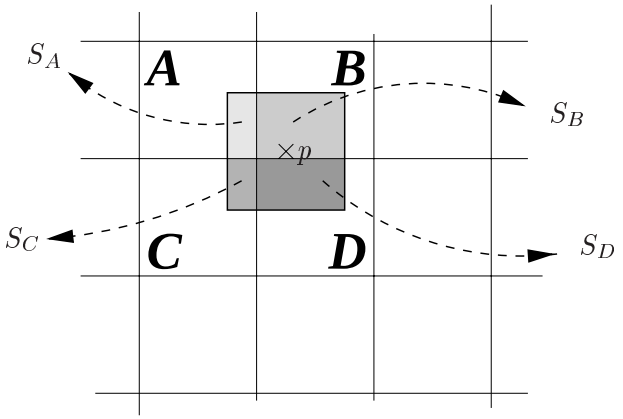


FIG. 4.28: *Projection d'un point* 3D *sur un pixel non-entier.*

Les pixels A, B, C et D prennent donc l'intensité I du point 3D, avec une pondération S_A, S_B, S_C ou S_D. Quand tous les points 3D du modèle sont projetés, alors chaque pixel

de l'image a accumulé une somme pondérée d'intensités, que l'on normalise ensuite. Par exemple, si les points 3D P_1 et P_2, d'intensités I_1 et I_2 se projettent près de A, alors le pixel A recevra une contribution $S_{A,1}$ de P_1, et une contribution $S_{A,2}$ de P_2. L'intensité finale de A sera $\frac{S_{A,1}I_1+S_{A,2}I_2}{S_{A,1}+S_{A,2}}$.

Cette forme d'anti-crénelage permet de combler une grande partie des zones non renseignées, puisque chaque point 3D se projette désormais sur 4 pixels, au lieu de 1.

Synthétiser une image 256×256 nécessite 0.25 seconde CPU avec la méthode 1 (projection sur un seul pixel), et 0.35 seconde CPU avec la méthode 2 (projection sur 4 pixels, avec contributions pondérées).

4.6.2.3 Cas étalonné

Dans le cas étalonné, nous disposons des 6 images de référence, et des 6 matrices de projection. Cela nous a permis de calculer un appariement dense multi-oculaire, puis une reconstruction robuste de tous les points 3D visibles au moins dans 2 images. Les points sont obtenus dans le repère 3D des images de référence, ce qui rend les comparaisons faciles : les transformations euclidiennes s'exprimant de la même façon pour les images calculées ou pour les images de référence, nous pourrons leur appliquer les mêmes transformations, afin de déterminer le domaine de l'espace où les images calculées restent valides.

Aussi, nous calculons avec POV–Ray 72 nouvelles vues de notre scène, que nous comparerons avec 72 vues calculées. Ces vues réparties en deux séries. Les vues de la série A sont disposées sur un cercle de centre O et de rayon 50 unités, contenu dans le plan OXZ, espacées de 10°, et visant le centre O de la scène (soient 36 vues). Les 36 vues de la série B sont disposées sur un cercle de centre O contenu dans le plan OXY, de la même manière (figure 4.29). Sur ces 72 vues, deux vues sont identiques : les points d'intersection des deux orbites. Cela ne constitue pas un échantillonnage régulier de l'espace, mais permet de mesurer la façon dont les résultats se dégradent à mesure qu'on s'éloigne des images de référence.

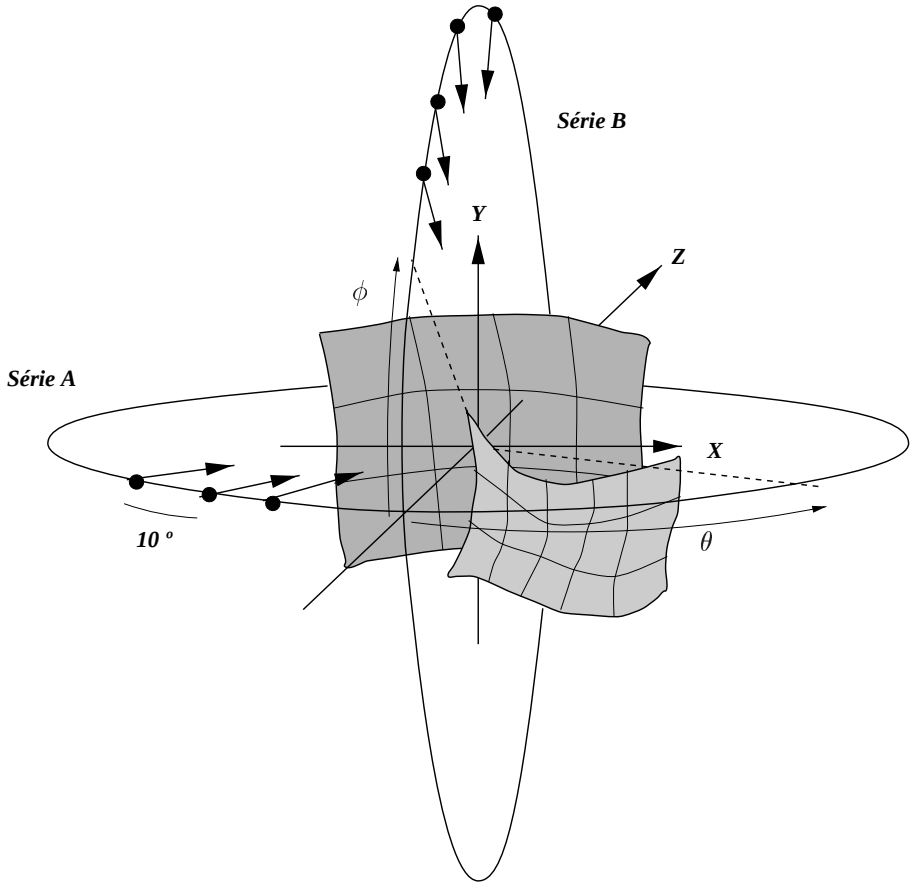


FIG. 4.29: *Disposition des 72 vues pour l'évaluation de la re-synthèse. Remarque : le repère de* POV–*Ray est gauche.*

L'erreur absolue moyenne EAM sur des images calculées par transfert par rapport aux images calculées par POV–Ray est donnée en figure 4.30 pour la série A (cercle horizontal), et en figure 4.31 pour la série B (cercle vertical). Rappelons que le critère EAM n'est calculé que sur les pixels renseignés de l'image synthétisée : celle-ci peut comporter de nombreux trous, sans que la valeur de EAM en soit affectée.

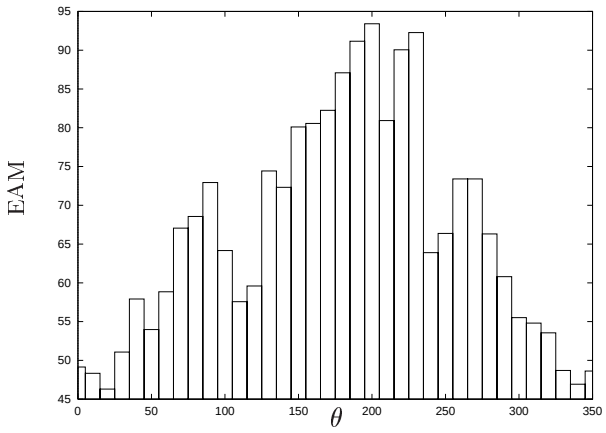


FIG. 4.30: *Critère* EAM *pour les 36 images de la série A.*

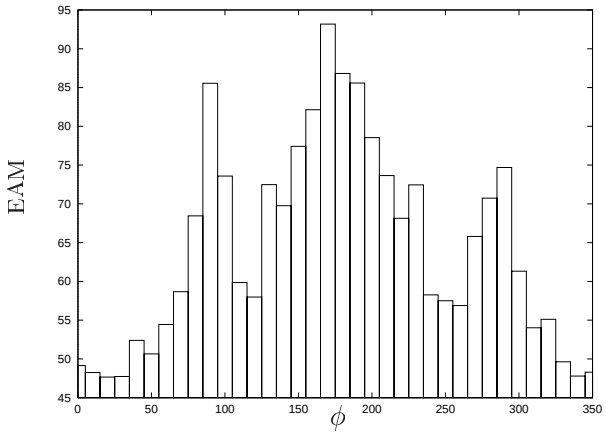


FIG. 4.31: *Critère* EAM *pour les 36 images de la série B.*

L'EAM reste inférieure à 50 pour $(\theta, \phi) = (-30.. + 20, 0)$, et pour $(\theta, \phi) = (0, -30.. + 30)$. Rappelons que les 6 images de référence correspondent aux positions $\phi = 0, \theta \in \{-12.5, -7.5, -2.5, +2.5, +7.5, +12.5\}$. Cela montre que ces images peuvent être extrapolées dans une zone approximativement égale à 2 fois le débattement maximal des images de référence. De plus, la disposition des images initiales permet de capturer le relief de façon suffisamment complète pour que des extrapolations en ϕ restent valables dans une gamme assez large. Le plus mauvais transfert a lieu pour l'image située sur l'orbite A, à $(\theta, \phi) = (-160, 0)$. La figure 4.32 montre l'image théorique calculée par POV–Ray, et l'image obtenue par transfert. Il n'est pas étonnant que ce cas soit la pire configuration, car elle correspond à la position de la caméra d'où l'arrière de la scène est vu de face ; or, cette zone n'était pas présente dans les images de référence, et n'a pas pu être capturée. Il y a donc un trou à cet endroit, et le reste de la scène transparaît (alors qu'il devrait être occulté).

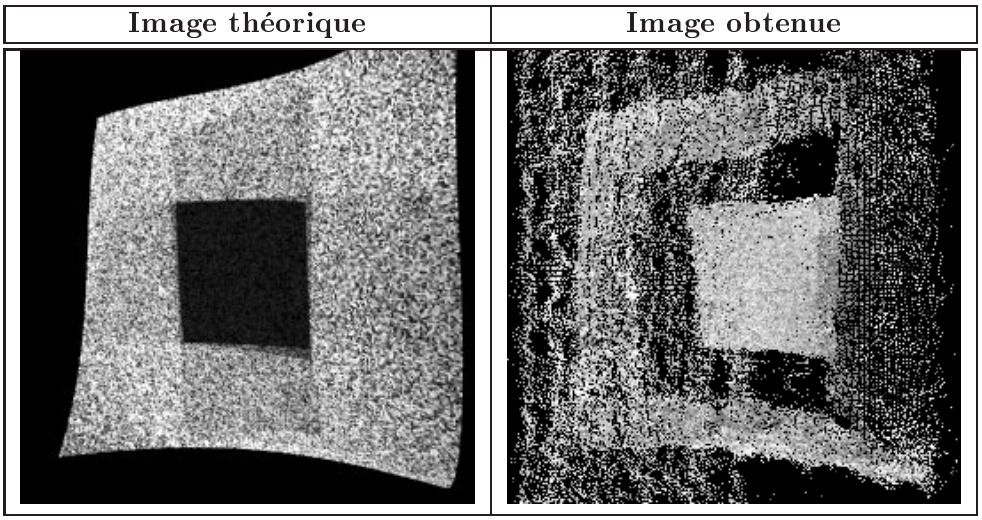


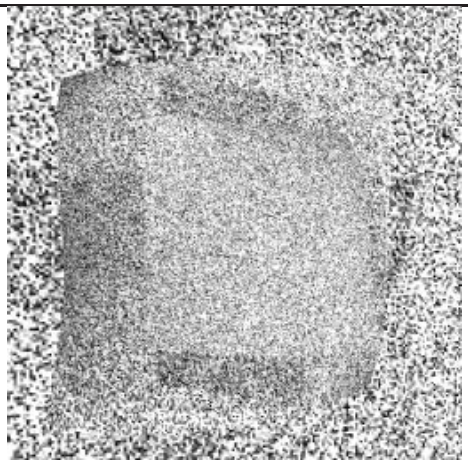
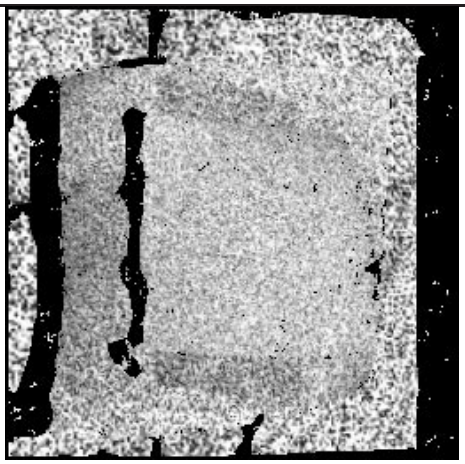
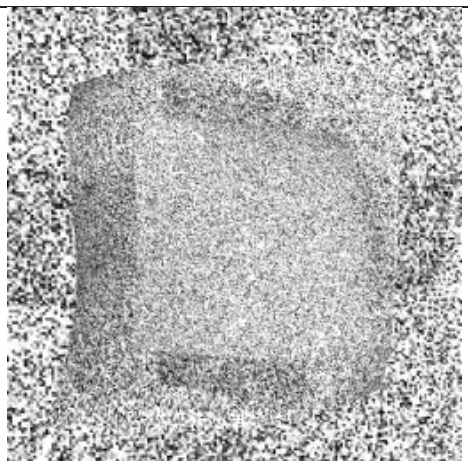
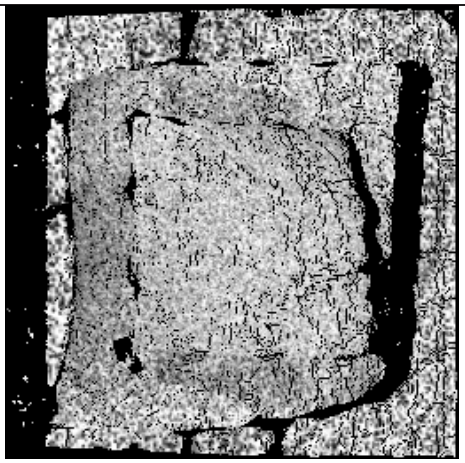
FIG. 4.32: *Pire transfert, obtenu pour* $(\theta, \phi) = (-160, 0)$.

Nous reviendrons sur les chiffres donnés ici lors des tests portant sur des textures réelles ; nous constaterons les mêmes différences de comportement que pour la construction de mosaïques.

4.6.2.4 Cas non étalonné

Dans le cas non étalonné, nous disposons des images de référence, mais pas des matrices de projection. Comme expliqué précédemment, nous ne traiterons que le cas binoculaire, pour des raisons de simplicité. Un appariement dense binoculaire a donc été calculé entre les images `im0` et `im1`, et la géométrie épipolaire $F_{0,1}$ déduite des images permet, à l'aide des paramètres intrinsèques, de calculer deux matrices de projection M_0 et M_1 dans les deux images permettant une triangulation et une reconstruction euclidienne. Les appariements ont subi un ajustement épipolaire au préalable.

Le repère de la reconstruction 3D obtenue est arbitraire, et ne correspond pas au repère utilisé par POV–Ray lors de la création des images de référence. Il est donc impossible de procéder comme dans le cas étalonné, et de comparer des images transférées à des images calculées à partir du modèle. Dans ce cas, nous nous contenterons donc de comparer les deux images transférées *via* M_0 et M_1 aux deux images de référence `im0` et `im1`. Les résultats sont en tableau 4.12.

Image	Image théorique	Image transférée
im0		 EAM = 11
im1		 EAM = 19

TAB. 4.12: *Cas binoculaire et non étalonné : critère* EAM *mesuré sur les deux images de référence.*

Nous retrouvons en noir les zones d'occultation, qui ne sont pas transférables, ainsi que quelques parties qui n'avaient pas été appariées. Le reste de l'image est convenable (EAM = 11 sur l'image 0, et EAM = 19 sur l'image 1).

4.6.3 Synthèse d'images à partir de triangles

Après la synthèse à partir de points 3D, nous examinons maintenant la construction d'un maillage triangulaire de ces points, en vue de réaliser un modèle à facettes triangulaires texturées. Nous sommes maintenant dans le cas étalonné, et nous utilisons les 6 images de référence, ainsi que les points 3D issus de la reconstruction euclidienne robuste. Un maillage est calculé sur `im0` selon l'algorithme proposé en 4.5.3 ; il est présenté en figure 4.33, superposé avec l'image `im0`.

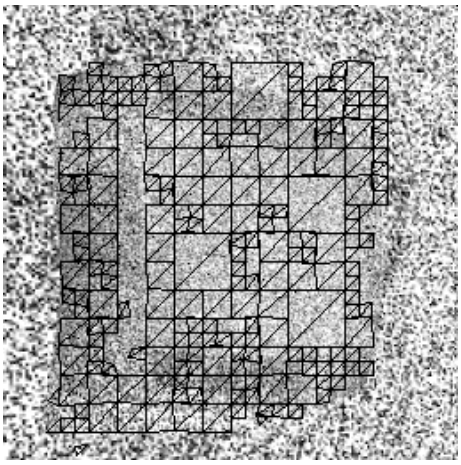


FIG. 4.33: *Maillage calculé sur l'image* im0*, à partir des appariements denses.*

Le processus de maillage évite de créer des triangles joignant les bords d'une zone occultée. Sur notre exemple, il existe néanmoins deux points de contact. Si nous superposons le maillage à l'image représentant les points visibles dans l'image im0 (figure 4.34), nous voyons que ces points de contact sont situés là où l'occultation est la plus étroite : à cet endroit, l'algorithme de maillage considère que les quelques pixels non-appariés résultent d'une erreur de mise en correspondance, et non d'une occultation réelle.

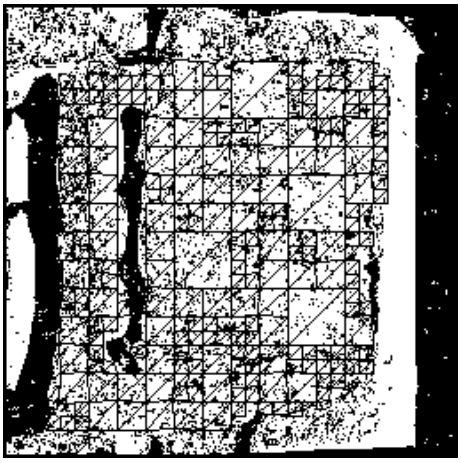


FIG. 4.34: *Maillage calculé sur l'image* im0*, superposé à l'image des points visibles depuis l'image* im0*.*

Les textures sont ensuite calculées automatiquement à partir de ce maillage, comme indiqué en 4.5.4, et un modèle VRML est généré. La figure 4.35 montre diverses vues de ce modèle, synthétisées par VRWeb.

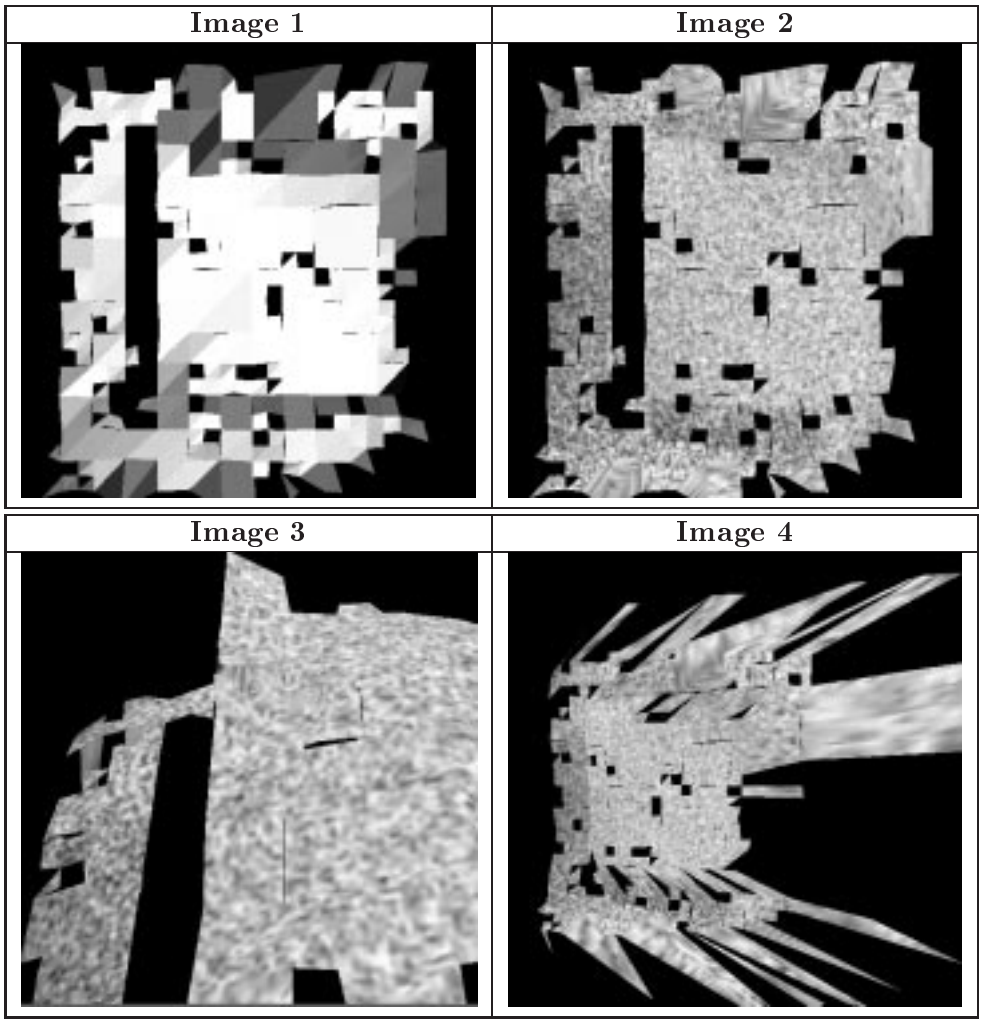


FIG. 4.35: *Quatre vues du modèle en triangles texturés, synthétisées par* VR*Web.*

Les images de la figure 4.35 montrent que de nombreuses zones de l'image ne sont pas couvertes par le maillage triangulaire. Ces zones n'avaient pas pu être appariées, et cela a deux causes : l'endroit considéré était occulté dans les images (bande noire, à gauche), ou l'algorithme de mise en correspondance n'a pas bien fonctionné à cet endroit, et/ou l'algorithme de reconstruction robuste a rejeté une partie des appariements obtenus (centre de l'image). Il est impossible de distinguer entre ces deux causes; si on veut que les zones occultées ne soient pas maillées, il faut aussi accepter que certaine parties de l'image ne soient pas couvertes, par manque d'informations d'appariement.

Enfin, certains triangles ne sont pas jointifs. Cela a les mêmes causes : des défauts d'appariement. En effet, si le sommet d'un triangle se trouve sur un point non renseigné, il est déplacé jusqu'au pixel renseigné le plus proche. Ceci entraîne une légère inclinaison des arêtes du triangle, qui peut faire apparaître un espace avec les triangles voisins.

Les quatre vues fournies montrent tout de même une assez bonne approximation du

modèle 3D sous-jacent, bien que nous ne puissions pas fournir d'évaluation quantitative sur ce point. Les textures seront plus aisées à apprécier dans les sections suivantes, sur les autres images-tests.

4.7 Évaluation sur images de synthèse — 2

Le processus est le même sur ces images :

1. reconstruction 3D ;
2. synthèse par points, et évaluation quantitative ;
3. synthèse par triangles, et évaluation qualitative.

Les résultats sont présentés et commentés ci-dessous.

4.7.1 Reconstruction 3D

Points 3D	**Nombre**	**Erreur**						**Temps**
		< 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**max**	**CPU**
Reconstruction	108039	7.47	17.39	76.17	89.17	93.26	668.11	14.78
Reconstruction robuste	96698	13.39	26.80	86.83	98.59	99.91	12.53	294.32

TAB. 4.13: *Erreur de reprojection après la phase de reconstruction* 3D*, versions standard ou robuste.*

Appariements	**Nombre**	**Erreur**					**Temps**
		< 0.05	< 0.10	< 0.50	< 1.00	< 2.00	**CPU**
Initiaux	108039	0.93	3.53	44.94	76.20	87.73	—
Après robustification	96698	0.93	3.63	48.06	80.80	92.61	294.32

TAB. 4.14: *Précision initiale de l'appariement, et précision des appariements restants, après la phase de reconstruction robuste.*

Comme pour la texture aléatoire, la phase d'appariement robuste a un effet positif sur les appariements 3D (tableau 4.14), mais améliore les reconstructions 3D de façon beaucoup plus radicale (tableau 4.13). Nous synthétisons de nouvelles images de la scène à partir du nuage des points 3D obtenus (cas étalonné et non étalonné), ou après construction d'un maillage triangulaire.

4.7.2 Synthèse d'images à partir de points

4.7.2.1 Cas étalonné

En utilisant les 6 images et leurs 6 matrices de projection, nous évaluons le processus de synthèse après reconstruction robuste, sur le même principe que pour les images à texture aléatoire. Les critères d'erreur sont montrés en 4.36 et 4.37).

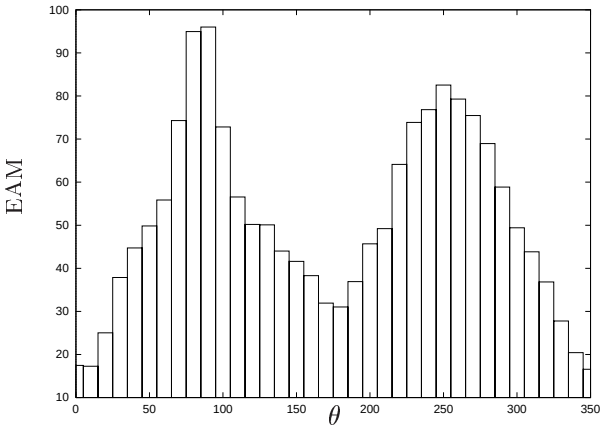


FIG. 4.36: *Critère* EAM *pour les 36 images de la série A.*

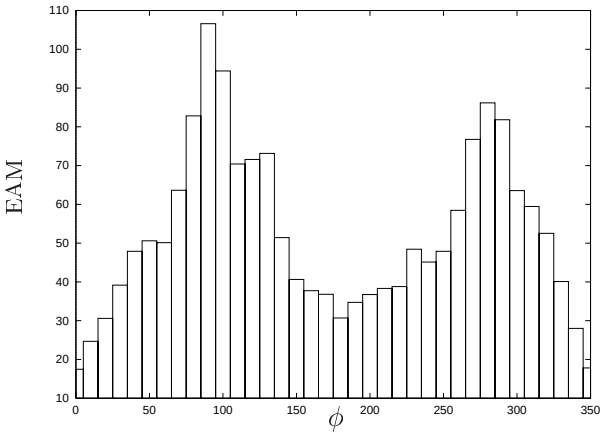


FIG. 4.37: *Critère* EAM *pour les 36 images de la série B.*

Les erreurs les plus élevées sont plus fortes que pour les images précédentes, et cela est dû à l'appariement, beaucoup plus délicat à mener en présence de zones uniformes. Le pire transfert correspond aux positions à angle droit, par exemple à $(\theta, \phi) = (0, 90)$ (figure 4.38).

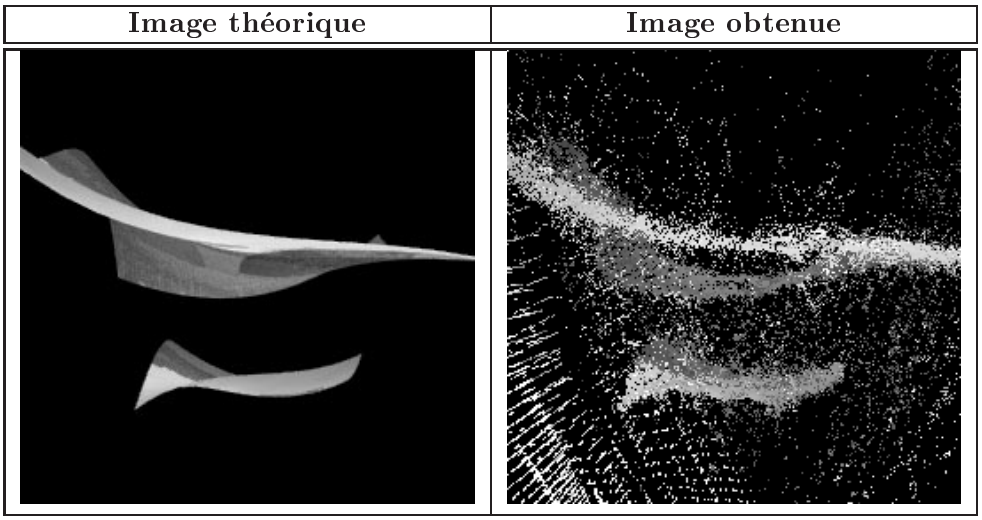


FIG. 4.38: *Pire transfert, obtenu pour* $(\theta, \phi) = (0, 90)$.

En revanche, les autres transferts sont en général meilleurs, même pour le point en opposition $(\theta, \phi) = (0, 180)$, qui constitue un minimum local. Cela est encore dû à la texture particulière des images, où des zones qui ne se correspondent pas peuvent avoir des apparences très similaires (figure 4.39).

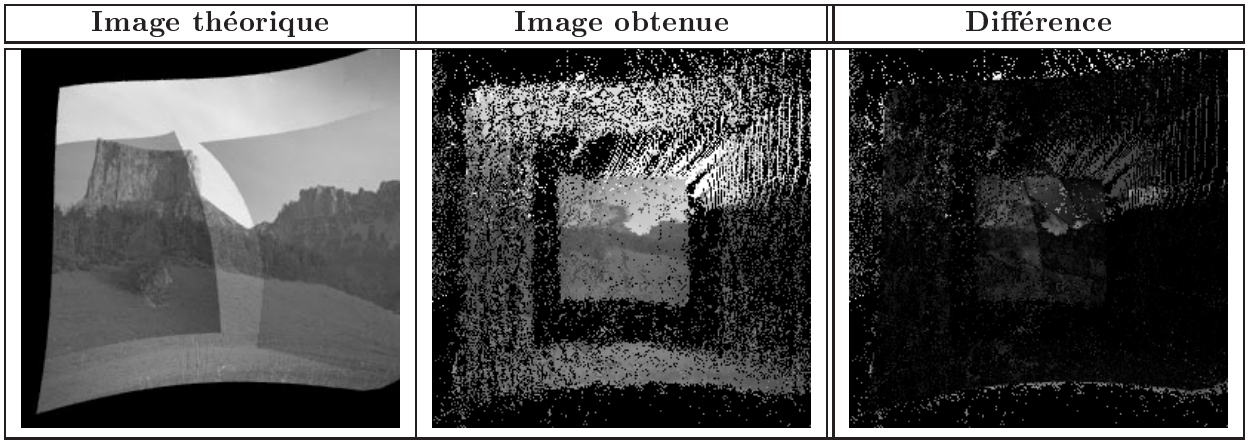


FIG. 4.39: *Un «bon» transfert :* EAM $= 31$. *Si on regarde l'image de différence, on s'aperçoit en effet que beaucoup de pixels concordent (différence nulle : zones noires). Rappelons que le critère* EAM*, comme l'image de différence, n'est calculé que sur les zones renseignées de l'image synthétisée.*

Enfin, nous montrons l'effet de notre algorithme de projection en figure 4.40, sur les images qui se projettent le mieux : $(\theta, \phi) = (0, 0)$.

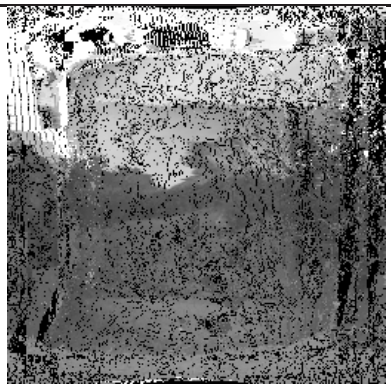
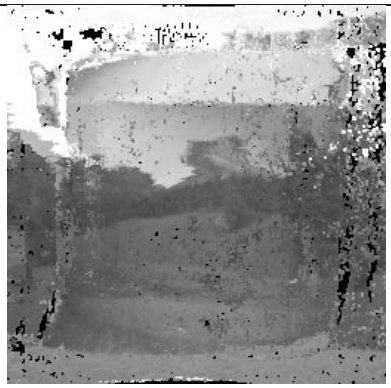
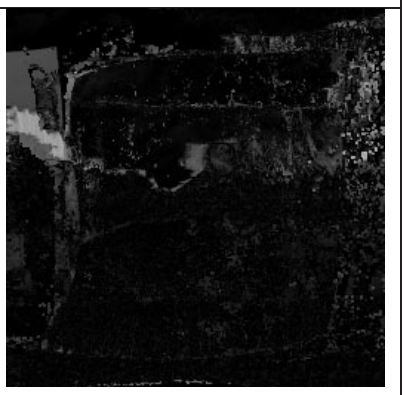
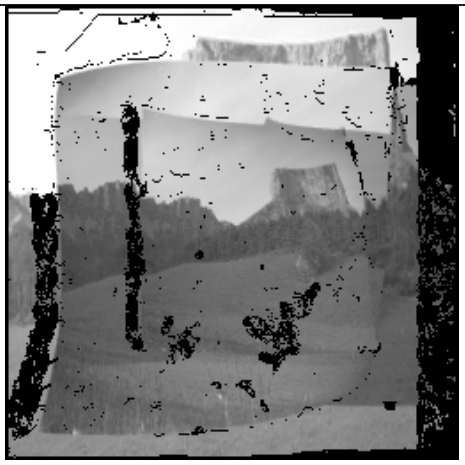
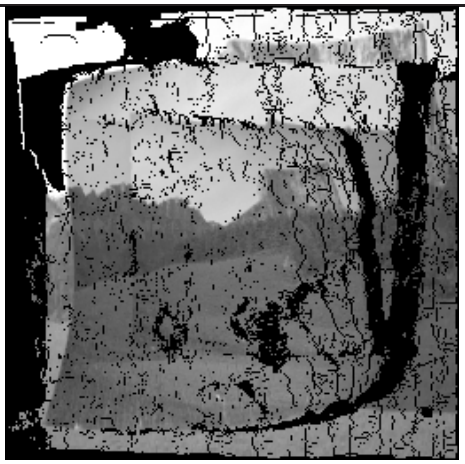
	Image théorique	Image obtenue	Différence
Projection classique			
Projection sur 4 pixels			

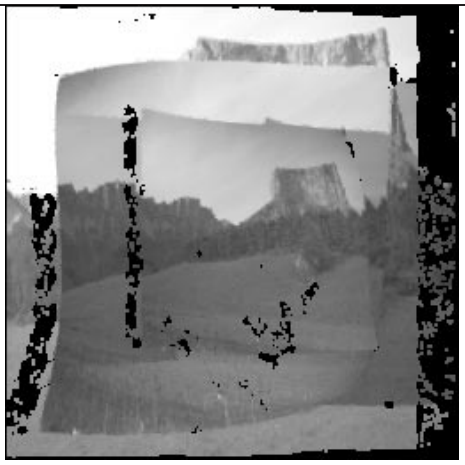
FIG. 4.40: *Comparaison de la projection simple (1 point* 3D *se projette sur 1 pixel), et de la projection décrite en 4.6.2.2, où chaque point* 3D *se projette sur les 4 plus proches pixels, avec une pondération adéquate.* $(\theta, phi) = (0, 0)$.

4.7.2.2 Cas non étalonné

Nous ne travaillons plus qu'à partir de deux images de référence, `im0` et `im1`, sans autre connaissance. De ces images, nous avons déjà calculé la matrice fondamentale $F_{0,1}$, et nous l'utilisons pour rectifier les appariements binoculaires, puis les reconstruire. La reconstruction obtenue, reprojetée *via* les matrices M_0 et M_1 (calculées à partir de $F_{0,1}$), est montrée en tableau 4.15, ainsi que l'effet de la projection sur 4 pixels (tableau 4.16).

Image	Image théorique	Image transférée
im0		 EAM = 1
im1		 EAM = 2

TAB. 4.15: *Cas binoculaire et non étalonné : critère* EAM *mesuré sur les deux images de référence, après projection classique.*

Image	Image théorique	Image transférée
im0		 EAM = 2
im1		EAM = 3

TAB. 4.16: *Cas binoculaire et non étalonné : critère* EAM *mesuré sur les deux images de référence, après projection sur les 4 plus proches pixels.*

4.7.3 Synthèse d'images à partir de triangles

La figure 4.41 montre le maillage calculé automatiquement sur les appariements issus de la reconstruction 3D robuste.

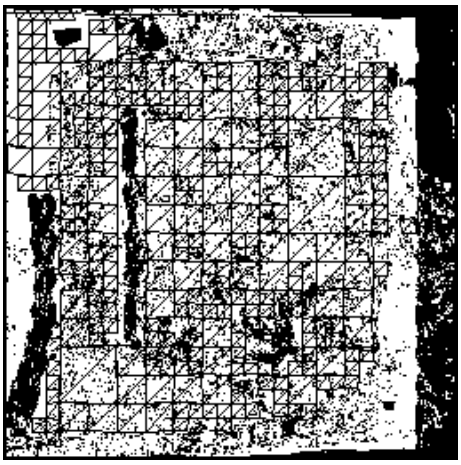


FIG. 4.41: *Maillage calculé sur l'image* $im0$*, superposé à l'image des points visibles depuis l'image* $im0$*. Les points visibles ne sont pas suffisamment denses.*

Comme nous le voyons, les points visibles ne sont pas suffisamment denses pour produire un maillage acceptable. Une solution est de régulariser les appariements obtenus, et nous effectuons 20 passes du calcul décrit en 3.5.1.5. L'algorithme de régularisation n'effectue des appariements qu'au pixel près, et nous devons donc appliquer une nouvelle passe d'affinage des appariements (AFFZD4). Ensuite, nous pouvons effectuer une reconstruction aux moindres carrés (non-robuste), suffisamment fiable à cette étape. Au total, le processus d'obtention des appariements a donc été le suivant :

1. calcul de la géométrie épipolaire ;
2. appariement dense avec contrainte épipolaire ;
3. régularisation ;
4. affinage ;
5. reconstruction robuste, pour rejeter les appariements peu vraisemblables ;
6. régularisation ;
7. affinage ;
8. reconstruction standard (moindres carrés).

Maintenant, le maillage obtenu est celui de la figure 4.42.

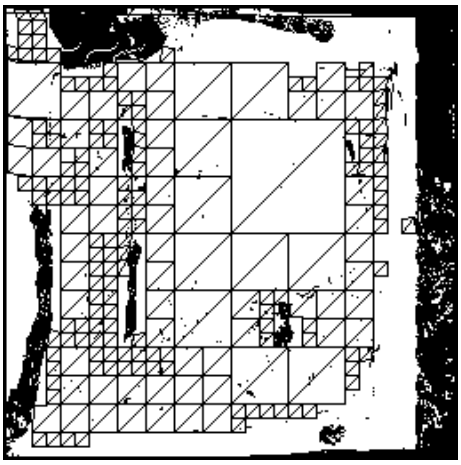


FIG. 4.42: *Maillage calculé sur l'image* $im0$*, superposé à l'image des points visibles depuis l'image* $im0$*, après une nouvelle étape de régularisation.*

La figure 4.43 montre des vues de ce modèle synthétisées par VRWeb.

Ici, le modèle obtenu est plus approximatif, car les appariements étaient de moins bonne qualité que pour les images à texture aléatoire. La zone occultée ayant été partiellement comblée par le processus de régularisation, quelques triangles apparaissent dans cette zone (ces triangles apparaissent nettement sur les images 1 et 3). Comme prévu, quelques triangles faux perturbent suffisamment la solution pour la rendre inacceptable, sous un angle de vue éloigné (image 3).

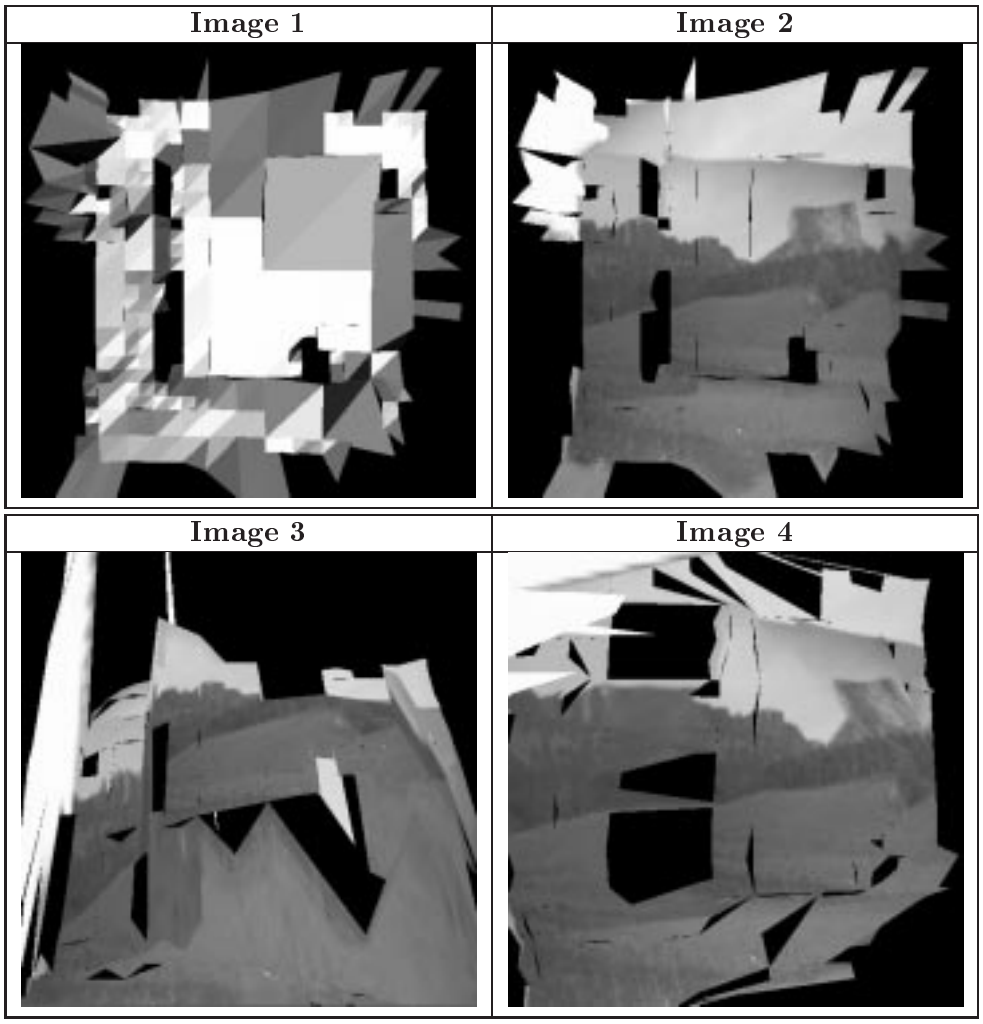


FIG. 4.43: *Quatre vues du modèle en triangles texturés, synthétisées par* VRWeb.

4.8 Évaluation sur images de synthèse — 3

Le même schéma de calcul mène aux résultats présentés ci-dessous.

4.8.1 Reconstruction 3D

Points 3D	Nombre	Erreur						Temps
		< 0.05	< 0.10	< 0.50	< 1.00	< 2.00	max	CPU
Reconstruction	105057	7.99	18.63	75.13	89.41	94.96	192.39	14.49
Reconstruction robuste	94245	13.83	28.03	85.59	97.48	99.79	12.28	283.10

TAB. 4.17: *Erreur de reprojection après la phase de reconstruction* 3D*, versions standard ou robuste.*

Appariements	Nombre	Erreur					Temps
		< 0.05	< 0.10	< 0.50	< 1.00	< 2.00	CPU
Initiaux	105057	1.38	4.85	46.94	75.63	88.31	—
Après robustification	94245	1.36	4.99	50.53	80.44	92.49	283.10

TAB. 4.18: *Précision initiale de l'appariement, et précision des appariements restants, après la phase de reconstruction robuste.*

Les tableaux 4.17 et 4.18 montrent les mêmes effets que précédemment (chiffres très similaires).

4.8.2 Synthèse d'images à partir de points

4.8.2.1 Cas étalonné

Les critères d'erreur sont montrés en 4.44 et 4.45).

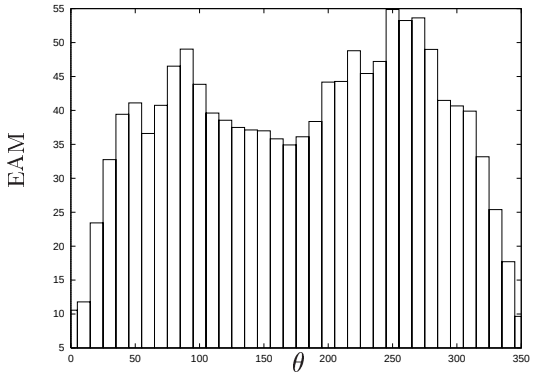


FIG. 4.44: *Critère* EAM *pour les 36 images de la série A.*

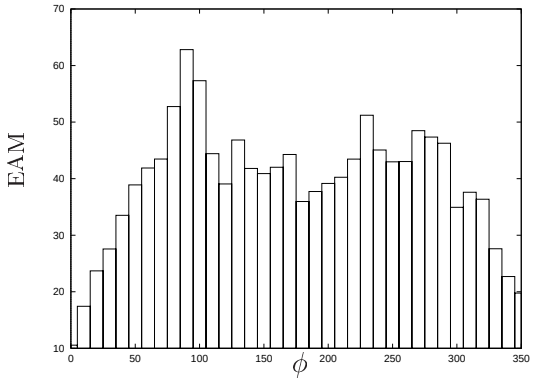


FIG. 4.45: *Critère* EAM *pour les 36 images de la série B.*

Les mêmes constatations que précédemment s'appliquent aussi : les configurations à

angle droit provoquent les erreurs les plus élevées, et le transfert à 180 ° donne artificiellement un assez bon score, parce que les textures sont assez uniformes sur toute la surface de l'image. La figure 4.46 compare de nouveau les deux méthodes de projection, pour la configuration $(\theta, \phi) = (0, 0)$.

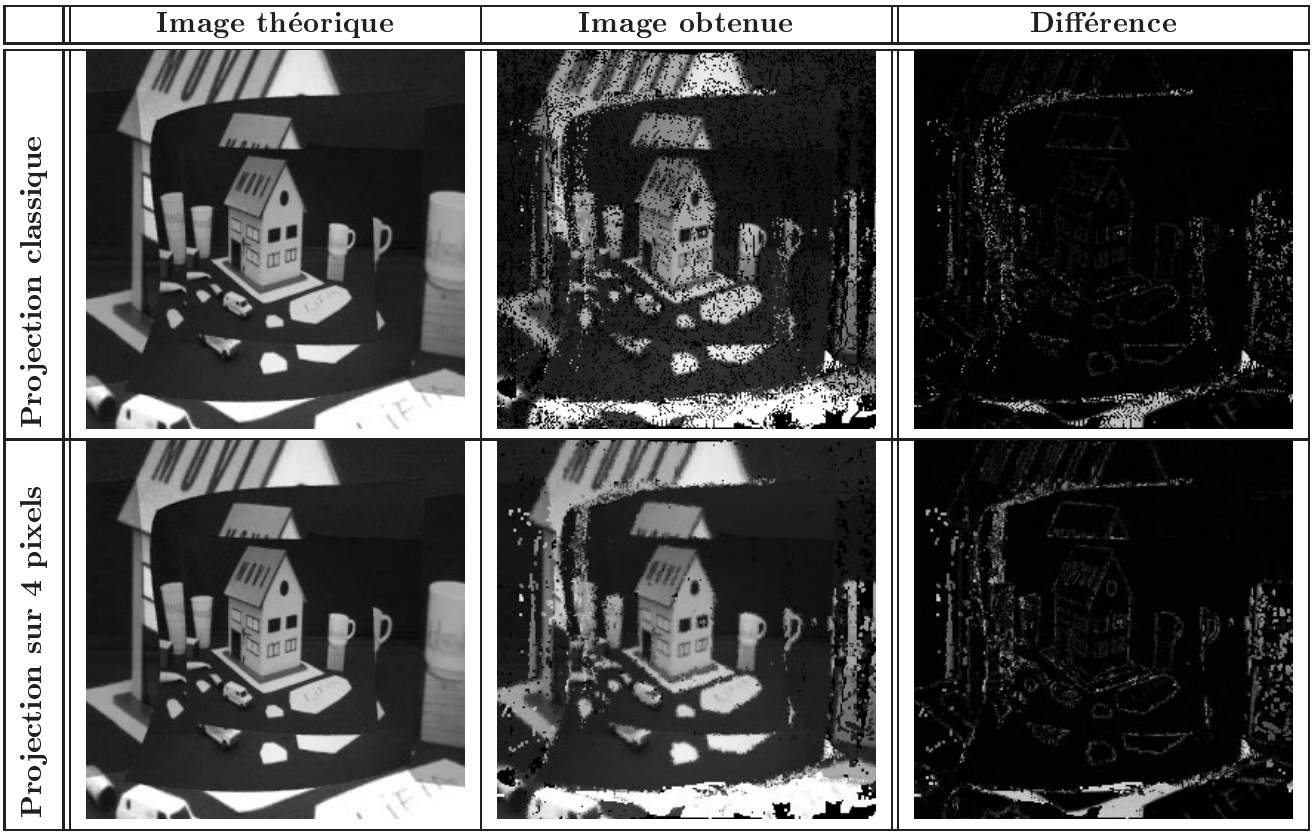
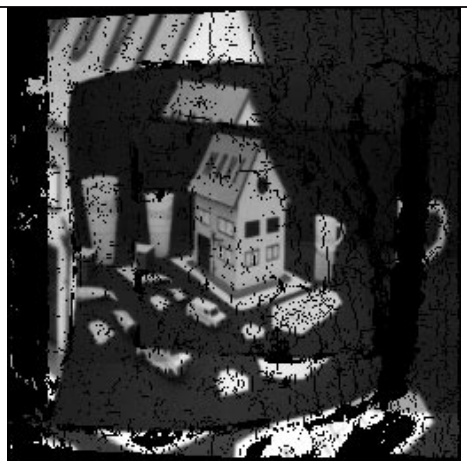


FIG. 4.46: *Comparaison de la projection simple (1 point* 3D *se projette sur 1 pixel), et de la projection décrite en 4.6.2.2, où chaque point* 3D *se projette sur les 4 plus proches pixels, avec une pondération adéquate.* $(\theta, \phi) = (0, 0)$.

4.8.2.2 Cas non étalonné

À partir de la seule donnée des images de référence `im0` et `im1` et des paramètres intrinsèques, nous obtenons les images montrées en tableaux 4.19 et 4.20.

Image	Image théorique	Image transférée
im0		EAM = 2
im1		 EAM = 3

TAB. 4.19: *Cas binoculaire et non étalonné : critère* EAM *mesuré sur les deux images de référence, après projection classique.*

Image	Image théorique	Image transférée
im0		EAM = 3
im1		EAM = 4

TAB. 4.20: *Cas binoculaire et non étalonné : critère* EAM *mesuré sur les deux images de référence, après projection sur les 4 plus proches pixels.*

4.8.3 Synthèse d'images à partir de triangles

La figure 4.47 montre le maillage calculé automatiquement sur les appariements issus de la reconstruction 3D robuste.

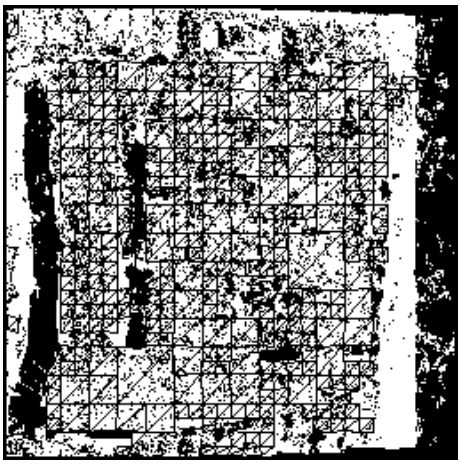


FIG. 4.47: *Maillage calculé sur l'image* im0*, superposé à l'image des points visibles depuis l'image* im0*. Les points visibles ne sont pas suffisamment denses.*

Comme précédemment, les points visibles ne sont pas suffisamment denses, et nous sommes contraints de procéder à une phase supplémentaire de régularisation. Le nouveau maillage obtenu est montré en figure 4.48.

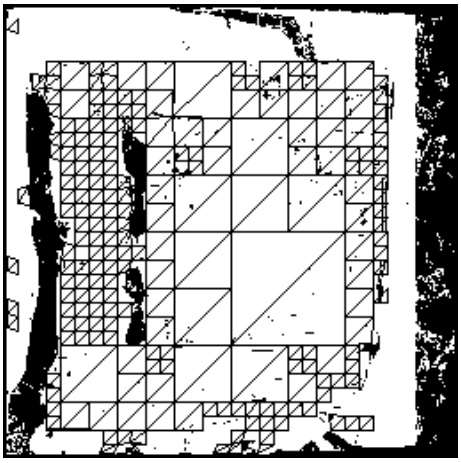


FIG. 4.48: *Maillage calculé sur l'image* im0*, superposé à l'image des points visibles depuis l'image* im0*, après une nouvelle étape de régularisation.*

Le maillage présenté en figure 4.48 est correct, mais nous sommes cette fois confrontés à un nouveau problème : les sommets 3D des triangles n'ont pas tous été correctement reconstruits. La figure 4.49 montre la structure tridimensionnelle du maillage obtenu à la figure 4.48. Quelques points faux suffisent à rendre le modèle inutilisable.

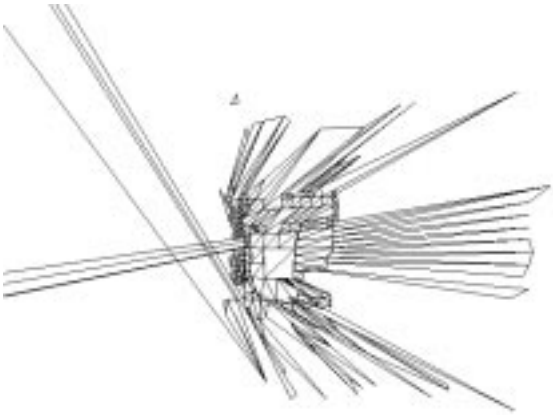


FIG. 4.49: *Une vue du modèle* 3D *obtenu ; quelques points faux suffisent à rendre le modèle inutilisable.*

Nous pourrions envisager de reprendre une phase de reconstruction 3D robuste, mais nous aboutirions dans une impasse : les points reconstruits ne seraient, une nouvelle fois, plus suffisamment denses pour une bonne triangulation. Aussi, nous décidons plutôt d'appliquer une méthode de rejet plus brutale, basée sur la position des points 3D dans l'espace : nous rejetons les 1 ou la plus faible composante ne X, Y ou Z. Le nouveau maillage est en figure 4.50, et les vues obtenues en 4.51.

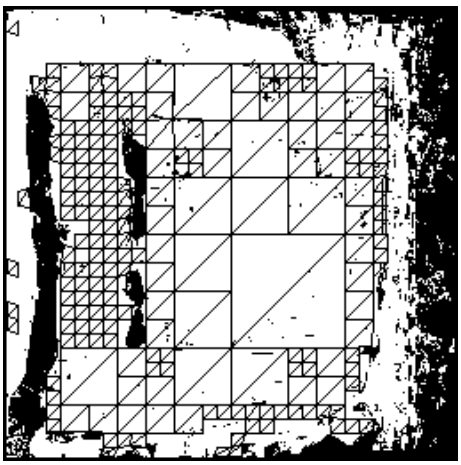


FIG. 4.50: *Maillage calculé sur l'image* $im0$*, superposé à l'image des points visibles depuis l'image* $im0$*, après régularisation et filtrage des points* 3D *extrêmes.*

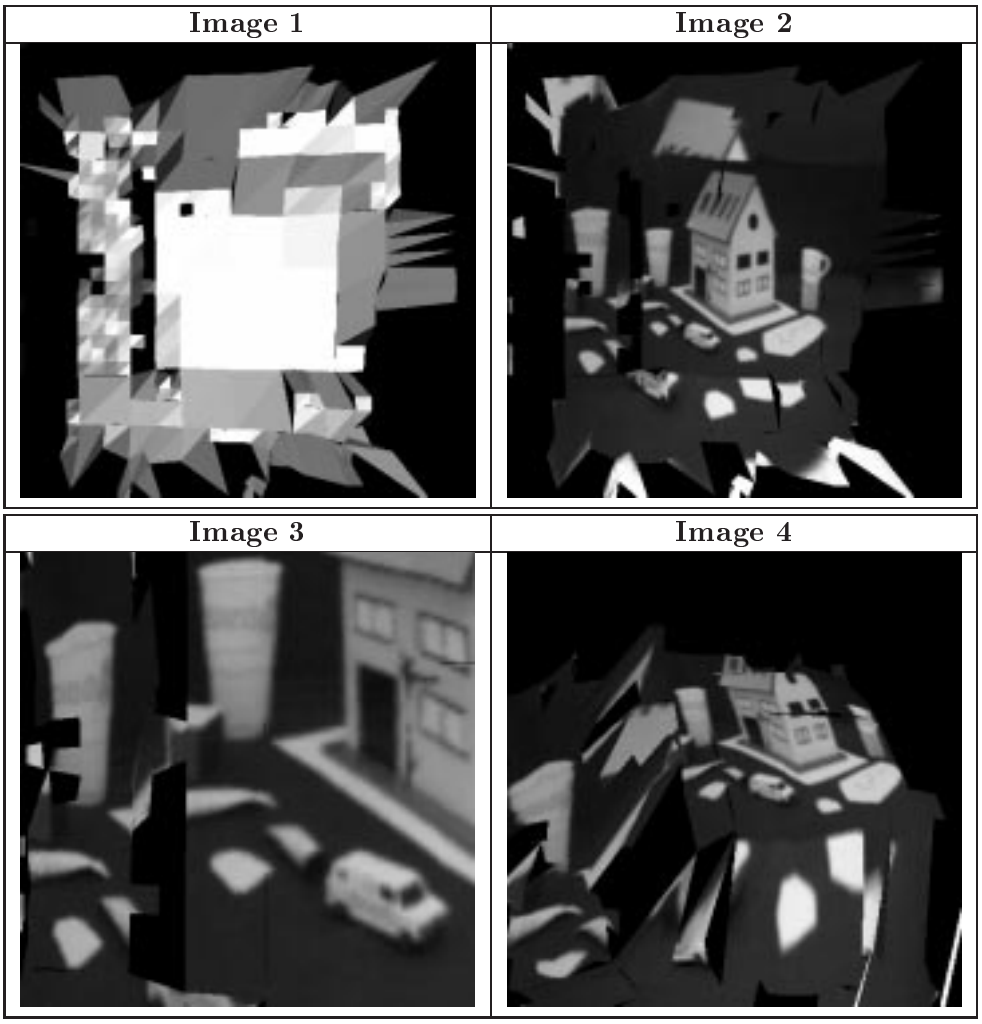


FIG. 4.51: *Quatre vues du modèle en triangles texturés, synthétisées par* VR*Web.*

Nous observons le même comportement que pour les images à texture d'extérieur : quelques triangles couvrent la zone occultée, et certains ne sont pas jointifs. L'appariement étant assez dense, seuls deux triangles sont absents de la zone centrale (carré noir, en haut et à gauche de la zone centrale de l'image 1). L'image 3 montre que les textures des triangles connexes sont bien représentées de façon continue, et sans distorsion. Enfin, l'image 4 montre que des triangles joignent la partie centrale de l'image avec le fond (bas de la maison, au premier plan de l'image). Cela est inévitable, car aucune occultation ne peut être perçue à cet endroit : les images sont prises en translation horizontale, et il est impossible de déterminer s'il s'agit seulement d'une rupture de disparité, ou d'une rupture de connexité. Aussi, c'est une brutale rupture de disparité qui a été modélisée. Le même phénomène est d'ailleurs observables sur les deux séries précédentes.

4.9 Évaluation sur images réelles

L'utilisation directe des appariements obtenus pour les images réelles ne donne pas de résultats satisfaisants : une petite proportion (de l'ordre de 5 %) des points 3D sont reconstruits à des positions très éloignées de leurs positions théoriques, ou même derrière les caméras. Ceci rend le maillage triangulaire inutilisable. Nous devons donc appliquer des étapes supplémentaires de filtrage et de régularisation. Plus précisément, les appariements obtenus à la fin des expérimentations du chapitre précédent suivent le traitement suivant :

1. reconstruction robuste, pour rejeter les appariements peu vraisemblables ;
2. régularisation ;
3. affinage ;
4. reconstruction standard (moindres carrés) ;
5. rejet des points se situant derrière les caméras.

4.9.1 Synthèse d'images à partir de points

Dans le cas binoculaire, le reconstruction robuste est impossible, mais nous pouvons effectuer un ajustement épipolaire, préalable à la reconstruction 3D. Les figures 4.52 et 4.53 montrent les images obtenues par reprojection des points 3D sur les plans des images de référence, pour les deux séries d'images réelles.

Les images des figures 4.52 et 4.53 montrent que les images synthétiques obtenues sont très semblables aux images de référence : l'EAM est au maximum de 9 niveaux d'intensité. Les points manquants correspondent à des zones non-appariées des images (voir en particulier le toit de la maison MOVI). La dernière image de chaque série est toujours moins bonne que les autres images, car c'est la seule à ne pas avoir été régularisée : quatrième image du château (EAM = 9), et seconde image de la maison (EAM = 9 aussi).

Ces images ne permettent que de vérifier la bonne marche du système, mais pas la qualité de reconstruction 3D : les points 3D obtenus étant reprojetés dans les images de référence, nous construisons presque nécessairement une image de bonne qualité. Aussi, nous présentons en figures 4.54 et 4.55 d'autres images de ces deux scènes, sous d'autres points de vue.

No.	Image théorique	Image obtenue	Différence
1			EAM = 7
2			EAM = 6
3			EAM = 6
4			EAM = 9

FIG. 4.52: *Projection des points reconstruits sur le plan des images de référence (projection sur 4 pixels). Première série.*

No.	Image théorique	Image obtenue	Différence
1			EAM = 9
2			EAM = 8

FIG. 4.53: *Projection des points reconstruits sur le plan des images de référence (projection sur 4 pixels). Seconde série.*

Image 1	Image 2

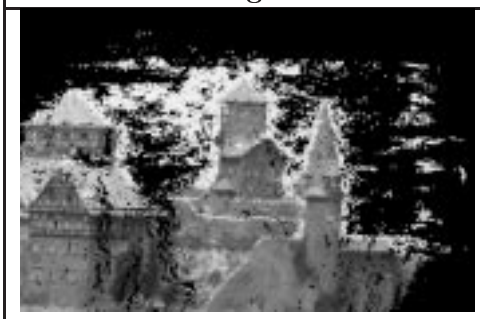



Image 3	**Image 4**

FIG. 4.54: *Quatre vues arbitraires de la scène du château.*

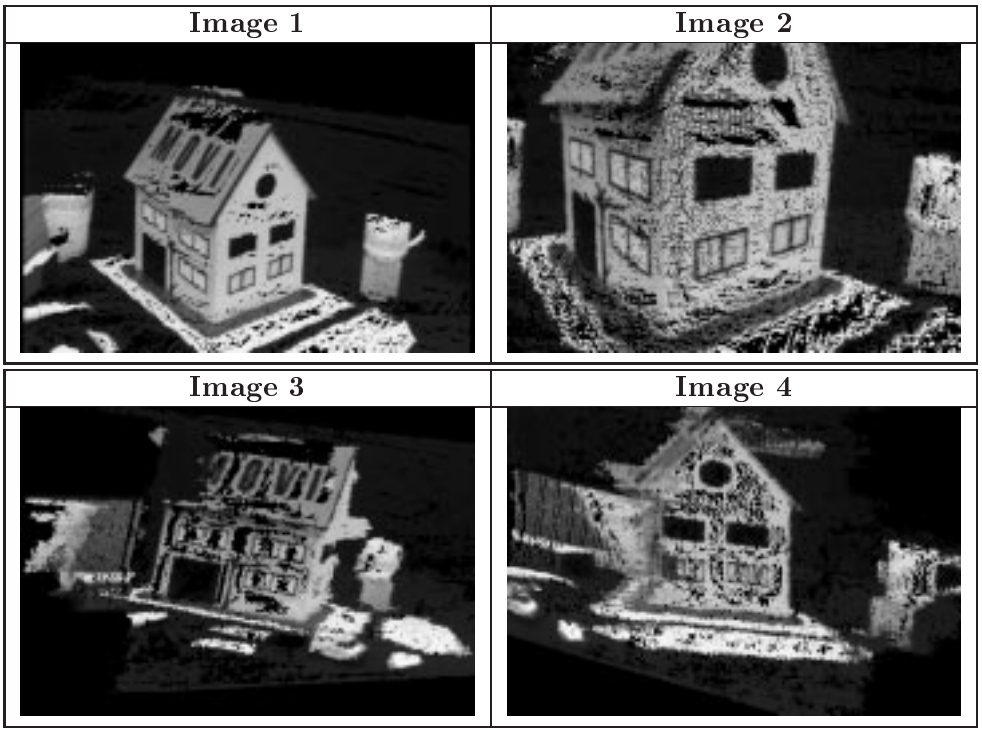


FIG. 4.55: *Quatre vues arbitraires de la scène de la maison* MOVI.

Les images obtenues en figure 4.54 et 4.55 révèlent les problèmes auxquels nous sommes confrontés lors de la synthèse à partir de points isolés : les pixels ne présentent aucune cohérence, et même en utilisant une méthode de projection sur 4 pixels, l'image générée est très «pointilliste». Ceci se voit en particulier sur les vues rapprochées : image 4 du château, et image 2 de la maison. Enfin, les erreurs d'appariement/reconstruction deviennent d'autant plus visibles que l'on s'écarte des points de vue de référence : l'image 2 du château montre que le dessus de la scène est mal capturé (toits des maisons, au premier plan), car les images de référence étaient toutes situées le long d'une même horizontale, et aucune vue de dessus de la scène n'était disponible. Les images de la maison confirment qu'une partie du toit était mal appariée, ce que nous avions déjà remarqué en examinant la carte de disparité. Toutes les zones uniformes présentent des problèmes de cet ordre.

Nous pouvons maintenant appliquer différents filtres pour améliorer les images obtenues. La figure 4.56 montre l'effet d'un filtrage médian sur quelques images synthétiques. Le filtrage médian a l'avantage de ne pas lisser les contours, et de préserver les zones de contraste franc.

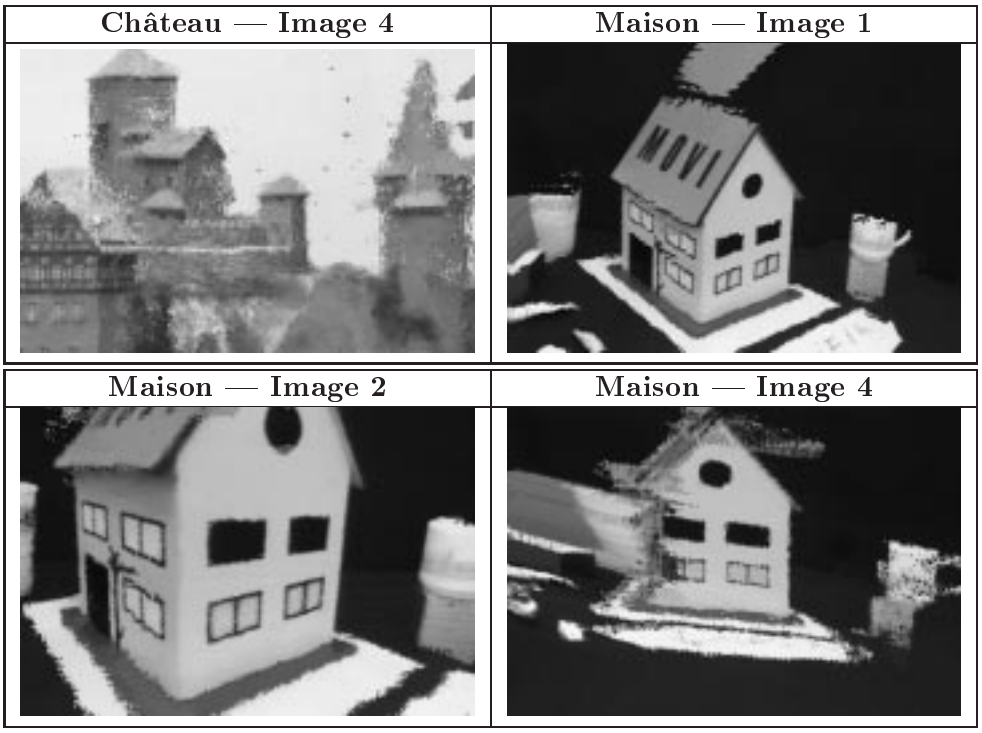


FIG. 4.56: *Effet du filtre médian sur les images synthétisées.*

4.9.2 Synthèse d'images à partir de triangles

Les figures 4.57 et 4.58 montrent des images synthétisées par VRWeb, à partir d'un modèle en triangles texturés.

La première image de chaque série montre la triangulation obtenue, sans texture. La première série (le château) compte 1246 triangles, et la seconde 1367 triangles (la maison MOVI). Les autres vues montrent que les erreurs d'appariement sont exacerbées, car les sommets des triangles sont mal placés : toit de la maison au premier plan de la scène du château (image 4 de la scène du château), petit donjon à l'arrière plan (image 3 de la scène du château), toit de la maison MOVI (image 4 de la scène de la maison). Les triangles manquants sont dus à des appariements absents. L'algorithme de triangulation ne peut en effet pas déterminer si cela correspond à des erreurs de l'algorithme de mise en correspondance, ou à des occultations.

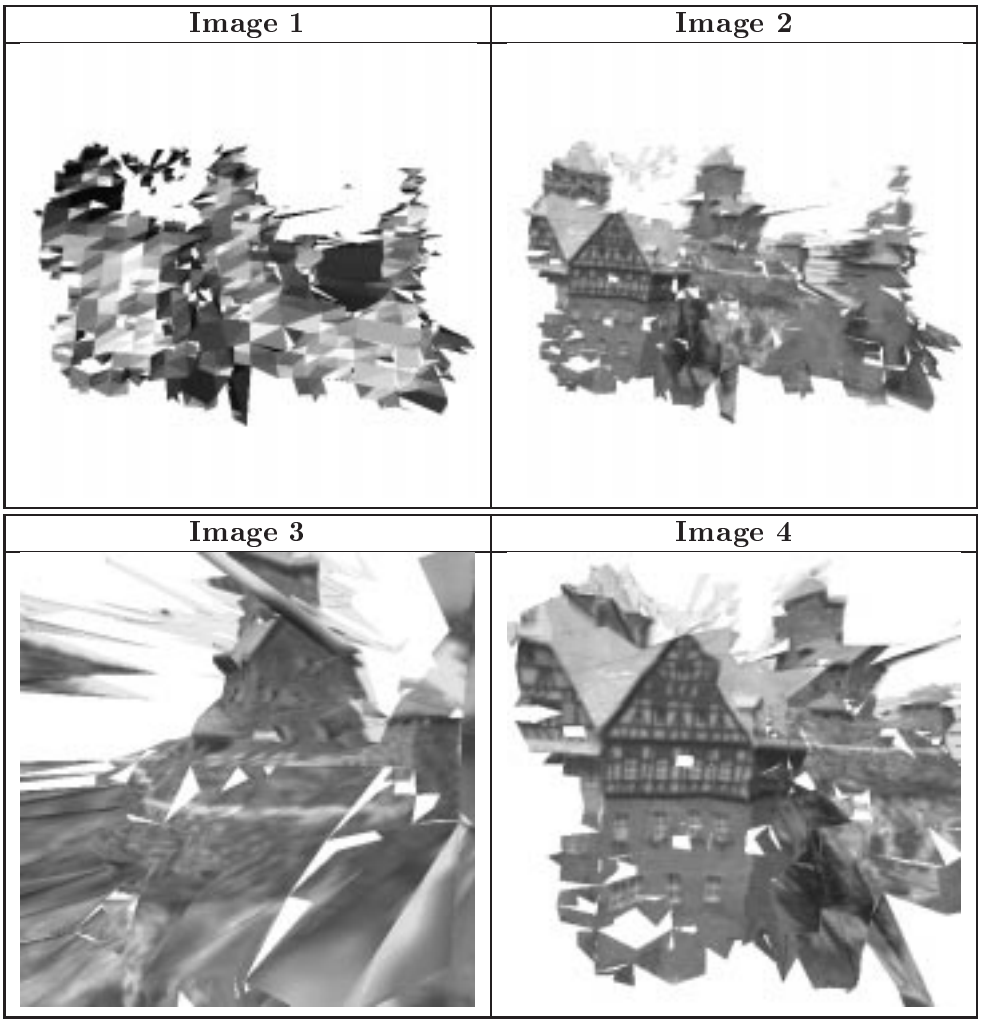


FIG. 4.57: *Quatre vues du modèle en triangles texturés, synthétisées par* VR*Web. Première série.*

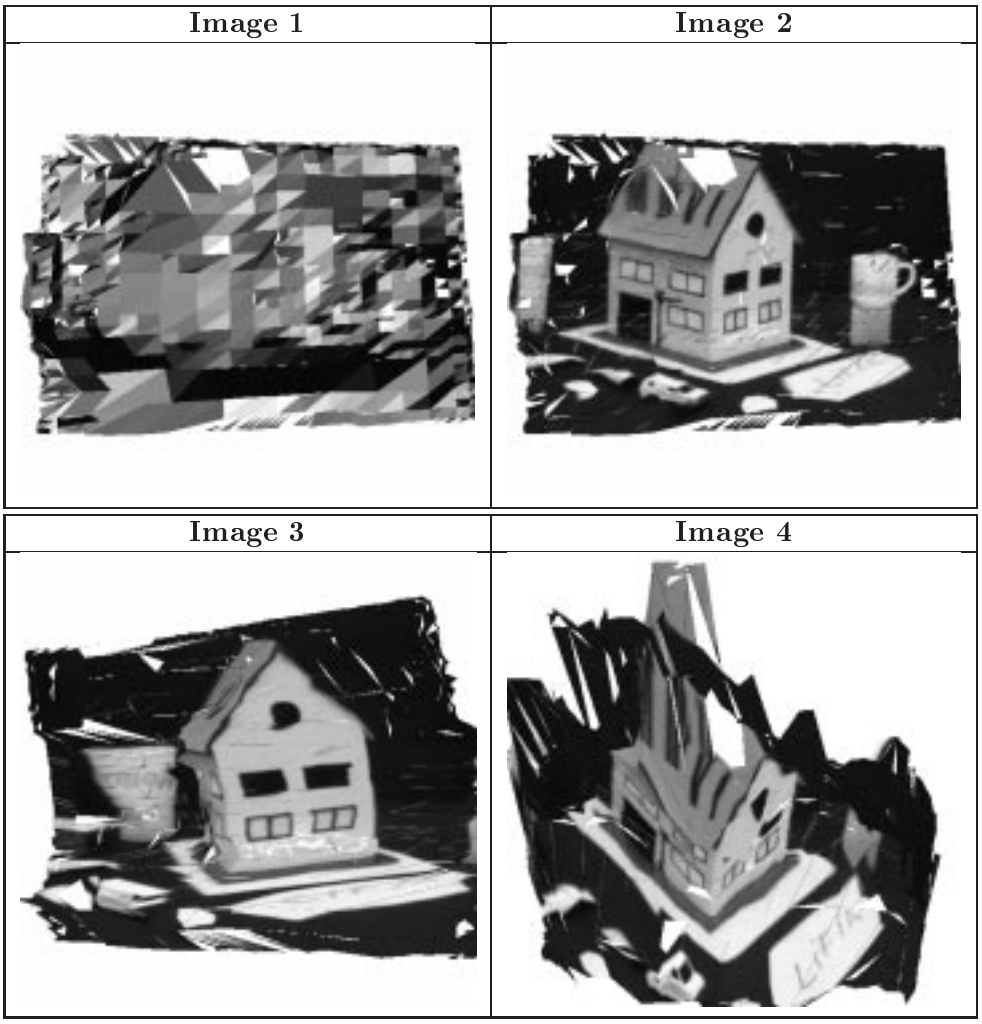


FIG. 4.58: *Quatre vues du modèle en triangles texturés, synthétisées par* VR*Web. Seconde série.*

Supposons maintenant qu'il n'existe pas d'occultation, mais que la scène puisse être recouverte par une seule nappe 3D. Alors l'algorithme de triangulation peut être considérablement simplifié, sous la forme suivante :

1. parmi les appariements denses, ne conserver que les points d'intérêt extraits dans les images ;

2. sur ces points 2D, effectuer une triangulation de Delaunay. Cela revient à faire l'hypothèse que les zones situées entre les points d'intérêt sont nécessairement planes.

Les points d'intérêt sont des points reconnaissables, car par définition, ils présentent des courbures très élevées de gradient de contraste (ils ont été extraits par un détecteur de Harris). Leurs appariements sont donc plus fiables que ceux des points uniformes. De plus,

partir d'un appariement dense permet de profiter de toutes les étapes de régularisation globale que nous avons pu effectuer en amont. Les images obtenues par ce procédé sont visibles en figures 4.59 et 4.60.

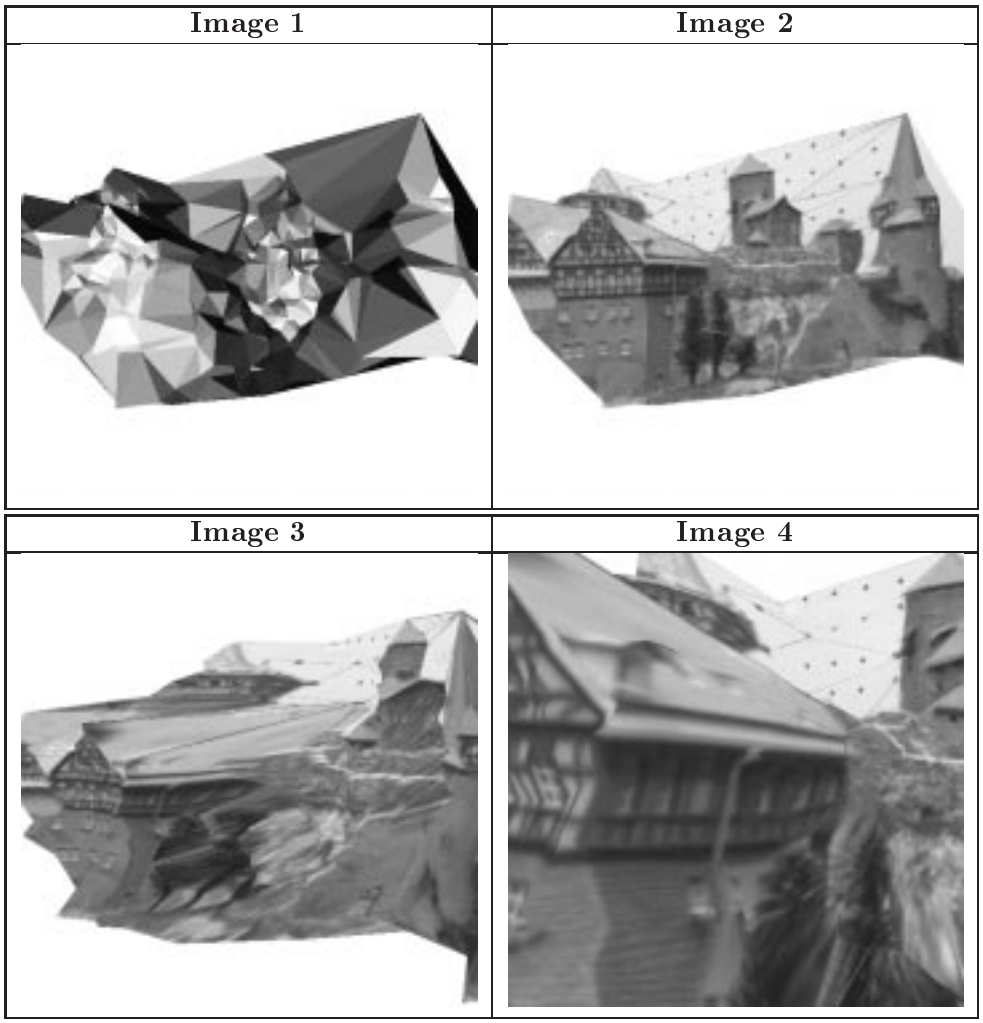


FIG. 4.59: *Quatre vues du modèle en triangles texturés, synthétisées par* VR*Web (première série). On ne s'appuie que sur des points d'intérêt, et les occultations ne sont pas prises en compte.*

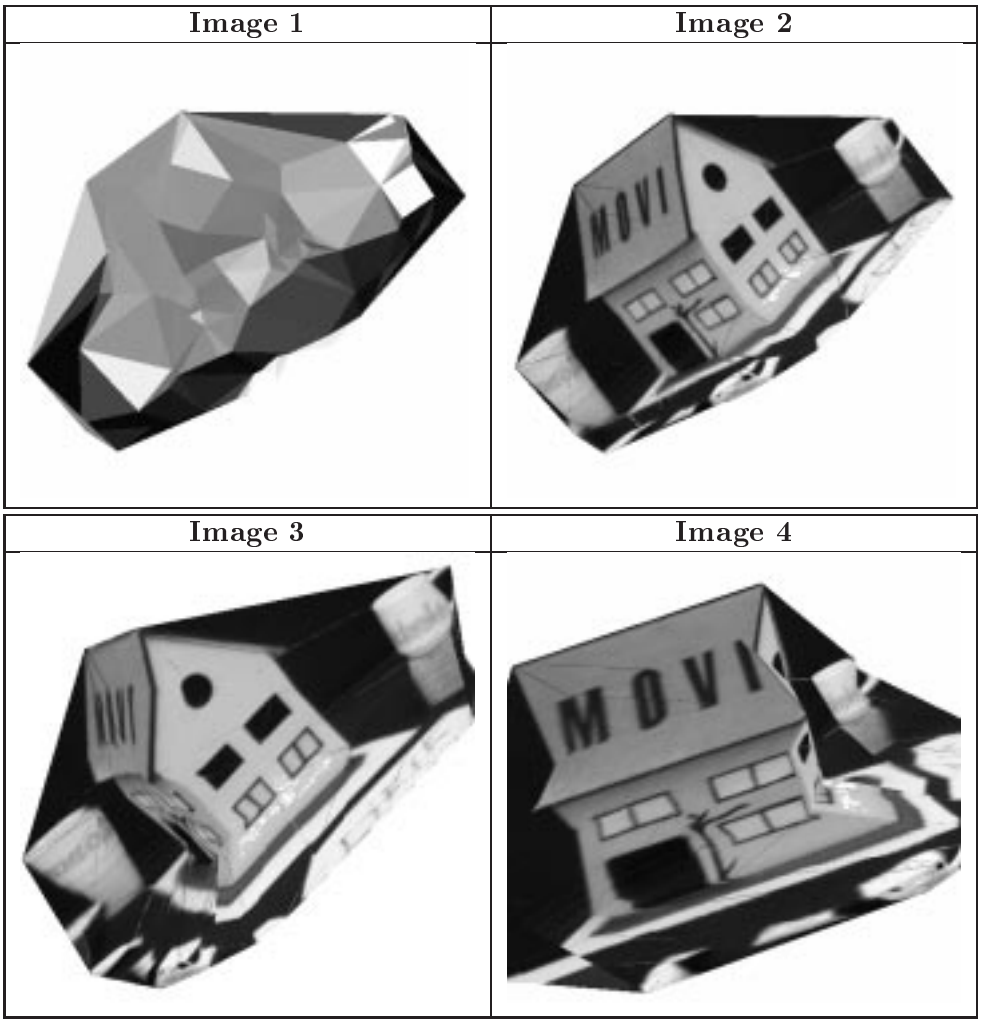


FIG. 4.60: *Quatre vues du modèle en triangles texturés, synthétisées par* VR*Web (seconde série). On ne s'appuie que sur des points d'intérêt, et les occultations ne sont pas prises en compte.*

Les triangulations sont très réduites par rapport aux expérimentations précédentes, puisque la première série compte 707 triangles, et la seconde 169 triangles. Les images obtenues sont aussi de meilleure qualité : même si le toit de la maison MOVI n'est pas plan (image 4 de la figure 4.60), les erreurs sont cependant beaucoup moins fortes sur la figure 4.58. Comme toujours, l'erreur est d'autant moins visible que les points de vue des images synthétisées sont proches des points de vue de référence.

4.10 Conclusion

Dans ce chapitre, nous avons généré de nouvelles images de scènes 3D, sur la base des appariements établis au chapitre précédent.

Nous avons exploré deux cas de figure : le cas étalonné, et le cas non-étalonné. Dans le cas étalonné, nous avons pu comparer les images synthétisées aux images théoriques, car les positions des caméras virtuelles étaient précisément connues. Dans le cas non-étalonné, nous avons dû nous contenter de remarques qualitatives.

Si les images de référence ne forment aucun couple stéréo, alors il n'est pas possible de calculer des informations tridimensionnelles. On peut cependant calculer de nouvelles images, par la technique de construction de mosaïques détaillée en 4.4.

Dans le cas stéréo enfin, nous avons utilisé deux techniques de synthèse : l'une s'appuyant sur un modèle constitué d'un nuage de points 3D isolés, l'autre faisant appel à un modèle en facettes triangulaires texturées.

Le tableau 4.21 résume les différents cas de figure, et les traitements que l'on peut envisager dans chacun des cas.

	Étalonné	**Non-étalonné**
Stéréo	points/triangles	étalonnage + points/triangles
Non-stéréo	mosaïque	mosaïque

TAB. 4.21: *Cas de figure envisageables pour la synthèse d'images à partir de photographies.*

Les remarques que nous pouvons tirer de nos expérimentations sont les suivantes.

- La construction de mosaïques est un processus simple et stable, qui donne de très bons résultats : après une phase initiale d'appariement, le calcul robuste d'homographie est suffisamment stable et précis pour conduire à de bons transferts. De notre point de vue, la technique est mature pour d'éventuels développements industriels. Des travaux plus récents, notamment ceux d'I. Zoghlami à l'INRIA, permettent même d'étendre la technique à des images de référence plus générales, par exemple ayant subi de fortes rotations ou changements de facteur d'échelle [Zog 97].

- L'utilisation d'un modèle 3D est plus complexe. Un nuage de points 3D dispersés est une bonne solution dans la majorité des cas ; des points de vue relativement éloignés des images de référence conduisent à des images de qualité acceptable (dans un débattement de l'ordre de deux fois celui des prises de vue) ; enfin, la dégradation de qualité est progressive, et croît régulièrement à mesure qu'on s'éloigne des positions de référence.

- La construction d'un modèle géométrique : facettes triangulaires texturées, est très complexe à mener à bien. Les données sont en effet en partie fausses, ou absentes. Lorsqu'elles sont absentes, cela peut être dû à une erreur d'appariement (zone uniforme), ou à une impossibilité (zone occultée). Nous ne pouvons pas aisément distinguer entre ces deux cas, et si nous ne voulons pas que des triangles soient construits sur les zones occultées, nous devons accepter que certaines zones non-appariées (pour d'autres raisons) ne soient pas couvertes. Si nous supposons que le modèle est connexe

de genre 0 (sans trou), il n'est pas nécessaire de tenir compte de ce problème, et nous pouvons mailler toute la surface de l'image, en ne nous appuyant que sur des points fiables : triangulation de Delaunay sur des points d'intérêt. Notons enfin que la synthèse d'un modèle en triangles texturés est souvent plus rapide que la projection d'un nuage de points, mais que cela dépend du nombre de triangles, et des possibilités spécifiques de la machine d'affichage ; une UltraSPARC 2, munie d'une carte d'affichage Creator 3D réalisant le mapping de texture, peut afficher 5000 triangles texturés par seconde. Sur cette machine, la synthèse de l'image du château de CMU à partir d'un nuage de points exige 0.3 seconde (projection simple) ou 0.5 seconde (projection sur 4 pixels) ; il suffit donc que le modèle de la scène soit composé de moins de 2500 triangles pour atteindre une vitesse comparable, ce qui est une hypothèse raisonnable pour cette scène.

- Les processus de reconstruction 3D ne sont pas automatisables, et ils nécessitent une grande part d'intervention humaine : décision d'appliquer des filtrages supplémentaires, des reconstructions robustes, de nouvelles étapes de régularisation, et réglage des différents seuils.

- Le problème de l'auto-étalonnage multi-oculaire n'est pas résolu en pratique. Il existe des algorithmes, mais ils ne sont pas adaptés à des données incomplètes, ou imprécises. En l'état, nous n'avons pu appliquer qu'une méthode d'étalonnage binoculaire, nécessitant une connaissance approximative des paramètres intrinsèques.

- Le principal problème reste l'appariement, et bien qu'on dispose de techniques adaptées (la reconstruction robuste en est une), de l'étape de mise en correspondance dépend la bonne marche de toute la chaîne de traitement.

Chapitre 5

Conclusion

Nous avons présenté dans ce mémoire des solutions au problème de synthèse de nouvelles vues d'une scène 3D à partir de vues existantes. Nous ne revenons pas sur les nombreuses applications de ce procédé, qui sont toutes celles de la Réalité Virtuelle, et dont la nouveauté consiste en la suppression de l'étape de modélisation de la scène, aujourd'hui manuelle et fastidieuse. Rappelons que notre problème peut présenter deux aspects :

$\mathcal{P}$1 : synthétiser de nouvelles images de la scène observée ;

$\mathcal{P}$2 : calculer une représentation tridimensionnelle de cette scène.

Les solutions de $\mathcal{P}2$ sont des solutions de $\mathcal{P}1$, car il est possible de synthétiser des vues quelconques d'une scène définie par un modèle 3D adéquat.

Notre contribution à ces problèmes est détaillée ci-dessous.

5.1 Contribution

Nous avons recherché des techniques applicables aux deux problématiques. Il est apparu que les solutions possibles étaient nécessairement basées sur le schéma suivant :

1. calculer des informations tridimensionnelles sur la scène ;
2. utiliser ces informations pour générer de nouvelles vues.

D'une certaine façon, cela correspond à la résolution du problème $\mathcal{P}2$, bien qu'à l'issue de la première phase, on ne puisse pas toujours parler de «représentation tridimensionnelle» au sens strict : nous ne disposons pas nécessairement d'un modèle 3D structuré, mais simplement d'«informations de relief», comme la position relative de certains points dans l'espace.

Nous avons séparément traité les deux phases, en considérant d'un côté les questions liées à l'appariement de structures dans N images (chapitre 3), et de l'autre, l'utilisation de ces appariements pour inférer des informations 3D, et calculer de nouvelles vues (chapitre 4). Les structures appariées se sont limitées à des points des images, car apparier d'autres structures (comme des segments) fait intervenir des pré-traitements (comme la détection de contours) ; la qualité de ces pré-traitements est complexe à définir et à évaluer, et des résultats incertains risquent d'obérer la qualité de tout le processus.

Sur le problème de l'appariement, notre contribution est la suivante :

- nous avons étudié les mesures de corrélation existantes ;
- nous avons proposé de nouvelles mesures de corrélation, dérivées de mesures et de procédés existants (RSAD), ou entièrement nouvelles (PRSAD) ;
- nous avons étudié les algorithmes d'appariement existants ;
- nous avons proposé de nouveaux algorithmes d'appariement, en particulier de nouveaux schémas pour la programmation dynamique ;
- nous avons proposé un algorithme de régularisation des résultats ;
- nous avons étudié les méthodes d'affinage d'appariements, afin d'obtenir une précision sous-pixellique ;
- nous avons proposé et appliqué une méthode d'évaluation numérique de toutes ces techniques, basée sur l'utilisation d'images de synthèse.

Les nombreuses utilisations possibles des mesures de corrélation et des algorithmes de mise en correspondance sont combinatoires. Pour pouvoir les évaluer, nous avons dû sérier les tests, et l'évaluation séquentielle de tous les paramètres disponibles nous amené aux conclusions qui seront présentées ci-dessous, en 5.2.

Concernant le transfert des informations de relief obtenues afin de synthétiser de nouvelles vues, nos apports sont les suivants :

- nous avons étudié et appliqué une méthode simple de calcul de mosaïques d'images, basée sur des calculs rapides, précis, et robustes ;
- nous avons étudié un algorithme de reconstruction 3D binoculaire, dans le cas non-étalonné, fonctionnant pour des données d'appariement imprécises (ajustement épipolaire) ;
- nous avons proposé un nouvel algorithme de reconstruction 3D robuste multi-oculaire, dans le cas étalonné, fonctionnant pour des données d'appariement imprécises *ou incorrectes* ;
- nous avons développé une méthode exacte de calcul de textures, faisant appel à des transformations projectives pour des résultats corrects (les méthodes standard n'utilisent que des transformations affines) ;

- nous avons évalué les images synthétisées, et leur domaine de validité : évolution de la qualité des résultats en fonction de la position de la caméra virtuelle par rapport à la position des caméras des vues de référence.

Ce qui nous semble le plus important est d'avoir pu évaluer quantitativement le processus d'appariement, de façon objective. Pour cela, nous avons créé et utilisé des images de synthèse ; cette méthode était déjà utilisée sur des stéréogrammes aléatoires représentant des plans parallèles, empilés et vus de face, mais ne menait qu'à des considérations qualitatives, menées manuellement. Nous avons généralisé cette approche, en l'étendant à des scènes non-planes, présentant des occultations importantes, et des déformations perspectives. Cela nous a permis d'évaluer précisément les apports des techniques précises et/ou robustes par rapport aux méthodes classiques, ce que nous n'aurions pas pu faire manuellement de façon rigoureuse.

Cependant, nous n'avons utilisé qu'une seule scène 3D, constituée de deux nappes bicubiques. Cela est insuffisant, et il faudra, dans de futures expérimentations, créer plusieurs scènes, plus complexes, représentant des conditions d'expérimentation variées : scènes polyédriques, ou surfaces dérivables, avec ou sans occultations, et de plus grande taille (pour une évaluation plus précise) ; surtout, les textures devront être plus soigneusement étudiées, afin de correspondre à des cas réels, où les lignes de fort gradient d'intensité correspondent souvent à des ruptures de disparité (ce qui n'est pas le cas dans nos images de synthèse).

Enfin, le critère d'évaluation de la qualité des images synthétisées pourrait être notablement amélioré, en prenant en compte des critères perceptuels : perception non seulement de la photométrie, mais aussi de la géométrie de l'ensemble.

5.2 Conclusions

Tirées de nos différentes évaluations, nos conclusions sont les suivantes.

- La technique de mosaïque est simple, robuste, rapide, précise, donne de bons résultats, et ne nécessite pas d'appariement dense. De notre point de vue, elle est immédiatement industrialisable.

- En revanche, les techniques de synthèse faisant appel à de véritables informations 3D nécessitent un appariement dense. En effet, un appariement épars ne fournirait pas suffisamment d'information, et obligerait à faire des hypothèses non vérifiables sur la structure de la scène, dans ses zones non-renseignées. Il faudrait par exemple supposer que tous les pixels situés entre des points appariés appartiennent à des facettes planes ; or cela n'est pas vérifiable, à moins de disposer d'un appariement dense.

- Les solutions au problème de l'appariement dense se situent dans la recherche d'algorithmes simples et coopérants, faisant intervenir plusieurs sources d'information. Dans notre cas, une simple mesure SAD, un simple algorithme d'appariement avec

vérification croisée CCR, suivis d'une étape de régularisation, puis d'affinage, fournissent des résultats acceptables pour notre tâche. Des algorithmes plus intégrés seraient certainement plus efficaces, et nous reviendrons sur ce point ci-dessous, en 5.3.

- Après appariement dense puis reconstruction 3D, la synthèse de nouvelles vues comme reprojection du nuage des points reconstruits reste une méthode simple et de bonne qualité. Certaines techniques, comme la projection sur un ensemble de pixels voisins, ou des filtrages *a posteriori*, permettent d'améliorer encore la qualité visuelle des images calculées. Le processus est robuste, car les points 3D incorrects sont imperceptibles individuellement.

- La construction d'un modèle constitué de triangles texturés est une tâche beaucoup plus complexe. Elle est pour l'instant affaire d'heuristiques, à base de filtres grossiers sur les données, et de réévaluations dirigées. Elle est délicate à mettre en œuvre, car un seul point 3D faux peut provoquer des distorsions catastrophiques dans le maillage généré. Le comportement est moins chaotique si l'on suppose que la surface maillée est connexe, car nous pouvons faire des hypothèses régularisatrices, lissant le résultat aux endroits non renseignés. Il apparaît pour l'instant hors de portée d'automatiser entièrement un tel processus.

5.3 Futurs développements

Le problème majeur reste l'appariement de structures entre plusieurs images. La qualité des appariements est en effet à la base du bon fonctionnement de tout le processus, et l'utilisation de techniques robustes n'apporte qu'un filtrage supplémentaire des données erronées.

Comme nous l'avons déjà évoqué, une solution pour obtenir un appariement dense et fiable réside dans l'utilisation combinée de nombreuses sources d'information. Il est nécessaire d'explorer les techniques de minimisation «très globales», faisant intervenir des données issues des images (intensités, gradients, textures, scores de corrélation, distances entre descripteurs locaux), des données 3D calculées à partir de ces images (positions de points, de plans, gradients, régularité des surfaces observées), et des données issues de l'utilisation des informations 3D (qualité du maillage, régularité, qualité des images synthétisées par rapport aux images originales). Tous ces paramètres sont liés, et la seule donnée invariante est constituée par les images de référence[1]. Un tel processus nécessite donc quelques hypothèses *a priori*, se renforçant mutuellement, ou se contrebalançant, au cours du calcul.

Nous croyons que par sa conception, un tel système pourrait améliorer notablement la qualité et la flexibilité des algorithmes sous-jacents de mise en correspondance. Bien sûr, il

1. Dans le cadre de la vision active, les images de référence ne constitueraient pas une donnée invariante, car les conditions de prise de vue pourraient être modifiées à la demande, en cours du calcul. Néanmoins, cela nous interdirait de travailler à partir de photographies.

faudrait développer ceci dans un cadre strict d'évaluation quantitative et rigoureuse, portant sur de nombreuses images, entièrement connues et représentant des domaines d'application variés : images aériennes, de bâtiments, de paysages, de surfaces continues et/ou dérivables, texturées ou non. Seule une évaluation quantitative nous permettra d'améliorer les processus en connaissance de cause, et de progresser vers une solution à ce problème.

Bibliographie

[Ack 84] F. Ackermann. Digital image correlation : Performance and potential application in photogrammetry. *Photogrammetric Record*, 64(11): 429–439, October 1984.

[Asc 92] P. Aschwanden and W. Guggenbühl. Experimental results from a comparative study on correlation-type registration algorithms. In Förstner and Ruwiedel, editors, *Robust Computer Vision*, pages 268–282. Wichmann, 1992.

[Bai 96] C. Baillard, O. Dissard, O. Jamet, and H. Maitre. Appariement stéréoscopique d'images aériennes en milieu péri-urbain. In *Actes du 10ème Congrès AFCET de Reconnaissance des Formes et Intelligence Artificielle, Rennes, France*, pages 247–256, January 1996.

[Bak 81] H.H. Baker and T.O. Binford. Depth from edge- and intensity- based stereo. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 631–636, August 1981.

[Bak 82] H.H. Baker and T.O. Binford. A system for automated stereo mapping, August 1982.

[Bal 96] J.F. Balaguer. VRML for LHC engineering. In *Journées Nationales du Groupe de Travail "Réalité Virtuelle" - Toulouse, France*, pages 67–72, October 1996.

[Bar 97] D. Barba and N. Bekkat. Modélisation psychovisuelle – représentation psychovisuelle et critère de qualité d'images. In A. Chéhikian, P.-Y. Coulon, and F. Luthon, editors, *École des Techniques Avancées Signal Image Parole – Grenoble, 1997*, pages 1–33. Institut National Polytechnique de Grenoble, September 1997. Volume 2.

[Bha 96] D.N. Bhat and S.K. Nayar. Ordinal measure for visual correspondence. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Francisco, California, USA*, pages 351 – 357, San Francisco, California, June 1996.

[Bla 94] J. Blanc. Reconstruction 3D pour la synthèse d'images - rapport de DEA, June 1994.

[Bla 95] J. Blanc and R. Mohr. Calcul de vues de scènes 3D. Application à la compression vidéo. In *Journées d'Études et d'Échanges*, pages 125–128. CCETT, January 1995.

[Bla 97] J. Blanc, P. Sturm, G. Giraudon, and R. Mohr. Étude bibliographique sur les systèmes de modélisation de bâtiments à partir d'images aériennes ou satellitaires, April 1997. Unpublished. Contact Jerome.Blanc@inrialpes.fr.

[Bob 96] P. Bobet, J. Blanc, and R. Mohr. Aspects cachés de la tri-linéarité. In *Actes du 10ème Congrès AFCET de Reconnaissance des Formes et Intelligence Artificielle, Rennes, France*, pages 137–146. LIFIA–IMAG–INRIA Rhône-Alpes, January 1996.

[Bou 94] B. Boufama. *Reconstruction tridimensionnelle en vision par ordinateur : cas des caméra non étalonnées.* Thèse de doctorat, Institut National Polytechnique de Grenoble, December 1994.

[Bra 95] P. Brand. *Reconstruction tridimensionnelle d'une scène à partir d'une caméra en mouvement : de l'influence de la précision.* Thèse de doctorat, Université Claude Bernard, Lyon I, October 1995.
ftp://ftp.imag.fr/pub/MOVI/theses/brand.ps.gz.

[Cas 97] D. Casasent. New techniques for object detection and recognition. In *Proceedings of the 10th Scandinavian Conference on Image Analysis, Lappeenranta, Finland*, pages 597–604, June 1997. Oral communication.

[Che 94] Y.Q. Cheng, R.T. Collins, A.R. Hanson, and E.M. Riseman. Triangulation without correspondences. In *Proceedings of* ARPA *Image Understanding Workshop, Monterey, California, USA*, pages 993–1000, November 1994.

[Che 95a] S.E. Chen. Quicktime VR - an image-based approach to virtual environment navigation. In SIGGRAPH *1995, Los Angeles*, pages 29–38, 1995.

[Che 95b] S.E. Chen and G.S.P. Miller. Cylindrical to planar image mapping using scanline coherence, March 1995. U.S. Patent No. 5,396,583.

[Chr 98] S. Christy. *Modélisation tridimensionnelle d'objets quelconques par vision dynamique.* Thèse de doctorat, Institut National Polytechnique de Grenoble, GRAVIR – IMAG – INRIA Rhône–Alpes, 1998. To appear.

[Col 96] R.T. Collins. A space-sweep approach to true multi-image matching. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Francisco, California, USA*, pages 358–363, June 1996.

[Cox 92] I.J. Cox, S. Hingorani, B.M. Maggs, and S.B. Rao. Stereo without regularization, October 1992.

[Cox 96] I.J. Cox, S.L. Hingorani, S.B. Rao, and B.M. Maggs. A maximum likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63(3): 542–567, May 1996.

[Deb 96a] P.E. Debevec, C.J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: a hybrid geometry-and image-based approach. Technical Report CSD-96-893, University of California, Berkeley, January 1996.

[Deb 96b] 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*, New Orleans*, August 1996.

[Fau 92] O. 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, Santa Margherita Ligure, Italy*, pages 563–578. Springer-Verlag, May 1992.

[Fau 93a] O. Faugeras and L. Robert. What can two images tell us about a third one? Technical report, INRIA, July 1993.

[Fau 93b] O. Faugeras, T. Viéville, E. Théron, J. Vuillemin, B. Hotz, Z. Zhang, L. Moll, P. Bertin, H. Mathieu, P. Fua, G. Berry, and C. Proy. Real time correlation-based stereo: Algorithm, implementations and applications. Technical Report 2013, INRIA, August 1993.

[Fau 94] O. Faugeras and L. Robert. What can two images tell us about a third one? In J.O. Eklundh, editor, *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 485–492. Springer-Verlag, 1994.

[Fau 95a] O. Faugeras, S. Laveau, L. Robert, G. Csurka, and C. Zeller. 3D reconstruction of urban scenes from sequences of images. Technical Report 2572, INRIA, June 1995.

[Fau 95b] O. Faugeras and B. Mourrain. About the correspondences of points between n images. In *Workshop on Representation of Visual Scenes, Cambridge, Massachusetts, USA*, pages 37–44, June 1995.

[Fau 95c] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 951–956, June 1995.

[Fua 91] P. Fua. Combining stereo and monocular information to compute dense depth maps that preserve discontinuities. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence, Sydney, Australia*, August 1991.

[Fua 94a] P. Fua. Reconstructing complex surfaces from multiple stereo views. Technical report, SRI International, November 1994.

[Fua 94b] P. Fua and Y.G. Leclerc. Registration without correspondences. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Seattle, Washington, USA*, pages 121–128, June 1994.

[Fua 94c] P. Fua and Y.G. Leclerc. Using 3-dimensional meshes to combine image-based and geometry-based constraints. In *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 281–291, May 1994.

[Fua 95a] P. Fua. Reconstructing complex surfaces from multiple stereo views. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, June 1995.

[Fua 95b] P. Fua. Surface reconstruction using 3-D meshes and particle systems. In *Third International Workshop on High Precision Navigation, Stuttgart, Germany.* Dümmler-Verlag, April 1995.

[Gei 92] D. Geiger, B. Ladendorf, and A. Yuille. Occlusions and binocular stereo. In G. Sandini, editor, *Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy*, pages 425–433. Springer-Verlag, 1992.

[Gru 85] A.W. Gruen. Adaptative least squares correlation: a powerful image matching technique. *S. Afr. Journal of Photogrammetry, Remote Sensing and Cartography*, 14(3): 175–187, 1985.

[Har 94] R. Hartley and P. Sturm. Triangulation. In *Proceedings of* ARPA *Image Understanding Workshop, Monterey, California, USA*, pages 957–966, November 1994.

[Har 95] R. Hartley. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 1064–1070, June 1995.

[Hut 96] D.P. Huttenlocher, R.H. Lilien, and C.F. Olson. Object recognition using subspace methods. In B. Buxton and R. Cipolla, editors, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 536–545. Springer-Verlag, April 1996.

[Kan 91] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. In *Proceedings of* IEEE *International Conference on Robotics and Automation, Sacramento, California, USA*, pages 1088–1095, April 1991.

[Kan 95] T. Kanade, P.J. Narayanan, and P.W. Rander. Virtualized reality: Concepts and early results. In *Workshop on Representation of Visual Scenes, Cambridge, Massachusetts, USA*, pages 69–76, June 1995.

[Koc 94] R. Koch. 3D scene modeling from stereoscopic image sequences. In *Image Processing for Broadcast and Video Production 1994, Hamburg, Germany*, pages 128–135, 1994.

[Koc 95] R. Koch. 3D surface reconstruction from stereoscopic image sequences. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 109–114, June 1995.

[Kum 94] R. Kumar, P. Anandan, and K. Hanna. Direct recovery of shape from multiple views: a parallax based approach. In *Proceedings of the 12th International Conference on Pattern Recognition, Jerusalem, Israel*, pages 685–688, 1994.

[Kum 95] R. Kumar, P. Anandan, M. Irani, J. Bergen, and K. Hanna. Representation of scenes from collections of images. In *Workshop on Representation of Visual Scenes, Cambridge, Massachusetts, USA*, pages 10–17, June 1995.

[Lan 97] Z.D. Lan. *Méthodes robustes en vision : application aux appariements visuels.* Thèse de doctorat, Institut National Polytechnique de Grenoble, 1997.

[Lav 94a] S. Laveau and O. Faugeras. 3D scene representation as a collection of images and fundamental matrices. Technical report, INRIA, February 1994.

[Lav 94b] S. Laveau and O.D. Faugeras. 3D scene representation as a collection of images. In *Proceedings of the 12th International Conference on Pattern Recognition, Jerusalem, Israel*, volume 1, pages 689–691, 1994.

[Lav 96] S. Laveau. *Géométrie d'un système de N caméras. Théorie, estimation, et applications.* Thèse de doctorat, École Polytechnique, May 1996.

[Lec 97] P. Lechat, G. Le Mestre, and D. Pelé. An approach for scene reconstruction from the analysis of a triplet of still images. In *Electronic Imaging 1997*, 1997. Yet unpublished.

[Lev 96] M. Levoy and P. Hanrahan. Light field rendering. In SIGGRAPH *1996, New Orleans*, pages 31–42, 1996.

[Lot 96] J.L. Lotti. *Mise en correspondance stéréo par fenêtres adaptives en imagerie aérienne haute résolution.* Thèse de doctorat, Université de Nice – Sophia Antipolis, February 1996.

[Luo 92] Q.T. Luong. *Matrice fondamentale et autocalibration en vision par ordinateur.* Thèse de doctorat, Université de Paris-Sud, Orsay, France, December 1992.

[McM 95] L. McMillan and G. Bishop. Plenoptic modelling: an image-based rendering system. In SIGGRAPH *1995, Los Angeles*, pages 39–46, 1995.

[Mes 96] G. Le Mestre and D. Pelé. Trinocular image analysis for virtual frame reconstruction. In *VCIP 1996, Orlando*, 1996.

[Moh 93] R. Mohr. Projective geometry and computer vision. In C.H. Chen, L.F.Pau, and S.P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision.* World Scientific Pub., 1993.

[Mur 95] H. Murase and S.K. Nayar. Visual learning and recognition of 3D objects from appearance. *International Journal of Computer Vision*, 14: 5–24, 1995.

[Nie 95] W. Niem and H. Broszio. Mapping texture from multiple camera views onto 3D-object models for computer animation. In *Proceedings of the International Workshop on Stereoscopic and Three Dimensional Imaging, Santorini, Greece*, September 1995.

[Oht 85] Y. Ohta and T. Kanade. Stereo by intra and inter-scanline search using dynamic programming. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 7(2): 139–154, 1985.

[Ois 96] L. Oisel, L. Morin, B. Gasnier, and C. Labit. Application de la géométrie projective à la compression en TV3D. In *Journées ORASIS 1996, Clermont-Ferrand, France*, pages 129–134, May 1996.

[Oku 93] M. Okutomi and T. Kanade. A multiple-baseline stereo. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 15(4): 353–363, April 1993.

[O'N 92] M.A. O'Neill and M.I. Denos. Practical approach to the stereo matching of urban imagery. *Image and Vision Computing*, 10(2): 89–98, March 1992.

[Oua 96] M.H. Ouali, H. Lange, and C. Laurgeau. An energy minimization approach to dense stereovision. In *Proceedings of the* IEEE *International Conference on Image Processing*, pages 841–846, September 1996.

[Pog 96] T. Poggio and A. Shashua. Image processing system with frame store and having recording and rendering systems, August 1996. U.S. Patent No. 5,550,641.

[Pol 85a] S.B. Pollard. *Identifying Correspondences in Binocular Stereo.* PhD thesis, University of Sheffield, 1985.

[Pol 85b] S.B. Pollard, J.E.W. Mayhew, and J.P. Frisby. PMF: A stereo correspondence algorithm using a disparity gradient constraint. *Perception*, 14: 449–470, 1985.

[Pra 78] W.K. Pratt. *Digital Image Processing.* Wiley-Interscience, 1978.

[Raj 87] G.V.S. Raju, T.O. Binford, and S. Shekhar. Stereo matching using the Viterbi algorithm. In *Proceedings of* DARPA *Image Understanding Workshop, Los Angeles, California, USA*, pages 766–776, February 1987.

[Rob 96] L. Robert and R. Deriche. Dense depth map reconstruction: a minimization and regularization approach which preserves discontinuities. In *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 439–451, April 1996.

[Rob 97] L. Robert. Modèles réalistes bâtis à partir de séquences d'images. In IMAGINA 1997*, Monaco*, February 1997.

[Rot 95] C. Rothwell, G. Csurka, and O. Faugeras. A comparison of projective reconstruction methods for pairs of views. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 932–937, June 1995.

[Saw 95a] H.S. Sawhney, S. Ayer, and M. Gorkani. Model-based 2D & 3D dominant motion estimation for mosaicing and video representation. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 583–590, June 1995.

[Saw 95b] H.S. Sawhney, S. Ayer, and M. Gorkani. Model-based 2D & 3D dominant motion estimation for mosaicing and video representation, 1995.

[Sch 96a] D. Scharstein. Stereo matching with non-linear diffusion. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Francisco, California, USA*, pages 343–350, June 1996.

[Sch 96b] C. Schmid. *Appariement d'images par invariants locaux de niveaux de gris.* Thèse de doctorat, Institut National Polytechnique de Grenoble, GRAVIR – IMAG – INRIA Rhône–Alpes, July 1996.

[Sei 95] S.M. Seitz and C.R. Dyer. Physically-valid view synthesis by image interpolation. In *Workshop on Representation of Visual Scenes, Cambridge, Massachusetts, USA*, pages 18–25, June 1995.

[Sei 96] S.M. Seitz and C.R. Dyer. View morphing. In SIGGRAPH *1996, New Orleans*, pages 21–30, 1996.

[Sei 97] S.M. Seitz and C.R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Puerto Rico, USA*, pages 1067–1073. IEEE Computer Society Press, June 1997.

[Ser 94] B. Serra and M. Berthod. Subpixel contour matching using continuous dynamic programming. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Seattle, Washington, USA*, pages 202–207, June 1994.

[Ser 95] B. Serra and M. Berthod. Optimal subpixel matching of contour chains and segments. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 402–407, June 1995.

[Sha 94] A. Shashua. Trilinearity in visual recognition by alignment. In J.O. Eklundh, editor, *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 479–484. Springer-Verlag, May 1994.

[Sir 87] L. Sirovitch and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America*, 2: 586–591, 1987.

[Str 95] A. Streilein. Videogrammetry and CAAD for architectural restitution of the Otto-Wagner-Pavillon in Vienna. In A. Gruen and Kahmen, editors, *Optical* 3D *Measurement Techniques III*, pages 305–314. Wichmann Verlag, Heidelberg, 1995.

[Stu 96] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In B. Buxton and R. Cipolla, editors, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, volume 1065 of *Lecture Notes in Computer Science*, pages 709–720. Springer-Verlag, April 1996.

[Stu 97] P. Sturm. Critical motion sequences and conjugacy of ambiguous euclidean reconstructions. In M. Frydrych, J. Parkkinen, and A. Visa, editors, *Proceedings of the 10th Scandinavian Conference on Image Analysis, Lappeenranta, Finland*, volume I, pages 439–446, June 1997.

[Sze 95a] R. Szeliski and S.B. Kang. Direct methods for visual scene reconstruction. In *Workshop on Representation of Visual Scenes, Cambridge, Massachusetts, USA*, pages 26–33, June 1995.

[Sze 95b] R. Szeliski, S.B. Kang, and H.Y. Shum. A parallel feature tracker for extended image sequences. Technical report, Digital Equipment Corporation, Cambridge Research Lab, May 1995.

[Tom 96a] C. Tomasi and R. Manduchi. Stereo without search. In *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 452–465, April 1996.

[Tom 96b] C. Tomasi and R. Manduchi. Stereo without search, 1996.

[Tri 97a] B. Triggs. Autocalibration and the absolute quadric. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Puerto Rico, USA*, pages 609–614. IEEE Computer Society Press, June 1997.

[Tri 97b] B. Triggs. Linear projective reconstruction from matching tensors. *Image and Vision Computing*, 15(8): 617–625, August 1997.

[Tsa 86] R.Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Miami Beach, Florida, USA*, pages 364–374, 1986.

[Tur 91] M. Turk and A. Pentland. Face recognition using eigenfaces. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Maui, Hawaii, USA*, pages 586–591, 1991.

[Ull 91] S. Ullman and R. Basri. Recognition by linear combinations of models. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 13(10): 992–1006, 1991.

[Wer 94] T. Werner. Rendering real-world objects without 3D model. Technical report, Czech Technical University, Dept. of Control, Faculty of Electrical Engineering, Czech Technical University, Karlovo nám. 13, 12135 Praha, Czech Republic, 1994.

[Wer 95] T. Werner, R.D. Hersch, and V. Hlavac. Rendering real-world objects using view interpolation. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 957–962, June 1995.

[Zab 94] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondance. In *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 151–158. Springer-Verlag, May 1994.

[Zah 92] M. Zahid. *Stéréovision en imagerie aérienne : reconstruction tridimensionnelle des bâtiments.* Thèse de doctorat, Université Paris 11 Orsay, May 1992.

[Zha 94] 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. Rapport de recherche 2273, INRIA, May 1994.

[Zog 97] I. Zoghlami, O. Faugeras, and R. Deriche. Using geometric corners to build a 2D mosaic from a set of images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Puerto Rico, USA*, pages 420–425, June 1997.

Résumé

La synthèse d'images a pour but de calculer des vues aussi réalistes que possible d'une scène tridimensionnelle définie par un modèle géométrique. Cette modélisation est effectuée manuellement, et pour synthétiser de façon réaliste une scène complexe, telle qu'un paysage, cette étape fastidieuse peut demander plusieurs hommes-mois.

Nous proposons d'automatiser cette tâche. En effet, quelques photographies du paysage suffisent à modéliser entièrement ses informations géométriques et photométriques : structure 3D, couleurs et textures. Aussi, en appliquant des techniques d'analyse d'images et de vision par ordinateur, nous pouvons générer automatiquement une représentation tridimensionnelle de la scène, et la visualiser sous d'autres points de vue.

Les algorithmes appropriés sont évalués et spécialement adaptés à notre problème. Des tests quantitatifs détaillés sont menés sur des données synthétiques et réelles, et la qualité finale des images produites est évaluée numériquement.

Mots-clés : vision par ordinateur, stéréovision, appariement, reconstruction 3D, synthèse d'images.

Abstract

The aim of image synthesis is to compute realistic views of a three-dimensional scene, defined by a geometric 3D model. Modeling is a manual task, and may need several man-months for a realistic and complex scene, as a landscape.

We propose to automate this task. A few photographs of the landscape are enough to entirely model its geometrical and photometric information: 3D structure, colors and textures. Therefore, using image analysis and computer vision techniques, we can automatically generate a three-dimensional representation of the scene, and view it under any angle.

Appropriate algorithms are evaluated and specifically adapted to our problem. Detailed quantitative tests are led on both synthetic and real data, and the final quality of the computed images is numerically evaluated.

Keywords: computer vision, stereovision, matching, 3D reconstruction, image synthesis.