N° d'ordre: 3612

THÈSE

Présentée devant

l'Université de Rennes 1

pour obtenir le grade de :

Docteur de l'Université de Rennes 1
Mention Informatique

par

Gaël Sourimant

Équipe d'accueil : Temics/IRISA
École Doctorale : Mathématiques, Informatique, Signal, Electronique, Télécommunications (MATISSE)
Composante universitaire : Institut de Formation Supérieure en Informatique et Communication (IFSIC)

Titre de la thèse :

Reconstruction de scènes urbaines à l'aide de fusion de données de type GPS, SIG et Vidéo

soutenue le 20 Décembre 2007 devant la commission d'examen

M.	Pascal	Guitton	Président, Rapporteur
Mme.	Marie-Odile	Berger	Rapportrice
M.	Vincent	Charvillat	Examinateur
M.	Jerôme	Royan	Examinateur
M.	Eric	Marchand	Membre Invité
Mme.	Luce	Morin	Encadrante
M.	Kadi	Bouatouch	Directeur de Thèse

Remerciements

Le Moteur

L'Étiquette voudrait que je commence à adresser mes remerciements aux membres de mon jury. J'ai jamais rien compris à toutes ces histoires d'étiquettes. C'est pourquoi votre serviteur préfère tout d'abord s'adresser aux personnes sans qui ce manuscrit serait resté à l'état de vague spéculation.

Luce, ça m'aurait fait mal d'avoir ton nom à une place autre que le sommet de cette page. Ces quatre ans où tu as dû me supporter m'ont énormément apporté, sur bien des plans. Et ce en dépit de toutes tes petites manies insuportables au possible, comme ce sourire permanent, cette pédagogie avec laquelle tu me mets l'air de rien sur la voie quand je suis à côté de mes pompes, tes gateaux aux noix et tout ce que je peux oublier d'autre. Merci Chef !

Merci à toi aussi Kadi. Tu as toujours été là pour dire ce qui allait ou non, pour me mettre discrètement des coups de pied où je pense quand il le fallait, et pour me dire d'arrêter quand j'en avais besoin. Je ne compte plus le nombre de fois où une simple discussion avec toi a relancé ma motivation pour continuer.

Une fois n'est pas coutume, je voudrais également ici avoir une pensée pour Huguette, qui est toujours là quand il faut pour s'occupper des ses p'tits, pousser de rares coups de gueule[i], et pour être aussi adorable, tout simplement. Je t'embrasse Huguette.

Les Sages

Je tiens également à remercier Marie-Odile Berger, pour avoir accepté de rapporter ce manuscrit, ainsi que Pascal Guitton, pour les mêmes raisons et pour avoir accepté également la charge de présider le jury.

[i] Oui : tout est relatif

Merci encore aux examinateurs — Vincent Charvillat et Jerôme Royan — d'une part pour avoir eu la patience de relire le manuscrit, et d'autre part pour les discussions fructueuses et agréables que l'on a pu avoir en dehors.
Un merci tout particulier à Eric Marchand, membre invité de ce jury, pour tous ses conseils et le partage de son expérience. Certains travaux de ce manuscrit n'auraient jamais abouti sans lui.
Enfin, un grand merci à Christine Guillemot pour m'avoir accepté dans son équipe pendant ces quatre années de stage et de thèse.

Les tripatouilleurs d'équations

Un grand merci à tous ces scientifiques de tout poil, ces pourvoyeurs d'idées, ces bidouilleurs d'équation, avec qui l'inspiration a pu venir au petit bonheur la chance (ou pas d'ailleurs).
Un grand merci en premier lieu à Thomas, mon fidèle ~~destrier~~ ~~esclave~~ stagiaire, qui en plus d'avoir fait un boulot au delà de toute attente, a eu le culot d'être quelqu'un de très agréable. Et ce malgré ses coups de folie. *Gniii !*
Un grand merci également à Julie — ou Jérôme selon l'humeur — pour toutes ces tergiversations, entre autres sur des sujets scientifiques, et parfois même sur le mien. La conclusion en restera toujours la même : *Ah ouais, ça a l'air marrant à faire ce truc ! Dommage que j'aie pas le temps d'essayer...*
En vrac, et dans un ordre plus ou moins pseudo-aléatoire, un grand merci également à : Alex (le pilleur de pots), Saïmone (l'incarnation même de la naïveté), Hervé (Ah tiens ! J'ai une question !), Vivien (Ah tiens ! J'ai une question ![ii]), François (Maître *ès* 'Pataphysique), Laurent (L'huile dans le moteur, c'est lui), Khaled (le collègue de bureau le plus patient du monde — merci pour la lampe au fait !), Aurélie (on a pas des verres à étrenner ?), Christian Bouville, Jerôme Royan, Raphaèle Balter et Patrick Gioia (La team FT), toute l'équipe des anciens et nouveaux de chez Temics : Christel, Denis, Olivier, Angélique (la courageuse remplaçante de Khaled, et qui a peur de ma boîte à mouette), Teddy, Caroline, Matthieu, Sebastien, Fabien, Florelle, Marion, Cécile, et tous les autres[iii].

La Famille

En toute simplicité, je voudrais embrasser tous les membres de ma famille, qui ont supporté sans trop rien dire ces quelques années de présence partielle : mes parents : Maman, Papa, Pierre, Danièle ; mes grand-parents : Yvonne, Joseph, Raymonde, Maurice, Monique ; ma fratrie : Romain, Morgane, John, Marion, Emilie ; mon onclerie[iv] : Pierre, Gwenaëlle, Yannick, Odile, François-Xavier, Roselyne ; et enfin ma cousinerie : Emmanuelle, Fabien, Aude, Natacha, Nicolas, Pierre-Emmanuel.
Ce manuscrit vous est dédié.

[ii] Ceci n'est pas une erreur de copier-coller
[iii] Je parie que j'en ai oublié
[iv] Les néologismes sont nos amis

Le Clan

Alors là on en arrive au point où je risque de froisser quelques susceptibilités en oubliant malgré moi certaines personnes. Je vais devoir faire en sorte de diffuser ce manuscrit le moins possible pour éviter les ennuis.
Toujours dans un ordre tenant plus de la théorie du chaos que de la classification réfléchie, je salue : ma p'tite Flo (évidemment), Camille (un croisement entre un écureuil et un rayon de soleil[v]), Poupou (le Moulin Rouge, les p'tites pépés, tout ça...), Christian (pardon encore pour ta porte mon vieux), Maëlle et Jildaz (la boîte à mouette, c'est eux !), F$ (merci d'avoir été là quand j'en avais besoin mon poulet), Romu (pour les mêmes raisons que F$, et pour m'avoir appris que l'éléphant était le seul animal qui ne pouvait pas sauter), G (mon poussin), J.ny et Solène (si si, elles arrivent les photos !), Lanou et toute sa famille (Mazel Tov !), Gaël et Laëtitia (pour la dernière fois : Non ! Il ne s'appelle pas Stéphane !), Doudou (un homme qui sait ce qu'il veut. Des fois. Et vraiment par hasard.), Igor (notre Chef spirituel à tous), Jerôme (la pile électrique qui parle sans s'arrêter, et à qui je dois énormément...), Aurélie (pour un tas de raisons que tu connais), ma Marie (pas la sienne), Polo (fini les douches après le bad' !), Seb, Rom, Philou, Pedro, Melissa, Titi et Soso, Germain, Filo, Raph, Agathe, tous les autres pandas, Hélène (poutit chat !), Mélanie, Zofia, Boris (ou Jerem[vi]), sa Marie (pas la mienne), Camille, son autre Marie (la voisine), les poulets Toulousains, Romu, Zazou, Gladys, ...
En cas de réclamation pour un oubli inopiné, merci de ne pas frapper sur la tête.

[v] C'est *trouu* mignonnn !!
[vi] Ou Boris

Table des matières

Introduction

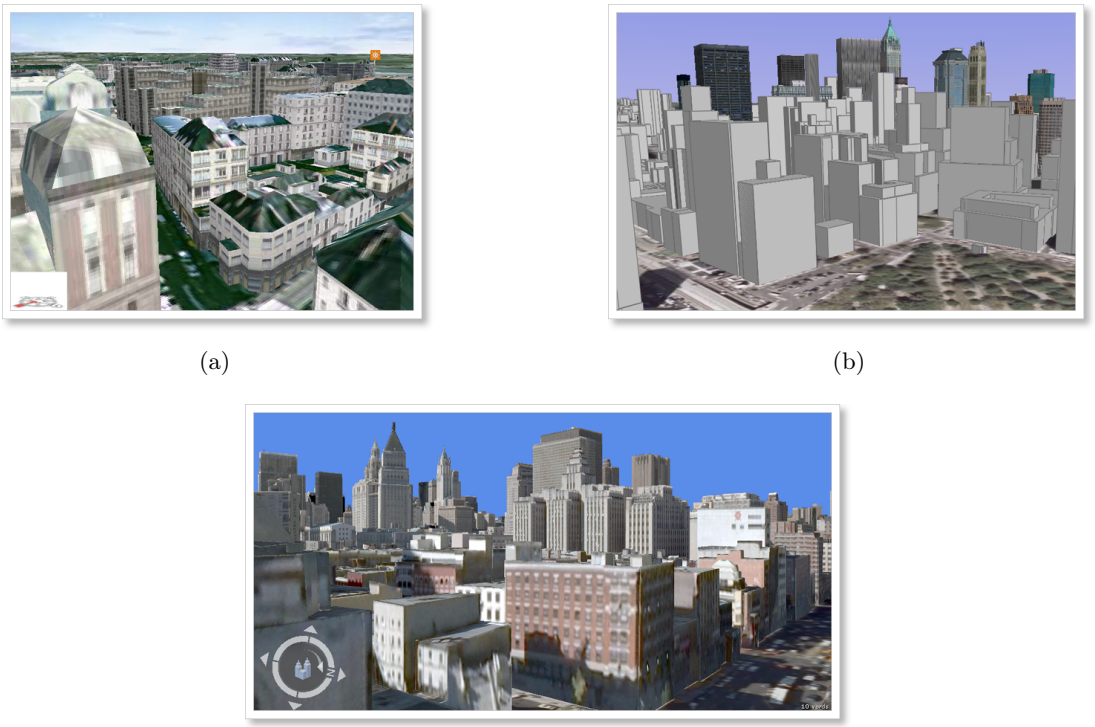
The most exciting phrase to hear in science, the one that heralds new discoveries, is not 'Eureka!' (I found it!) but 'Oh! That's funny...' - **Isaac Asimov**

Contexte

Le succès récent de plates-formes telles que Google Earth[vii], Virtual Earth 3D[viii] ou V3D[ix] montre que l'utilisation de textures photo-réalistes sur une carte géographique 2D apporte beaucoup d'information pour l'utilisateur par rapport à une carte symbolique traditionnelle. Les fonctionnalités nouvelles offertes par ces outils, telles que la navigation ou la représentation en 3D des bâtiments, sont une autre raison de leur succès. Cependant, les modèles 3D fournis sont souvent simples et les images plaquées sur les façades manquent souvent de réalisme ou de précision. Dans Google Earth, seuls quelques bâtiments connus sont texturés. Dans V3D, des textures prédéfinies sont utilisées. Elles sont générées synthétiquement à partir des données du cadastre (année de construction, nombre d'étages, etc.). Enfin, dans Virtual Earth, des photos aériennes obliques sont utilisées pour texturer tous les bâtiments. Elles ne sont donc pas de très bonne résolution, et des erreurs d'alignement apparaissent parfois. En ce qui concerne la géométrie des bâtiments, ces derniers sont représentés par des polyèdres simples dans tous les cas. Seuls les bâtiments les plus significatifs sont parfois modélisés plus finement, mais une telle modélisation est manuelle. Les fonctionnalités de visualisation 3D de ces plates-formes sont illustrées sur la figure 1.

L'objectif de ces travaux de thèse est de proposer une méthode permettant d'estimer des textures photo-réalistes plus précises de bâtiments. Ceux-ci étant modélisés par des polyèdres simples, nous souhaitons également raffiner leur géométrie pour ajouter aux modèles plans de façades des détails géométriques tels que les portes ou les fenêtres. On

[vii] `http://maps.google.com`
[viii] `http://maps.live.com`
[ix] `http://v3d.pagesjaunes.fr`



(a) (b)

(c)

FIG. 1 – Visualisation de zones urbaines en 3D via les plateformes (a) V3D, (b) Google Earth et (c) Virtual Earth

parle de la "micro-structure " des bâtiments.
De nombreux modèles 3D simples de bâtiments sont d'ores et déjà disponibles[x]. Ils sont construits en utilisant des données aériennes (photographies, images radar, etc.) et permettent une reconstruction des villes à grande échelle. Par contre, pour estimer la micro-structure des façades ainsi que des textures correspondantes de façon plus précise, des données acquises au niveau du sol sont nécessaires, puisque les façades ne sont que très peu visibles dans des données aériennes. Notre approche est donc basée sur la mise en correspondance de données multimodales, à savoir des vidéos prises au sol de bâtiments, une base de donnée de type SIG (*Système d'Informations Géographiques*) composée d'un ensemble de modèles 3D géo-référencés de bâtiments décrits par leur empreinte au sol et leur élévation, ainsi que des mesures de positionnement par GPS acquises conjointement avec les vidéos, qui font la passerelle entre les vidéos et les modèles 3D. Si la modélisation 3D extraite de la base SIG est pertinente pour une visualisation aérienne, elle n'est pas satisfaisante pour une navigation 3D au niveau du sol car elle n'offre pas un degré de réalisme suffisant. La vidéo et la base SIG contiennent donc des informations complémentaires : la vidéo fournit le photo-réalisme et les détails géomé-

[x]Par exemple, la visualisation 3D des villes françaises sera publique en 2007, selon l'IGN

triques des bâtiments, tandis que les modèles SIG donnent une géométrie "propre" et complète de la scène, structurée en bâtiments individuels.
Pour pouvoir combiner ces différents types de données, il est nécessaire de les mettre en correspondance dans le même système de coordonnées. Une telle relation entre chaque image de la vidéo et les modèles 3D de bâtiments permet de déterminer quelles images sont utilisables pour avoir la possibilité ensuite de texturer et raffiner telle ou telle façade. Cette mise en correspondance est de fait le point sensible du système. Ce recalage de données 2D-3D se rattache au domaine de la vision par ordinateur.
La seconde étape de notre méthode consiste à exploiter le recalage estimé précédemment entre les modèles 3D et la vidéo pour en extraire des textures photo-réalistes des bâtiments et estimer leur micro-structure. Cette partie est également associée au domaine de la vision mais également aux domaines du traitement et de la synthèse d'image.

Contributions et organisation du manuscrit

Nos contributions se décomposent en trois parties :

1. *Recalage automatique* SIG-*vidéo*
   Pour chaque image de la séquence, on détermine la position et l'orientation de la caméra dans le repère géo-référencé, de telle sorte que la projection perspective du modèle 3D dans le plan image de la caméra soit alignée avec les contours des bâtiments dans l'image. En particulier, une méthode d'initialisation automatique du recalage pour la première image est proposée. Elle précède une phase de suivi temporel du recalage des modèles pour toutes les images restantes.
2. *Extraction et fusion de textures*
   Le recalage entre les images source et la projection du modèle 3D correspondant est exploité pour calculer la texture finale de chaque façade visible du modèle. Chaque façade visible permet de générer une texture de celle-ci à partir de chaque image de la vidéo dans laquelle elle est visible. Cette texture sera généralement incomplète, masquée par des objets du premier plan, et plus ou moins floue selon la méthode d'extraction et la configuration géométrique de la caméra. Une méthode de fusion de textures pour le calcul de la texture finale de la façade est alors proposée.
3. *Calcul de la micro-structure*
   Nous présentons une étude préliminaire sur la faisabilité de l'extraction des micro-structures des façades, en utilisant les textures calculées à l'étape précédentes. Cette estimation se base sur le calcul d'un champ de mouvement dense entre les-dites textures.

Le plan de l'étude détaillant ces contributions est donc le suivant : dans la partie I, nous présentons tout d'abord un état de l'art sur les méthodes de reconstruction 3D de zones urbaines dans un cadre plus général, et donnons une vue globale de notre méthode ainsi que des données utilisées.
La partie II traite du recalage entre les données SIG et vidéo, en détaillant plus particulièrement la phase d'initialisation du recalage pour la première image puis celle de

suivi de ce recalage au cours du temps.
Enfin, la partie III détaille l'exploitation de ce recalage en vue d'extraire de la vidéo les différentes textures des façades visibles, puis de repérer et d'estimer tous les détails de la micro-structure des bâtiments qui ne sont pas modélisés dans la base SIG.
Pour chaque étape, des résultats sont présentés pour des séquences de test.

Première partie

Reconstruction de zones urbaines

Chapitre 1

Reconstruction de zones urbaines

We've heard that a million monkeys at a million keyboards could produce the complete works of Shakespeare ; now, thanks to the Internet, we know that it is not true - **Robert Wilensky**

1.1 Introduction

Depuis quelques années, la reconstruction 3D de zones urbaines est un domaine de plus en plus exploré. Son attrait est dû aux possibilités d'exploitation des modèles 3D résultants, en particulier pour des applications de tourisme virtuel, de réalité augmentée ou de planification architecturale. En fonction de l'application visée, divers niveaux de qualité de la reconstruction peuvent être recherchés. Les méthodes de reconstruction et les données utilisées en entrée peuvent donc être très différentes selon que l'on recherche à obtenir un ensemble de modèles simples mais couvrant une très large zone géographique, ou que l'on souhaite disposer de modèles de bâtiments plus spécifiques mais beaucoup plus précis.
Nous présentons dans ce chapitre un bref survol des méthodes de reconstruction 3D de bâtiments. Le type de reconstruction visée (à grande ou à petite échelle) déterminant généralement quel type de données sont utilisées en entrée, nous classifions les travaux existants en fonction de celles-ci :

1. Reconstruction à partir de données aériennes
2. Reconstruction à partir de données acquises au sol
3. Reconstruction à partir de données hybrides, combinant données aériennes et données au sol

1.2 Reconstruction avec données aériennes

Les données qualifiées d'*aériennes* peuvent être de nature très diverse : photographies acquises depuis un avion ou un satellite, images radar, LIDAR, laser, etc. Ces données pouvant couvrir une large zone géographique, elles ont l'avantage de permettre une reconstruction 3D à grande échelle et à moindre coût. Cependant, ce point de vue plus

macroscopique des zones urbaines limite les possibilités de mesure des textures des façades, et ne permet pas l'estimation de leur micro-structure.
Ces approches comprennent généralement deux étapes distinctes :

1. *Détection des bâtiments :* une première phase consiste à segmenter les images fournies en entrée pour séparer les bâtiments du reste de l'environnement. On obtient à l'issue de cette étape leur zone d'occupation au sol (ou leur empreinte au sol), qui peut être définie par une liste fermée de points ou un ensemble de segments.
2. *Calcul de la hauteur des bâtiments :* l'exploitation de plusieurs images d'une même zone, acquises depuis des points de vue différents, permet par stéréovision de déterminer la hauteur des bâtiments, et donc par extrusion de l'empreinte calculée précédemment de donner *in fine* un modèle 3D simple et non texturé de la zone urbaine visualisée.

1.2.1 Détection de bâtiments

[EBM02] présente un algorithme de type *split & merge*. Il se décompose en trois étapes : l'image est découpée récursivement en régions plus petites jusqu'à ce qu'un critère d'homogénéité soit satisfait. Puis les régions adjacentes sont regroupées en *super-régions* selon leur similarité avec leur voisinage. Enfin les plus petites régions restantes sont soit éliminées soit fusionnées avec d'autres régions plus grandes. [HMS02] propose une idée un peu similaire, mais en considérant le problème inverse : les régions sont créées en regroupant les pixels petit à petit selon un critère similaire d'homogénéité. L'originalité vient ici de l'introduction de trois paramètres (échelle, couleur et forme) pour guider la segmentation. Une telle segmentation des images nécessite alors une phase de classification pour distinguer les bâtiments de leur environnement. Dans [EBM02], cela est fait en attribuant à chaque région deux attributs. Le premier mesure la linéarité des frontières de régions en extrayant les lignes correspondantes avec une transformation de Hough, puis en mesurant la déviation des points effectifs par rapport aux lignes estimées. Le deuxième utilise la projection d'un DEM (Digital Elevation Model) sur l'image pour déterminer la hauteur moyenne de la région. Enfin un réseau de neurones à deux couches permet d'utiliser ces attributs pour classifier les régions en "bâtiments" ou "environnement".

Dans [SV00][SV02], les informations d'empreinte au sol contenues dans une base SIG sont utilisées. Elles sont projetées dans des photos aériennes obliques dont la pose est connue, pour délimiter les bâtiments dans l'image. A partir de hauteurs hypothétiques minimale et maximale de ceux-ci, leur contour est recherché localement par mise en correspondance de points et de lignes entre deux images de la même zone. On connaît alors la projection complète du bâtiment, et non uniquement son empreinte au sol.

Une telle détection peut également se faire en extrayant les lignes principales dans les images. Dans [TR03], deux types d'images sont utilisées pour une approche en deux temps. Tout d'abord, les lignes principales sont extraites d'une image radar. Ensuite,

les extrémités de ces lignes sont conservées puis des segments en sont déduits et projetés sur une image optique classique. Deux types de raffinement sont alors possibles, le premier étant rapide mais ne détectant que les bâtiments de forme rectangulaire, et le second plus lent mais permettant d'extraire des bâtiments aux formes plus complexes. Dans le premier cas, on recherche un rectangle en extrayant des lignes avec un détecteur de Canny-Deriche et en essayant d'en trouver une parallèle et deux orthogonales au segment initial. Dans le deuxième cas, un bâtiment est considéré comme un ensemble de coins joints les uns aux autres. Pour chaque segment extrait de l'image radar, deux coins hypothétiques sont sélectionnés (les extrémités), et un coin est finalement choisi s'il est à l'intersection de deux segments extraits (pas forcément orthogonaux). Enfin la forme du bâtiment est extraite en recherchant un chemin fermé entre les différents coins et suivant les lignes principales de l'image.

1.2.2 Calcul de la hauteur des bâtiments

A partir des régions calculées sur les images, on peut extraire la géométrie des bâtiments. Dans [EBM02] les polygones correspondant aux toits sont déduits des images segmentées, en utilisant les intersections entre les lignes de la région. Le modèle 3D final est calculé par stéréovision en mettant en correspondance ces polygones, puis des points au sein de ces polygones, entre les différentes images en entrée.

Un autre type d'approche consiste à utiliser une méthode de mise en correspondance entre les données extraites de l'image et des modèles synthétiques, soit simples soit utilisant une représentation d'arbre CSG (*Constructive Solid Geometry*) composée de plusieurs modèles simples afin de gérer les bâtiments aux formes complexes [SV02]. Pour cela le modèle synthétique est projeté dans la zone d'intérêt de l'image, et ses paramètres optimaux sont recherchés en maximisant une mesure de vraisemblance qui compare généralement les frontières projetées du modèle avec les lignes extractibles de l'image.

1.3 Reconstruction avec des données au sol

Un autre type d'approche consiste à synthétiser les modèles 3D souhaités à partir de vues *au sol*, c'est-à-dire acquises depuis un point de vue à l'intérieur de la zone. On a ainsi accès aux informations photométriques (textures des bâtiments, etc.) et géométriques (position et forme des portes, fenêtres, etc.) de façon précise, mais cela reste une approche très localisée. Reconstruire entièrement une ville de cette façon requiert énormément de travail, que ce soit du point de vue acquisition comme du point de vue traitement des données.
On peut distinguer deux manières d'appréhender ce problème :

- *Reconstruction sans à priori sur le contenu :* seules les images utilisées en entrée sont utilisées pour estimer la géométrie des bâtiments visualisés.
- *Reconstruction basée modèles :* des modèles 3D simples de bâtiments, de façades,

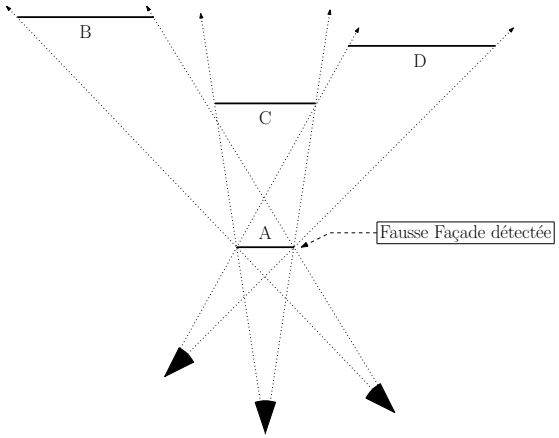


FIG. 1.1 – Apparition de fausses façades

ou de micro-structure sont mis en correspondance avec les images pour aider à la reconstruction finale.

1.3.1 Reconstruction sans à priori sur le contenu

Le projet *City Scanning*, conduit par Seth Teller et le premier projet ayant donné lieu à une reconstruction à grande échelle. On peut en trouver les grandes lignes dans [Tel97, TC98, Tel98, TC99]. Les données acquises consistent en un ensemble de *nœuds*, contenant comme informations la pose et le calibrage de la caméra, ainsi qu'un ensemble de prises de vues regroupées en une mosaïque hémisphérique. Les plans principaux correspondant aux façades sont calculés à partir de l'extraction d'un ensemble de lignes non verticales extraites des images et de la pose estimée de la caméra. Il se peut que l'on obtienne à ce stade de fausses façades (voir figure 1.1). Pour les éliminer, l'algorithme proposé consiste à ordonner toutes les façades, puis à les intégrer au modèle final dans cet ordre, les lignes ayant donné naissance à ces façades étant retirées des façades suivantes. Elles sont classées par ordre décroissant de longueur totale des segments leur ayant donné naissance. Enfin, la dernière étape de l'algorithme consiste à calculer la texture et la micro-structure de chaque façade (voir chapitre 7).

Dans [Tsa02], Tsay présente brièvement une formulation originale de la reconstruction. Au contraire d'approches basées modèles classiques, où l'on suppose que l'on peut reconstruire un bâtiment à l'aide de polyèdres simples, ici la modélisation de surface est basée ondelettes et ajustement aux moindres carrés contraint. La raison pour laquelle les ondelettes sont employées est que les surfaces de villes sont plus complexes que de simples polyèdres (exemple cité : temples à Taïwan) et contiennent autre chose que des bâtiments simples (jardins...).

1.3.2 Reconstruction basée modèles

Dans [DTM96a, DTM96b], support du logiciel *Façade*, Debevec présente une méthode de reconstruction de scènes architecturales quelconques. L'idée est de fusionner deux approches : la première est la reconstruction basée géométrie, c'est-à-dire type CAO, où l'utilisateur modélise les scènes voulues ; la seconde est basée images, c'est le type d'approche classique de reconstruction par stéréovision. En entrée on a un ensemble d'images assez éparse de la scène à reconstruire. L'utilisateur intervient à la main pour positionner des primitives géométriques volumétriques sur chaque image, qui doivent correspondre à une approximation du modèle à générer. Il peut également spécifier des contraintes liant ces primitives (parallélisme, positionnement relatif, etc.). Enfin il définit des segments dans les images et leurs correspondants dans la base de primitives. Puis l'algorithme minimise une fonction qui somme la disparité entre les bords du modèle projeté et les bords marqués. Les positions des différentes caméras ainsi que les données inconnues du modèle sont ainsi déduites. On a à cet instant un modèle brut ainsi que les positions de caméra par rapport à ce modèle.
Les images de texture de chaque caméra ne couvrant qu'une partie de la scène, si un rendu à une position virtuelle est effectué, la texture finale est une image composée des différentes images des caméras initiales projetées sur le modèle. La composition correspond à une pondération dépendant de l'angle entre les axes de visée de la caméra virtuelle et des caméras physiques alentours.

Dans [SB03], la reconstruction est faite de façon incrémentale : on construit un nuage de points de la scène à partir de plusieurs images, dont on déduit les plans principaux et donc un modèle brut du bâtiment, puis on recherche sur chaque plan les zones correspondant aux portes et aux fenêtre grâce à une approche basée modèle. Pour cela, on balaye chaque plan sur ses deux axes pour détecter les régions d'intérêt en observant les zones de variation de gradient dans l'image correspondante. Ces zones sont regroupées en zones rectangulaires de façon robuste à l'aide d'un algorithme de type Ransac pour détecter les portes et les fenêtres. A chaque zone détectée dans l'image est associé un modèle 3d prédéfini. Dans un dernier temps, le modèle 3d brut est raffiné en utilisant le nuage de points estimé pour déterminer les profondeurs des zones de micro-structure détectées.
Dans [WZ02a], une détection des plans principaux est tout d'abord effectuée en utilisant une estimation de la position des points de fuite et un nuage de points mesuré à partir des images. Dans un second temps, les contours des différentes zones d'intérêt sont détectés en fonction des variations du gradient de luminosité dans l'image. Enfin, le modèle 3d brut composé des plans principaux est raffiné en utilisant le nuage de points. Dans [WZ02b] le raffinement est fait en appariant les points du nuage en dehors des plans principaux avec des modèles 3d de portes, fenêtre, etc. prédéfinis.

Dans [TDC00], Torr *et al.* présentent une représentation de la scène par couches. Ils supposent que la scène est principalement planaire, et que les micro-structures qui peuvent apparaître sont d'autres couches superposées. Chaque couche $\mathcal{L}$ possède des paramètres

de positionnement, taille, orientation, texture, etc. Une couche particulière, $\mathcal{L}_0$, représente le plan principal de la scène. A chaque couche est associé un modèle $\mathcal{M}$, parmi une banque de modèle paramétriques préexistants (fenêtre rectangulaire, arche, etc.). De plus, des *hyperparamètres* permettent d'utiliser certaines particularités des scènes architecturales. Par exemple, un ensemble de couches $\mathcal{L}_i$ aura un hyperparamètre y modélisant le fait qu'elles soient alignées horizontalement. Les couches sont initialement détectées en estimant un nuage de points 3D de la scène.

1.4 Reconstruction avec des données hybrides

En vue d'obtenir des modèles 3D à grande échelle mais localement plus précis, des approches hybrides de reconstruction ont été proposées. Elles combinent à la fois des données acquises au sol et des données aériennes.

Une fusion d'images acquises au sol avec des images aériennes et des données LIDAR est présentée dans [HYN06]. L'empreinte au sol des bâtiments est déterminée de façon interactive dans l'image aérienne. Les différents pans de toits sont également définis à la main lors de cette étape. Ces empreintes sont ensuite mises en correspondance avec des données LIDAR pour reconstruire le modèle 3D des bâtiments. Des images au sol sont alors utilisées pour texturer les bâtiments. La pose de la caméra les ayant acquises est calculée en utilisant les points de fuite pour estimer l'orientation, et des correspondances entre des points 2D et 3D pour le calcul de la translation. Cette méthode ne permet cependant pas d'obtenir une géométrie précise des façades considérées.

Dans [FZ03a, FZ03b, FZ04], les zones urbaines sont modélisées en utilisant comme données acquises au sol deux lasers et une caméra. L'approche se décompose comme suit : acquisition des données brutes, estimation de pose, texturage automatique des façades, puis fusion des données au sol avec un DTM (*Digital Terrain Model*) pour la modélisation des toit.
Pour l'acquisition, deux lasers et une caméra sont synchronisés matériellement. Le laser vertical mesure un nuage de points 3D correspondant aux façades, et l'horizontal permet une estimation de pose des données acquises. La caméra sert à texturer le modèle 3D acquis.
Le calcul de pose permettant de recaler les données entre elles s'effectue en deux temps. Une première étape consiste à mettre en correspondance les mesures du laser horizontal pour obtenir le déplacement point à point du véhicule. Cette estimation étant sensible à la dérive, elle est recalée dans un deuxième temps par mise en correspondance des scans horizontaux avec une carte de contours des bâtiments issue d'un DSM (*Digital Surface Map*, provenant d'images aériennes laser), en utilisant un algorithme de filtrage particulaire (MCL, *Monte Carlo Localization*).
Enfin, la pose étant déterminée précisément, le nuage de point mesuré au sol par laser est recalé puis fusionné avec le modèle DTM pour modéliser les toits des bâtiments en plus des façades.

1.5 Conclusion

La reconstruction 3D de zones urbaines se base soit sur des données aériennes pour une synthèse de modèles peu précis mais à grande échelle, soit sur des données acquises au sol permettant une modélisation beaucoup plus fine, mais souvent trop coûteuse pour être appliquée à des villes entières. Les deux approches sont complémentaires, et dans les travaux combinant images aériennes et images au sol, la complexité est reportée sur le processus d'acquisition lui-même (capteurs laser synchronisés avec une caméra vidéo par exemple).
Il serait souhaitable de pouvoir combiner les deux types de données sans pour autant se contraindre à disposer d'un mécanisme d'acquisition trop lourd. C'est à ce problème que l'on essaye de répondre dans ces travaux de thèse. Nous présentons donc maintenant les données que nous avons utilisées en entrée ainsi qu'un aperçu global de la méthodologie développée.

Chapitre 2

Données utilisées

It is a capital mistake to theorize before one has data. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts - **Sir Arthur Conan Doyle**

2.1 Introduction

De nombreux types de données différents peuvent être utilisés afin de reconstruire des zones urbaines en trois dimentions. Photographies au sol, images aériennes, images radar ou mesures laser en sont les exemples les plus courants. L'imagerie aérienne permettant d'un côté une reconstruction à grande échelle mais peu précise, l'imagerie au sol quant à elle autorise une reconstruction plus détaillée mais qui reste généralement plus locale à cause des dérives dans l'estimation de la scène, brisant ainsi la cohérence globale nécessaire à une reconstruction à grande échelle.

Dans notre approche du problème, nous essayons de combiner les deux méthodes utilisant ces deux types d'images pour n'en garder que les avantages. De la reconstruction aérienne, nous conservons une reconstruction déjà effectuée de bâtiments, qui est géoréférencée. Elle est fournie par l'IGN (Institut Géographique National). Elle consiste en un ensemble de modèles polyédriques simples non texturés, mais globalement cohérents entre eux. En utilisant des images acquises au sol, nous souhaitons améliorer la géométrie de ces modèles et les texturer, sous la contrainte de conserver leur géométrie globale. Nous devons donc trouver une relation entre deux types de données. Pour cela, nous utilisons des mesures GPS, effectuées conjointement avec l'acquisition des images au sol.

Nous présentons dans ce chapitre les spécificités des trois types de données utilisées en entrée de notre méthode, à savoir le SIG, les vidéos et le GPS, ces spécificités étant une base nécessaire à la compréhension de la suite de l'exposé.

2.2 Sig

Des modèles 3D d'environnement urbain à grande échelle existent déjà. C'est sur de tels modèles que notre approche est basée. Nous détaillons donc ici leurs spécificités et limitations, ainsi que leur contenant : le SIG.

2.2.1 Définition

Un SIG, ou *Système d'Information Géographique*, est un système permettant de gérer et visualiser des données géo-référencées. Son rôle est de proposer une représentation d'un environnement géographique, à l'aide de primitives simples telles que des points, des vecteurs, des polygones ou des maillages.
A ces représentations diverses sont en général associées des informations contextuelles sur leur nature : routes, bâtiments, hydrographie, etc., informations qui sont regroupées par *couches*.
On peut voir par exemple sur la figure 2.1 différentes couches d'un quartier décrivant le réseau routier (2.1(a)), le réseau hydrographique (2.1(b)), l'emplacement des bâtiments (2.1(c)) ou une photo aérienne de la zone (2.1(d)).

2.2.2 Description de la couche utilisée

Nous avons vu qu'un SIG pouvait être constitué de différentes couches décrivant des données géo-référencées de natures diverses. Dans notre cas, nous nous intéressons à une couche particulière, qui contient une description de bâtiments. Les données que l'on peut en extraire consistent, pour chaque bâtiment, en une liste fermée de points correspondant à son empreinte au sol, ainsi qu'en son altitude (par rapport au niveau de la mer) et sa hauteur.
Les positions géo-référencées des points sont exprimées dans le repère métrique UTM (*Universal Transverse Mercator*), où les coordonnées $(X, Y)_{utm}$ correspondent au plan du sol et la coordonnée Z_{utm} indique une altitude. Un modèle simple de bâtiment est donc complètement décrit par les données suivantes (voir également figure 2.2) :

- Une liste de points 2D dont les coordonnées sont exprimées dans le plan $(X, Y)_{utm}$ pour l'empreinte au sol
- Une altitude a_{utm} correspondant à l'altitude de cette empreinte
- Une hauteur h_{utm} depuis la base jusqu'au sommet du bâtiment

Une telle représentation permet d'obtenir une modélisation géométrique globalement cohérente et à grande échelle d'une zone urbaine.

On peut toutefois mettre en lumière plusieurs limitations à une telle représentation si on l'exploite en tant qu'ensemble de modèles 3D de bâtiments.

1. **Géométrie approximative.** Les façades des bâtiments sont définies à partir d'une extrusion de leur empreinte, de hauteur h_{utm}. Elles sont donc parfaitement planes, et ne décrivent aucunement les éventuels détails géométriques tels que les portes ou les fenêtres. De plus, les bâtiments présentant des courbes sur leur pourtour

(a) Réseau routier (b) Réseau hydrographique

(c) Bâtiments (d) Photo aérienne

FIG. 2.1 – Différentes couches d'un SIG, pour un même quartier

sont également mal modélisées en général, faute d'un nombre de points suffisant pour décrire la finesse d'une telle empreinte.

2. **Erreurs de topologie.** Une telle représentation ne permet pas non plus de modéliser les modèles 3D "troués" qu'impliquent des arches, porches ou préaux.
3. **Modélisation du sol.** L'empreinte étant définie dans le plan $(X, Y)_{utm}$, les bâtiments construits sur un sol non globalement plan seront également mal modélisés.
4. **Pas d'information de texture.** L'une des plus grosses lacunes de cette représentation est qu'elle ne définit aucune information photométrique de la scène : les textures des façades sont inconnues.

On peut voir par exemple sur la figure 2.3 l'affichage via une interface OpenGL de modèles de bâtiments lus dans une telle base SIG (les toits et l'altitude des bâtiments ne sont pas modélisés ici). La pauvreté esthétique des polyèdres gris qui les représentent est manifeste.

C'est pour pallier ces lacunes que nous introduisons la vidéo (ou séquence d'images)

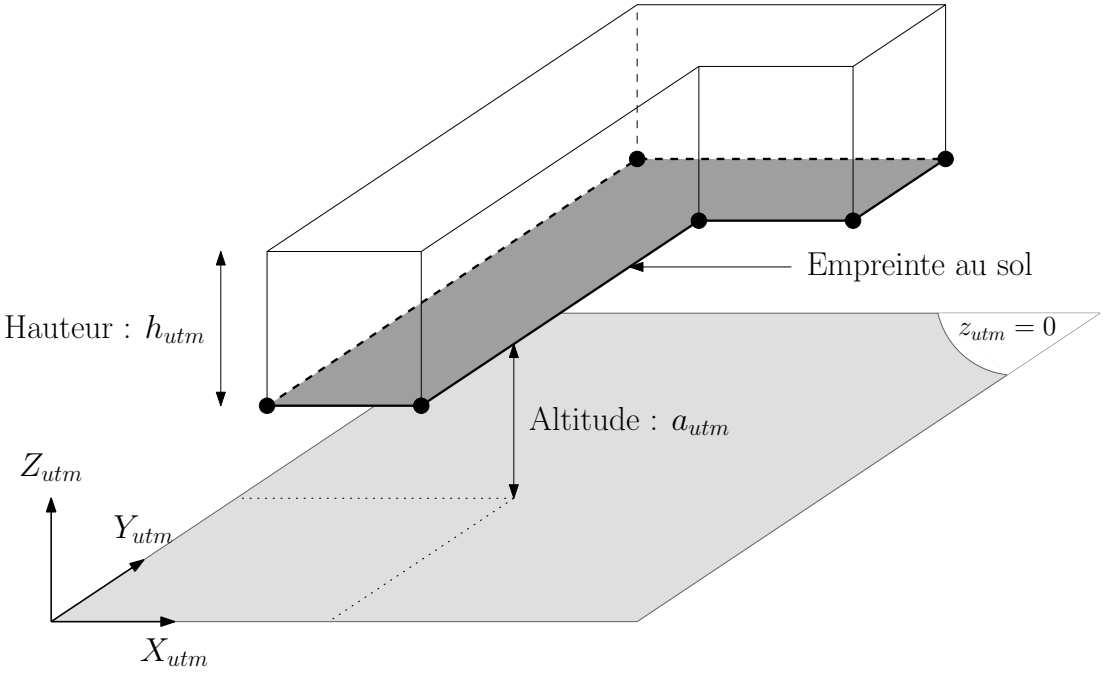


FIG. 2.2 – Paramètres extractibles du SIG pour décrire un bâtiment

dans notre boucle de modélisation. En recalant les images des bâtiments ainsi visualisés avec la projection de leur modèle 3D correspondant, notre but est de pouvoir extraire les textures des façades tout en calculant les éventuelles micro-structures— à l'aide de méthodes de reconstruction basées images — et ainsi améliorer, tant géométriquement que photogramétriquement, les modèles 3D simples extraits du SIG.

2.3 Vidéos

Des informations géométriques et photométriques sont ajoutées aux bâtiments de la base SIG en utilisant un ensemble d'images de ces mêmes bâtiments. Les images sont acquises au niveau du sol, en utilisant une caméra vidéo numérique du commerce.
Dans l'idéal, notre méthode doit permettre de recaler les modèles SIG avec la vidéo dans des conditions d'acquisition critiques, comme par exemple à main levée avec un PDA pour des applications de réalité augmentée sur terminaux mobiles. Pour cette raison, nous posons comme hypothèse forte que la caméra n'est pas calibrée de façon très précise (comme elle pourrait l'être en utilisant une mire de calibration par exemple). On suppose que seules les données constructeur sont disponibles pour déterminer les paramètres internes de la caméra.

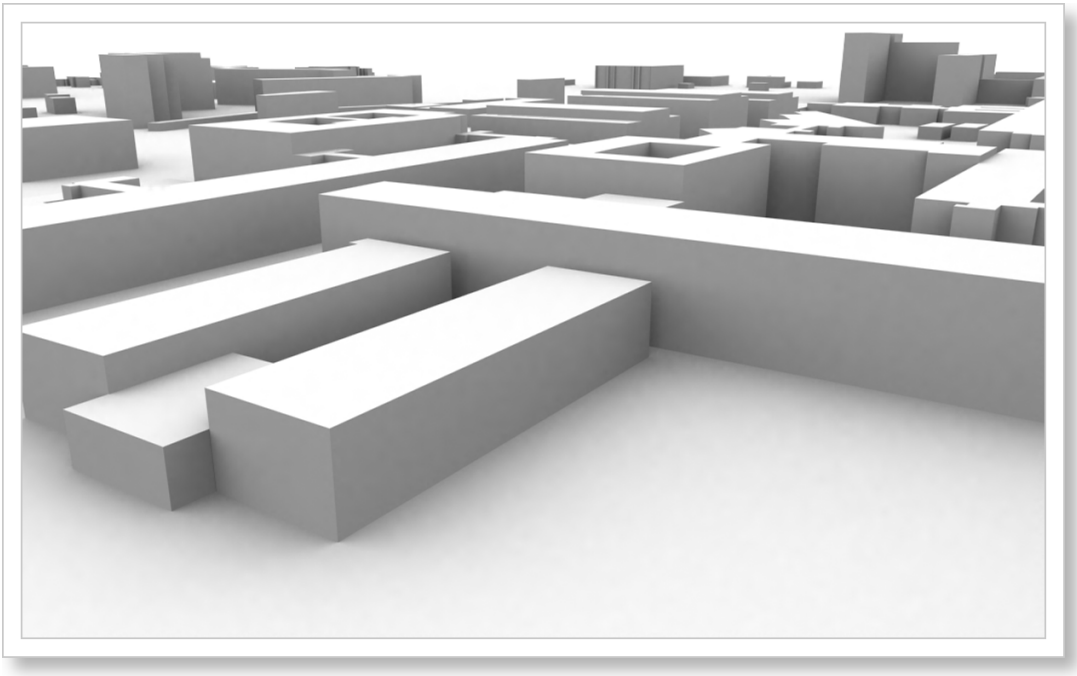


FIG. 2.3 – Exemple d'exploitation de la base SIG : rendu graphique des modèles polyédriques simples des bâtiments

2.4 Gps

Dans l'optique de mettre en relation images et modèles 3D, nous voulons ajouter à ces images une information de géo-localisation pour qu'elles soient décrites dans le même système de coordonnées que les modèles. Nous effectuons donc des mesures de positionnement par GPS conjointement aux prises de vue. Cette section est donc dédiée à la description des propriétés et limitations des données GPS, afin de les exploiter de façon pertinente dans notre approche.

2.4.1 Définition

Le GPS (*Global Positioning System*) est un système de navigation par satellites militaires, dont l'accès est permis aux civils. Ce réseau a été développé par le *Department of Defense* des États-Unis et est la propriété du *Department of Transports*. Il est composé de 24 satellites répartis sur 6 orbites circulaires, inclinées de 55° par rapport au plan équatorial, à 20200 km et d'une durée de 24 heures, comme on peut le voir sur la figure 2.4. Il y a donc 4 satellites par orbite, et 21 sont réellement utilisables (3 sont des satellites de secours). Cette disposition permet de capter en permanence entre 5 et 8 satellites en tout point du globe. Enfin, ils émettent en permanence sur deux fréquences porteuses différentes (l'une à 1575.42 MHz et l'autre à 1227.6 MHz, nous verrons tout à l'heure pourquoi) des signaux contenant :

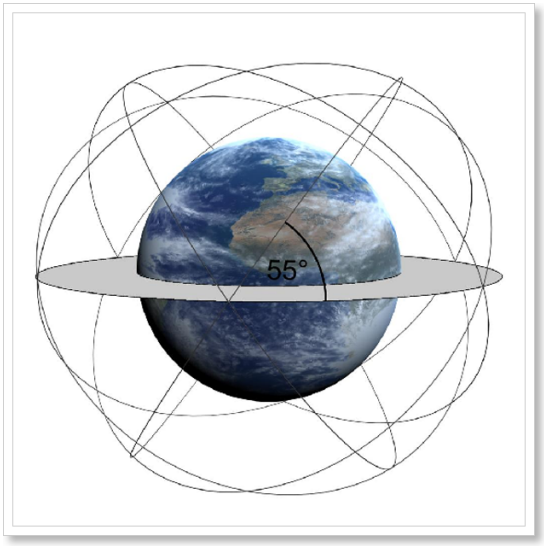


FIG. 2.4 – Illustration des 6 orbites différentes de la constellation GPS

- une information de l'état de santé du satellite
- les informations nécessaires à l'acquisition du code du message
- les informations de précision du satellite
- les éphémérides du satellite

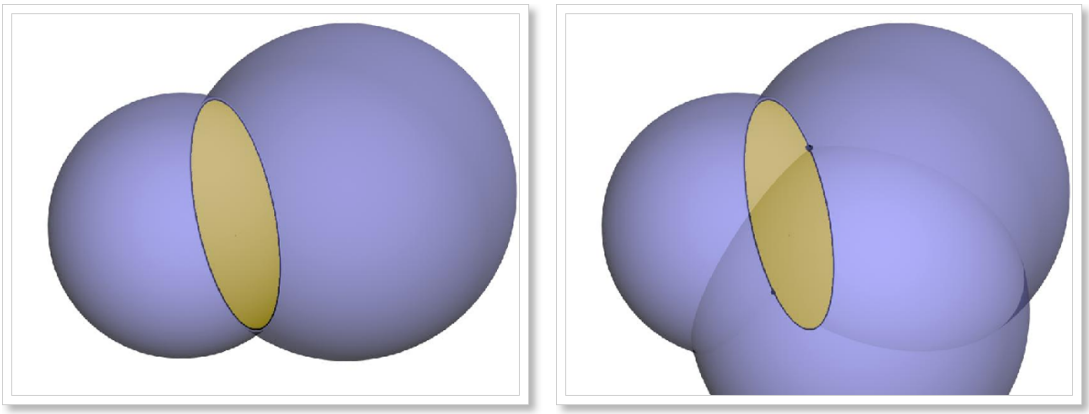
A ce réseau satellitaire s'ajoutent cinq stations au sol (Îles Hawaï et Marshall (Pacifique), Ascension (Atlantique), Diego Garcia (Océan Indien) et Colorado Springs (USA)), dont le rôle est de piloter la constellation de satellites. Elles enregistrent les signaux émis 24 heures sur 24, calculent les positions orbitales (éphémérides), recalent les horloges atomiques et les corrigent si nécessaire. Ce sont donc les seules installations capables d'envoyer des informations aux satellites.

2.4.2 Méthode de positionnement

2.4.2.1 Calcul des distances

Nous cherchons ici à estimer la distance d entre le satellite S et la position courante P. Le récepteur captant le signal de S, il connaît la position de celui-ci. Il a donc besoin de connaître le temps Δt du trajet de l'onde pour connaître précisément la distance qui l'en sépare. Or chaque satellite a un code propre pour le signal qu'il envoie, donc si le récepteur sait avec quel satellite il communique il ne lui reste plus qu'à synthétiser ce même code et à mesurer le déphasage entre signal reçu et signal synthétisé pour avoir Δt. On a alors $d = c.\Delta t$, c étant la vitesse supposée de propagation de l'onde, que l'on assimile à la vitesse de la lumière dans le vide (soit environ 300000 km/s).

Avec une orbite à 20200 km, le temps de propagation de l'onde du satellite au récepteur est en théorie compris entre 67 et 86 ms, selon que le satellite se trouve plus ou moins à



(a) L'intersection de deux sphères est un cercle

(b) L'intersection de trois sphères est un ensemble de deux points

FIG. 2.5 – Le calcul d'intersection entre des sphères permet de calculer la position du récepteur par rapport aux satellites environnants

l'horizon ou au zénith de P. D'où l'utilité d'avoir des horloges précises pour mesurer le déphasage et donc la distance d. A titre d'exemple, on peut estimer qu'une milliseconde d'erreur sur l'estimation de Δt implique une erreur d'estimation de 300 km sur la distance PS.

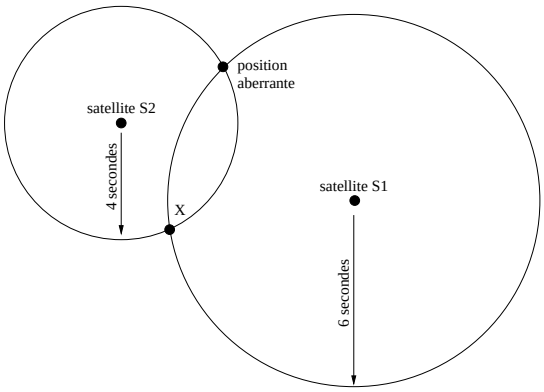
2.4.2.2 Calcul de la position du récepteur

Pour connaître la position du récepteur dans un système de coordonnées cartésiennes centrées sur le centre terrestre, il faut connaître la distance à trois satellites différents pour résoudre le système suivant :

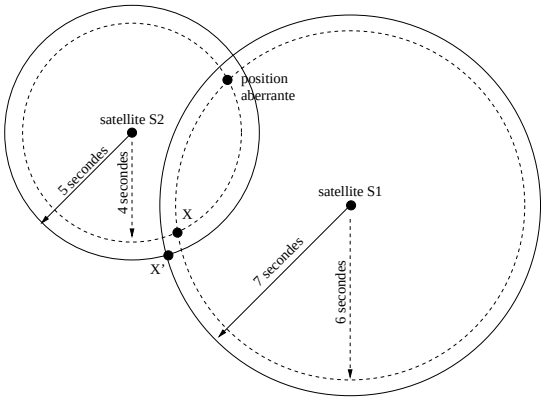
$$\{_i (x-a_i)^2 + (y-b_i)^2 + (z-c_i)^2 = {d_i}^2, i \in \{1,2,3\} \tag{2.1}$$

où (a_i, b_i, c_i) sont les coordonnées de la position du satellites S_i et d_i sa distance au récepteur en P, dont les coordonnées sont (x, y, z). Ce système se justifie par le fait que l'on sait que P se situe sur une sphère de rayon d_i par rapport au satellite S_i. Si l'on prend les deux premiers satellites, connaissant leur distance au récepteur on peut en déduire que celui-ci se situe à l'intersection de deux sphères, soit un cercle (voir figure 2.5(a)). Avec un troisième satellite, l'intersection se résume alors à deux points (voir figure 2.5(b)). Sachant évidemment que le récepteur se situe quelque part sur la surface terrestre, un des deux points est aberrant et la solution est trouvée.

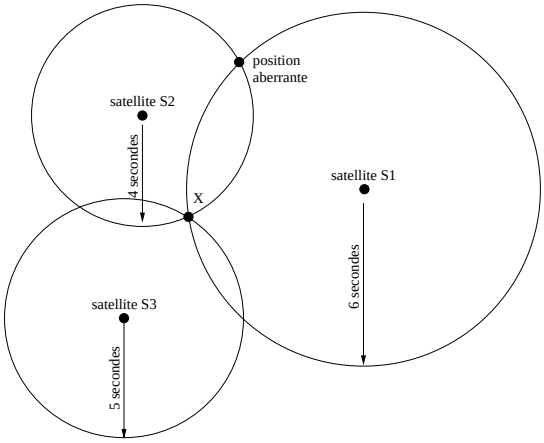
Une correction d'erreur s'avère nécessaire pour l'estimation des distances car il peut y avoir eu une défaillance mineure de l'horloge lors de la synchronisation des signaux. On peut trouver sur la figure 2.6 une schématisation du principe de correction en le simplifiant au cas 2D, avec 3 satellites. On voit alors pourquoi l'estimation de la position en



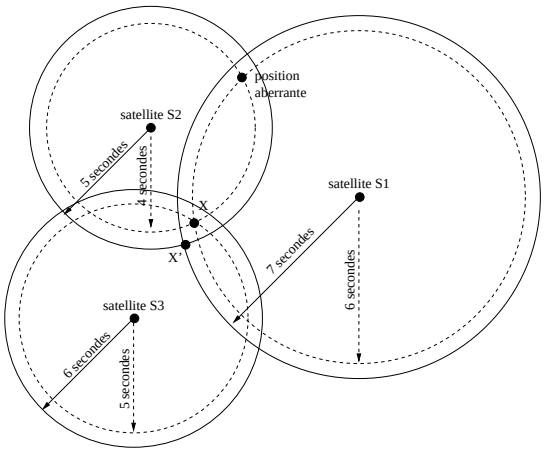
(a) En 2D, sans erreur de la part du récepteur, les 2 satellites donnent la position du récepteur



(b) Une erreur d'une seconde est faite sur l'estimation du temps de propagation. La position du récepteur est donc estimée en X' à la place de X



(c) L'utilisation d'un troisième satellite, sans erreur, se contente de confirmer l'estimation faite par les deux autres



(d) Le troisième satellite permet de définir une zone dans laquelle est située la position véritable du récepteur. L'erreur est alors estimée en essayant de faire se recouper en un point les 3 cercles, et la position finale est alors trouvée

FIG. 2.6 – L'intérêt de recourir à un satellite en plus pour l'estimation de la position

3D nécessite une synchronisation avec 4 satellites.

Pour être exploitables, les coordonnées doivent être exprimées en général dans un système basé $(latitude, longitude, altitude)$ plutôt que dans repère centré sur le centre de la Terre. La conversion des coordonnées cartésiennes en coordonnées polaires s'effectue

de la façon suivante :

$$\begin{cases} latitude = \arctan\frac{y}{x} & \text{elle est dite positive si } y \geq 0 \\ & \text{elle est dite négative si } y < 0 \\ longitude = \arctan\frac{z}{x} & \text{elle est dite positive si } z \geq 0 \\ & \text{elle est dite négative si } z < 0 \\ altitude = \sqrt{x^2+y^2+z^2} - 6380 & , \end{cases} \quad (2.2)$$

où 6380 est le rayon en kilomètres de la Terre au niveau de la mer.

2.4.3 Précision théorique et sources d'erreurs

Nous avons vu que l'ajout d'une ou plusieurs mesures satellites permettait de calculer avec plus de précision la position estimée. Voyons alors quelles sont ces sources d'erreur, et s'il existe des moyens pour les contourner plus directement.

2.4.3.1 SA (*Selective Availability*, Disponibilité Sélective)

Cette méthode de dégradation volontaire du signal a été mise en œuvre jusqu'au 1er Mai 2000 pour les civils, en modifiant de façon aléatoire l'horloge atomique des satellites. Elle limitait en principe la précision à 100 mètres dans 95% des cas, et à 300 mètres au pire. L'abandon du SA permet alors en théorie d'offrir aux civils une précision de 5 mètres dans 95% des cas.

2.4.3.2 Perturbations atmosphériques

Une autre source d'erreur est liée au trajet emprunté par le signal. En effet, il doit traverser différentes couches atmosphériques, dont l'ionosphère et la troposphère, qui sont des "nuages" de particules chargées électriquement (ions et électrons). Comme toute onde se propage moins vite dans un milieu conducteur, on a un délai supplémentaire à estimer pour le temps de transmission du signal. Il est toutefois possible de contourner le problème en émettant sur deux fréquences différentes. Ainsi en faisant une combinaison linéaire des celles-ci, on peut éliminer le facteur dispersif induisant le retard de mesure.

2.4.3.3 Trajets multiples

Enfin, on peut considérer que le signal accuse un retard à cause de multiples réflexions avant d'arriver au récepteur (contre un montagne, des bâtiments, etc.). Ce cas est rencontré très fréquemment dans des environnements urbains.

2.4.3.4 Dilution géométrique de la précision

La dilution géométrique de la précision (en anglais Dop : *Dilution of Precision*, ou Gdop : *Geometric Dilution of Precision*) est une mesure permettant d'évaluer l'influence de la configuration géométrique des satellites sur la précision d'une mesure Gps. Quand les satellites visibles sont proches les uns des autres, la géométrie est dite *faible*,

Dop	Qualité	Description
1	Idéal	Le plus haut niveau possible de confiance
2-3	Excellent	A ce niveau, la confiance dans la mesure est suffisamment importante pour toute application non sensible
4-6	Bon	Le niveau demandé pour faire de la navigation routière
7-8	Acceptable	Les mesures de position restent utilisables, mais la qualité peut être améliorée
9-20	Mauvais	Les mesures de positions ne devraient pas être prises en compte, sauf pour donner une approximation très grossière de la position
21-50	Très mauvais	L'imprécision est trop grande pour que les mesures soient utilisables

TAB. 2.1 – Interprétation qualitative des différentes valeurs possibles de DOP

et la valeur de DOP est importante. Réciproquement, s'ils sont éloignés, la géométrie est *forte* et la valeur de DOP est faible. Une faible valeur de DOP est donc associée à une meilleure précision de la mesure de positionnement, du fait d'un angle de séparation plus important entre les satellites utilisés pour calculer la position. Cette mesure est particulièrement utile en milieu urbain, certaines parties du ciel étant occultées par des bâtiments. Tous les satellites potentiellement visibles ne peuvent alors pas forcément être utilisés, et la valeur de DOP sera plus importante.
Cette mesure peut être décomposée en HDOP, VDOP, PDOP et TDOP, respectivement pour *Horizontal*, *Vertical*, *Position* (3D) et *Time Dilution of Precision*. Une interprétation des valeurs de DOP est donnée dans le tableau 2.1[i].

2.4.4 Systèmes de correction d'erreurs

Afin de corriger les différentes erreurs menant à un positionnement approximatif, plusieurs systèmes ont été développés. La plupart se basent sur des stations aux sol connaissant précisément leur propre position, qui mesurent les erreurs du signal GPS puis les transmettent aux récepteurs afin qu'ils corrigent leur position estimée.

2.4.4.1 GPS Différentiel

Le GPS Différentiel (ou DGPS pour *Differential* GPS) est un système d'amélioration des mesures GPS, qui utilise un réseau de stations de référence au sol. Historiquement, le DGPS était utilisé principalement pour la navigation maritime précise, ce pourquoi les stations sont situées en grande majorité sur des zones côtières. Ces stations connaissent avec précision leur propre position, et captent en permanence les signaux GPS des satellites visibles.
L'hypothèse fondatrice de ce système est qu'en des points voisins géographiquement et temporellement, les erreurs de mesure sont très semblables. Si P^S est la position réelle

[i] sources : [Mac02] et Wikipedia

(et connue) d'une station, et $\widehat{P_t^S}$ la position Gps que cette station mesure au temps t, alors elle peut estimer l'erreur sur le signal Gps en cet instant :

$$\Delta P_t^S = P^S - \widehat{P_t^S} \tag{2.3}$$

L'erreur ΔP_t^S est alors diffusée par ondes radio dans le voisinage de la station. Tout récepteur Gps classique permettant de capter ce signal radio peut alors corriger sa propre mesure de position. Soit P^R la position réelle du récepteur (inconnue), et ΔP_t^R l'erreur (inconnue également) sur la mesure Gps $\widehat{P_t^R}$. Les erreurs étant proches géographiquement et temporellement, on a $\Delta P_t^R = \Delta P_t^S + \epsilon$, avec ϵ très faible. Le récepteur connaissant $\widehat{P_t^R}$ et ΔP_t^S, il peut alors calculer sa position suivante réelle avec une erreur très faible (ϵ) :

$$\widehat{P_t^R} + \Delta P_t^S = \widehat{P_t^R} + \Delta P_t^R - \epsilon = \boxed{P^R - \epsilon} \tag{2.4}$$

Même si ces mesures sont de bonnes qualité, le système possède plusieurs inconvénients :

- Tous les récepteurs Gps ne sont pas équipés pour capter la diffusion radio de correction d'erreur.
- Les stations sont peu nombreuses et réparties en majorité sur les côtes, et donc ne sont pas toujours accessibles.

Ce système de correction d'erreur ne sera donc pas utilisé dans notre cadre d'étude.

2.4.4.2 SBAS - EGNOS

Il existe depuis quelques années de nouveaux systèmes de correction d'erreurs, qui fonctionnent sur un principe similaire au Dgps. On parle de Sbas, pour *Satelite Based Augmentation System*. Il en existe actuellement trois implantations différentes :

- Waas (*Wide Area Augmentation System*) aux États-Unis
- Egnos (*European Geostationary Navigation Overlay System*) en Europe
- Msas (*Multi-functional transport Satelite based Augmentation System*) au Japon

La principale différence avec le Dgps est qu'en plus de se baser sur des stations au sol, le système utilise également des satellites géo-stationnaires pour relayer les informations de correction. Nous nous restreindrons ici à la présentation du système européen Egnos.

Les stations de référence basées au sol (une quarantaine) reçoivent les signaux de tous les satellites Gps en vue, puis calculent les corrections différentielles correspondantes. Ces corrections sont alors transmises aux trois satellites du système, qui eux-mêmes les diffusent sous forme d'un signal similaire au signal Gps. Les récepteurs Gps classiques, s'ils sont configurés pour capter le signal Egnos (c'est le cas pour une très grande majorité des récepteurs récents), réservent un canal à cet effet et conservent les autres canaux pour les signaux Gps classiques. Ils peuvent alors appliquer les corrections différentielles reçues pour avoir une mesure de position plus précise.

Le système Sbas à l'avantage par rapport au Dgps de fournir les corrections différentielles dans des zones où le Dgps n'est pas accessible.

2.5 Aperçu de notre approche

Le principe du recalage entre les diverses données utilisées est illustré sur la figure 2.7. La première étape de notre algorithme, illustré sur cette figure par la boîte labélisée *Initialisation de la pose basée* GPS , consiste à utiliser conjointement les données GPS et SIG pour trouver une première approximation de la position de la caméra d'acquisition par rapport aux bâtiments environnants. On associe alors à chaque image de la vidéo une position et une orientation très approximative de la caméra. Puisque seule une approximation de la pose est recherchée à ce stade, une erreur de quelques mètres sur la mesure de positionnement par GPS n'a que peu d'influence.

L'étape suivante, illustrée sur la figure 2.7 par la boîte intitulée *Recalage images-modèle*, consiste à mettre en correspondance des primitives extraites de images avec les primitives correspondantes dans le modèle 3D pour calculer la pose précise de la caméra pour toutes les images de la vidéo. A ce stade, on peut faire un rendu en images de synthèse des modèles 3D de bâtiments dont on dispose, de telle sorte que la projection des frontières de ce modèle (*i.e.* des segments de droite) se superpose exactement avec les limites des bâtiments dans les images.

La relation entre la projection du modèle 3D et les images de la vidéo étant établie, les textures des façades visibles sont extraites (partie *Extraction de textures*) dans leur propre *espace texture*, qui correspond au plan principal de la façade décrit par le modèle de la base SIG.

La partie *Fusion de textures* utilise l'ensemble des textures extraites de la vidéo d'une même façade pour construire une nouvelle texture unique de celle-ci. L'apport de cette partie tient au fait que des objets peuvent venir masquer tout ou partie de la façade en premier plan dans la vidéo — des véhicules en stationnement, des panneaux de signalisation, des arbres, etc. — et se retrouvent visibles dans les textures extraites à l'étape précédente. Ils sont ici supprimés pour ne garder que l'information photométrique utile de la façade.

Ces textures sont également utilisées pour la dernière partie de notre algorithme : *Calcul de la micro-structure* . Nous présentons à ce point une étude préliminaire sur la faisabilité de l'extraction des structures particulières des façades, comme leurs fenêtres, uniquement à partir des images.

2.6 Conclusion

Nous avons présenté ici les différents types de données utilisés en entrée de notre méthode de reconstruction de zones urbaines.
Malgré leur hétérogénéité — mesures de position par GPS, modèles 3D polyédriques et vidéos — des relations peuvent être établies entre elles. Une des principales difficultés consiste à effectuer une telle mise en correspondance en dépit de leurs imprécisions in-

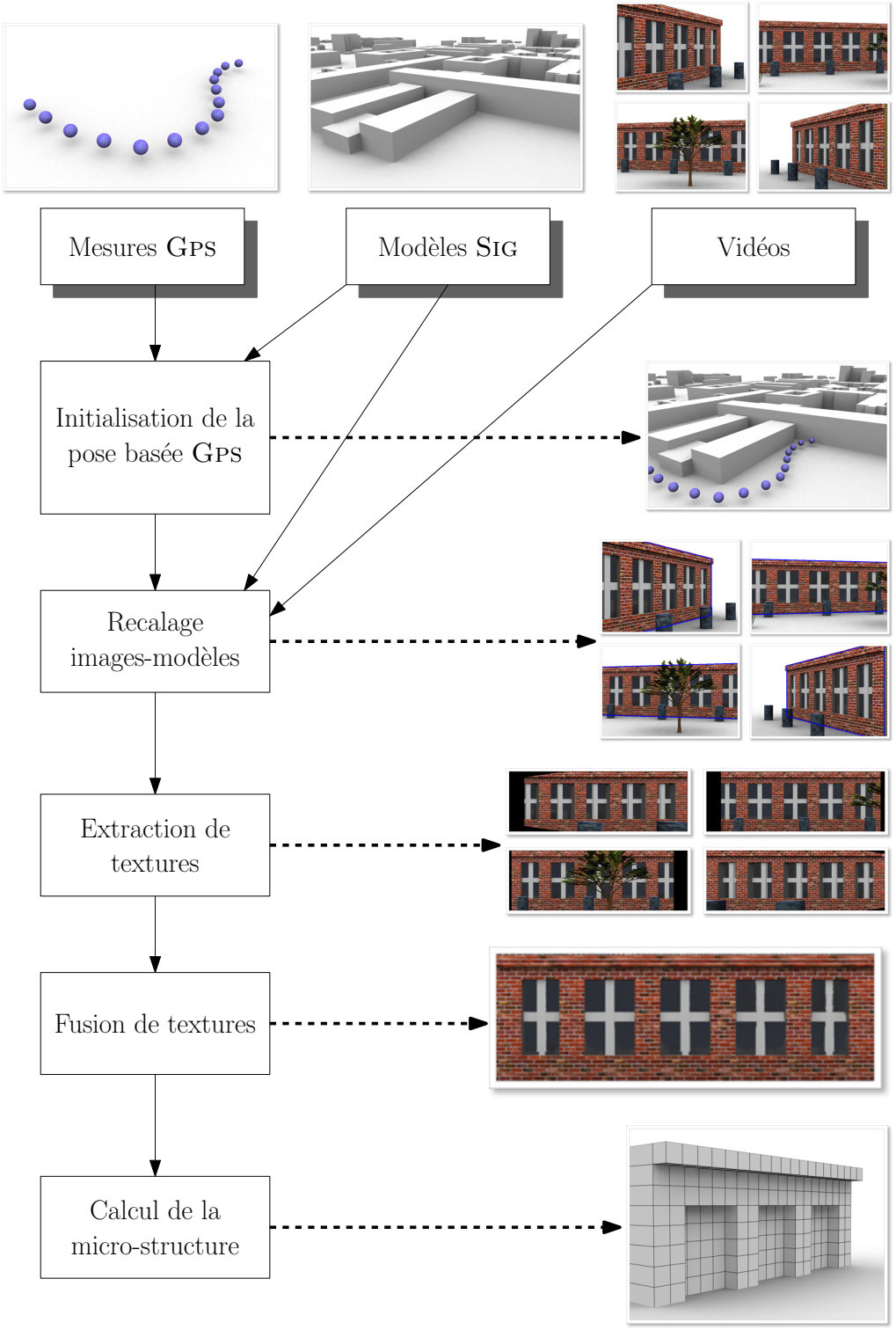


FIG. 2.7 – Aperçu global de notre approche

trinsèques : erreurs de positionnement pour le GPS, simplicité géométrique des modèles 3D et utilisation d'une caméra qui n'est pas calibrée avec une grande précision.
Un pré-traitement de ces données est alors nécessaire au bon conditionnement du recalage entre images et modèles 3D, et donc à la qualité du raffinement final des-dits modèles, à l'aide de la vidéo.

Chapitre 3

Prétraitement des données

Why think ? Why not try the experiment ? - **John Hunter**

3.1 Introduction

Dans notre cadre de raffinement de modèles 3D existants nous avons vu que toutes nos données en entrée sont entachées d'incertitudes ou d'erreurs (chapitre 2). L'altitude des bâtiments du SIG est en général estimée avec une erreur due au mode de calcul de ces modèles 3D, basé sur des images aériennes. Les mesures GPS sont entachées d'erreurs de positionnement à cause de la dispersion atmosphérique, de la visibilité des satellites ou de l'activation ou non des algorithmes de correction d'erreur. Le modèle de la caméra ayant acquis les données vidéo est quant à lui supposé non calibré, et ses paramètres internes ne sont donc pas connus avec une extrême précision.

Avant de recaler les données SIG avec les données vidéo en utilisant les mesures GPS, on peut toutefois effectuer certains prétraitements sur ces données pour mieux les conditionner. Nous n'essayons cependant pas de corriger la hauteur des bâtiments du SIG si elle est erronée. On la supposera suffisamment proche de sa vraie valeur.

3.2 Calibrage de la caméra

Nous souhaitons être contraints le moins possible par l'acquisition de données. Cela implique que l'enregistrement des vidéos peut être fait à des résolutions, fréquences et qualités hétérogènes. Plus concrètement, cela signifie que l'on ne peut contraindre la caméra à être calibrée de façon extrêmement précise, c'est-à-dire disposer d'une estimation précise de la matrice $\mathbf{K}$ ou des paramètres intrinsèques décrits dans l'équation 4.3, estimation donnée généralement par un processus de calibrage utilisant une mire dont on connaît les propriétés avec une grande précision.

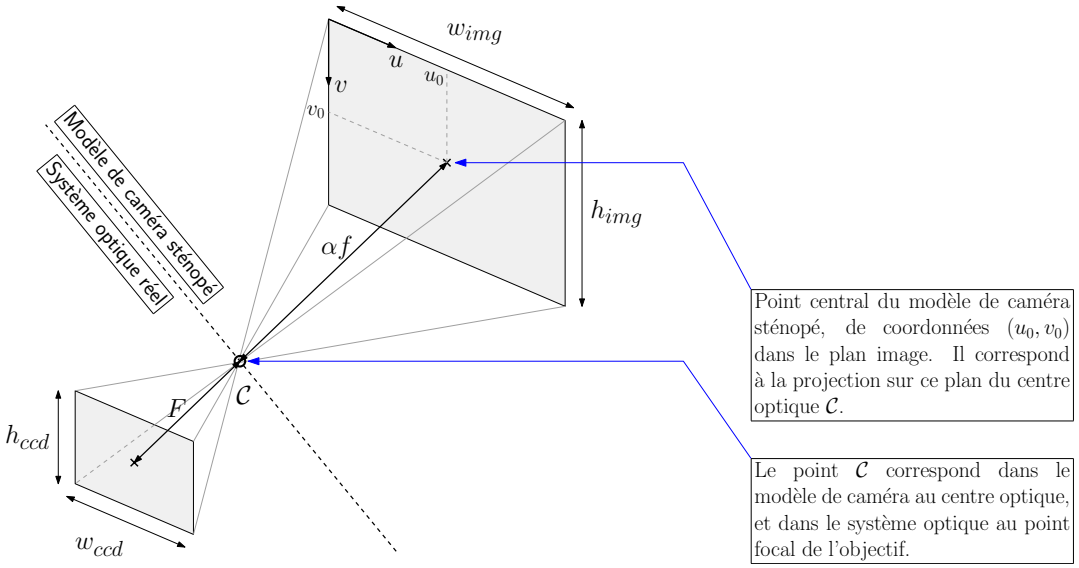


FIG. 3.1 – Relation géométrique entre modèle de caméra projective et système optique du matériel d'acquisition

3.2.1 Calibrage de la caméra utilisant les données du constructeur

Dans notre cadre d'étude, on suppose que les paramètres donnés par le constructeur du matériel d'acquisition suffisent à l'estimation de $\mathbf{K}$. Les données nécessaires à ce calcul sont :

- La focale F de l'objectif utilisé (en millimètres)
- Les dimensions (w_{ccd}, h_{ccd}) du capteur CCD (en millimètres)
- Les dimensions (w_{img}, h_{img}) des images acquises (en pixels)

On fait les hypothèses suivantes :

- Le point central est au centre de l'image
- La focale est fixe durant l'acquisition
- Les pixels sont généralement carrés, *i.e.* $\alpha_u = \alpha_v = \alpha$

A partir de ces données constructeur on détermine les valeurs des paramètres (f_u, f_v) du modèle de la caméra, qui correspondent à la décomposition selon les deux axes de l'image de la focale f de ce modèle (voir figure 3.2). La relation géométrique entre la modélisation de la caméra et le système optique correspondant est illustrée sur la figure 3.1.
Si les images sont redimentionnées sans respecter l'*aspect ratio* du capteur CCD, ou si les pixels du capteur CCD ne sont pas carrés, alors $\alpha_u \neq \alpha_v$ et les deux termes f_u et f_v de $\mathbf{K}$ sont évalués séparément (voir figure 3.2) :

$$\begin{cases} f_u = F\frac{w_{img}}{w_{ccd}} \\ f_v = F\frac{h_{img}}{h_{ccd}} \end{cases} \tag{3.1}$$

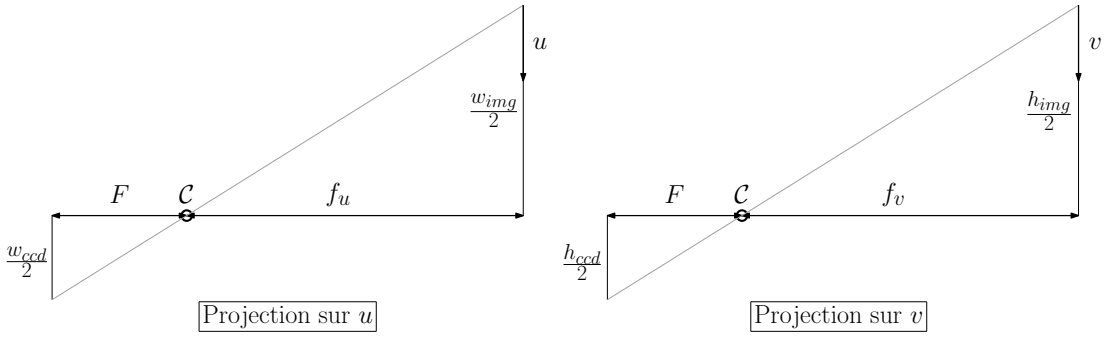


FIG. 3.2 – Pixels non carrés : projections selon u et v

Par contre, si les pixels sont carrés, alors $w_{img}/h_{img} = w_{ccd}/h_{ccd}$: l'*aspect ratio* du CCD est le même que celui des images. On peut alors écrire :

$$F\frac{w_{img}}{w_{ccd}} = F\frac{h_{img}}{h_{ccd}} = f\alpha \tag{3.2}$$

3.2.2 Calibrage de la caméra utilisant les modèles SIG

Afin de valider une telle approximation des paramètres intrinsèques de la caméra, nous avons développé une procédure simple de calibrage se basant sur la connaissance des dimensions des modèles SIG et de leur projection dans les images. Cette méthode s'inspire des travaux de Wilczkowiak *et al.* [WSB05].

3.2.2.1 Image de la conique absolue

En utilisant la modélisation de la matrice $\mathbf{K}$ (pour plus de précisions, voir l'équation 4.3 dans le chapitre 4), on peut définir également la matrice $\omega \sim \mathbf{K}^{-\top}\mathbf{K}^{-1}$ ($\sim$ étant l'opérateur d'égalité à un facteur d'échelle près), qui représente l'image de la conique absolue (IAC) utilisée courrament dans la littérature pour définir des contraintes sur les paramètres intrinsèques de la caméra :

$$\omega \sim \mathbf{K}^{-\top}\mathbf{K}^{-1} = \begin{bmatrix} 1 & 0 & -u_0 \\ 0 & \tau^2 & -\tau^2 v_0 \\ -u_0 & -\tau^2 v_0 & \tau^2 f_v^2 + u_0^2 + \tau^2 v_0 \end{bmatrix}, \quad \text{avec } \tau = \frac{f_u}{f_v} \tag{3.3}$$

3.2.2.2 Paramétrisation des parallélépipèdes

On considère ici qu'un parallélépipède est défini par 12 paramètres : 6 paramètres extrinsèques correspondant à sa pose dans le repère objet, ainsi que 6 paramètres intrinsèques définissant sa forme (voir figure 3.3) :

- 3 paramètres pour les longueurs des côtés (l_1, l_2 et l_3, $l_i > 0$)
- 3 paramètres pour les angles entre les différents côtés (θ_{12}, θ_{13} et θ_{23} ; $\theta_i \in]0;\pi[$)

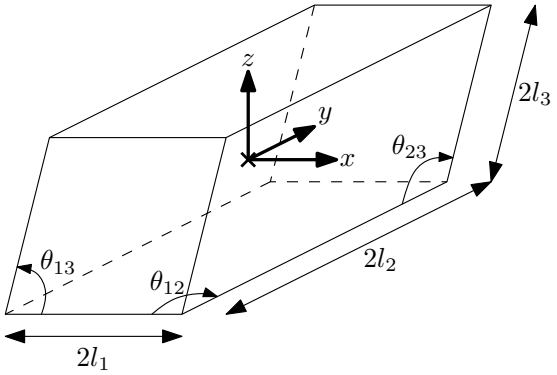


FIG. 3.3 – Paramétrisation d'un parallélépipède

On peut alors représenter un parallélépipède par la matrice compacte $\mathbf{Q}$:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{S} & \mathbf{v} \\ \mathbf{0} & 1 \end{bmatrix} \widetilde{\mathbf{L}}, \quad \text{avec } \widetilde{\mathbf{L}} = \begin{bmatrix} l_1 & l_2 c_{12} & l_3 c_{13} & 0 \\ 0 & l_2 s_{12} & l_3 \frac{c_{23} - c_{13} c_{12}}{s_{12}} & 0 \\ 0 & 0 & l_3 \sqrt{\frac{s_{12}^2 - c_{13}^2 s_{12}^2 - (c_{23} - c_{13} c_{12})^2}{s_{12}^2}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.4}$$

où $\mathbf{S}$ est l'orientation du parallélépipède, $\mathbf{v}$ sa position, et où $\widetilde{\mathbf{L}}$ représente une transformation affine entre le cube canonique et le parallélépipède ($c_{ij} = \cos\theta_{ij}$, $s_{ij} = \sin\theta_{ij}$). Elle joue le même rôle que la matrice $\mathbf{K}$ pour la caméra. Les sommets $\mathbf{X}_{i,i\in\{1..8\}}$ de ce dernier dans le repère objet sont donc définis entièrement par les sommets $\mathbf{C}_{i,i\in\{1..8\}}$ du cube canonique associés à la matrice $\mathbf{Q}$:

$$\forall i \in \{1..8\}, \quad \mathbf{X}_i = \mathbf{Q}\mathbf{C}_i \tag{3.5}$$

De façon analogue à l'image de la conique absolue ω pour la caméra, on peut définir pour les parallélépipèdes la matrice μ :

$$\mu \sim \mathbf{L}^\top \mathbf{L} \sim \begin{bmatrix} l_1^2 & l_1 l_2 \cos\theta_{12} & l_1 l_3 \cos\theta_{13} \\ l_1 l_2 \cos\theta_{12} & l_2^2 & l_2 l_3 \cos\theta_{23} \\ l_1 l_3 \cos\theta_{13} & l_2 l_3 \cos\theta_{23} & l_3^2 \end{bmatrix}, \tag{3.6}$$

où $\mathbf{L}$ est la sous-matrice 3×3 supérieure gauche de $\widetilde{\mathbf{L}}$.

3.2.2.3 Utilisation de la dualité entre K et L

On peut déduire des équations 4.1 et 3.5 qu'un point $\mathbf{C}_i$ du cube canonique se projette en $\mathbf{x}_i$ dans l'image de la façon suivante (voir également figure 3.4) :

$$\mathbf{x}_i \sim \mathbf{P}\mathbf{X}_i = \mathbf{P}\mathbf{Q}\mathbf{C}_i = \widetilde{\Gamma}\mathbf{C}_i \tag{3.7}$$

Cette matrice $\widetilde{\Gamma}$ est appelée matrice de projection canonique. En connaissant au moins

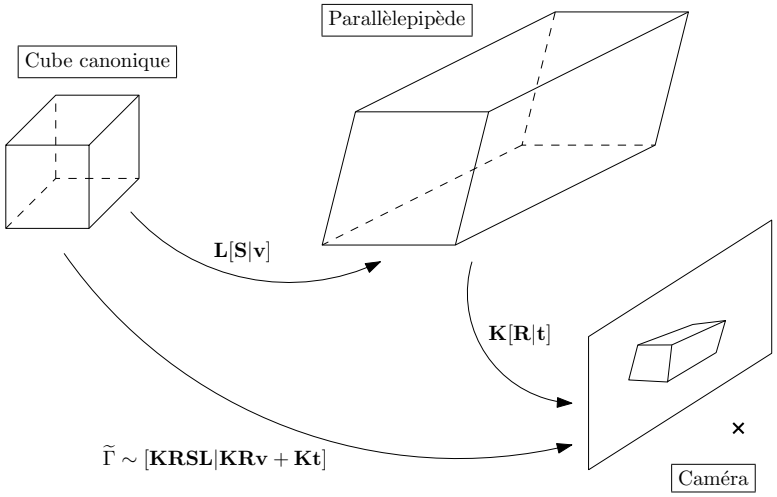


FIG. 3.4 – Projection canonique

6 correspondances entre des sommets $\mathbf{C}_i$ du cube canonique et leurs correspondants $\mathbf{x}_i$ dans une image, cette matrice peut être estimée sans connaissance a priori sur les poses respectives du parallélépipède et de la caméra. On peut en effet réécrire l'équation 3.7 sous la forme d'un système de type $\mathbf{A}\widetilde{\gamma} = \mathbf{0}$, où $\widetilde{\gamma}$ est un vecteur contenant les paramètres de $\widetilde{\Gamma}$. La solution est donnée à partir de la décomposition en valeurs singulières (SVD, *Singular Values Decomposition*) de $\mathbf{A}$, et correspond au vecteur propre de $\mathbf{A}$ associé à sa valeur propre minimale.

Connaissant $\widetilde{\Gamma}$, on peut en extraire la sous-matrice supérieure gauche Γ, de taille 3×3 puis dériver la relation suivante :

$$\begin{array}{rrcl} & \Gamma^\top \omega \Gamma & = & [\mathbf{KRSL}]^\top \omega [\mathbf{KRSL}] \\ & & = & \left[\mathbf{L}^\top \mathbf{S}^\top \mathbf{R}^\top \mathbf{K}^\top\right] \mathbf{K}^{-\top} \mathbf{K}^{-1} [\mathbf{KRSL}] \\ & & = & \mathbf{L}^\top \mathbf{L} \quad \text{(du fait de l'orthogonalité de } \mathbf{R} \text{ et } \mathbf{S}) \\ \Leftrightarrow & \Gamma^\top \omega \Gamma & \sim & \mu \end{array} \tag{3.8}$$

De même que pour $\widetilde{\Gamma}$, en connaissant Γ et μ, on trouve ω par SVD, obtenant ainsi un vecteur $\mathbf{w} = [a\ b\ c\ d\ e]^\top$ contenant les paramètres de ω. La matrice de paramètres intrinsèques $\mathbf{K}$ se déduit alors de $\mathbf{w}$ en utilisant la définition de ω (équation 3.3) :

$$\omega \sim \begin{bmatrix} a & 0 & b \\ 0 & c & d \\ b & d & e \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & b' \\ 0 & c' & d' \\ b' & d' & e' \end{bmatrix} \Rightarrow \left\{ \begin{array}{rcl} u_0 & = & -b' \\ v_0 & = & -d'/c' \\ f_u & = & \sqrt{e' - b'^2 - d'^2/c'} \\ f_v & = & f_u/\sqrt{c'} \end{array} \right. \tag{3.9}$$

3.2.3 Expérimentation

Nous avons implanté cette méthode de calibrage pour comparer les valeurs obtenues avec celles calculées à partir des données constructeur. Les deux cas $f_u = f_v$ et $f_u \neq f_v$ sont étudiés. Les images fournies en entrée représentent un bâtiment parallélépipède de la base SIG dont au six sommets sont visibles. Les correspondances entre ces sommets et leur projection dans l'image sont données à la main à l'aide d'une interface utilisateur. Deux estimations différentes peuvent donc utiliser en entrée des correspondances quelques peu distinctes, et donc générer des résultats différents.

3.2.3.1 Cas $f_u = f_v$

L'image utilisée a été acquise à l'aide d'un appareil photo numérique Nikon D70s, avec une focale de 18 millimètres. Le capteur CCD et l'image ont une taille respectivement de $23,7 \times 15,8$ millimètres et 900×600 pixels. Pour minimiser les erreurs de mise en correspondance faites à la main, dix estimations sont effectuées.
On peut déduire des données constructeur la matrice de paramètres intrinsèques suivante :

$$\widetilde{\mathbf{K}} = \begin{bmatrix} 683,544 & 0 & 450,0 \\ 0 & 683,544 & 300,0 \\ 0 & 0 & 1 \end{bmatrix}$$

En faisant la moyenne pour chaque paramètre (f_u, f_v, u_0, v_0) des dix estimations effectuées, on obtient la matrice suivante :

$$\mathbf{K} = \begin{bmatrix} 620,094 & 0 & 442,632 \\ 0 & 658,760 & 295,637 \\ 0 & 0 & 1 \end{bmatrix}$$

Les écarts-types mesurés sur chaque paramètre sont les suivants :

- $\sigma_{f_u} = 1,982$
- $\sigma_{f_v} = 4,179$
- $\sigma_{u_0} = 5,119$
- $\sigma_{v_0} = 3,230$

On peut observer à partir des mesures de $\mathbf{K}$ et des σ_i que l'erreur entre les paramètres estimés et les paramètres théoriques n'est pas négligeable.

3.2.3.2 Cas $f_u \neq f_v$

L'image utilisée a été acquise à l'aide d'un Olympus μ600, avec une focale de 5,8 millimètres. Le capteur CCD et l'image ont une taille respectivement de $5,76 \times 4,29$ millimètres et 800×600 pixels. Contrairement au cas précédent, l'image a été redimentionnés sans respecter l'aspect ratio original. Le protocole de mesure reste le même, et les résultats obtenus sont les suivants :

$$\widetilde{\mathbf{K}} = \begin{bmatrix} 805,556 & 0 & 400,0 \\ 0 & 811,189 & 300,0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{K} = \begin{bmatrix} 745,765 & 0 & 419,889 \\ 0 & 806,435 & 285,350 \\ 0 & 0 & 1 \end{bmatrix}$$

- $\sigma_{f_u} = 4,145$
- $\sigma_{f_v} = 6,090$
- $\sigma_{u_0} = 16,998$
- $\sigma_{v_0} = 6,628$

Là encore, on note une différence notable entre les valeurs théoriques et les valeurs estimées.

3.2.3.3 Interprétation

Les valeurs de paramètres intrinsèques estimés avec la méthode décrite précédemment restent assez éloignées des valeurs théoriques calculées à l'aide des données constructeur. Les erreurs de mesure sur le point principal ayant une influence négligeable dans notre cadre d'étude [GSB05], nous le fixons malgré tout au centre de l'image. Nous préférons toutefois pour le suivi utiliser la focale estimée plutôt que la focale théorique, les erreurs étant à notre sens trop importantes.

Ces erreurs sont cependant à relativiser. L'estimation est en effet basée sur l'hypothèse forte que les dimensions du parallélépipède sont connues avec précision. Or, dans la base SIG, il reste une incertitude sur l'exactitude de la hauteur estimée des bâtiments, la reconstruction ayant été calculée en se basant sur des images aériennes.

3.3 Étude sur la précision des mesures GPS

Afin de déterminer les conditions acceptables d'utilisation d'un récepteur GPS en milieu urbain, nous avons effectué différentes mesures sur points fixes, dans plusieurs conditions potentielles d'utilisation. Le récepteur étant immobile, une combinaison des conditions suivantes est testée :

- Environnement occultant / environnement dégagé (trajets multiples et visibilité réduite)
- Pas de correction différentielle / correction SBAS

On rappelle par ailleurs que les effets de la dispersion atmosphérique ne sont pas évalués car ils sont déjà corrigés par l'émission du signal sur deux fréquences différentes.
Des résultats de mesure de positionnement sont présentés sur la figure 3.6. Les mesures exprimées en $(latitude, longitude, altitude)$ sont tout d'abord converties en (X, Y, Z) dans le repère UTM (voir Annexe B). Les courbes représentent l'évolution de ces positions (X, Y, Z) dans le temps. Chaque tracé est centré sur sa moyenne. Les lieux d'acquisition des mesures — avec un environnement dégagé et avec un environnement occultant — sont illustrés sur la figure 3.5.
L'ensemble des mesures est résumé dans le tableau 3.1. Nous y indiquons les écarts-types sur les mesures en (X, Y, Z), la valeur moyenne de HDOP qui donne une indication sur la qualité horizontale de la mesure, ainsi qu'une mesure de surface d'imprécision au sol définie comme la surface de l'ellipse dont les axes sont les écarts-types σ_X et σ_Y.
On peut remarquer tout d'abord que de façon attendue, les mesures sont beaucoup plus

(a) Environnement dégagé

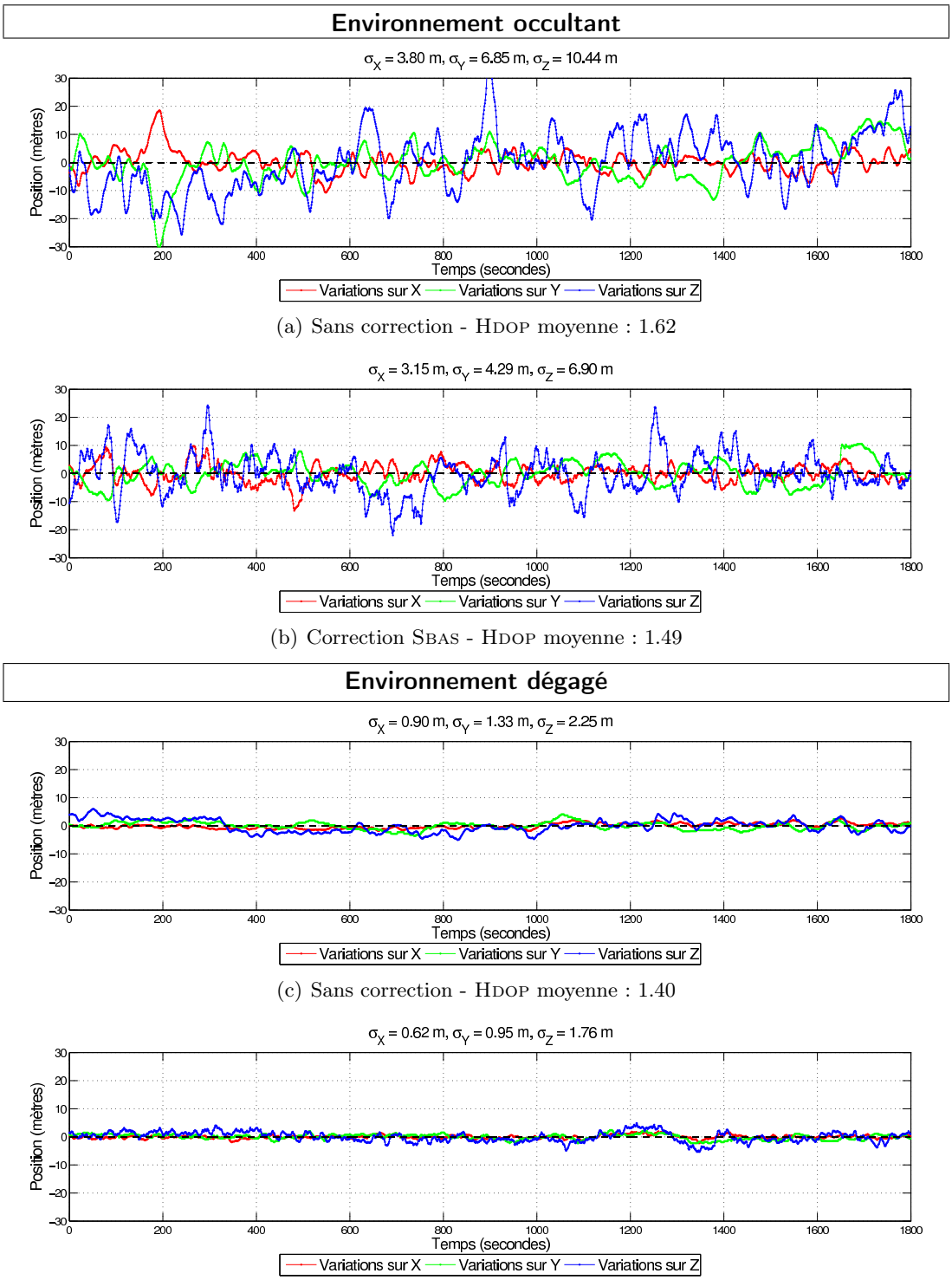
(b) Environnement occultant

FIG. 3.5 – Lieux d'acquisition des mesures GPS sur point fixe

précises dans un environnement dégagé que dans un environnement de type urbain, où des bâtiments entourent le point de mesure (surface d'imprécision allant de 10 à 20 mètres carré dans ce cas, contre moins d'un mètre carré dans le cas d'un environnement dégagé). De plus, pour un environnement particulier, les mesures effectuées à l'aide du système de correction SBAS sont toujours plus précises que celles effectuées sans correction. Cette correction a toutefois ses limites car l'imprécision est dans ce cas quand même supérieure à un mètre. Enfin, comme attendu, l'imprécision en altitude est toujours environ une fois et demie plus importante qu'au sol. Par contre, le fait que l'imprécision selon Y soit toujours plus importante que selon X pour ces mesures reste inexpliqué.

On peut déduire des mesures présentées ici principalement deux choses :

- Pour effectuer des mesures GPS, que ce soit en milieu urbain ou dégagé, l'utilisation du système de correction SBAS peut être toujours utilisée pour réduire l'erreur sur le positionnement.
- Les mesures d'altitude étant beaucoup plus imprécises, la possibilité de ne pas les prendre en compte est à étudier si l'on peut trouver un moyen de s'en passer.



(a) Sans correction - Hdop moyenne : 1.62

(b) Correction Sbas - Hdop moyenne : 1.49

(c) Sans correction - Hdop moyenne : 1.40

(d) Correction Sbas - Hdop moyenne : 1.02

Fig. 3.6 – Mesure Gps sur point fixe - Environnement occultant

	Environnement occultant		Environnement dégagé	
	Sans correction	SBAS	Sans correction	SBAS
σ_X	3.80 m	3.15 m	0.90 m	0.62 m
σ_Y	6.85 m	4.29 m	1.33 m	0.95 m
σ_Z	10.44 m	6.90 m	2.25 m	1.76 m
HDOP	1.62	1.49	1.40	1.02
Imprécision	20.44 m^2	10.61 m^2	0.94 m^2	0.46 m^2

TAB. 3.1 – Résumé des mesures GPS effectuées sur point fixe

3.4 Conclusion

Les sources d'erreur et d'imprécision sur les données vidéo et GPS utilisées en entrée on été décrites dans ce chapitre. La connaissance (au moins approximative) des paramètres internes de la caméra, ainsi que les erreurs attendues de positionnement sur les mesures GPS, sont à prendre en compte pour l'estimation de la poses des différentes caméras et donc pour le recalage entre les images et les modèles 3D. En particulier, les paramètres constructeur de la caméra doivent être connus, et la correction SBAS activée lors de l'acquisition de données GPS.

Deuxième partie

Recalage images-modèles

Introduction

Don't reinvent the wheel, just realign it - **Anthony J. D'Angelo**

Afin de pouvoir raffiner les modèles 3D de bâtiments extraits de la base SIG à partir des données vidéo, on doit être capable de déterminer à quelle partie du modèle correspondent les différentes zones dans chaque image de la vidéo. Une solution à ce problème consiste à aligner la projection des modèles sur les images de bâtiments. Or, pour effectuer une telle projection, on doit connaître à la fois les paramètres intrinsèques (focale, centre de projection, etc.) et extrinsèques (position et orientation) de la caméra pour toutes les images de la vidéo. On a déjà vu que les paramètres intrinsèques sont déterminés par les données constructeur de la caméra (voir chapitre 3). On appelle *pose* de la caméra les paramètres extrinsèques exprimés dans le même repère que la base SIG, à savoir le repère UTM.

Nous présentons donc dans cette partie une méthode automatique permettant de calculer de façon précise la pose de la caméra pour toutes les images de la vidéo considérée. Ce recalage entre images et modèles 3D est possible sans poser de contraintes sur le mouvement de la caméra d'acquisition. On peut également l'estimer dans le cas où des objets présents dans la vidéo mais pas le modèle 3D (comme des arbres par exemple) masquent partiellement les façades.

Principe de la méthode Le calcul de pose pour l'ensemble des images se décompose en trois étapes principales :

1. *Initialisation de la position.* La position des caméras est initialisée en utilisant les mesures GPS acquises. Celles-ci sont interpolées temporellement car la fréquence d'acquisition GPS est souvent inférieure à celle de la vidéo utilisée.
2. *Calcul de pose pour la première image.* Un algorithme basé sur les premières images et les mesures GPS permet de calculer précisément la pose pour la première image de la vidéo, pour laquelle on ne connaissait à ce stade qu'une approximation de

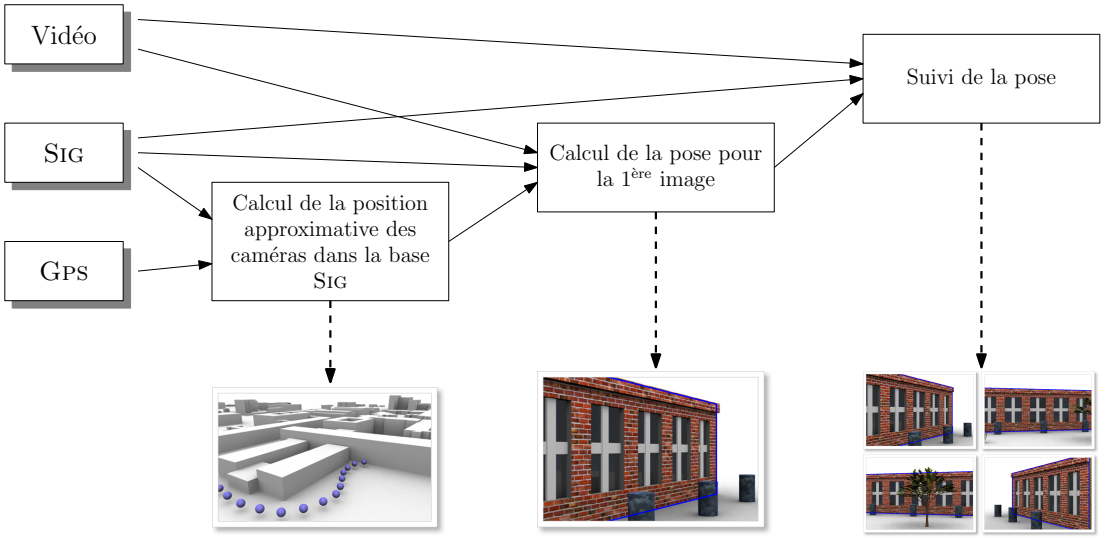


Fig. 3.7 – Chaîne algorithmique pour le calcul de poses de la caméra pour toutes les images

la position.

3. *Suivi de la pose.* La mise en correspondance entre des primitives extraites des images et le modèle 3D, associée au suivi de ces primitives dans la vidéo, permet de suivre la projection du modèle recalée pour la première image.

Plan Cette partie est organisée comme suit : nous présentons tout d'abord les outils mathématiques nécessaires dans notre cadre de recalage images-modèles dans le chapitre 4. Ce chapitre est suivi par la présentation du calcul de pose pour la première image (chapitre 5). Enfin, nous présentons la phase de suivi de la pose dans les images restantes (chapitre 6). La figure 3.7 reprend la chaîne nécessaire au recalage images-modèles souhaité.

Chapitre 4

Fondements théoriques

In theory, there is no difference between theory and practice. In practice, there is - **Chuck Reid**

4.1 Introduction

Ce chapitre présente un état de l'art (non exhaustif) des méthodes de recalage images-modèles, ainsi que les éléments théoriques et méthodologiques largement utilisés en recalage qui nous seront utiles pour la présentation de la méthode proposée.

La littérature sur le sujet étant assez conséquente, nous nous restreindrons à l'étude des méthodes de recalage existantes utilisant en entrée des séquences vidéo (donc avec un faible déplacement inter-images de la caméra) ainsi qu'une connaissance à priori de la structure 3D de la scène visualisée. Souhaitant une approche entièrement automatisée, nous concentrons également notre étude sur les méthodes requérant un minimum d'intervention manuelle.

L'hypothèse de faibles déplacements de la caméra entre les images successives (on parle aussi de faible *baseline*) induite par l'utilisation de séquences d'images implique en général une décomposition du recalage en deux phases distinctes. Si une image de la séquence est déjà recalée, alors la pose estimée constitue généralement une bonne initialisation au calcul du recalage de l'image suivante. On se trouve alors dans un contexte de suivi du recalage. Au contraire, pour la première image de la séquence, la pose de la caméra étant complètement inconnue, le recalage est beaucoup plus complexe à effectuer. Sans hypothèses fortes sur la structure de la scène ou la pose, ce recalage en particulier est souvent fait manuellement, et la littérature reste assez pauvre dans ce domaine en particulier.

En fonction de la structure des modèles 3D utilisés, différentes primitives peuvent être utilisées pour le recalage : points, lignes, plans, ellipses, etc. Nous détaillons plus particulièrement quelques techniques d'extraction et de suivi de points (section 4.3) et de lignes (section 4.4). Des méthodes de calcul de pose utilisant ces primitives sont décrites

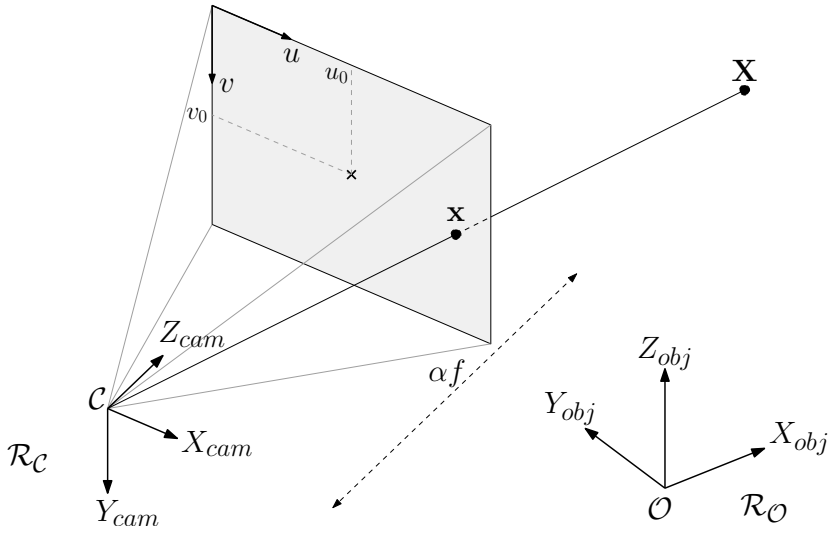


FIG. 4.1 – Modèle de caméra sténopé

en section 4.5. De nombreuses sources d'erreur pouvant entacher le succès du recalage, nous présentons en section 4.6 des méthodes d'estimation dites robustes intervenant soit en amont, soit pendant le recalage, et dont l'objectif est de minimiser la dérive induite par les erreurs pouvant être introduites.
Enfin, pour une étude plus approfondie et complète sur le sujet, le lecteur pourra se référer à [LF06]

4.2 Modélisation de la caméra

Un certain nombre d'hypothèses sont généralement faites sur le processus d'acquisition des données, en particulier sur le processus de formation de l'image sur le capteur de la caméra. Plusieurs modèles de caméra existent dans la littérature, comme par exemple les modèles de projection orthographique, para-perspective ou sténopé. C'est plus particulièrement à ce dernier que nous nous intéressons (voir figure 4.1).
Un point 3D $\mathbf{X} = [X\ Y\ Z\ 1]^\top$ se projette dans le plan image en $\mathbf{x} = [u\ v\ 1]$ de la façon suivante (les points sont exprimés en coordonnées homogènes) :

$$\mathbf{x} \sim \mathbf{P}\mathbf{X}, \tag{4.1}$$

où $\mathbf{P}$ est la matrice 3×4 de projection, définie à un facteur d'échelle près ($\sim$ étant l'opérateur d'égalité à un facteur d'échelle près). $\mathbf{P}$ peut être décomposée en deux matrices définissant plus précisément les paramètres de la caméra :

$$\mathbf{P} = \mathbf{K}\,[\mathbf{R}|\mathbf{t}]\,, \tag{4.2}$$

où :

- $\mathbf{K}$ est une matrice 3×3 définissant les paramètres internes, ou *intrinsèques* de la caméra. On a pour le modèle sténopé :

$$\mathbf{K} = \begin{bmatrix} f\alpha_u & s & u_0 \\ 0 & f\alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

  Les coordonnées du point central dans l'image sont données par $[u_0\ v_0]^\top$. Le couple $[f\alpha_u\ f\alpha_v]^\top$ modélise la focale f de la caméra multipliée par la taille des pixels, respectivement selon les axes u et v du plan image. Enfin, s défini le *skew*, et est non nul uniquement si les axes de la caméra ne sont pas perpendiculaires, ce qui devient extrêmement rare avec les caméras actuelles. $\mathbf{K}$ définit donc la matrice de passage entre le repère caméra $\mathcal{R}_\mathcal{C} = (\mathcal{C}, X_{cam}, Y_{cam}, Z_{cam})$ et le plan image.
- $[\mathbf{R}|\mathbf{t}]$ est une matrice 3×4 définissant les paramètres externes, ou *extrinsèques*, de la caméra. Plus précisément, $\mathbf{R}$ est une matrice de rotation 3×3 et $\mathbf{t}$ un vecteur 3×1. $[\mathbf{R}|\mathbf{t}]$ définit la matrice de passage entre le repère objet $\mathcal{R}_\mathcal{O} = (\mathcal{O}, X_{obj}, Y_{obj}, Z_{obj})$ et repère caméra $\mathcal{R}_\mathcal{C}$. Les colonnes de $\mathbf{R}$ sont les axes de $\mathcal{R}_\mathcal{O}$ exprimés dans $\mathcal{R}_\mathcal{C}$ et $\mathbf{t}$ est la position de l'origine $\mathcal{O}$ de $\mathcal{R}_\mathcal{O}$ exprimée dans $\mathcal{R}_\mathcal{C}$. La position $\mathcal{C}$ de la caméra dans le repère objet est donc donnée par la relation :

$$\mathcal{C} = -\mathbf{R}^\top \mathbf{t} \tag{4.4}$$

Avec cette formulation du modèle sténopé, il est fait abstraction de la distorsion lenticulaire, introduite par exemple par des objectifs grand angle. Si $[\tilde{u}\ \tilde{v}]^\top$ représentent les coordonnées d'un point 2D dans le cas idéal (*i.e.* sans distorsion) et $[u_d\ v_d]^\top$ ses coordonnées mesurées (avec distorsion), on peut les relier par la fonction de distorsion radiale $\mathcal{L}$, qui est un bonne modélisation de la distorsion lenticulaire :

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = \mathcal{L}(\tilde{r}) \begin{bmatrix} \tilde{u} \\ \tilde{v} \end{bmatrix}, \tag{4.5}$$

où $\tilde{r}$ est la distance radiale $\sqrt{\tilde{u}^2 + \tilde{v}^2}$ au centre de distorsion radiale (pris généralement comme étant le point principal $[u_0\ v_0]^\top$, même s'ils peuvent ne pas coïncider exactement). $\mathcal{L}(\tilde{r})$ est une fonction définie pour des valeurs positives de $\tilde{r}$ avec $\mathcal{L}(0) = 1$. Une approximation de $\mathcal{L}(\tilde{r})$ peut être donnée par le développement de Taylor :

$$\mathcal{L}(\tilde{r}) = 1 + \kappa_1 r + \kappa_2 r^2 + \kappa_3 r^3 + \dots \tag{4.6}$$

Les coefficients de la distorsion radiale sont alors donnés par $\{\kappa_1, \kappa_2, \kappa_3, \dots, u_0, v_0\}$ et sont considérés comme partie intégrante des paramètres intrinsèques de la caméra.
On considère généralement que la distorsion est négligeable ou corrigée, selon les cadres applicatifs. Le logiciel de [Int] par exemple permet une estimation de la distorsion radiale en utilisant une méthode dérivée de [HS97].

4.3 Recalage basé points

Les méthodes de suivi par points reposent généralement sur la mise en correspondance de pixels dans les images avec des points 3D appartenant au modèle correspondant. Les primitives extraites ne dépendent que de la scène visualisée. Nous ne nous intéresserons donc pas ici aux approches utilisant la détection et le suivi de marqueurs particuliers présents dans la scène. On considère en effet qu'il n'est pas envisageable d'installer de tels marqueurs dans le contexte de la reconstruction de scènes urbaines à grande échelle. Ces méthodes reposent également sur la mise en correspondance de points 2D entre les différentes images, indépendamment les uns des autres. Elles peuvent ainsi être rendues facilement robustes aux occultations partielles, aux erreurs de mise en correspondance, ou aux changements d'illumination.
Un point suivi est donc défini par sa position 2D dans les images et son correspondant 3D dans le modèle. Une telle correspondance 2D-3D est en général calculée lors d'une phase d'initialisation, où le modèle est recalé à la main ou à l'aide d'une méthode ad-hoc. Les points 2D sont ensuite suivis dans les différentes images pour calculer la nouvelle pose.

4.3.1 Détection de points d'intérêt

Le suivi d'un sous-ensemble de pixels plutôt que de l'ensemble d'entre eux (comme dans les techniques basées flot optique) permet d'une part de réduire la complexité calculatoire de l'algorithme, et d'autre part de choisir explicitement les points qui seront suivis de façon plus fiable. On appelle ces points des *points d'intérêt*. Pour être suivis efficacement dans les images successives, ces points d'intérêt doivent posséder les propriétés suivantes :

- leur voisinage doit être suffisement texturé pour qu'ils puissent être mis en correspondance facilement.
- ils doivent être suffisement différents de leurs voisins pour éviter les ambiguïtés de mise en correspondance.
- La détection doit être répétable : les mêmes points doivent être repérés dans des images différentes, malgré la distorsion perspective ou le bruit dans les images.

Dans les premiers travaux sur le sujet, les points ayant la plus grande variance d'intensité lumineuse dans les quatre directions sont retenus. Les détecteurs utilisés actuellement, comme ceux de Harris & Stephens [HS88] ou de Shi-Tomasi [TK91, ST94], se reposent quand à eux sur la matrice d'auto-corrélation :

$$\mathbf{A} = \begin{bmatrix} \sum\limits_{\mathcal{W}} I_u^2 & \sum\limits_{\mathcal{W}} I_u I_v \\ \sum\limits_{\mathcal{W}} I_u I_v & \sum\limits_{\mathcal{W}} I_v^2 \end{bmatrix} \tag{4.7}$$

Cette matrice est calculée pour chaque pixel de l'image, sur une fenêtre locale $\mathcal{W}$. I_u et I_v sont les dérivées partielles de l'image selon respectivement les axes u et v de l'image. Un pixel sera considéré comme point d'intérêt si l'image possède de forts gradients dans les deux directions. Cette information est déterminée par l'analyse de l'ordre de grandeur des valeurs propres λ_1 et λ_2 ($\lambda_1 < \lambda_2$) de $\mathbf{A}$:

- Si $\lambda_1 \approx 0$ et $\lambda_2 \approx 0$, le pixel considéré n'est pas un point d'intérêt (son voisinage est très peu texturé)
- Si $\lambda_1 \approx 0$ et λ_2 a une valeur positive importante, alors le pixel se trouve sur un contour
- Si λ_1 et λ_2 on des valeurs positives importantes et distinctes, le pixel considéré est un point d'intérêt

On peut ajouter que pour le calcul de $\mathbf{A}$, les dérivées partielles peuvent être pondérées par un noyau Gaussien, ou calculée sur une image filtrée par une Gaussienne, afin d'augmenter la résistance de la détection au bruit.

Harris & Stephens considèrent que le calcul exact des valeurs propres de $\mathbf{A}$ introduit une trop grande complexité algorithmique, et suggèrent d'utiliser à la place une mesure $\mathfrak{c}$ de *cornerness* dépendant des valeurs propres elles-mêmes :

$$\begin{aligned} & \mathfrak{c} = \lambda_1\lambda_2 - \kappa \cdot (\lambda_1 + \lambda_1)^2 \\ \Leftrightarrow \quad & \mathfrak{c} = \det(\mathbf{A}) - \kappa \cdot \mathrm{trace}(\mathbf{A})^2, \end{aligned} \tag{4.8}$$

κ étant un paramètre permettant de déterminer dans quelle mesure le détecteur rejette les points se situant sur des contours. Il doit être déterminé expérimentalement, et des valeurs de 0.04 à 0.15 sont généralement utilisées dans la littérature.

Pour le détecteur de Shi-Tomasi, les auteurs montrent que dans le cas de transformations affines des fenêtre de correlation $\mathcal{W}$ (dans un cadre de suivi) la mesure $\min(\lambda_1, \lambda_2)$ est plus efficace que $\mathfrak{c}$.

4.3.2 Mise en correspondance de points d'intérêt

Pour estimer le mouvement, on peut mettre en correspondance des ensembles $\mathbf{x}$ et $\mathbf{x}'$ de points d'intérêt extraits de deux images acquises depuis des points de vue similaires, voire successifs. Pour chaque point $\mathbf{x}_i$ de la première image, on recherche son correspondant $\mathbf{x'}_i$ de la deuxième image dans une fenêtre locale $\mathcal{W}$ centrée sur la position de $\mathbf{x}_i$. La recherche est basée sur une mesure de similarité, qui peut être calculée par exemple en maximisant une corrélation croisée centrée normalisée (ZNCC, *Zero mean Normalized Cross Correlation*) :

$$\mathrm{ZNCC}(\mathbf{x}_i, \mathbf{x'}_i, \mathcal{W}) = \frac{\sum\limits_{i \in \mathcal{W}} (\mathbf{x}_i - \bar{\mathbf{x}}) \cdot (\mathbf{x'}_i - \bar{\mathbf{x}}')}{\sqrt{\sum\limits_{j \in \mathcal{W}} (\mathbf{x}_j - \bar{\mathbf{x}})^2 \cdot (\mathbf{x'}_j - \bar{\mathbf{x}}')^2}}, \tag{4.9}$$

où $\bar{\mathbf{x}}$ dénote la moyenne des valeurs d'intensité lumineuse calculée sur la fenêtre $\mathcal{W}$ centrée sur $\mathbf{x}_i$. Une telle mesure de similarité est invariante aux transformations affines d'intensité lumineuse locale, et rend la procédure robuste aux changements d'illumination. On peut également, pour obtenir un ensemble plus fiable de correspondances, inverser le rôle des deux images et ne retenir que les couples $\mathbf{x}_i \leftrightarrow \mathbf{x'}_i$ qui se sélectionnent réciproquement.

4.4 Recalage basé lignes

Dans la même optique que pour le recalage basé points, le recalage basé lignes se repose sur la mise en correspondance entre des lignes ou segments extraits des images et les segments su modèle projetés sur le plan image. Deux catégories d'approches se dégagent de la littérature :

- celles qui recherchent les forts gradients dans l'image autour d'une première estimation de la pose, sans extraire explicitement les contours
- celles qui extraient explicitement les segments de l'image pour les mettre en correspondance avec ceux du modèle

Nous nous intéressons plus particulièrement à cette seconde catégorie d'approches, qui sont plus robustes que les méthodes se basant uniquement sur les gradients de l'image.

4.4.1 Extraction de lignes

Pour chaque image, les lignes ou segments sont extraits, généralement en utilisant une combinaison des algorithmes de Canny et Hough.

4.4.1.1 Détection de contours

De nombreux algorithmes pour la détection des contours d'une image existent dans la littérature [Gen92][KDN93][KN95][Low92][RTHN97]. Un des plus populaires est celui de Canny-Deriche [Can86][Der90]. Il se décompose en quatre étapes successives :

1. Un filtre Gaussien 2D est appliqué pour réduire le bruit dans l'image originale. Cette étape permet d'éliminer les pixels isolés qui pourraient conduire à de fortes réponses lors du calcul du gradient, conduisant ainsi à de faux-positifs.
2. Une carte des gradients d'intensité lumineuse est alors calculée pour chaque pixel de l'image, à l'aide d'un opérateur de Sobel.
3. Seuls les points correspondant à des maxima locaux de la carte des gradients sont considérés comme appartenant à des contours. On ne conserve donc que les points dont la dérivée s'annule.
4. La dernière étape consiste en une différenciation des contours par seuillage à hysteresis (un seuil bas s_{bas} et un seuil haut s_{haut}). On sélectionne les points de contours finals en fonction de l'intensité i de leur gradient de la façon suivante :
   - Si $i \leq s_{bas}$ le point est rejeté
   - Si $i \geq s_{haut}$ le point est considéré comme faisant partie d'un contour
   - Si $s_{bas} < i < s_{haut}$ le point est sur un contour s'il est connecté à un point de contour

Le résultat final de cet algorithme est une carte binaire décrivant les contours de l'image.

4.4.1.2 Détection de lignes

Comme pour les contours, de nombreuses techniques de détection de lignes existent en traitement d'images. Nous nous intéressons ici à la transformée de Hough [DH72].
Les lignes de l'image sont définies par deux paramètres ρ et θ plutôt que par a et b pour avoir un domaines de caractérisation borné (voir figure 4.2) :

- ρ est la distance entre l'origine du repère image et la droite considérée. On a $\rho \in \mathbb{R}^+$.
- θ est l'angle formé entre l'axe $\vec{u}$ du plan image et le vecteur formé de l'origine et du point de la droite le plus proche de l'origine. On a $\theta \in [0; 2\pi]$.

Dans l'espace de Hough (ρ, θ), une droite l est représentée par un point $[\rho_l \ \theta_l]^\top$, et son équation dans le plan image est la suivante :

$$\rho_l = x \cos\theta_l + y \sin\theta_l \tag{4.10}$$

La détection est basée sur le principe qu'un point dans l'image peut être considéré comme l'intersection d'une infinité de droites. Si un point a les coordonnées $[x_p \ y_p]^\top$ dans l'image, l'infinité de droites qui passent par ce point est définie par l'équation :

$$\rho(\theta) = x_p \cos\theta + y_p \sin\theta \tag{4.11}$$

Cette équation correspond à une sinusoïde dans l'espace de Hough. Tout point de l'image peut donc être représenté par une sinusoïde dans cet espace dual.
Enfin, si on considère qu'une ligne de l'image est composée d'un ensemble de points, alors l'ensemble de ces points exprimé dans l'espace de Hough correspond à un ensemble de sinusoïdes, qui s'intersectent toutes en un point $[\rho_l \ \theta_l]^\top$ qui correspond à la ligne de l'image en question. La détection de lignes se ramène donc à la recherche du point d'intersection d'un ensemble de sinusoïdes.

Concrètement, une carte des contours de l'image est tout d'abord calculée (par exemple avec l'algorithme de Canny-Deriche), puis pour chaque point de cette carte on trace la sinusoïde correspondante dans un tampon accumulateur défini comme une discrétisation fine de l'espace de Hough (les valeurs du tampon sont initialisées à 0 puis chaque sinusoïde incrémente les points par lesquels elle passe de 1). Les lignes de l'image correspondant aux intersections de sinusoïdes que l'on détecte sont alors celles dont les valeurs en (ρ, θ) sont suffisamment importantes dans l'accumulateur.

4.4.2 Mise en correspondance de lignes

Dans le contexte du recalage 2D-3D, les segments du modèle 3D sont tout d'abord projetés dans l'image par rapport à l'initialisation de la pose à estimer. La mise en correspondance en elle-même est basée sur la distance de Mahalanobis, calculée sur les attributs des segments. Ceux-ci sont généralement exprimés en coordonnées polaires (ρ, θ) dans le repère image, ou par exemple par un vecteur $\mathbf{S} = (c_u, c_v, \theta, l)$ défini par les coordonnées du point milieu, l'orientation et la longueur du segment [KDN93]. Si $\mathbf{S}_m$

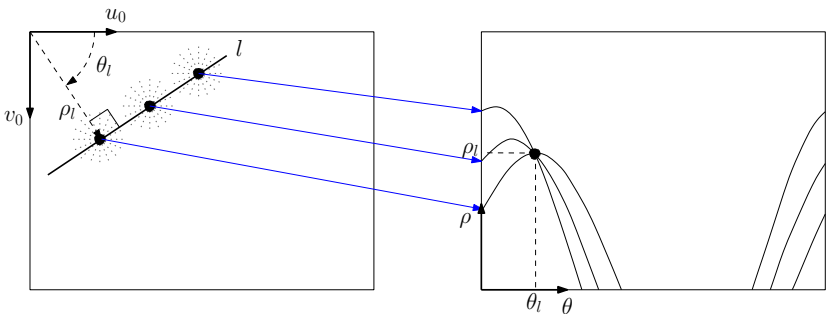


FIG. 4.2 – Transformée de Hough

est la paramétrisation d'un segment du modèle projeté et $\mathbf{S}_i$ celle d'un segment extrait de l'image, la distance d de Mahalanobis entre les deux est définie par

$$d = \sqrt{[\mathbf{S}_i - \mathbf{S}_m]^\top \, {\Gamma_{\mathbf{S}_i,\mathbf{S}_m}}^{-1} \, [\mathbf{S}_i - \mathbf{S}_m]}, \tag{4.12}$$

où $\Gamma_{\mathbf{S}_i,\mathbf{S}_m}$ est la matrice de covariance des vecteurs $\mathbf{S}_i$ et $\mathbf{S}_m$.

On utilise une procédure itérative pour trouver les meilleures correspondances $\mathbf{S}_i \leftrightarrow \mathbf{S}_m$, en général en seuillant la distance d maximale autorisée entre les deux segments. Dans [KDN93], la pose $\mathcal{P}$ est alors estimée par minimisation :

$$\hat{\mathcal{P}} = \underset{\mathcal{P}}{\operatorname{argmin}} \left(\sum_k \sqrt{\left[\mathbf{S}_i^k - \mathbf{S}_m^k(\mathcal{P})\right]^\top {\Gamma^k_{\mathbf{S}_i,\mathbf{S}_m}}^{-1} \left[\mathbf{S}_i^k - \mathbf{S}_m^k(\mathcal{P})\right]} \right), \tag{4.13}$$

où $\mathbf{S}_m^k(\mathcal{P})$ représente les attributs du segment du modèle $\mathbf{S}_m^k$ projeté par rapport à la pose $\mathcal{P}$. On notera qu'en règle générale, les lignes ne sont pas suivies d'une image à l'autre, mais ré-extraites à chaque nouvelle image.

4.5 Méthodes d'estimation de pose

Nous présentons ici plusieurs méthodes pour estimer les paramètres extrinsèques de la caméra, sans connaissances a priori sur sa position, mais en connaissant des correspondances entre des points 3D exprimés dans le repère objet, et leur projection 2D dans le plan image. L'estimation des paramètres extrinsèques quand les paramètres intrinsèques sont connus sera appelée dans la suite du manuscrit *estimation de pose*.

On considère ici n correspondances entre des points 3D $\mathbf{X}_i$ et leurs projections $\mathbf{x}_i$, et on recherche la matrice de projection $\mathbf{P}$ envoyant $\mathbf{X}_i$ sur $\mathbf{x}_i$. On recherche donc $\mathbf{P}$ telle que :

$$\forall i, \quad \mathbf{x}_i \sim \mathbf{P}\mathbf{X}_i \tag{4.14}$$

Avec $\mathbf{K}$ connue, $n = 3$ correspondances $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$ donnent quatre solutions possibles pour $\mathbf{P}$. Quand $n = 4$ ou $n = 5$, [FB81] montre qu'il y a au moins deux solutions dans

les configurations classiques, mais que si les points sont coplanaires et qu'il n'y a pas trois points collinéaires, alors la solution est unique si $n \geq 4$. Pour $n \geq 6$, la solution est unique.

4.5.1 Estimation par minimisation algébrique

4.5.1.1 Direct Linear Transformation

La DLT (*Direct Linear Transformation*) a été introduite dans la communauté vision ([Fau93], [HZ04]) pour estimer entièrement $\mathbf{P}$ à partir des correspondances entre $\mathbf{X}_i = [X_i\ Y_i\ Z_i\ 1]^\top$ et $\mathbf{x}_i = [u_i\ v_i\ 1]^\top$, même dans le cas où $\mathbf{K}$ est inconnue. En utilisant l'équation 4.14, chaque correspondance $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$ donne deux équations linéairement indépendantes sur les entrées de $\mathbf{P}$:

$$\begin{aligned} \frac{\mathbf{P}_{11}X_i + \mathbf{P}_{12}Y_i + \mathbf{P}_{13}Z_i + \mathbf{P}_{14}}{\mathbf{P}_{31}X_i + \mathbf{P}_{32}Y_i + \mathbf{P}_{33}Z_i + \mathbf{P}_{34}} &= u_i \\ \frac{\mathbf{P}_{21}X_i + \mathbf{P}_{22}Y_i + \mathbf{P}_{23}Z_i + \mathbf{P}_{24}}{\mathbf{P}_{31}X_i + \mathbf{P}_{32}Y_i + \mathbf{P}_{33}Z_i + \mathbf{P}_{34}} &= v_i \end{aligned} \tag{4.15}$$

Ces équations peuvent être réécrites sous la forme d'un système de la forme $\mathbf{Ap} = \mathbf{0}$, $\mathbf{p}$ étant un vecteur composé des coefficients de $\mathbf{P}$. La solution à ce système est déduite de la décomposition en valeurs singulières de $\mathbf{A}$ comme étant le vecteur propre associé à la plus petite valeur propre.

Si $\mathbf{K}$ est connue, les paramètres extrinsèques peuvent être déduits de $\mathbf{P}$ à un facteur d'échelle près :

$$[\mathbf{R}|\mathbf{t}] \sim \mathbf{K}^{-1}\mathbf{P} \tag{4.16}$$

La matrice $\mathbf{R}$ trouvée n'est pas forcément une matrice de rotation, mais elle peut être corrigée en ce sens [Zha00]. En conditions réelles, les mesures $\mathbf{x}_i$ sont généralement bruitées (le bruit étant modélisé par un bruit Gaussien de moyenne nulle). L'erreur minimisée par la DLT étant algébrique plutôt qu'une mesure ayant une signification physique, comme une distance géométrique, les paramètres de la caméra estimés avec cette méthode doivent généralement être raffinés par une phase d'optimisation non linéaire de l'erreur de reprojection.

4.5.1.2 POSIT

POSIT [DD95] est un algorithme très utilisé pour l'estimation de pose dans le cas où $n \geq 4$ (avec au moins quatre points non coplanaires). Le processus est itéré jusqu'à convergence, et chaque itération se décompose en deux étapes :

- POS (*Pose from Orthography and Scaling*), qui approche la projection perspective par une projection orthographique mise à l'échelle, et donne une première estimation de la position et de l'orientation de la caméra en résolvant un système linéaire.
- POSIT (POS *with ITerations*), où le facteur d'échelle de la projection orthographique est mis à jour itérativement afin de faire converger la projection orthographique vers

la projection perspective, permettant ainsi de déterminer la "vraie" pose de l'objet visualisé.

Cette technique est simple à mettre en œuvre [Int], mais elle est relativement sensible au bruit. Une version modifiée est présentée dans [DDDS04] pour traiter à part le cas des points coplanaires.

4.5.1.3 Estimation de pose à partir d'un plan 3D

Si $\mathbf{K}$ est connue, la pose de la caméra peut également être estimée à partir d'une structure plane. La relation entre un plan 3D et sa projection peut être décrite par une matrice $\mathbf{H}$ d'homographie, de taille 3×3. En effet, si on considère le plan d'équation $Z = 0$, la matrice $\mathbf{H}$ qui projette un point 3D $\mathbf{X} = [X\ Y\ 0]^\top$ de ce plan sur le point 2D $\mathbf{x} = [u\ v]^\top$ de l'image peut être déduite de la façon suivante :

$$\begin{array}{rrcl} & \mathbf{x} & \sim & \mathbf{P}\mathbf{X} \\ \Leftrightarrow & \mathbf{x} & \sim & \mathbf{K}[\mathbf{R}^1\mathbf{R}^2\mathbf{R}^3|\mathbf{t}][X\ Y\ 0\ 1]^\top \\ \Leftrightarrow & \mathbf{x} & \sim & \mathbf{K}[\mathbf{R}^1\mathbf{R}^2|\mathbf{t}][X\ Y\ 1]^\top \\ \Leftrightarrow & \mathbf{x} & \sim & \mathbf{H}[X\ Y\ 1]^\top, \end{array} \tag{4.17}$$

où $\mathbf{R}^i$ représente la $i^{\text{ème}}$ colonne de la matrice de rotation $\mathbf{R}$. Inversement, si $\mathbf{H}$ et $\mathbf{K}$ sont connues, la pose peut être retrouvée puisque :

$$\mathbf{H} = \mathbf{K}\left[\mathbf{R}^1\mathbf{R}^2|\mathbf{t}\right] \Leftrightarrow \left[\mathbf{R}^1\mathbf{R}^2|\mathbf{t}\right] = \mathbf{K}^{-1}\mathbf{H} \tag{4.18}$$

$\mathbf{H}$ peut être estimée par DLT en utilisant quatre correspondances $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$. On connaît alors le vecteur $\mathbf{t}$ et les deux premières colonnes de $\mathbf{R}$. $\mathbf{R}^3$ est déduite comme étant le produit vectoriel de $\mathbf{R}^1$ et $\mathbf{R}^2$, puisque $\mathbf{R}$ est une matrice de rotation, et est donc orthonormale.

Cette technique a été appliquée par exemple dans [SFZ00][SB02], pour un suivi de plans 3D à des fins de réalité augmentée.

4.5.2 Estimation par minimisation géométrique

L'avantage des techniques décrites précédemment est qu'elles sont rapides et ne requièrent pas d'initialisation de la pose. Elles sont cependant relativement sensibles au bruit, et de ce fait manquent de précision. Il est préférable pour obtenir de meilleurs résultats de minimiser une erreur géométrique, à savoir la distance euclidienne (ou son carré) entre les points 2D mesurés dans l'image (qui sont bruités) et les points 3D projetés correspondants. On a alors :

$$\left[\tilde{\mathbf{R}}|\tilde{\mathbf{t}}\right] = \underset{[\mathbf{R}|\mathbf{t}]}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \text{distance}^2 \left(\mathbf{P}\mathbf{X}_i, \mathbf{x}_i\right) \right) \tag{4.19}$$

La pose estimée est optimale sous la condition que les erreurs de mesure sont indépendantes et suivent une distribution Gaussienne. Elle est généralement calculée itérativement par une méthode de minimisation aux moindres carrés (Gauss-Newton, Levenberg-Marquardt, etc.), qui nécessite par contre une initialisation de la pose proche de la solution pour ne pas converger vers un minimum local.

4.5.2.1 Asservissement visuel virtuel

Une méthode classique dans le domaine de la robotique, et qui vise à calculer la pose par minimisation de l'erreur de projection, est l'asservissement visuel virtuel [MC02, Com05a, CMPC06]. Contrairement à l'asservissement visuel classique, où le but est d'aligner une caméra physique sur une caméra virtuelle, on veut ici estimer la pose de la caméra virtuelle correspondant à celle de la caméra physique. L'avantage de cette méthode est qu'elle est générique par rapport aux primitives utilisées, que ce soit des points, des lignes, des ellipses, des plans, etc.

Soit $\mathbf{S}_i$ les primitives extraites des images, et $\mathbf{S}_m$ les primitives 3D correspondantes. On note $\mathcal{P}(\mathbf{S}_m, \mathbf{R}, \mathbf{t})$ la fonction de projection des $\mathbf{S}_m^k$ dans l'image, fonction qui dépend de la pose $[\mathbf{R}|\mathbf{t}]$ que l'on cherche à estimer. La pose est initialisée puis mise à jour au sein d'une procédure itérative jusqu'à convergence en utilisant la loi de commande :

$$\mathbf{v} = -\lambda \mathbf{L}_\mathbf{S}^+ \left(\mathcal{P}\left(\mathbf{S}_m, \mathbf{R}, \mathbf{t}\right) - \mathbf{S}_i \right), \tag{4.20}$$

où $\mathbf{v} = [r_1\ r_2\ r_3\ t_1\ t_2\ t_3]^\top$ est un vecteur contenant la pose en cours d'estimation (trois paramètres r_i de rotation et trois paramètres t_i de translation), et où λ est un gain scalaire permettant de régler la vitesse de convergence de l'algorithme. $\mathbf{L_S}$ est une matrice d'interaction dépendant non seulement des primitives projetées $\mathcal{P}\left(\mathbf{S}_m, \mathbf{R}, \mathbf{t}\right)$ (le type de primitive déterminant la paramétrisation de $\mathbf{L_S}$) mais aussi de la profondeur relative entre la caméra et l'objet visualisé. Si $[\tilde{x}_k\ \tilde{y}_k]^\top$ sont les coordonnées 2D normalisées du $k^{\text{ième}}$ point 3D $\mathbf{S}_k = [X_k\ Y_k\ Z_k]^\top$, alors $\mathbf{L}_{\mathbf{S_k}}$ est définie comme :

$$\mathbf{L}_{\mathbf{S_k}} = \begin{bmatrix} -\frac{1}{Z_k} & 0 & \frac{\tilde{x}_k}{Z_k} & \tilde{x}_k\tilde{y}_k & -(1+\tilde{x}_k^2) & \tilde{y}_k \\ 0 & -\frac{1}{Z_k} & \frac{\tilde{y}_k}{Z_k} & 1+\tilde{x}_k^2 & -\tilde{x}_k\tilde{y}_k & -\tilde{x}_k \end{bmatrix} \tag{4.21}$$

La matrice finale $\mathbf{L_S}$ est définie par empilage des différentes matrices $\mathbf{L}_{\mathbf{S_k}}$, et $\mathbf{L}_\mathbf{S}^+$ dénote la pseudo-inverse de $\mathbf{L_S}$:

$$\mathbf{L}_\mathbf{S}^+ = \left(\mathbf{L}_\mathbf{S}^\top \mathbf{L_S} \right)^{-1} \mathbf{L_S} \tag{4.22}$$

4.6 Estimations robustes

L'estimation robuste de la pose dans le cadre du suivi est souvent indispensable, pour permettre de gérer les erreurs grossières sur les données en entrée, comme de fausses correspondances $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$. RANSAC et les M-Estimateurs sont deux techniques populaires pour éviter la dérive due à ce type d'erreurs. RANSAC est généralement appliqué sur la

mise en correspondance ; il ne requiert pas de bonne initialisation et donne une solution au problème posé sans utiliser toutes les données fournies en entrée. Les M-Estimateurs sont quant à eux plutôt utilisés lors du calcul de pose en lui-même ; ils sont utiles pour trouver une solution correcte avec des données aberrantes en entrée, mais nécessitent une bonne initialisation de la solution.

4.6.1 Ransac

Ransac (*RANdom SAmple and Consensus*) [FB81] est une méthode d'estimation robuste utilisée généralement en amont d'une phase de minimisation, pour éviter d'avoir en entrée des données trop aberrantes. De l'ensemble des données disponibles, Ransac extrait des sous-ensembles de taille minimale par rapport à la modélisation du problème considéré (par exemple quatre points non coplanaires dans le cas de Posit). Cela permet de maximiser la probabilité qu'au moins un de ces sous-ensembles ne contienne pas d'erreur grossière, et donc qu'il puisse permettre de calculer une solution correcte au problème.

Plus formellement, nous voulons estimer les paramètres $\mathbf{p}$ d'un modèle à partir d'un ensemble $\mathbf{m}$ de mesures, certaines étant erronées. Si n mesures sont nécessaires pour estimer les paramètres du modèle, N sous-ensembles de n mesures sont sélectionnés aléatoirement. Chacun de ces sous-ensembles est utilisé pour estimer les paramètres $\mathbf{p}_i$ du modèle, ainsi que le sous-ensemble $\mathbf{m}_i \subseteq \mathbf{m}$ de données valides (ou *inliers*, par opposition aux *outliers* qui sont les données aberrantes). La sélection qui produit le $\mathbf{m}_i$ de taille maximale est conservée, puis la solution est raffinée par minimisation aux moindres carrés en utilisant uniquement les données de $\mathbf{m}_i$.

Plusieurs paramètres sont à prendre en compte pour Ransac, en particulier le seuil d'erreur tolérée permettant de décider si une donnée est cohérente ou non avec le modèle considéré (*i.e.* le seuil définissant quels sont les *inliers* et quels sont les *outliers*). Il est généralement fixé à un multiple de l'écart-type sur les erreurs de mesure.
Un autre paramètre à fixer est N. Dans [FB81], une formule est donnée pour calculer le nombre d'essais minimal permettant d'assurer avec une probabilité p que l'on ait au moins un tirage parmis N qui ne contient aucun *outlier*. Si w est la probabilité de tirer un *inlier*, alors la probabilité de ne jamais tirer un ensemble contenant au moins un *inlier* est donné par :

$$1 - p = (1 - w^n)^N \tag{4.23}$$

En fixant p, on peut alors en déduire le nombre minimal N de tirages nécessaires pour assurer cette probabilité p :

$$N = \frac{\log(1-p)}{\log(1-w^n)} \tag{4.24}$$

La valeur de w n'est en général pas connue, mais une approximation grossière de sa valeur est souvent suffisante. Par exemple, si on prend $w = 50\%$, $\text{card}(\mathbf{S}) = 20$ mesures et $n = 4$, une recherche exhaustive implique $N = \text{C}_4^{20} = 4845$ tirages. Par contre, si

on veut que $p = 99\%$ (ce qui correpond généralement à la valeur prise pour p), seuls $N = 72$ tirages sont nécessaires.

4.6.2 M-Estimateurs

4.6.2.1 Principe

Une estimation par minimisation aux moindres carrés peut également être vue comme la maximisation d'une fonction de vraissemblance, sous l'hypothèse que les erreurs de mesure sont indépendantes et suivent une distribution gaussienne. Dans un cadre général, si $\hat{s}$ est la solution au problème considéré, $d(s)$ les données dépendant du modèle, d les données correspondantes observées et n le nombre de mesures, on a :

$$\hat{s} = \underset{s}{\operatorname{argmax}} \left(\prod_{i=1}^{n} \exp \left[-\frac{1}{2} \left(\frac{d_i(s) - d_i}{\sigma_i} \right)^2 \right] \delta d_i \right), \tag{4.25}$$

où σ_i est l'écart-type sur les erreurs de mesure. En prenant le log de cette fonction et en faisant abstraction des données constantes n et δd, on a :

$$\hat{s} = \underset{s}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \frac{(d_i(s) - d_i)^2}{2\sigma_i} \right) \tag{4.26}$$

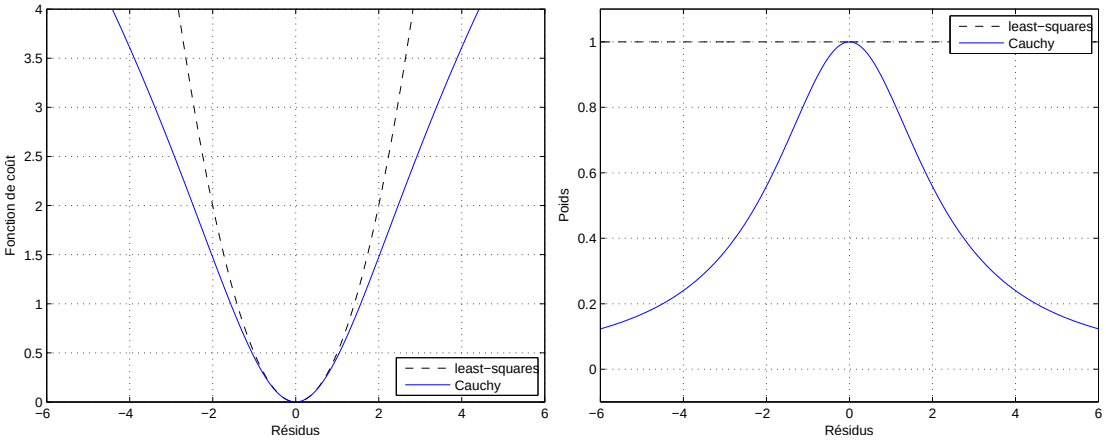
C'est l'hypothèse de distribution Gaussienne et d'indépendance des erreurs de mesure qui permet de poser que $\forall i, j \in \{1..n\}^2, i \neq j : \sigma_i = \sigma_j$. On peut alors supprimer le terme σ_i de l'équation 4.26.

En cas d'erreurs grossières dans les mesures (par exemple une mauvaise estimation 3D de certains points du modèle lors d'un calcul de pose), celles-ci ne suivent plus une distribution Gaussienne et on ne peut pas supprimer le σ_i de l'équation 4.26. Dans le cadre d'une M-estimation [Hub81] la quantité à minimiser devient alors :

$$\hat{s} = \underset{s}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \rho \left(\frac{(d_i(s) - d_i)}{\sigma_i} \right) \right), \tag{4.27}$$

où $\rho(x)$ est une fonction de coût robuste qui croît de façon sub-quadratique et est monotoniquement non décroissante pour des valeurs de $|x|$ croissantes. Son rôle est d'écarter de l'estimation les mesures entachées d'erreurs grossières. Les algorithmes de type Gauss-Newton ou Levenberg-Marquardt sont toujours utilisables pour minimiser la somme 4.27 des résidus d'erreur, même si le M-estimateur introduit est une fonction complexe. Cela est mis en œuvre en pondérant les résidus d'erreur $r_i = (d_i(s) - d_i)$ à chaque itération, à l'aide d'une fonction de pondération $w(_r i)$. Chaque r_i est remplacé par r'_i de telle sorte que :

$$r'_i = w(r_i) r_i = \frac{d\rho(r_i)}{dr_i} \tag{4.28}$$



(a) Fonction de coût (b) Poids associés

FIG. 4.3 – Cauchy : fonction de coût robuste et poids associés

Les poids $w(r_i)$ sont donc calculés en fonction de ρ :

$$w(r_i) = \frac{1}{r_i}\frac{d\rho(r_i)}{dr_i} \tag{4.29}$$

Le problème de minimisation aux moindres carrés devient alors un problème de minimisation aux moindres carrés pondérés. L'exemple le plus simple est de prendre le cas où les erreurs suivent une distribution normale, et où on a alors $\rho(r_i) = r_i^2/2$. Dans ce cas, on a $\forall i, w(r_i) = \rho'(r_i)/r_i = 1$. Il s'agit de l'estimation classique aux moindres carrés ; toutes les données en entrée contribuent avec le même poids à la recherche de la solution.

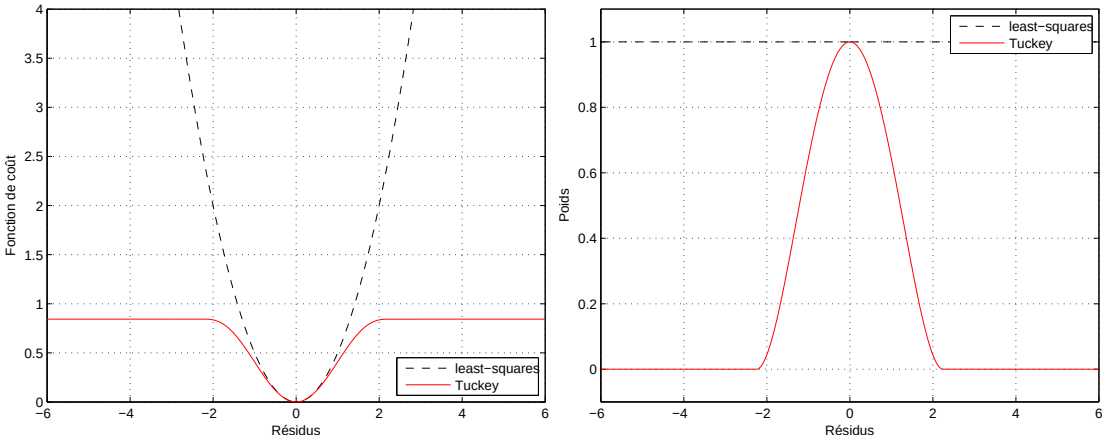
4.6.2.2 Exemples de fonctions robustes

Parmi les fonctions de coût robustes existantes, on peut citer par exemple celles de Cauchy ou de Tuckey. La fonction de Cauchy est de type *soft re-descending* (*i.e.* $\lim_{r_i\to\infty} w(r_i) = 0$), alors que celle de Tuckey est de type *hard re-descending* (*i.e.* $w(r_i) = 0$ pour $|r_i| > c$) :

$$\rho_{\text{Cauchy}}(r_i) = \frac{c^2}{2}\log\left(\frac{1}{1+\left(\frac{r_i}{c}\right)^2}\right) \tag{4.30}$$

$$\rho_{\text{Tuckey}}(r_i) = \begin{cases} \frac{c^2}{6}\left(1-\left(1-\left(\frac{r_i}{c}\right)^2\right)^3\right) & \text{si} \quad |r_i| \le c \\ \frac{c^2}{6} & \text{si} \quad |r_i| > c \end{cases} \tag{4.31}$$

Le paramètre c est un scalaire pris en général comme un multiple de l'écart-type mesuré des résidus d'erreur sur les *inliers*. Il est utilisé pour déterminer à quel moment un point est considéré comme étant un *outlier*, et donc assigné à un poids très faible ou rejeté



(a) Fonction de coût (b) Poids associés

FIG. 4.4 – Tuckey : fonction de coût robuste et poids associés

(*i.e.* $w(r_i) = 0$).
On peut déduire de l'équation 4.29 les fonctions de poids correspondantes :

$$w_{\text{Cauchy}}(r_i) = \frac{1}{1 + \left(\frac{r_i}{c}\right)^2} \tag{4.32}$$

$$w_{\text{Tuckey}}(r_i) = \begin{cases} \left(1 - \left(\frac{r_i}{c}\right)^2\right)^2 & \text{si} \quad |r_i| \leq c \\ 0 & \text{si} \quad |r_i| > c \end{cases} \tag{4.33}$$

Les figures 4.3 et 4.4 présentent le tracé des deux fonctions de coût présentées, ainsi que les poids correspondants, en fonction des résidus d'erreur. On a pris pour cet exemple $c = 2,25$ dans les deux cas. L'estimateur aux moindres carrés classique $r_i^2/2$ est également tracé à titre de comparaison. On peut noter par exemple que pour une même valeur de c, la fonction de Cauchy est beaucoup plus permissive que celle de Tuckey, dans le sens où elle intègre beaucoup plus les données de résidu élevé (et ne leur associe pas de poids nul au-delà d'un certain seuil comme le fait la fonction de Tuckey).

4.7 Conclusion

Nous avons présenté dans ce chapitre différents outils mathématiques de base pour le recalage entre images et modèles 3D, et plus particulièrement pour le calcul de pose de la caméra virtuelle associée à ce recalage.
Les techniques de calcul de pose se basent sir la mise en correspondance de primitives, d'un côté celles extraites des images elles-mêmes, et de l'autre celles correspondant à la projection du modèles 3D pour une pose donnée.
L'extraction de ces primitives, qui peuvent être de nature très diverses (points, lignes, ellipses, etc.), est généralement sensible au bruit ou à la modélisation des données en entrée. Nous présentons donc dans le chapitre suivant les prétraitements appliqués aux

données que l'on utilise pour le recalage entre des modèles 3D simples de bâtiments et des vidéos de ceux-ci.

Chapitre 5

Initialisation du recalage

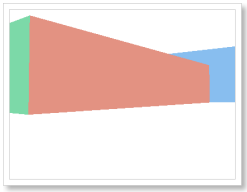
In the begining there was nothing, and it exploded - **Terry Pratchett**

5.1 Introduction

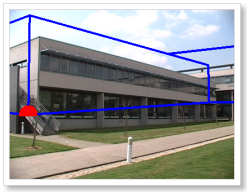
Le recalage entre les données vidéo et SIG est une étape nécessaire pour l'extraction des textures de bâtiments et le calcul de leur micro-structure. Pour chaque image de la séquence, on doit déterminer la position et l'orientation de la caméra dans le repère géo-référencé, de telle sorte que la projection perspective du modèle 3D dans le plan image de la caméra soit alignée avec les contours des bâtiments dans l'image.
Dans le cas des vidéos ou séquences d'images, le recalage est effectué sur un schéma de suivi temporel : des primitives du modèle (coins, arêtes des bâtiments) sont identifiées dans la première image puis suivies dans les suivantes. La correspondance entre les primitives 3D et leur projeté 2D dans une image de la vidéo permet d'estimer la pose (position et orientation) de la caméra et donc de réaliser le recalage images-modèle recherché.

L'initialisation du recalage consiste à estimer simultanément la pose de la caméra pour la première image à partir d'un ensemble de primitives 2D-3D en correspondances. C'est un problème délicat pour lequel on trouve de nombreuses contributions dans la littérature. Une solution consiste à éliminer une des deux inconnues (correspondance ou pose) grâce à une intervention manuelle ou à du matériel de mesure. Ainsi, dans [DTM96a] et [KBKS02] l'utilisateur indique lui-même les correspondances. Dans [TAB+03] et [RD06], c'est la pose qui est directement mesurée à l'aide d'un matériel de navigation (GPS + centrale inertielle). D'autres solutions sont proposées à partir de modèles plus riches comme un modèle texturé [RD06] ou un modèle provenant d'un scanner 3D [LS05]. Le modèle 3D dont nous disposons ne contient que les contours des bâtiments. Dans ce cas, deux méthodes existent. Elles ont chacune leur inconvénient et nécessitent qu'une pose approximative soit disponible. La première utilise l'algorithme RANSAC [FB81], elle est efficace si l'ensemble de primitives est petit et possède peu d'*outliers*. La deuxième méthode est basée sur la minimisation d'une fonction d'énergie [DDDS04], cette méthode

1: Pose approximative



2: Color coding



3: Détection / m.e.c. des points visibles

4: Calcul de la pose

FIG. 5.1 – Calcul de la pose semi-automatique pour la première image avec mise en correpondance manuelle entre l'image et le modèle 3D correspondant

peut ne pas converger du fait de la non-linéarité de la fonction de coût si la pose approximative fournie en entrée est trop éloignée de la solution (problème des minima locaux).
Dans ce chapitre, nous présentons d'abord une méthode semi-automatique (section 5.2), puis une méthode automatique pour réaliser le recalage initial. L'objectif est d'estimer la pose (position et orientation) précise de la caméra pour la première image de la séquence vidéo. La méthode proposée pour le recalage de la première image comprend deux étapes :

- *Estimation grossière de la pose, permettant une bonne superposition des champs de vue image et modèle* SIG *projeté* (section 5.3) : à l'issue de cette étape, les primitives visibles du modèle SIG projeté avec la pose estimée sont les mêmes que celles présentes dans l'image. Par contre elles ne se superposent pas de façon exacte. Cette première estimation utilise uniquement les données GPS et la vidéo et ne fait pas intervenir le modèle SIG.
- *Estimation précise de la pose* (section 5.4) : l'estimation grossière précédente permet d'établir automatiquement des correspondances 2D-3D qui sont ensuite utilisées pour estimer précisément la pose. Cette estimation utilise le modèle SIG et la première image de la séquence vidéo.

5.2 Recalage semi-automatique

Nous décrivons ici la méthode semi-automatique permettant de recaler le SIG avec la première image de la séquence. Les différentes étapes de la procédure sont illustrées sur la figure 5.1.

A ce point, seules une position et une orientation approximatives de la caméra sont connues pour la première image. La position est donnée par les mesures GPS, et l'orientation est initialisée dans le sens du déplacement de la caméra, également indiqué par la suite de mesures GPS. L'utilisateur corrige tout d'abord ces valeurs grâce à une interface OpenGL, qui affiche à la fois l'image et la projection du modèle SIG des bâtiments visualisés. Ces derniers sont rendus en mode filaire à l'aide d'une caméra virtuelle. L'utilisateur translate et oriente cette caméra virtuelle manuellement de telle sorte que le

modèle projeté soit visuellement proche du contenu de l'image (figure 5.1, partie 1). La pose initiale de la caméra est raffinée en utilisant des correspondances 2D-3D. Les seuls points 3D présents dans les modèles SIG sont les coins des bâtiments (*i.e.* les points au niveau du sol et du toit qui appartiennent à l'empreinte des bâtiments). Ceux qui sont visibles dans le rendu en filaire sont automatiquement détectés en utilisant une procédure de *color coding*. Une version polygonale de la base SIG est stockée dans la mémoire graphique, et à chaque façade est associée une unique couleur RVB :

$$R = b_i \div 256 \qquad V = b_i \text{ modulo } 256 \qquad B = f_i$$

où b_i est l'index du bâtiment dans la base, f_i l'index de la façade dans l'empreinte du bâtiment, et $\div$ l'opérateur de division entière. La couleur noire est réservée pour dénoter l'absence de bâtiment. Ce modèle coloré est rendu dans le tampon arrière d'OpenGL à la pose courante approximative. Lire le contenu de ce tampon permet d'identifier quelles sont les façades visualisées, et le couple (b_i, f_i) donne alors un accès direct dans la base SIG aux coordonnées 3D des coins des façades. Ceux qui se projettent en dehors de l'image ou qui sont occultés par une autre façade sont éliminés automatiquement.

On dispose alors de la liste $\mathbf{X}_i$ des coins de bâtiments du SIG visibles dans l'image pour la pose courante. Pour chaque point 3D $\mathbf{X}_i$ sélectionné, l'interface affiche un marqueur dans le modèle SIG, et attend que l'utilisateur fournisse en cliquant sur l'image son correspondant 2D $\mathbf{x}_i$. Une fois que toutes les correspondances 2D-3D sont effectuées, la pose est calculée automatiquement en utilisant un algorithme d'asservissement visuel virtuel à partir de l'équation 4.20. Au moins quatre de ces correspondances sont nécessaires pour calculer la pose, le résultat étant plus pertinent dans le cas de points non coplanaires.

5.3 Recalage automatique : approximation

Nous abordons dans cette section la première partie de notre algorithme de calcul automatique de la pose pour la première image de la séquence. Le but est d'obtenir une estimation approximative de la pose, permettant une bonne superposition des champs de vue image et modèle SIG projeté. A l'issue de cette étape, les primitives visibles du modèle SIG projeté avec la pose estimée sont les mêmes que celles présentes dans l'image. Cette première estimation utilise uniquement les données GPS et vidéo. Elle ne fait pas intervenir le modèle SIG.

5.3.1 Principe

L'inconnue recherchée est la pose $\mathcal{P}_{1_{utm}}$ de la caméra $\mathcal{C}_1$, exprimée dans le repère UTM ($\mathcal{R}_{utm}$) :

$$\mathcal{P}_{1_{utm}} = [\mathbf{R}_1|\mathbf{t}_1] \tag{5.1}$$

$\mathbf{t}_1$ est la translation du repère UTM au repère caméra. $\mathbf{R}_1$ est la rotation du repère UTM au repère caméra.

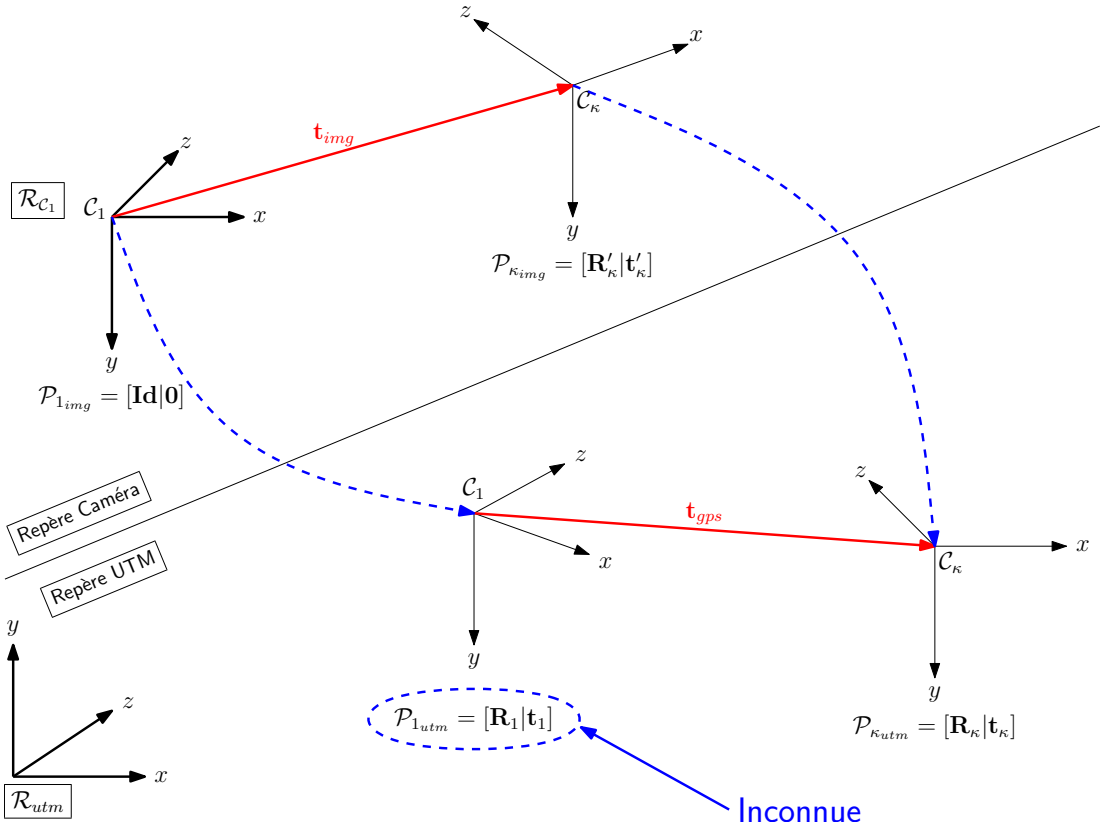


FIG. 5.2 – Mise en relation des poses exprimées dans les repères $\mathcal{R}_{\mathcal{C}_1}$ et $\mathcal{R}_{utm}$ pour estimer $\mathbf{R}_1$

La solution proposée consiste à faire coïncider les informations de translation contenues dans la séquence vidéo avec celles contenues dans les données GPS. Elle se décompose en trois étapes successives :

1. Une image I_κ (appelée "image clé") est sélectionnée dans la séquence vidéo, puis la translation relative, appelée $\mathbf{t}_{img}$, entre la caméra $\mathcal{C}_1$ et la caméra $\mathcal{C}_\kappa$ est estimée grâce aux images 1 et κ dans le repère de la caméra 1 ($\mathcal{R}_{\mathcal{C}_1}$).
2. La translation $\mathbf{t}_{gps}$ est mesurée entre les positions GPS des caméras $\mathcal{C}_1$ et $\mathcal{C}_\kappa$, exprimées dans le repère UTM. La rotation $\mathbf{R}_1$ est alors calculée par identification entre les vecteurs $\mathbf{t}_{img}$ et $\mathbf{t}_{gps}$, qui sont sencés représenter la même translation mais exprimée dans deux repères différents.
3. La translation $\mathbf{t}_1$ est déduite de $\mathbf{R}_1$ et de la position $\mathcal{C}_1$ donnée par le GPS dans le repère UTM.

Explication géométrique Considérons, sur la figure 5.2, la translation $\mathbf{t}_{img}$ entre la caméra $\mathcal{C}_1$ et la caméra $\mathcal{C}_\kappa$, exprimées dans le repère $\mathcal{R}_{\mathcal{C}_1}$. Considérons également la translation $\mathbf{t}_{gps}$ donnée par le GPS entre la caméra $\mathcal{C}_1$ et la caméra $\mathcal{C}_\kappa$ dans le repère

$\mathcal{R}_{utm}$.
Puisque les poses des caméras $\mathcal{C}_1$ et $\mathcal{C}_\kappa$ forment un système rigide, si l'on peut trouver la rotation permettant d'aligner les vecteurs $\mathbf{t}_{img}$ et $\mathbf{t}_{gps}$, alors on peut en déduire la rotation $\mathbf{R}_1$ entre le repère UTM et le repère de la caméra 1.

Il reste toutefois un degré de liberté qui est la rotation de ce système rigide $(\mathcal{R}_{\mathcal{C}_1}, \mathcal{R}_{\mathcal{C}_\kappa})$ autour du vecteur $\mathbf{t}_{gps}$. En effet, même si les directions des vecteurs $\mathbf{t}_{img}$ et $\mathbf{t}_{gps}$ sont alignées, la caméra peut néanmoins tourner autour de cet axe. Pour éliminer ce degré de liberté, l'orientation de la caméra est contrainte à une rotation plane autour de l'axe vertical (*i.e.* l'axe $\vec{Y}$ de la caméra $\mathcal{C}_1$ doit rester orthogonal au plan (X_{utm}, Y_{utm})). La rotation d'angle θ entre $\mathbf{t}_{img}$ et $\mathbf{t}_{gps}$ est ainsi estimée en utilisant les composantes horizontales de ces vecteurs.

La contrainte imposée revient à faire l'hypothèse que la caméra est tenue en "position naturelle" c'est-à-dire qu'elle ne doit être dirigée ni vers le sol, ni vers le ciel, mais dans une direction parallèle au sol. Cette hypothèse est acceptable car, d'après [HEH05], sur un ensemble de 300 images de scènes extérieures collectées avec la recherche "Google Image", la tenue de la caméra est dans la plupart des cas parallèle au sol à 15° près.

Explication mathématique Nous démontrons ici comment la connaissance des vecteurs $\mathbf{t}_{img}$ et $\mathbf{t}_{gps}$ permet de déduire la matrice $\mathbf{R}_1$ que l'on cherche à estimer.

La pose $\mathcal{P}_{1_{utm}}$ étant définie comme $\mathcal{P}_{1_{utm}} = [\mathbf{R}_1|\mathbf{t}_1]$ la matrice de passage entre le repère $\mathcal{R}_{utm}$ et le repère $\mathcal{R}_{\mathcal{C}_1}$ est donnée par :

$$^{\mathcal{R}_{\mathcal{C}_1}}\mathcal{T}_{\mathcal{R}_{utm}} = \begin{bmatrix} \mathbf{R}_1 & \mathbf{t}_1 \\ \mathbf{0} & 1 \end{bmatrix} \tag{5.2}$$

On peut en déduire la position de $\mathcal{C}_\kappa$ exprimée dans le repère $\mathcal{R}_{\mathcal{C}_1}$ en fonction de sa position exprimée dans $\mathcal{R}_{utm}$ (qui est donnée par GPS) :

$$\mathcal{C}_\kappa^{\mathcal{R}_{\mathcal{C}_1}} = ^{\mathcal{R}_{\mathcal{C}_1}}\mathcal{T}_{\mathcal{R}_{utm}} \mathcal{C}_\kappa^{\mathcal{R}_{utm}} = \mathbf{R}_1 \mathcal{C}_\kappa^{\mathcal{R}_{utm}} + \mathbf{t}_1 \tag{5.3}$$

Or, comme par définition $\mathcal{C}_1^{\mathcal{R}_{utm}} = -\mathbf{R}_1^\top \mathbf{t}_1$, on en déduit que :

$$\mathbf{t}_1 = -\mathbf{R}_1 \mathcal{C}_1^{\mathcal{R}_{utm}} \tag{5.4}$$

En substituant la valeur de $\mathbf{t}_1$ formalisée dans l'équation 5.4 à l'intérieur de l'équation 5.3, on obtient :

$$\mathcal{C}_\kappa^{\mathcal{R}_{\mathcal{C}_1}} = \mathbf{R}_1 \left(\mathcal{C}_\kappa^{\mathcal{R}_{utm}} - \mathcal{C}_1^{\mathcal{R}_{utm}} \right) = \mathbf{R}_1 \mathbf{t}_{gps} \tag{5.5}$$

Or par définition $\mathcal{C}_\kappa^{\mathcal{R}_{\mathcal{C}_1}} = \mathcal{C}_\kappa^{\mathcal{R}_{\mathcal{C}_1}} - \mathcal{C}_1^{\mathcal{R}_{\mathcal{C}_1}} = \mathbf{t}_{img}$. On a donc bien :

$$\mathbf{t}_{img} = \mathbf{R}_1 \mathbf{t}_{gps} \quad \square \tag{5.6}$$

Le vecteur $\mathbf{t}_{gps}$ étant connu, si on peut estimer $\mathbf{t}_{img}$ à partir des images, alors on peut en déduire $\mathbf{R}_1$, puis $\mathbf{t}_1$ en utilisant les équations 5.4 et 5.6.

5.3.2 Calcul de $\mathbf{t}_{img}$

Ce calcul fait intervenir les notions présentées dans [HZ04], chapitres 9 et 11. La méthode se décompose en 4 étapes :

1. Choix de l'image clé κ correspondant à la pose $\mathcal{P}_\kappa$ de la caméra $\mathcal{C}_\kappa$.
2. Détection et suivi de points d'intérêts entre la première et la $\kappa^{\text{ième}}$ image.
3. Calcul de la matrice fondamentale $\mathbf{F}$, dont on déduit la matrice essentielle $\mathbf{E}$.
4. Décomposition de la matrice essentielle $\mathbf{E}$ en $\mathbf{t}'_i$ et $\mathbf{R}'_i$, dont on déduit $\mathbf{t}_{img}$.

Choix de l'image clé κ Le choix de l'image κ est important pour le calcul de la matrice fondamentale $\mathbf{F}$ qui représente la relation géométrique entre les deux caméras. En effet, il existe certaines configurations dégénérées pour lesquelles la matrice $\mathbf{F}$ ne peut pas être déterminée précisément. Par exemple, le cas où la caméra se déplace uniquement vers l'avant (translation selon $\vec{Z}$), ou encore le cas où le centre optique de la caméra est fixe.
Dans la solution proposée ici, κ est fixée manuellement de façon à ce que le mouvement de la caméra entre la première et la $\kappa^{\text{ième}}$ image ne soit pas un cas particulier.

Détection et suivi de points d'intérêts Dans cette étape, des correspondances de points entre la première et la $\kappa^{\text{ième}}$ image sont établies. Nous utilisons pour cela l'algorithme de Kanade-Lucas-Tomasi [ST94].
Le nombre de points suivis initialement doit être suffisamment élevé pour qu'il reste au moins 8 points à la fin du suivi, c'est-à-dire le nombre de points minimum pour le calcul de la matrice fondamentale $\mathbf{F}$ avec l'algorithme des 8-points et RANSAC.

Calcul de la matrice essentielle Les points suivis entre les images I_1 et I_κ sont utilisés pour calculer la matrice fondamentale $\mathbf{F}$ en utilisant l'algorithme des 8-points et l'estimation robuste RANSAC. Une fois $\mathbf{F}$ obtenue, $\mathbf{E}$ est calculée en utilisant la matrice de paramètres intrinsèques $\mathbf{K}$ de la façon suivante :

$$\mathbf{E} = \mathbf{K}^\top \mathbf{F} \mathbf{K} \tag{5.7}$$

Calcul de $\mathbf{t}_{img}$ La matrice essentielle $\mathbf{E}$ est décomposée en $\mathbf{t}'_i$ et $\mathbf{R}'_i$ (voir [HZ04]). Puisque $\mathbf{t}_{img} = \mathcal{C}_\kappa^{\mathcal{R}_{\mathcal{C}_1}} - \mathcal{C}_1^{\mathcal{R}_{\mathcal{C}_1}} = \mathcal{C}_\kappa^{\mathcal{R}_{\mathcal{C}_1}}$, et que par définition $\mathcal{C}_\kappa^{\mathcal{R}_{\mathcal{C}_1}} = -{\mathbf{R}'_i}^\top \mathbf{t}'_i$, on en déduit finalement que :

$$\mathbf{t}_{img} = -{\mathbf{R}'_i}^\top \mathbf{t}'_i \quad \square \tag{5.8}$$

5.3.3 Calcul de $\mathbf{R}_1$

Connaissant $\mathbf{t}_{img}$ et $\mathbf{t}_{gps}$, il nous reste à calculer $\mathbf{R}_1$, qui correspond à la rotation entre ces deux vecteurs (équation 5.6). Nous la formalisons comme étant une rotation d'angle θ autour de l'axe orthogonal à ces deux vecteurs. Puisque de plus nous souhaitons contraindre cette rotation à rester dans le plan (X, Y), nous utilisons les projections horizontales des vecteurs $\mathbf{t}_{img}$ et $\mathbf{t}_{gps}$ pour calculer $\mathbf{R}_1$.

Formalisation de $\mathbf{R}_1$ Le vecteur $\mathbf{t}_{img}$ est exprimé dans le repère $\mathcal{R}_{\mathcal{C}_1}$, et sa projection dans le plan horizontal est donnée par :

$$\mathbf{t}_{img} \cdot [1\ 0\ 1]^\top = \left[\mathbf{t}_{img_x}\ 0\ \mathbf{t}_{img_z}\right]^\top$$

Le vecteur $\mathbf{t}_{gps}$ est exprimé dans le repère $\mathcal{R}_{utm}$, et sa projection dans le plan horizontal est donnée par :

$$\mathbf{t}_{gps} \cdot [1\ 1\ 0]^\top = \left[\mathbf{t}_{gps_x}\ \mathbf{t}_{gps_y}\ 0\right]^\top$$

Par conséquent, l'équation 5.6 devient :

$$\begin{aligned} \left[\mathbf{t}_{img_x}\ 0\ \mathbf{t}_{img_z}\right]^\top &= \mathbf{R}_1 \left[\mathbf{t}_{gps_x}\ \mathbf{t}_{gps_y}\ 0\right]^\top \\ \Leftrightarrow \quad \left[\mathbf{t}_{img_x}\ 0\ \mathbf{t}_{img_z}\right]^\top &= \mathbf{R}_\theta^y \mathbf{R}_{\pi/2}^x \left[\mathbf{t}_{gps_x}\ \mathbf{t}_{gps_y}\ 0\right]^\top = \mathbf{R}_\theta^y \left[\mathbf{t}_{gps_x}\ 0\ \mathbf{t}_{gps_y}\right]^\top \end{aligned} \tag{5.9}$$

où $\mathbf{R}_{\pi/2}^x$ est la rotation d'angle $\pi/2$ autour de $\vec{X}$ qui permet de placer l'axe $\vec{Y}$ à la verticale et $\mathbf{R}_\theta^y$ est la rotation horizontale d'angle θ autour de $\vec{Y}$.
Finalement, la rotation initiale de la caméra, exprimée en fonction de θ, est donnée par :

$$\begin{aligned} \mathbf{R}_1 = \mathbf{R}_\theta^y \mathbf{R}_{\pi/2}^x &= \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \\ \Leftrightarrow \quad \mathbf{R}_1 &= \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ 0 & 0 & -1 \\ -\sin\theta & \cos\theta & 0 \end{bmatrix} \end{aligned} \tag{5.10}$$

Dans cette matrice $\mathbf{R}_1$, les lignes représentent les axes de la caméra $\mathcal{C}_1$ exprimés dans le repère $\mathcal{R}_{utm}$.

Calcul de θ L'angle θ formé par les projections horizontales des vecteurs $\mathbf{t}_{gps}$ et $\mathbf{t}_{img}$ est calculé en utilisant les produits scalaire et vectoriel de ces vecteur, qui sont préalablement normalisés. Les vecteurs 2D projetés correspondants sont notés :

$$\begin{aligned} \mathbf{t}_{img}^{horiz} &= \left[\mathbf{t}_{img_x}\ \mathbf{t}_{img_z}\right]^\top \\ \mathbf{t}_{gps}^{horiz} &= \left[\mathbf{t}_{gps_x}\ \mathbf{t}_{gps_y}\right]^\top \end{aligned} \tag{5.11}$$

L'angle formé est déduit de leur produit scalaire :

$$\theta = \arccos\left(\mathbf{t}_{img_x}\mathbf{t}_{gps_x} + \mathbf{t}_{img_z}\mathbf{t}_{gps_y}\right) \tag{5.12}$$

Le sens de rotation est quant à lui donné par le signe du produit vectoriel $\mathbf{t}_{img}^{horiz} \times \mathbf{t}_{gps}^{horiz}$.

5.3.4 Résultats

Les séquences de test utilisées pour l'évaluation de cet algorithme de calcul approximatif de la pose initiale sont présentées en annexe A.

Observations La figure 5.3 illustre l'alignement obtenu à l'issue du calcul de $\mathcal{P}_{1_{utm}}$ pour les séquences *Ifsic* et *Beaulieu 1*, pour différents choix de l'image clé I_κ. On peut y voir la première image des séquences avec en surimpression le modèle SIG projeté en mode fil de fer (en bleu).

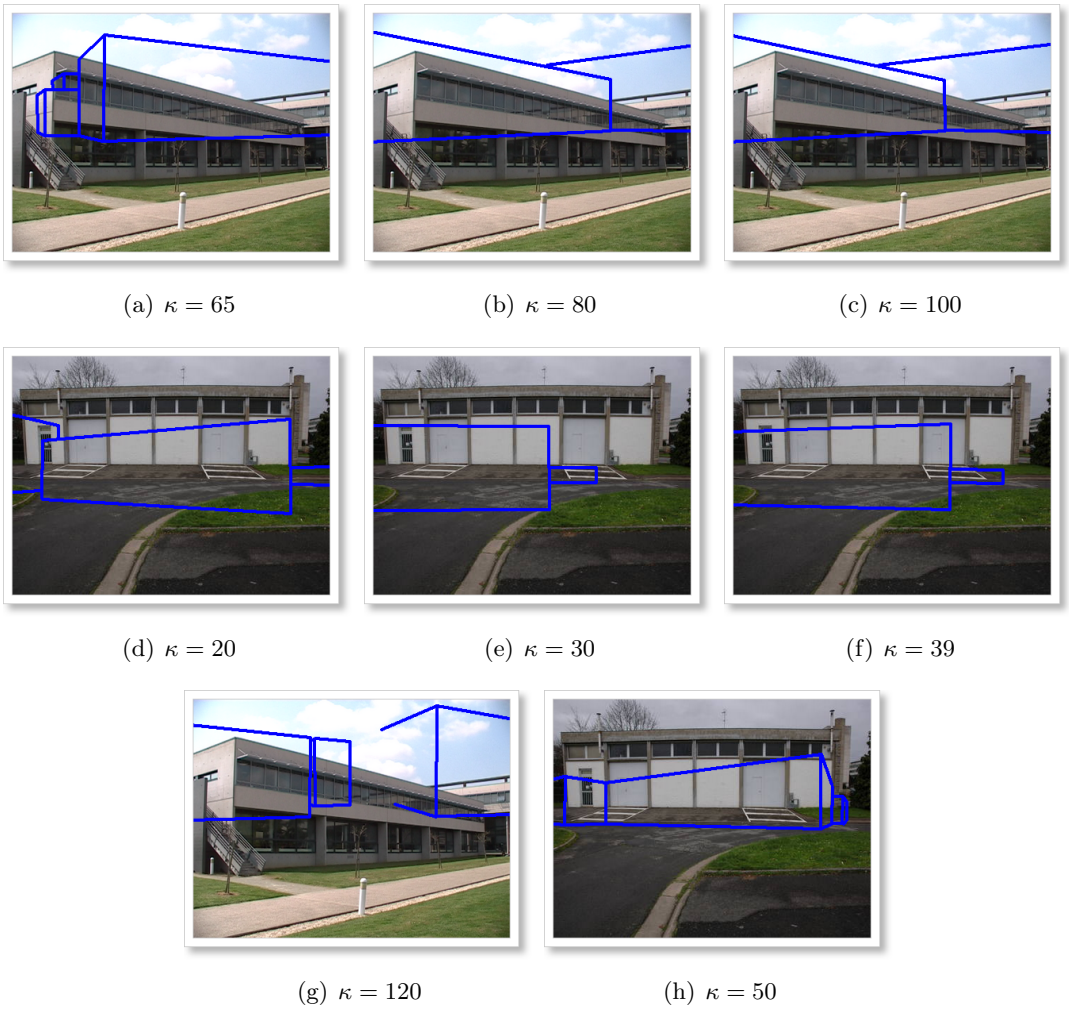
Dans les images 5.3(a), 5.3(b), 5.3(c), 5.3(d), 5.3(e) et 5.3(f), on remarque que les façades visibles du modèle SIG correspondent à celles visibles dans l'image. En ce sens, l'objectif de la première étape est atteint car une mise en correspondance est possible. Par contre, pour les images 5.3(g) et 5.3(h) l'image clé I_κ a été mal choisie et on peut voir que les façades du modèle 3D ne correspondent pas à celles de l'image.
Dans les images 5.3(a) et 5.3(d), certaines façades du modèle apparaissent en reprojection, alors qu'elles ne sont pas visibles dans l'image. Cela fait apparaître des contours du modèle pour lesquels aucune correspondance ne peut être trouvée dans l'image.

Interprétation La bonne réalisation de la première étape de l'algorithme dépend de plusieurs facteurs :

- Le facteur le plus important est le choix de l'image κ dans la séquence vidéo. En effet, nous avons vu que le mouvement de la caméra entre les deux images choisies doit être assez important et non dégénéré pour permettre l'estimation de $\mathbf{F}$, mais il faut aussi qu'au moins 8 correspondances de points entre les deux images soient disponibles, ce qui limite le nombre d'images pouvant être suivies. Il y a donc un compromis sur le choix de la valeur de κ. Malgré ce compromis, les résultats précédent ont montré que de nombreuses valeurs pour l'image clé permettent d'obtenir un alignement approximatif satisfaisant ce qui confirme le bon fonctionnement de cette première étape.
- Le deuxième facteur concerne l'hypothèse de position "naturelle" de la caméra. Cette contrainte est respectée dans la plupart des cas.
- Ensuite, l'erreur sur la position initiale $\mathbf{t}_1$ dépend de la précision du GPS, et plus particulièrement de son biais global.
- Enfin, l'erreur sur la translation $\mathbf{t}_{gps}$ dépend aussi de la précision du GPS, mais ici de la dérive de l'erreur par rapport au biais.

5.3.5 Résumé

Nous avons estimé dans cette section des valeurs approximatives de la position et de l'orientation initiales de la première caméra par rapport au repère UTM. L'orientation $\mathbf{R}_1$ est estimée grâce aux propriétés de la géométrie épipolaire dans la séquence d'image et grâce aux données GPS. La position $\mathbf{t}_1$ est quant à elle déduite de $\mathbf{R}_1$ et de la position



(a) $\kappa = 65$ (b) $\kappa = 80$ (c) $\kappa = 100$

(d) $\kappa = 20$ (e) $\kappa = 30$ (f) $\kappa = 39$

(g) $\kappa = 120$ (h) $\kappa = 50$

FIG. 5.3 – Estimation approximative de la pose initiale. (a,b,c,d,e,f) : Mise en correspondance possible. (g,h) : Aucune mise en correspondance possible.

$\mathcal{C}_1$ de la première caméra donnée par GPS. L'ensemble de la procédure est résumée dans l'algorithme 5.1.
L'erreur d'alignement obtenue est suffisamment faible pour que des correspondances de primitives puissent être établies et que le calcul de pose soit effectué, ce qui nous amène à la deuxième étape de l'algorithme, qui consiste à raffiner cette estimation de la pose $\mathcal{P}_{1_{utm}}$.

5.4 Recalage automatique : raffinement

Cette étape a pour but d'estimer de façon précise la pose $\mathcal{P}_{1_{utm}}$ de la caméra afin de recaler le modèle SIG projeté avec les bâtiments présents dans la première image. L'idée

Objectif
Connaissant les mesures GPS pour chaque image, on recherche la pose approximative $\mathcal{P}_{1_{utm}} = [\mathbf{R}_1|\mathbf{t}_1]$ de la caméra $\mathcal{C}_1$, qui permet d'obtenir les mêmes primitives entre l'image et le modèle SIG projeté.

Algorithme

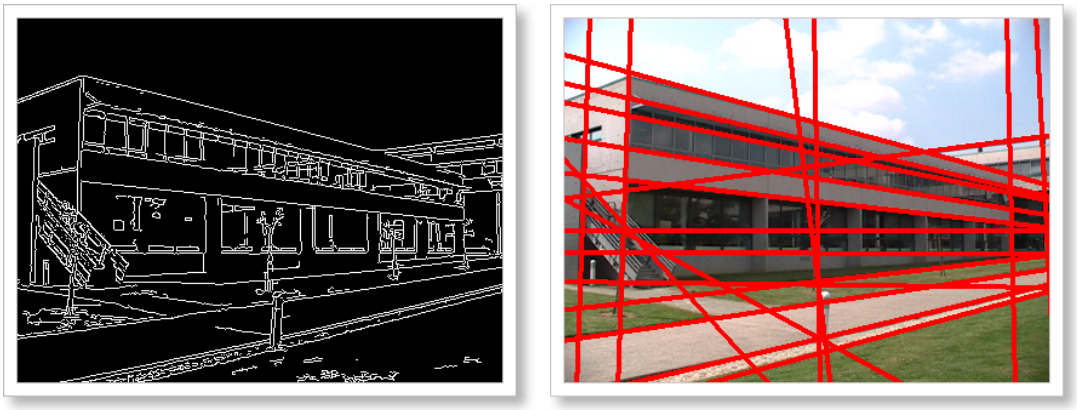
1. Sélection de l'image clé I_κ
2. Suivi de points d'intérêt entre I_1 et I_κ
3. Calcul de la matrice fondamentale $\mathbf{F}$ définie entre I_1 et I_κ, en utilisant les points suivis
4. Calcul de la matrice essentielle $\mathbf{E} = \mathbf{K}^\top \mathbf{F} \mathbf{K}$
5. Décomposition de $\mathbf{E}$ en $\mathbf{R}'_\kappa$ et $\mathbf{t}'_\kappa$
6. Calcul de la translation entre $\mathcal{C}_1^{\mathcal{R}_{\mathcal{C}_1}}$ et $\mathcal{C}_\kappa^{\mathcal{R}_{\mathcal{C}_1}}$: $\mathbf{t}_{img} = -{\mathbf{R}'_\kappa}^\top \mathbf{t}'_\kappa$
7. Calcul de $\mathbf{R}_1$ par identification entre $\mathbf{t}_{img}$ et $\mathbf{t}_{gps}$, qui est la translation entre $\mathcal{C}_1^{\mathcal{R}_{utm}}$ et $\mathcal{C}_\kappa^{\mathcal{R}_{utm}}$
8. Calcul de $\mathbf{t}_1 = -\mathbf{R}_1 \mathcal{C}_1^{\mathcal{R}_{utm}}$

ALG 5.1: Recalage approximatif de la pose pour la première image

est la suivante : lorsque l'orientation approximative de la caméra est connue (section 5.3), il est possible d'établir des correspondances de primitives entre le modèle et l'image afin de mesurer l'erreur d'alignement entre la projection du modèle 3D et l'image. Cette erreur est alors minimisée en modifiant la pose de la caméra. En considérant qu'une correspondance est établie si la distance entre deux primitives est inférieure à un certain seuil, la difficulté de cette étape repose dans la détermination d'un critère permettant de savoir si cette correspondance est correcte ou non.

5.4.1 Primitives

L'information structurelle contenue dans le modèle 3D permet pour la mise en correspondance de primitives l'utilisation de points, de lignes, de segments, de quadrilatères ou une combinaison de ces quatre choix. Par contre, l'information de texture contenue dans l'image ne peut pas être utilisée pour la mise en correspondance car elle n'existe pas dans le modèle 3D. Nous utilisons ici des lignes. L'avantage des lignes par rapport aux points est qu'elles permettent de mieux conserver l'information structurelle des bâtiments dans l'image. Utiliser des segments serait moins fiable car la longueur d'un segment est très sensible au bruit et aux occultations. Enfin, la détection de quadrilatères serait trop restrictive car certaines arêtes de bâtiments peuvent être cachées, ce qui empêche leur détection et élimine toute l'information apportée par la façade correspondante.
Les lignes que nous utilisons par la suite sont représentées sous forme polaire (ρ, θ).



(a) Canny (b) Hough : 21 lignes détectées

FIG. 5.4 – Détection de lignes pour la première image de la séquence *Ifsic*

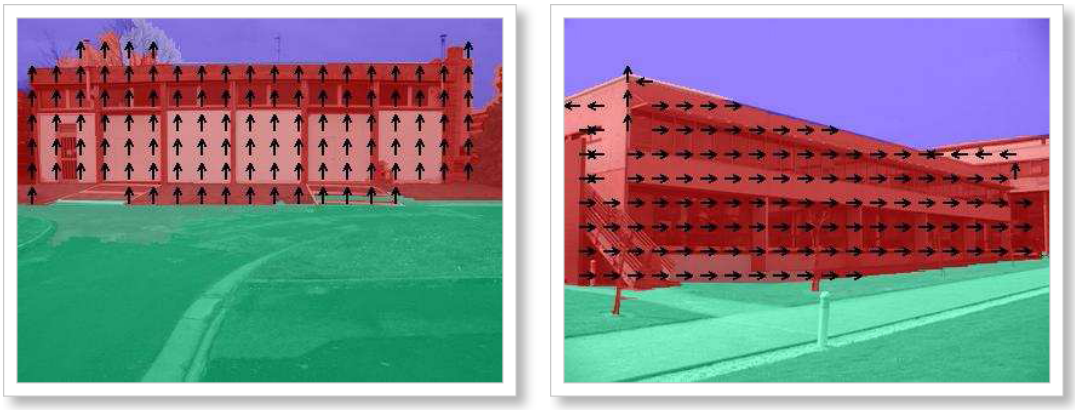
5.4.1.1 Lignes extraites de l'image

Détection Dans une image, les lignes sont détectées en trois étapes. Les contours de l'image sont tout d'abord extraits en utilisant l'algorithme de Canny. Le résultat est une image binaire dans laquelle toute l'information de texture est supprimée (figure 5.4(a)). A partir des contours, les groupes de pixels qui sont sur une même ligne sont ensuite détectés grâce à la transformée de Hough probabiliste. Enfin, lorsqu'une nouvelle ligne est proche d'une autre ligne précédemment détectée, alors elle n'est pas conservée. Sinon elle est ajoutée à l'ensemble des lignes détectées (figure 5.4(b)).

Labélisation L'ensemble de lignes obtenu précédemment contient celles correspondant aux arêtes des bâtiments — c'est-à-dire celles qui nous intéressent — mais contient aussi de nombreuses autre lignes situées à l'intérieur des façades ou au niveau du sol dans l'image. Il est souhaitable d'éliminer autant que possible ces dernières afin de diminuer la combinatoire lors de la recherche de correspondances et aussi pour éviter qu'une des ces lignes soit considérée à tort comme une arête de bâtiment.

Pour cela, nous utilisons le fait qu'une arête supérieure de bâtiment délimite la frontière ciel/bâtiment, et qu'une arête inférieure de bâtiment délimite la frontière sol/bâtiment. Ainsi, en classifiant les zones de l'image en trois zones (*ciel*, *vertical* ou *sol*), on peut éliminer toutes les lignes horizontales qui sont éloignées de la frontière ciel/bâtiment ou sol/bâtiment. De plus, on peut aussi labéliser les lignes que l'on conserve selon leur proximité à l'une des deux frontières, ce qui permettra dans la suite d'établir les correspondances uniquement entre lignes de même label.

La classification est réalisée en utilisant uniquement la première image, avec la méthode d'estimation du contexte géométrique proposée dans [HEH05] et dont le programme



(a) *Beaulieu 1* (b) *Ifsic*

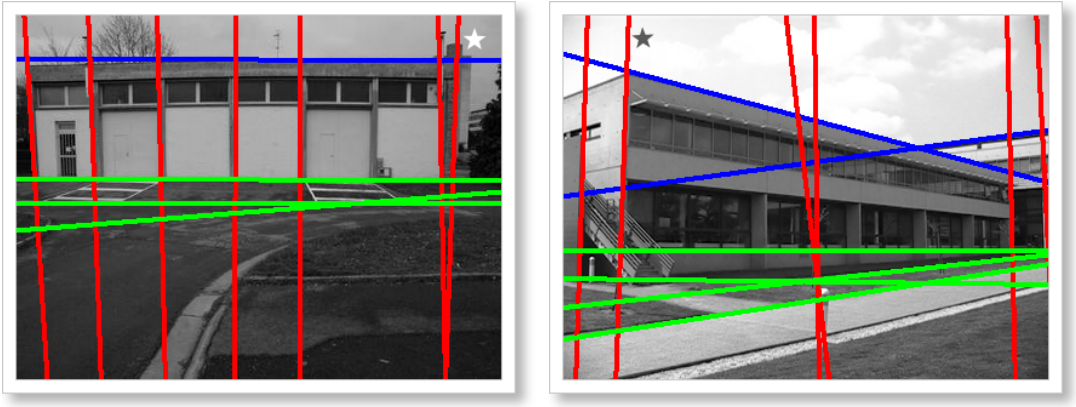
FIG. 5.5 – Estimation du contexte géométrique

est disponible en ligne[i]. Cet algorithme estime, grâce à des techniques d'apprentissage statistique, l'orientation approximative de larges surfaces contenues dans des scènes extérieures. Chaque pixel est classifié comme appartenant soit au sol, soit a une surface verticale, soit au ciel. De plus, chaque surface verticale est subdivisée en trois surfaces planes selon leur orientation par rapport à la caméra (vers la droite, face à la caméra, ou vers la gauche), et en deux surfaces non planes : poreuse (e.g. la végétation), ou solide (e.g. un tronc d'arbre ou une personne). La figure 5.5 montre les résultats de l'extraction d'un tel contexte. Le sol est représenté en vert, les surfaces verticales en rouge, et le ciel bleu. Les surfaces planes sont symbolisées par des flèches, les surfaces poreuses par des ronds, et les surfaces solides par des croix.

Nous utilisons cette classification pour ne conserver que les lignes horizontales les plus proches des frontières entre les zones *ciel* et *sol*. Nous leur attribuons alors le label correspondant (*ciel* ou *sol*).
Par contre, la subdivision en zones verticales est approximative et ne reflète pas complètement les limites verticales des bâtiments dans l'image. C'est pourquoi, au lieu d'éliminer certaines lignes verticales comme nous l'avons fait pour les lignes horizontales, il est préférable de les conserver toutes et d'attribuer une pondération plus importante à celles qui sont proches des frontières verticales extraites du contexte géométrique. De plus, toujours dans le but de limiter par la suite les mauvaises correspondances pour le calcul de la pose, nous attribuons aux lignes verticales le label *verticale* pour les différencier des autres lignes de l'image.

La figure 5.6 montre l'ensemble des lignes conservées et labélisées. Les lignes proches du ciel sont en bleu, celles proches du sol sont en vert, et les lignes verticales sont en

[i] `http://cs.cmu.edu/~dhoiem/projects/context/index.html`



(a) *Beaulieu 1* : 11 Lignes conservées et labélisées (b) *Ifsic* : 12 Lignes conservées et labélisées

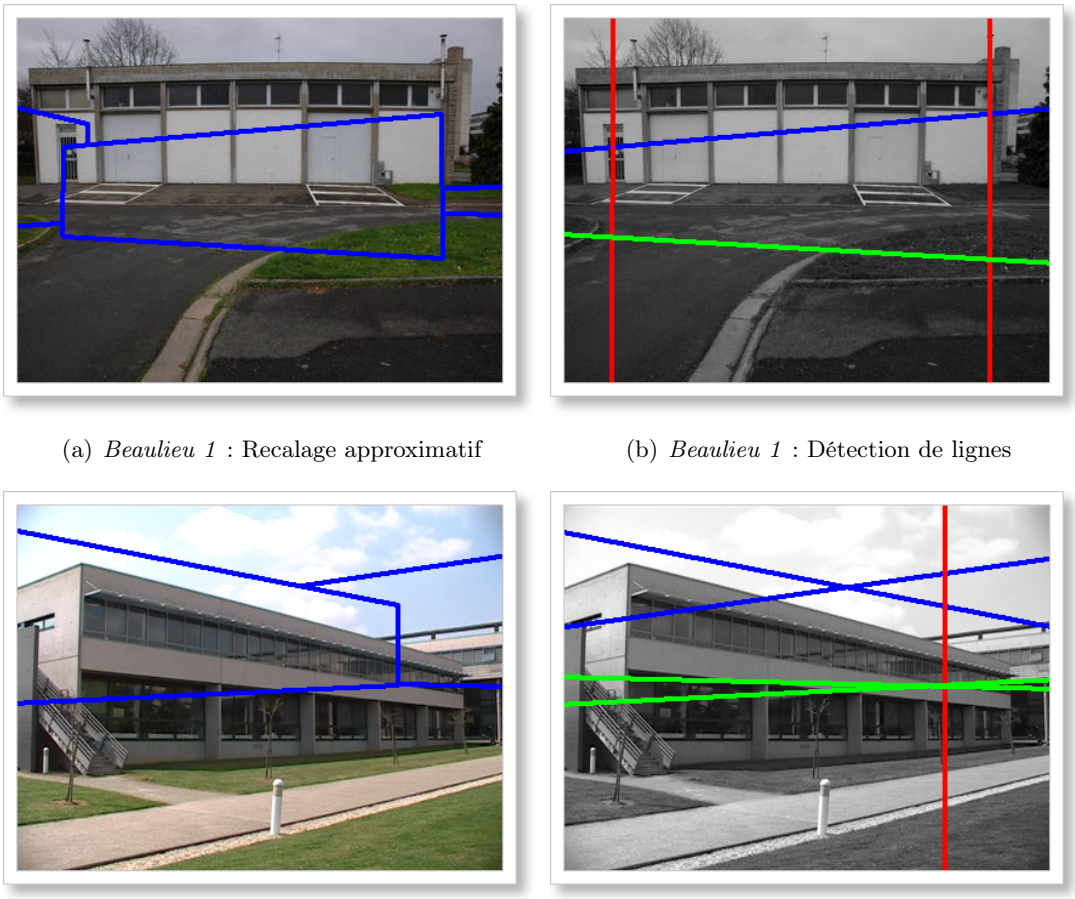
FIG. 5.6 – Labélisation des lignes extraites en fonction du contexte géométrique. En bleu : les lignes correspondant aux frontières entre bâtiments et ciel. En vert : frontières entre bâtiments et sol. En rouge : lignes verticales.

rouge. Une étoile signale les lignes verticales qui ont une pondération plus importantes que les autres lignes verticales.
Sur ces images, on peut voir que la frontière ciel/bâtiment est très précise ce qui entraîne une bonne détection des arêtes supérieures des bâtiments représentées par les lignes bleues. La frontière sol/bâtiment est moins précise mais permet néanmoins de limiter le nombre de lignes pouvant être une arête inférieure de bâtiment. Enfin, toutes les lignes verticales sont conservées et le contexte géométrique a permis d'attribuer une pondération plus forte à une seule de ces lignes, or cette ligne correspond bien à une arête verticale de bâtiment.

5.4.1.2 Lignes extraites du modèle

Dans le modèle 3D issu de la base SIG, les lignes sont détectées en projetant les coins des bâtiments dans le plan image. L'équation des droites est alors obtenue en utilisant les couples de coins projetés correspondants.

Du fait de l'alignement approximatif obtenu lors de la première partie, il est possible que certaines lignes du modèle ne soient pas présentes dans l'image, ce qui veut dire qu'aucune correspondance ne peut être trouvée pour ces lignes. De plus, pour éviter de prendre en compte des façades visibles dans le modèle projeté mais pas dans l'image, nous ne considérons que les façades dont l'aire de la projection dans l'image est suffisamment importante. Ainsi, les petites parties de façades qui apparaissent à cause du mauvais alignement initial sont éliminées. Le seuil choisi ici empiriquement est fonction de la taille de l'image (Aire($Image$)/25). De plus, nous ne considérons pas les lignes qui sont cachées par d'autres façades. En revanche, nous conservons les lignes qui, une fois projetées, sortent du cadre de l'image car ces lignes peuvent apparaître lors de la



(a) *Beaulieu 1* : Recalage approximatif

(b) *Beaulieu 1* : Détection de lignes

(c) *Ifsic* : Recalage approximatif

(d) *Ifsic* : Détection de lignes

FIG. 5.7 – Détection des lignes du modèle 3D. Colonne de gauche : projection du modèle 3D suivant la pose approximative estimée. Colonne de droite : labélisation des lignes de ce modèle projeté.

correction de la pose. Enfin, comme pour la détection des lignes de l'image, chaque ligne détectée à partir du modèle 3D reçoit un label en fonction de sa position sur la façade (*ciel*, *verticale* ou *sol*).

La figure 5.7 donne les résultats de cette détection. On peut y voir dans la colonne de gauche le recalage approximatif obtenu lors la première partie (section 5.3), et dans la colonne de droite les lignes détectées. La couleur des lignes correspond à leur label. On remarque dans les images 5.7(a) et 5.7(b) que la façade du modèle 3D qui apparaît sur la gauche n'est pas prise en compte car son aire est trop faible. De plus, dans les images 5.7(c) et 5.7(d), la ligne verticale de la façade de droite située en arrière plan n'est pas non plus prise en compte ce qui respecte les contraintes que nous avons imposé.

5.4.2 Mise en correspondance et calcul de pose

5.4.2.1 Mise en correspondance

La mise en correspondance est effectuée directement entre les lignes de même label. Grâce à la réduction du nombre de lignes présentées précédemment, le nombre de correspondances pour les lignes du ciel et pour celles du sol est assez faible. En revanche, pour limiter le nombre de correspondances verticales, nous appliquons un seuil d'erreur sur le paramètres ρ. Considérons :

- $l_{mod} = (\rho_{mod}, \theta_{mod})$
- $l_{img} = (\rho_{img}, \theta_{img})$
- s_ρ le seuil sur ρ

Une paire de lignes verticales sera ajoutée dans l'ensemble des correspondances si :

$$|\rho_{mod} - \rho_{img}| < s_\rho \tag{5.13}$$

Nous avons choisi $s_\rho = 0.75 * \text{Largeur}(Image)$. Ce seuil est très large pour éliminer les correspondances improbables entre lignes de part et d'autre de l'image.

De plus, les correspondances sont pondérées différemment selon leur label. Ainsi, la frontière ciel/bâtiment étant généralement bien nette, les correspondances labélisées *ciel* ont une pondération plus importante que celles labélisées *verticale* ou *sol*. De même, les lignes verticales détectées grâce au contexte géométrique sont plus fiables que les lignes verticales simples. Enfin, la frontière sol/bâtiment étant difficile à détecter précisément (à cause de l'ombre du bâtiment ou d'un véhicule garé devant par exemple), les correspondances labélisées *sol* ont une pondération plus faible, tout comme les lignes verticales simples qui sont en nombre important. Les pondérations adoptées sont les suivantes :

- *ciel* : 3
- *verticale* (avec contexte géométrique) : 2
- *verticale* (simple) : 1
- *sol* : 1

Ces pondérations seront utiles lors du calcul de pose, et plus particulièrement dans la phase de validation de la pose calculée pour chaque itération de RANSAC. Le tableau 5.1 précise le nombre total de correspondances trouvées pour les images *Ifsic* (image clé utilisée : $\kappa = 70$) et *Beaulieu 1* ($\kappa = 39$), ainsi que le nombre de correspondances par label. La dernière ligne du tableau donne le pourcentage d'*outliers* contenu dans l'ensemble de correspondances. On remarque que malgré la réduction du nombre de lignes, il y a encore jusqu'à 75% d'*outliers* dans l'ensemble de correspondances.

5.4.2.2 Calcul de pose

Nous avons vu que la présence d'*outliers* dans l'ensemble de correspondance est quasiment inévitable. Or, pour que le calcul de pose soit précis, tous ces *outliers* doivent être

Nombre de correspondances	*Ifsic*	*Beaulieu 1*
ciel	4	1
verticale (contexte géométrique)	2	2
verticale	5	12
sol	8	3
Total	19	16
% *outliers*	75%	75%

TAB. 5.1 – Nombre de correspondances en fonction du contexte géométrique

détectés et éliminés. Pour cela, nous utilisons un algorithme de calcul de pose basé sur la méthode d'estimation robuste RANSAC (chapitre 4).

Sélection aléatoire des correspondances Nous souhaitons que le sous-ensemble de correspondances sélectionnées ne donne pas de configuration dégénérée, c'est-à-dire une configuration qui n'apporte pas suffisamment d'information pour le calcul précis de la pose. Par exemple, un sous-ensemble ne contenant que des lignes verticales est dégénéré. Ainsi, nous contraignons chaque sous-ensemble généré aléatoirement à contenir au moins deux labels différents. Cette vérification permet d'éviter des calculs inutiles et permet donc de converger plus rapidement vers la solution.

Calcul de pose par asservissement La méthode d'asservissement visuel virtuel présentée dans le chapitre 4 est utilisée pour le calcul de la pose de la caméra. Pour résoudre le système à six inconnues de l'équation 4.20, six équations indépendantes sont nécessaires, ce qui revient à dire que la matrice d'interaction $\mathbf{L_s}$ doit être de rang 6. Etant donné qu'une ligne fournit deux équations indépendantes, nous en déduisons que trois lignes placées dans une "configuration générale" donnent six équations indépendantes et donc une matrice $\mathbf{L_s}$ de rang plein.
Il existe des configurations particulières pour lesquelles 3 lignes ne donnent pas 6 équations indépendantes. C'est le cas par exemple lorsque ces 3 lignes sont parallèles. Pour éviter ces cas particuliers, il suffit de calculer le rang de la matrice d'interaction et de s'assurer qu'il est bien de rang égal à 6.
Il faut donc en théorie un minimum de 3 correspondances de lignes. Néanmoins, en pratique, l'alignement précis de 3 correspondances de lignes ne permet pas d'aligner correctement la totalité du modèle 3D sur les bâtiments de l'image. Cette imprécision est due au bruit présent dans les lignes détectées et elle rend difficile la recherche d'*inliers* avec l'algorithme de RANSAC. La figure 5.8 montre cette imprécision, on voit que les trois lignes utilisées pour le calcul de pose sont alignées avec le bâtiment mais que la quatrième ligne est assez éloignées de l'objectif réel. C'est pourquoi nous utilisons un minimum de 4 correspondances de lignes lors de la sélection aléatoire de correspondances avec RANSAC (paramètre $n = 4$).

FIG. 5.8 – Alignement peu précis en utilisant seulement trois lignes : le côté gauche du bâtiment dans l'image n'est pas correctement aligné avec le modèle 3D correspondant.

Test de la pose pour chaque sélection A partir de la nouvelle pose obtenue, nous classifions les correspondances en *inliers* ou *outliers*, et conservons l'ensemble d'*inliers* si celui-ci est le meilleur ensemble trouvé jusqu'à présent. Pour cela, on calcule l'erreur de projection pour toutes les correspondances. Si l'erreur de projection d'une correspondance est telle que :

$$|\rho_{m3d} - \rho_{img}| < s_{\rho_{min}} \quad \text{et} \quad |\theta_{m3d} - \theta_{img}| < s_{\theta_{min}} \ , \tag{5.14}$$

alors cette correspondance est un *inlier*, sinon, c'est un *outlier*. Les seuils $(s_{\rho_{min}}, s_{\theta_{min}})$ sont fixés au double de l'écart-type sur les erreurs de mesure.

A chaque *inlier*, on associe une pondération qui dépend de son label et de son erreur. On obtient ainsi un ensemble d'*inliers* dont la qualité est mesurée par sa taille, sa pondération totale et son erreur totale. Enfin, pour valider ou non cet ensemble d'*inliers*, on le compare avec le meilleur ensemble estimé jusqu'à présent en commençant par comparer leur taille, puis leur pondération, et enfin leur erreur totale de projection. Au final, seul le meilleur ensemble d'*inliers* est conservé.

5.4.3 Résultats

Nous présentons à présent les résultats obtenus après le recalage précis du modèle 3D. Les données fournies en entrées sont :

- La première image de la séquence vidéo.
- La pose approximative du modèle 3D obtenue à l'issue de la première étape (section 5.3).
- Les données du modèle 3D.

Observations Sur les figures 5.9 et 5.10, on peut voir sur la première ligne les différents exemples de recalage approximatif, et sur la deuxième le recalage final. Les tableaux 5.2

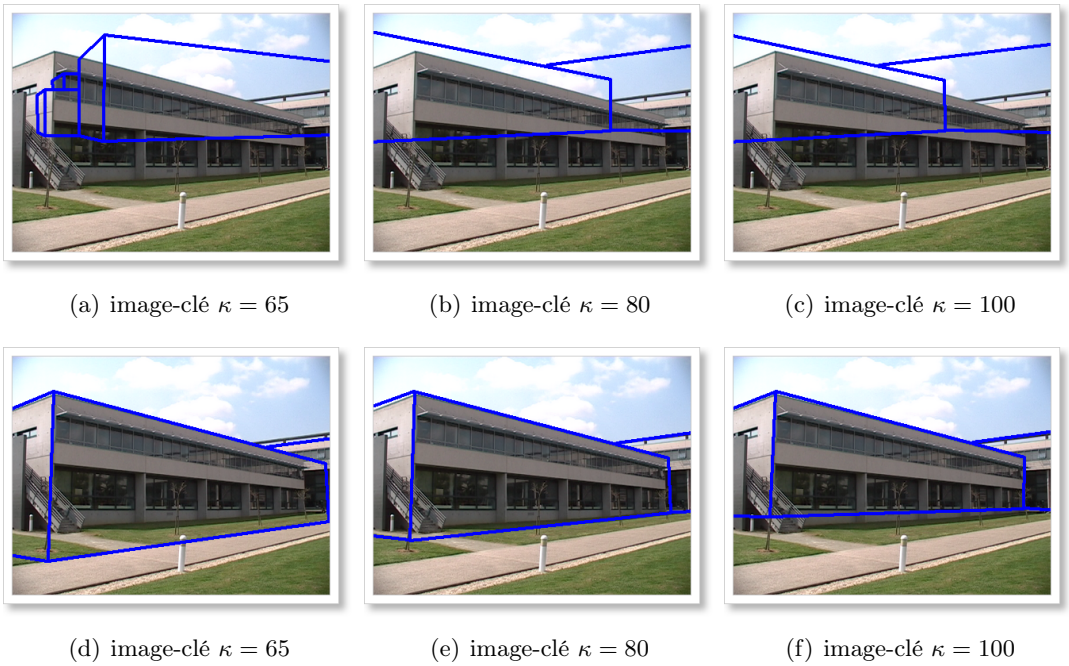
	Itérations RANSAC	Nb. inliers	Pondération	Erreur
image-clé $\kappa = 65$	512	4	7	0.00001
image-clé $\kappa = 80$	958	5	10	0.0001
image-clé $\kappa = 100$	1177	5	10	0.0005

TAB. 5.2 – Performances *Ifsic* : nombre d'itérations, d'*inliers*, pondération totale et erreur totale en fonction du choix de l'image-clé κ utilsée.

	Itérations RANSAC	Nb. inliers	Pondération	Erreur
image-clé $\kappa = 20$	908	4	7	0.0002
image-clé $\kappa = 30$	511	4	7	0.0002
image-clé $\kappa = 39$	1176	4	7	0.0002

TAB. 5.3 – Performances *Beaulieu 1* : nombre d'itérations, d'*inliers*, pondération totale et erreur totale en fonction du choix de l'image-clé κ utilsée.

et 5.3 donnent quand à eux les performances obtenues en terme d'itérations de RANSAC et de qualité d'alignement avec le nombre d'*inliers*, leur pondération et leur erreur.
Dans les trois images de la figure 5.9, le recalage du modèle est assez proche de celui recherché, néanmoins aucun de ces trois exemples ne donne un alignement parfait du modèle sur le bâtiment dans l'image. Cela vient du fait que les lignes détectées au niveau du sol ne correspondent pas à l'arête inférieure du bâtiment mais sont suffisamment proches de la frontière sol/bâtiment pour être considérées à tort comme *inliers*. La faiblesse vient donc ici de la difficulté à détecter l'arête inférieure du bâtiment qui se trouve dans l'ombre et qui est occultée par des arbres et un escalier.
Dans les images (a) et (b), le recalage approximatif est tel qu'une seule façade est prise en compte pour le recalage (celle dont l'aire est suffisamment grande), il en résulte que les 4 arêtes de la façade doivent être détectées pour permettre un alignement correcte, or ce n'est pas le cas car, comme nous l'avons déjà précisé, les lignes détectées au niveau du sol ne correspondent pas à l'arête inférieure de la façade. En conséquence, parmi les 4 *inliers* trouvés, 2 d'entre eux sont mauvais et cela malgré le faible seuil utilisé pour classifié les *inliers*.
A l'inverse, dans les images (c), (d), (e), (f), les deux façades présentent une aire suffisamment grande pour être prise en compte. L'algorithme trouve ainsi 5 *inliers* parmi lesquels celui correspondant au sol est mauvais.
Les images de la figure 5.10 montrent de meilleurs résultats que la figure précédente. En effet, l'alignement du modèle est quasiment parfait et l'erreur restante vient seulement du bruit de détection des lignes dans l'image (principalement dû à la distorsion radiale de la caméra). Dans ces exemples, une seule façade est prise en compte, mais cela est suffisant car les 4 arêtes de cette façade ont été détectées dans l'image.
En ce qui concerne les temps d'exécution, Le nombre d'itérations maximum est de 1177 ce qui correspond à un temps d'exécution d'une dizaine de secondes avec un Pentium 4 à 3GHz. De plus, l'algorithme complet nécessite au maximum 1 minute, il n'est donc pas destiné pour l'instant à des applications temps réel.



(a) image-clé $\kappa = 65$ (b) image-clé $\kappa = 80$ (c) image-clé $\kappa = 100$

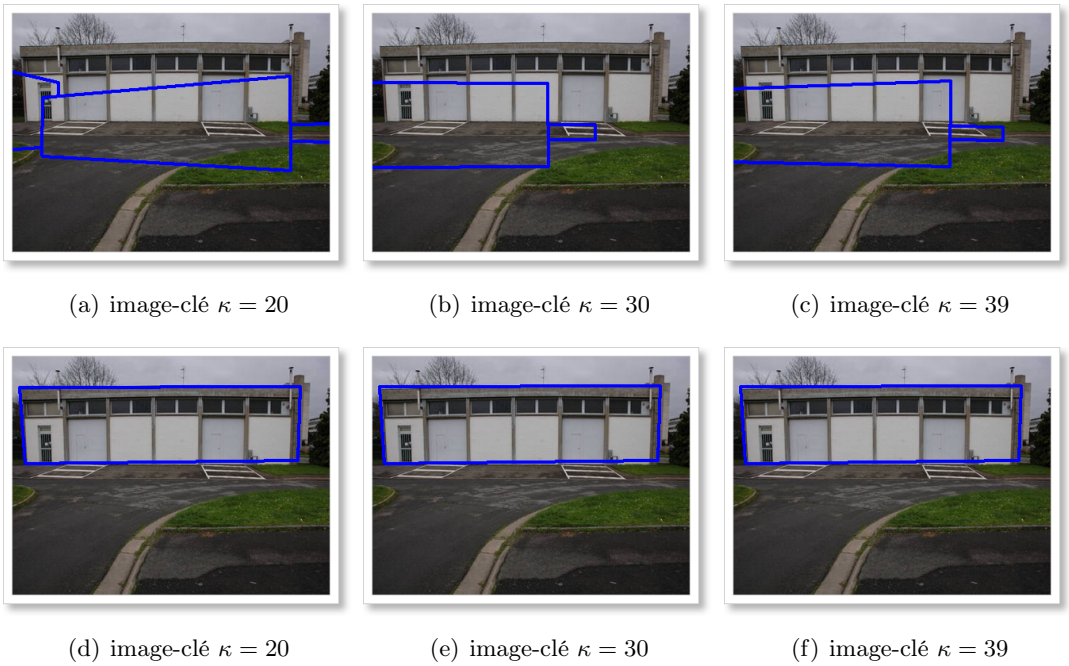
(d) image-clé $\kappa = 65$ (e) image-clé $\kappa = 80$ (f) image-clé $\kappa = 100$

FIG. 5.9 – *Ifsic* - Première ligne : recalage approximatif en fonction de l'image-clé κ utilisée. Deuxième ligne : recalage final correspondant.

Interprétation Un recalage précis a pu être atteint pour l'exemple de la séquence *Beaulieu 1*, mais pas pour la séquence *Ifsic*, bien que l'alignement du modèle soit proche de celui désiré, une faible erreur est toujours présente. La principale condition à la bonne réalisation du recalage précis est la détection dans l'image des arêtes des bâtiments (au moins 4). Ensuite, si cette condition est réalisée alors les chances de trouver la bonne solution sont élevées, même en présence de nombreux *outliers*.
La deuxième condition importante est bien sûr que le recalage approximatif fourni en entrée soit suffisamment proche de l'objectif pour permettre la mise en correspondance. Au vu des résultats obtenus avec les séquences *Ifsic* et *Beaulieu 1*, cette condition peut être facilement respectée.

5.5 Conclusion

Nous avons présenté une méthodologie permettant le recalage initial automatique d'un modèle SIG avec une vidéo. La solution proposée se décompose en deux parties qui sont, premièrement, l'estimation approximative de la pose du modèle en utilisant deux images et les données GPS, et deuxièmement, l'estimation précise de la pose par la mise en correspondance des lignes du modèle avec les lignes de l'image. Les résultats ont montré que l'estimation approximative de la pose permet bien de superposer les champs de vue image et modèle SIG projeté dans l'optique d'établir une mise en correspondance de



(a) image-clé $\kappa = 20$ (b) image-clé $\kappa = 30$ (c) image-clé $\kappa = 39$

(d) image-clé $\kappa = 20$ (e) image-clé $\kappa = 30$ (f) image-clé $\kappa = 39$

FIG. 5.10 – *Beaulieu 1* - Première ligne : recalage approximatif en fonction de l'image-clé κ utilisée. Deuxième ligne : recalage final correspondant.

lignes. Ensuite, le recalage précis du modèle est obtenu en combinant le calcul de pose et la recherche de correspondances grâce à l'algorithme RANSAC. La méthode proposée ne fonctionne pas pour n'importe quelle image de bâtiment (couplée avec des mesures GPS), en effet il faut qu'au moins quatre arêtes de bâtiment soient présentes dans la première image et il faut que ces arêtes soient relativement facile à détectées (peu d'occlusions, présence du ciel dans l'image, bon éclairage). Cette méthode possède aussi l'avantage de fournir un ensemble de correspondances de lignes 2D-3D entre le modèle SIG et l'image. Ces correspondances peuvent être utilisées pour le tracking du modèle dans la séquence vidéo.

Quelques améliorations devront encore être apportées afin que l'initialisation soit totalement automatique. Dans la première partie, nous avons vu que le choix de l'image κ doit prendre en compte le mouvement de la caméra et le nombre de correspondances disponibles entre les deux images. Ce choix peut être effectué automatiquement en suivant la méthode proposée par Galpin [Gal02]. Un choix plus pertinent de cette image-clé devrait par ailleurs permettre de lever l'hypothèse sur la rotation autour de l'axe vertical de la caméra et estimer le dernier degré de liberté restant. Enfin, nous envisageons d'améliorer la mise en correspondance 2D-3D en utilisant des techniques de *Structure from Motion* pour extraire des informations 3D sur les primitives présentes dans les images.

Chapitre 6

Suivi du recalage

If the facts don't fit the theory, change the facts - **Albert Einstein**

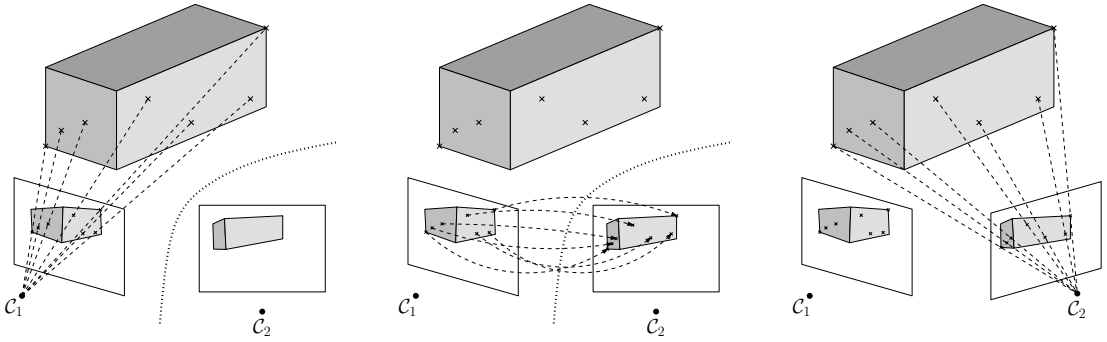
6.1 Introduction

Nous avons vu dans le chapitre 5 une méthode permettant d'obtenir la pose géo-référencée de la première image d'une séquence. Or nous voulons calculer une telle pose pour toutes les images en entrée. Il est peu judicieux d'utiliser la méthode décrite précédemment pour toutes les images. En effet, celle-ci s'est avérée nécessaire car les seules données GPS fournissent une initialisation de la pose bien trop éloignée de la solution. Puisque pour une séquence d'images donnée on suppose de faibles déplacements inter-images, et que le modèle SIG se superpose correctement avec la première image, le calcul de la pose pour les images restantes peut se restreindre à un problème de suivi de ce recalage.

Plusieurs approches ont été envisagées pour effectuer un tel suivi. Nous avons retenu une méthode basée sur l'asservissement visuel virtuel décrit dans le chapitre 4. Les conditions d'utilisation de la méthode étant particulières (modèles 3D très approximatifs, partiellement visibles, etc.), nous proposons une variation dite *robuste* de l'algorithme, adaptée à notre contexte d'étude. Des résultats de suivi sur des séquences complètes sont alors présentés en fin de chapitre.

6.2 Principe du suivi par asservissement visuel virtuel

Le calcul de pose par asservissement visuel virtuel nécessite un ensemble de correspondances entre des primitives 3D appartenant au modèle à recaler, et la projection 2D de ces primitives dans les images. De par la simplicité de leur modélisation, de leur extraction et de leur suivi, nous utilisons des points pour cette mise en correspondance de primitives. Pour assurer le suivi du modèle 3D tout au long de la séquence d'images, nous utilisons un schéma de transfert de points d'une image à l'autre.



Étape 1 : extraction de correspondances 2D-3D dans l'image recalée

Étape 2 : suivi des points 2D dans l'image non recalée

Étape 3 : transfert des correspondances et calcul de pose pour l'image non recalée

FIG. 6.1 – Les principales étapes du suivi de recalage par transfert de points

6.2.1 Schéma de transfert de points

Soit I_1 la première image pour laquelle le recalage avec le modèle est connu, c'est-à-dire pour laquelle on connaît la pose de la caméra dans le repère SIG. I_2, l'image suivante dans la séquence, est celle pour laquelle on veut retrouver la pose de la caméra. On doit donc pour cela retrouver des correspondances 2D-3D dans l'image I_2. La méthode utilisée, et illustrée sur la figure 6.1, consiste en un transfert de correspondances (ou de points dans notre cas) d'une image à l'autre. Elle est décomposable en trois principales étapes que nous détaillons ci-après.

6.2.1.1 Étape 1 : construction de correspondances 2D-3D

La pose de la caméra étant connue pour l'image I_1, on peut construire des correspondances 2D-3D aisément. Lors de cette première étape des points d'intérêt $\mathbf{x}^1$ sont détectés dans I_1. Nous utilisons pour cela le détecteur utilisé dans le KLT (Kanade, Lucas, Tomasi [i] [TK91][ST94]).
Pour calculer les positions 3D correspondantes $\mathbf{X}$ de ces points, une première solution consiste à calculer, dans le repère SIG $\mathcal{R}_{utm}$, les intersections entre le modèle 3D visualisé et les lignes de vue passant par le centre de la caméra $\mathcal{C}_1$ et les point 2D projetés $\mathbf{x}^1$ sur le plan image par lancer de rayons. Avec un grand nombre de points, cette solution peut s'avérer coûteuse en terme de temps de calcul. Nous avons donc opté pour une méthode basée sur le *z-buffering*. Pour afficher la projection du modèle 3D, le moteur OpenGL calcule des informations de profondeur dans le repère caméra $\mathcal{R}_{\mathcal{C}_1}$, pour tous les pixels de l'image (principe du z-buffer). Connaissant la position de la caméra $\mathcal{C}_1$ par rapport au modèle, la distance de chaque pixel projeté à la caméra (donnée par le z-buffer), et la matrice de projection de la caméra, on connaît potentiellement pour chaque pixel du

[i] http ://www.ces.clemson.edu/∼stb/klt/

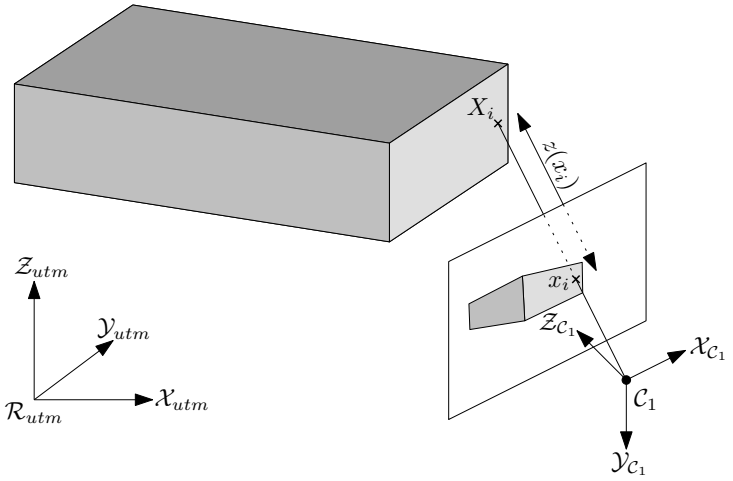


FIG. 6.2 – Le calcul de X_i se déduit de son correspondant 2D x_i, de la position $\mathcal{C}_1$ de la caméra ainsi que de la distance $z(x_i)$ lue dans le z-buffer

projeté du modèle 3D sa position correspondante dans le repère $\mathcal{R}_{utm}$[ii](voir figure 6.2). La détection des points 2D se faisant indépendamment du modèle, on peut avoir des points qui n'ont pas de correspondant 3D, puisque le modèle ne se projette pas forcément sur l'image entière. Pour faire une segmentation des points appartenant ou non au modèle, une simple lecture de la valeur correspondante dans le z-buffer suffit. Si pour un pixel x_i la valeur de z-buffer $z(x_i)$ correspondante est nulle, alors aucune partie du modèle 3D ne se projette en ce point.
On a donc construit à ce point des correspondances 2D-3D pour l'image I_1, *qui est déjà recalée avec le modèle*. L'avantage du z-buffering par rapport à un lancer de rayons est que le calcul des profondeurs des points est à la fois plus simple et plus rapide.

6.2.1.2 Étape 2 : suivi de points d'intérêt

La deuxième étape consiste à transférer ces correspondances à l'image I_2. Les points 2D $\mathbf{x}^1$ détectés précédemment sont suivis à l'aide de l'algorithme de suivi de points également implanté dans le KLT, pour trouver leur position correspondance $\mathbf{x}^2$ dans l'image I_2.
Dans un cadre plus général, l'algorithme du KLT fonctionne de la façon suivante pour suivre les points $\mathbf{x}$ dans les images $I_k, k \in \{1..N\}$:

- Entre I_k et I_{k+1}, les correspondances de points sont trouvées en minimisant l'erreur quadratique moyenne entre les fenêtres $\mathcal{W}_i$ autour des points x_i, en utilisant un

[ii]Concrètement, nous utilisons la fonction `gluUnproject` d'OpenGL qui inverse la matrice de projection en x_i pour calculer la ligne de vue correspondante, et utilise la valeur du z-buffer $z(x_i)$ pour retrouver la position X_i sur cette ligne

modèle de mouvement translationnel (2 paramètres (Δ_x, Δ_y) à estimer).

$$I_{k+1}\left(\left[\begin{array}{c} x_{x_i^{k+1}} \\ y_{x_i^{k+1}} \end{array}\right]\right) = I_k\left(\left[\begin{array}{c} x_{x_i^k} + \Delta_x \\ y_{x_i^k} + \Delta_y \end{array}\right]\right)$$

$$\left[\begin{array}{c} \Delta_x \\ \Delta_y \end{array}\right] = \underset{\delta_x,\delta_y}{argmin}\left(\frac{1}{|\mathcal{W}|}\sum_{\mathcal{W}}\sqrt{\left(I_{k+1}\left(\left[\begin{array}{c} x_{x_i^k} + \delta_x \\ y_{x_i^k} + \delta_y \end{array}\right]\right) - I_k\left(\left[\begin{array}{c} x_{x_i^k} \\ y_{x_i^k} \end{array}\right]\right)\right)^2}\right)$$

- Les correspondants $(\mathbf{x}^k, \mathbf{x}^{k+1})$ étant trouvés, une mesure de dissimilarité d est calculée entre $\mathbf{x}^1$ et $\mathbf{x}^{k+1}$. Au delà d'un certain seuil τ de dissimilarité, une correspondance est considérée comme fausse et est abandonnée.

  $$d(x_i^1, x_i^{k+1}) > \tau \Rightarrow \text{abandon de la correspondance } (x_i^1, x_i^{k+1})$$

  Cette dissimilarité d est mesurée en utilisant non plus un modèle translationnel (deux paramètres), mais un modèle affine (six paramètres).

  $$I_{k+1}\left(\left[\begin{array}{c} x_{x_i^{k+1}} \\ y_{x_i^{k+1}} \end{array}\right]\right) = I_k\left(\mathbf{A}\left[\begin{array}{c} x_{x_i^k} \\ y_{x_i^k} \end{array}\right] + \left[\begin{array}{c} \Delta_x \\ \Delta_y \end{array}\right]\right)$$

  $\mathbf{A}$ étant une matrice de déformation 2×2.

6.2.1.3 Étape 3 : calcul de pose

Ayant des correspondances entre $\mathbf{x}^1$ et $\mathbf{X}$, et entre $\mathbf{x}^1$ et $\mathbf{x}^2$, on peut alors en déduire des correspondances entre $\mathbf{x}^2$ et $\mathbf{X}$, qui sont utilisées en entrée de l'algorithme d'asservissement visuel virtuel pour calculer la pose de l'image I_2.

6.2.1.4 Images suivantes

Pour les images I_k $(k > 2)$, le recalage s'effectue de façon similaire, en se passant toutefois de l'étape 1. En effet la construction des correspondances $(\mathbf{x}^{k-1}, \mathbf{X})$ étant déjà définie, il ne reste plus qu'à suivre les points 2D (étape 2) et à calculer la pose correspondante (étape 3).

6.2.1.5 Perte de points

Lors de l'étape de suivi des points $\mathbf{x}^k$ d'une image à l'autre, certains points sont perdus. Ces pertes sont principalement dues à la disparition de certaines zones de l'image du fait du mouvement de la caméra, ainsi qu'au bruit ou spécularités qui empêchent de retrouver le correspondant de certains points. Ces pertes peuvent devenir critiques lorsqu'il n'y a plus assez de points pour construire le nombre minimum de correspondances 2D-3D (*i.e.* 4) pour calculer la pose par asservissement.

Pour pallier ce problème, nous fixons un seuil $s_{\mathbf{x}}$ au delà duquel on estime ne plus avoir

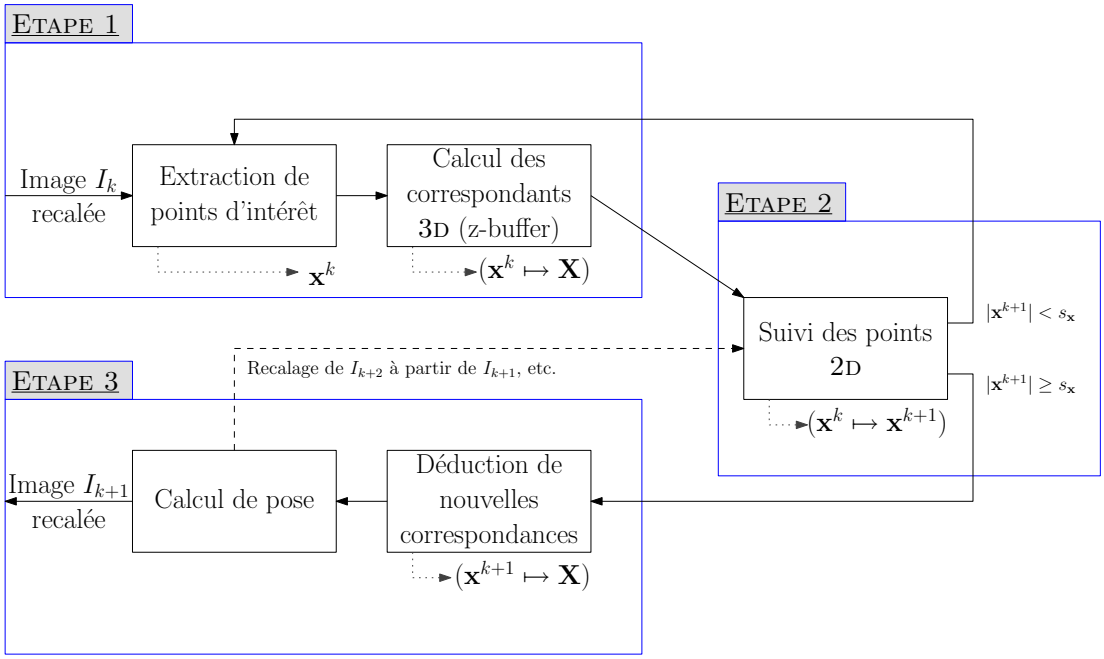


FIG. 6.3 – Résumé complet de la procédure de suivi de modèle par transfert de points

assez de points $\mathbf{x}^k$ pour calculer la pose de manière fiable. Dans ce cas de nouvelles correspondances 2D-3D doivent être générées à partir de nouveaux points 2D. Pour plus de clarté, nous pouvons illustrer ce cas particulier par l'exemple suivant :

1. La pose vient d'être calculée avec succès pour l'image I_k.
2. Les points $\mathbf{x}^k$ sont suivis dans I_{k+1}, et on a $|\mathbf{x}^{k+1}| < |\mathbf{x}^k|$ ($|\mathbf{x}|$ dénotant ici le cardinal de l'ensemble $\mathbf{x}$) du fait de points perdus lors du suivi.
3. $|\mathbf{x}^{k+1}| < s_{\mathbf{x}}$: on n'a plus assez de points.
4. On retourne à l'étape 1 de l'algorithme, en se basant sur la dernière image dont la pose est connue, c'est-à-dire I_k. De nouveaux points 2D $\mathbf{x}'^k$ sont extraits avec le KLT. On garde cependant les anciens points $\mathbf{x}^k$. On contraint les points nouvellement extraits à être suffisamment éloignés (dans l'image) des anciens, puis on calcule leurs correspondants 3D.
5. Les points $\mathbf{x}'^k$ sont suivis dans I_{k+1} à nouveau, puis la pose est calculée pour cette image.

L'ensemble de la procédure de suivi par transfert de points est résumée sur la figure 6.3.

6.2.2 Résultats de suivi

L'algorithme de suivi présenté dans la section 6.2.1 a été appliqué à plusieurs séquences réelles. Nous en présentons trois, ces séquences étant à notre avis représentatives des

FIG. 6.4 – Résultats de suivi pour la séquence *Beaulieu 1*

situations réelles les plus courantes. Une illustration des résultats est par ailleurs fournie sur les figures 6.4, 6.5 et 6.6.

Séquence *Beaulieu 1* (figure 6.4) La vidéo représente la visualisation d'une façade quasi-plane d'un petit bâtiment. La façade est visible dans toutes les images, et aucun obstacle non modélisé dans la base SIG (voiture, poteau, etc.) n'est visible. Dans ce cas presque idéal, on observe que notre méthode suit avec succès le bâtiment dans toutes les images.

Séquence *Ifsic* (figure 6.5) Plusieurs bâtiments sont visualisés, et le mouvement de la caméra est générique : aucune façade n'est visée explicitement à des vues de reconstruction. Le bâtiment visible dans la première image disparaît au cours de la séquence pour réapparaître à la fin. Le mouvement de la caméra étant très bruité, et un grand nombre de spécularités étant présentes, de nombreux points sont perdus lors du suivi 2D. Comme pour la séquence *Beaulieu 1*, peu d'objets non modélisés viennent perturber le suivi.
On observe que, bien que globalement cohérent malgré la disparition puis la réapparition du bâtiment principalement visible, une dérive dans le suivi apparaît. A la fin de la séquence, le modèle SIG n'est plus recalé correctement avec les images correspondantes. La pose n'étant pas suffisamment précise, l'extraction des textures et de la micro-structure des façades ne pourra pas être effectuée.

Séquence *Beaulieu 2* (figure 6.6) La caméra effectue un mouvement parallèle à la façade visée, comme pour la séquence *Beaulieu 1*. La principale différence vient de la

FIG. 6.5 – Résultats de suivi pour la séquence *Ifsic*

voiture garée devant le bâtiment, qui n'est pas modélisée dans la base SIG. Des points d'intérêt y sont détectés, et leur profondeur mal estimée puisque l'on fait l'hypothèse que ces points se situent sur la façade. Il en résulte un ensemble de fausses correspondances 2D-3D, qui font que le suivi de la pose diverge complètement : il échoue à suivre le modèle dans la séquence.

6.2.3 Interprétation

Les résultats présentés précédemment valident la faisabilité de la méthode proposée, mais montrent également les limites de l'application simple d'un algorithme d'asservissement visuel virtuel au suivi de façades de bâtiments. Cet échec est dû à la combinaison de plusieurs facteurs.

- Spécularités (séquences *Ifsic* et *Beaulieu 2*). Dans un contexte de reconstruction urbaine, de nombreuses fenêtres sont présentes, impliquant de nombreuses zones de spécularité. Des erreurs sur le suivi des points 2D sont alors générées, et il sera difficile de s'en débarrasser.
- Coplanarité (toutes les séquences). Il est rare d'avoir plusieurs façades visualisées au même instant. De fait, les points 2D et 3D extraits sont généralement coplanaires. Bien que la pose puisse être calculée dans cette configuration (voir [ODD96]), le suivi est en général plus instable dans ce cas. Ce problème de coplanarité est illustré sur la figure 6.7.
- Occultations non modélisables (séquences *Ifsic* et *Beaulieu 2*). On appelle *occultations non modélisables* tous les objets non visibles dans la base SIG, comme les voitures, poteaux électriques, piétons, etc. Quand des points 2D sont détectés sur ces objets dans les images, les correspondances 3D calculées sont alors fausses car

FIG. 6.6 – Résultats de suivi pour la séquence *Beaulieu 2*

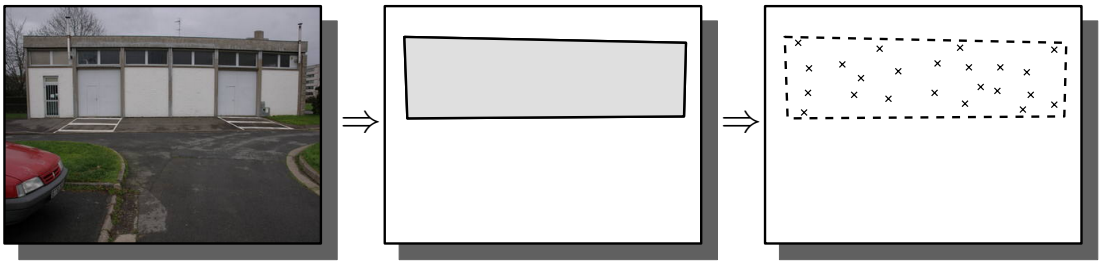


Image pour laquelle une configuration de points coplanaires apparaît.

Façade plane recalée avec l'image. Un seul plan est présent.

Tous les points 3D extraits se trouvent dans le même plan.

FIG. 6.7 – Configurations pour lesquelles des points coplanaires sont extraits

il leur est assigné la profondeur d'un points appartenant à la façade du modèle 3D. C'est la plus grande source d'erreur contribuant à l'échec du suivi du modèle.

6.3 Vers un suivi robuste

Les causes d'erreur entraînant un échec du suivi ayant été identifiées (section 6.2.3), nous présentons dans cette section les améliorations apportées à notre méthode. Un algorithme de suivi plus robuste aux erreurs est obtenu en modifiant les parties suivantes :

- Sélection des points d'intérêt 2D
- Calcul de la pose par asservissement

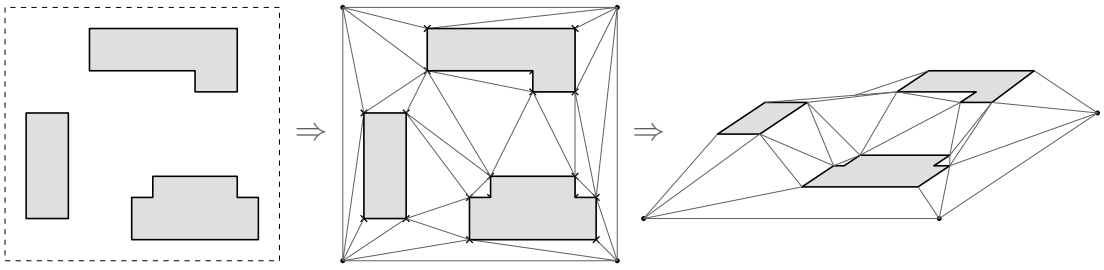


FIG. 6.8 – Modélisation du sol par triangulation de Delaunay

6.3.1 Extraction des points au sol

La base SIG ne contenant que les bâtiments, nous ne gardons pour les correspondances 2D-3D que les points 2D qui se projettent sur une façade (section 6.2.1.1). C'est ce qui entraîne l'extraction de points en général coplanaires. Si l'on fait l'hypothèse d'un sol localement plan près des bâtiments, on peut alors générer un modèle 3D de ce sol, par interpolation de tous les points de l'empreinte au sol des bâtiments.
Un tel modèle peut se calculer à l'aide d'une triangulation de Delaunay [Del34]. Un maillage 2D est tout d'abord estimé dans le plan $(X, Y)_{utm}$ en utilisant comme nœuds les points des empreintes au sol. Puis chaque nœud du maillage se voit assigner l'altitude du point auquel il correspond (voir figure 6.8).
Ce maillage nous permet de lever en partie le problème de coplanarité. En plus des points projetés sur une façade, on peut désormais utiliser les points qui se projettent sur le sol, puisque celui-ci est modélisé par un maillage triangulaire, tout comme les bâtiments. Les limitations de cette modélisation sont atteintes dans les cas suivants :

- Différence importante d'inclinaison entre le modèle du sol et le sol réel (beaucoup de fausses correspondances 3D)
- Utilisation pour le suivi de nouvelles zones où se situent de nouvelles occultations non modélisables (trottoirs, bordures, plantes, etc.)

6.3.2 Loi de commande robuste

La précision de la pose calculée par asservissement visuel virtuel est très sensible aux erreurs introduites par de fausses correspondances 2D-3D. Ces erreurs apparaissent non seulement en 2D (bruit dans les images, spécularités, etc.) mais aussi en 3D, à cause par exemple des occultations non modélisables. Cette sensibilité est due au fait que la loi de commande utilisée pour faire converger la pose (équation 4.20) prend en compte toutes les correspondances passées en entrée de manière équitable. Une loi de commande robuste a pour objectif de faire en sorte que les fausses correspondances faisant diverger l'estimation de la pose ne soient que peu ou pas prises en compte dans le calcul de pose.

L'introduction d'un M-estimateur ([Hub81] et section 4.6.2) dans la loi de commande contribue à résoudre ce problème [CMC06]. L'idée est d'associer un poids $w_i \in [0;1]$ à chaque correspondance $(\mathcal{P}(X_i), x_i)$, traduisant quantitativement sa contribution dans le calcul de la pose, en fonction de la confiance qui lui est accordée. Un *outlier* (une "fausse" correspondance) aura un poids très faible ou nul, contrairement à un *inlier* (une "vraie" correspondance). La loi de commande de l'équation 4.20 devient alors :

$$\mathbf{v} = -\lambda \left(\mathbf{DL}\right)^{+} \mathbf{D} \left(\mathcal{P}\left(\mathbf{X}\right) - \mathbf{x}\right), \tag{6.1}$$

avec $\mathbf{D} = diag(w_1, ..., w_N)$. Le calcul de la pose par minimisation aux moindres carrés de la distance entre les primitives 3D projetées $\mathcal{P}(\mathbf{X})$ et leur correspondant $\mathbf{x}$ dans l'image devient une minimisation aux moindres carrés pondérée.

Nous utilisons pour ρ la fonction de coût robuste de Cauchy présentée dans la section 4.6.2. [Com05b] propose de fixer le paramètre c (section 4.6.2) à la valeur de 2.3849. A chaque itération, les résidus $r_i = \mathcal{P}(\mathbf{X}_i) - \mathbf{x}_i$ sont remplacés par leur version pondérée $r'_i = w_i r_i$. L'équation 4.29 nous donne alors la formule de calcul des poids w_i :

$$w_i = \frac{1}{1 + \left(\frac{\delta_i}{\sigma c}\right)^2} \tag{6.2}$$

δ_i est un résidu d'erreur normalisé donné par :

$$\delta_i = \mathcal{P}(X_i) - x_i - med\left(\mathcal{P}\left(\mathbf{X}\right) - \mathbf{x}\right) \tag{6.3}$$

et σ est un facteur d'échelle correspondant à une estimation robuste de l'écart-type sur les *inliers*. σ est estimé à l'aide d'une fonction statistique robuste faisant l'hypothèse que les *outliers* représentent moins de 50% des correspondances : le *Median Absolute Deviation* (MAD).

$$\sigma = MAD\left(\delta_i\right) = med\left(\left|\delta_i - \underset{j}{med}\left(\delta_j\right)\right|\right) \tag{6.4}$$

6.3.3 Résultats de suivi robuste

Nous comparons ici les résultats de la méthode de suivi robuste à celle présentée dans la section 6.2. Rappelons que la méthode ne fonctionnait que dans le cas quasi-idéal, et dérivait ou échouait dans les cas plus généraux.

6.3.3.1 Vérité terrain

Afin d'évaluer l'apport de la méthode robuste en terme de qualité de recalage, des séquences de synthèse sont utilisées. On dispose alors des véritables poses de la caméra pour les comparer à celles estimées avec notre algorithme.
La scène synthétique utilisée ici se compose d'un unique bâtiment. Il est visible dans toutes les images. Le trajet de la caméra est soit fluide, soit bruité. L'algorithme est testé également en présence ou non d'objets occultants devant la façade. Des images de

la séquence avec mouvement fluide et objets occultants sont données en annexe, sur la figure A.1.
Sur la figure 6.9, nous présentons l'évolution temporelle de l'erreur de projection entre des points de l'image projetés en utilisant soit les poses de la vérité terrain, soit les poses que l'on a estimées. Les points utilisés sont ceux extraits des images lors du recalage robuste. Les quatre courbes présentées résument les quatre cas générés par les deux hypothèses suivantes :

- Présence ou non d'objets occultants
- Trajet de caméra fluide ou bruité

Pour chaque graphe, on compare l'erreur de projection dans le cas non robuste et dans le cas robuste.
On remarque que dans tous les cas, on obtient de bien meilleures performances avec le recalage robuste. Pour les séquences testées, l'erreur de projection moyenne est en effet toujours inférieure à un pixel dans le cas robuste, alors qu'elle varie entre 3 et 25 pixels dans le cas non robuste. On note également que malgré une dérive importante en début de séquence, l'algorithme non robuste converge de nouveau vers une solution plus correcte en fin de séquence.

6.3.3.2 Séquences réelles

Des résultats visuels de suivi sont présentés sur les figures 6.10, 6.11 et 6.12. Le modèle recalé est affiché en fil de fer bleu. La zone verte correspond au modèle de l'estimation du sol.

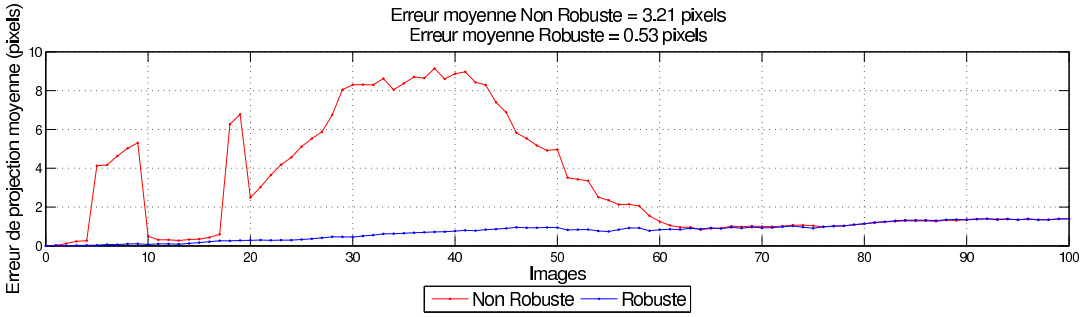
Séquence *Beaulieu 1* (figure 6.10) Comme dans le cas du suivi non robuste, le bâtiment est ici recalé correctement pour toutes les images.

Séquence *Ifsic* (figure 6.11) Par rapport à la version non robuste, on constate que la majeure partie de la dérive a disparu. On remarque cependant qu'il reste une petite erreur de recalage dans les dernières images.

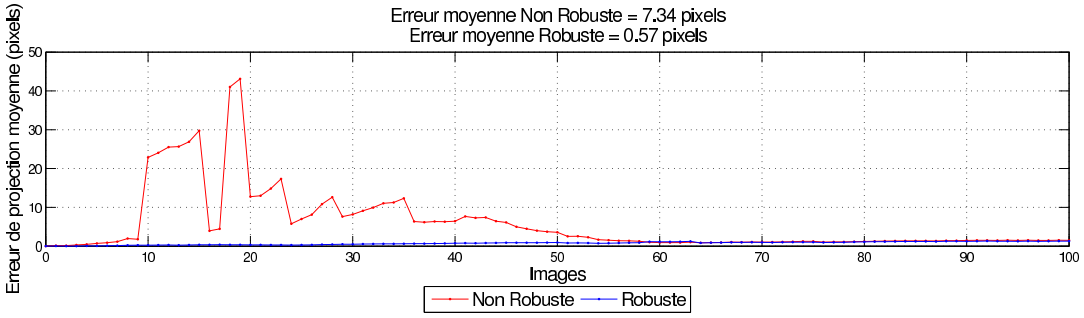
Séquence *Beaulieu 2* (figure 6.12) Sur cette séquence, la version non robuste de l'algorithme ne permet pas de suivre le bâtiment à cause des occultations non modélisables (la voiture par exemple). On constate au contraire ici que le bâtiment est correctement suivi tout au long de la séquence.

6.3.3.3 Temps de calcul

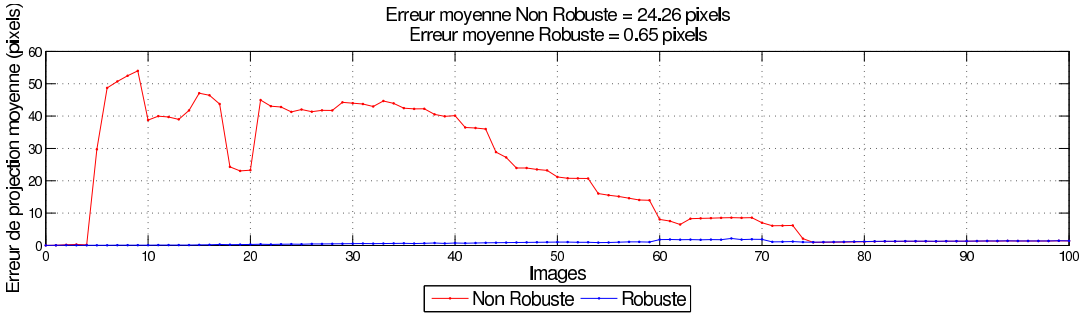
Les temps de calcul en secondes nécessaires au suivi de séquences complètes sont présentés dans le tableau 6.1. Nous indiquons également entre parenthèses le nombre d'images suivies par secondes, ainsi que le nombre maximal de points suivis dans la séquence. Le symbole $\varnothing$ indique que le suivi a échoué. Les temps indiqués correspondent à la moyenne sur dix exécutions, et les tests ont été effectués sur un Intel Core 2 Duo 6700,



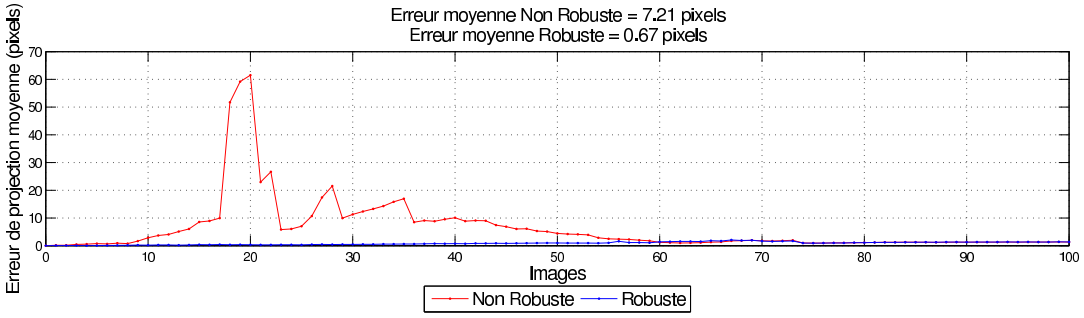
(a) Trajet fluide - pas d'occultations



(b) Trajet fluide - occultations



(c) Trajet bruité - pas d'occultations



(d) Trajet bruité - occultations

FIG. 6.9 – Comparaison de l'erreur de projection entre les méthodes robuste et non robuste

FIG. 6.10 – Résultats de suivi robuste pour la séquence *Beaulieu 1*

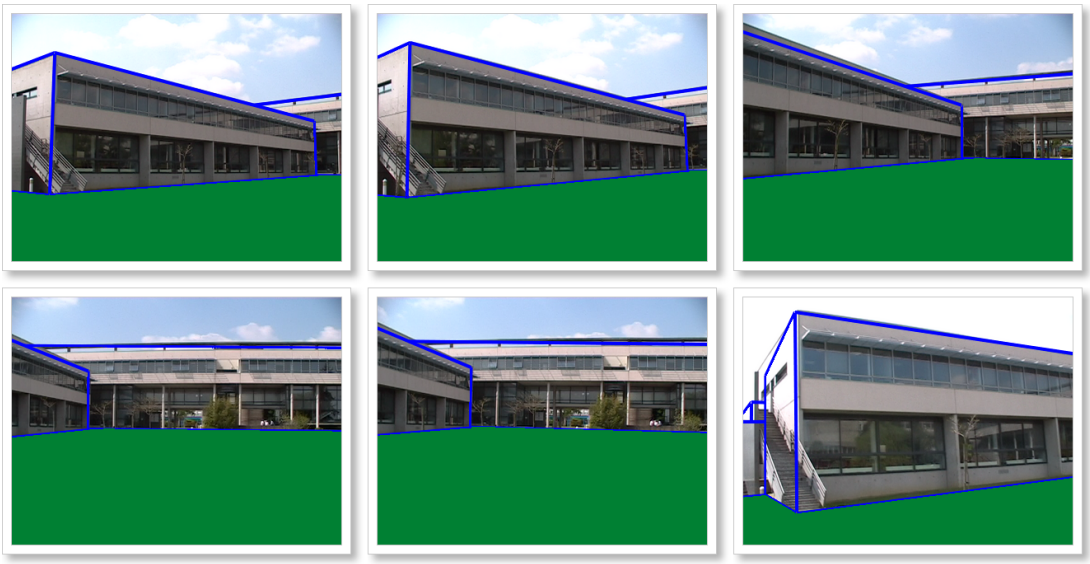


FIG. 6.11 – Résultats de suivi robuste pour la séquence *Ifsic*

avec 2 Go de mémoire vive et une Quadro FX 3500 pour l'affichage.
On remarque que sans surprise, le suivi robuste est plus lent que le suivi non robuste (environ deux fois plus). Ceci est dû au fait que l'ajout du modèle du sol permet de rejeter moins de points pour les correspondances 2D-3D d'une part, et d'autre part au fait du calcul du poids des différents points dans la phase d'asservissement puis des différentes itérations pour les nouvelles pondérations successives.

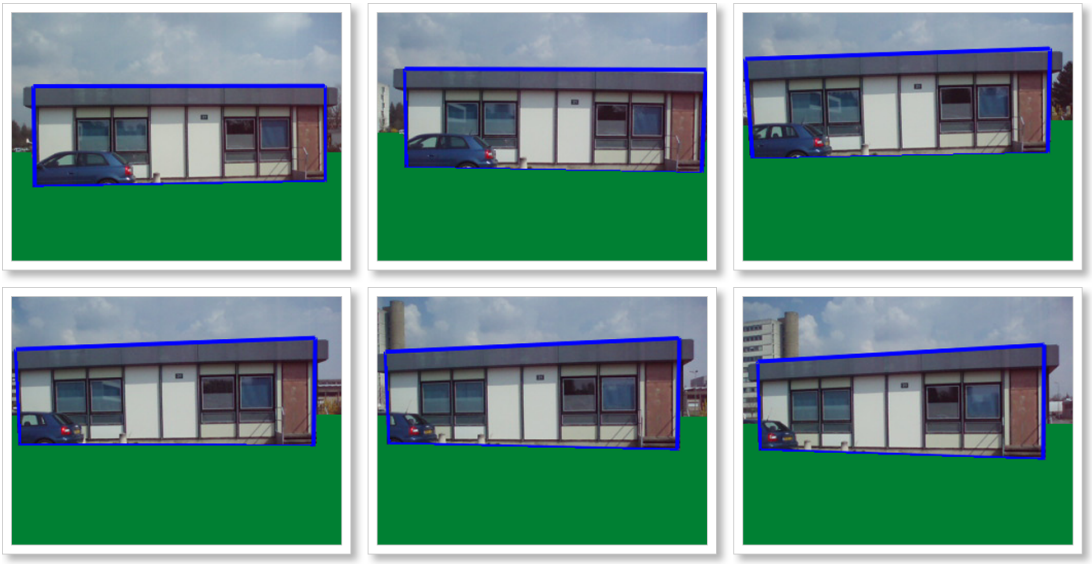


FIG. 6.12 – Résultats de suivi robuste pour la séquence *Beaulieu 2*

	Synth Rec	*Beaulieu 1*	*Ifsic*	*Beaulieu 2*
Non Robuste	23.66 (4.27)	8.48 (8.14)	118.57 (5.48)	∅
Robuste	50.13 (2.01)	17.61 (3.92)	185.10 (3.51)	57.42 (4.18)
Nb points	200	75	75	75

TAB. 6.1 – Temps de calcul en secondes du recalage pour les différentes séquences présentées, en fonction du type de méthode de suivi utilisée (robuste ou non)

Excepté pour la séquence *Beaulieu 1* qui est acquise à basse fréquence temporelle, le recalage ne s'effectue pas en temps réel.

6.4 Conclusion

Nous avons présenté dans ce chapitre une méthode de recalage entre des images de bâtiments et le modèle 3D d'une base SIG correspondante.
L'algorithme est basé sur de l'asservissement visuel virtuel utilisant des points comme primitives. L'adaptation de cet algorithme à notre contexte (estimation du sol, suppression des *outlier* par M-estimation) permet de calculer le recalage de façon beaucoup plus robuste à la fois à la coplanarité de la scène visualisée, aux objets occultants non modélisés dans la base SIG, mais aussi aux mouvements de caméra plus généraux (par opposition aux mouvements de caméra prévus pour bien conditionner le recalage).

Un premier défaut de cette approche est qu'il peut subsister une dérive dans le recalage. L'utilisation conjointe de lignes et de points pour le calcul de pose par asservissement,

plutôt que de points seuls, devrait permettre de limiter cette dérive, sous la condition que les frontières des bâtiments soient visibles dans les images.
Un autre défaut de l'algorithme est le temps d'exécution requis, qui interdit son utilisation en temps réel dans la majeure partie des cas. Le recalage étant calculé pour toutes les images, une solution consisterait à ne le calculer que toutes les k images ($k > 1$), et d'interpoler la pose pour les images restantes.

Connaissant désormais la pose, et donc la correspondance entre la projection du modèle Sig et les images, on peut calculer la texture des façades visualisées. C'est ce que nous présentons dans la partie suivante.

Troisième partie

Extraction de textures et calcul des micro-structures

Introduction

It is a mistake to think you can solve any major problems just with potatoes - **Douglas Adams**

La pose de la caméra étant connue pour toutes les images de la vidéo, on est désormais capable d'aligner la projection du modèle 3D de la base SIG sur les images de façades correspondantes.
Cette information sur une unique image permet déjà d'extraire une texture pour les façades visibles, afin de les plaquer sur le modèle 3D. Cependant, les textures ainsi calculées ne sont souvent que partiellement visibles, et leur résolution spatiale peut varier en fonction de la projection perspective (c'est-à-dire que certaines parties de la texture sont plus floues que d'autres). De plus, si l'on cherche à reconstruire la micro-structure uniquement à partir des images, alors une seule de ces images est insuffisante pour remonter à l'information 3D voulue. C'est pourquoi nous extrayons de la vidéo plusieurs textures d'une même façade. Celles-ci sont fusionnées pour calculer la texture finale des façades en question, et elles sont comparées entre elles pour déterminer la micro-structure des bâtiments.

Plan Cette partie est organisée de la façon suivante : des méthodes existantes sur le calcul de textures de façades ainsi que sur le calcul de leur micro-structure est présenté dans le chapitre 7. Puis notre algorithme d'extraction et de fusion de textures est développé dans le chapitre 8. Enfin, une étude sur la faisabilité du calcul de la micro-structure des façades en question est explicité dans le chapitre 9. Pour les deux derniers chapitres, des résultats sur des séquences de synthèse et réelles sont présentés.

Chapitre 7

Raffinement des modèles 3D - état de l'art

Computers make it easier to do a lot of things, but most of the things they make it easier to do don't need to be done - **Andy Rooney**

7.1 Introduction

L'objectif principal de ces travaux de thèse consiste à raffiner à la fois géométriquement et photogramétriquement des modèles 3D de bâtiments existants qui sont extraits d'une base SIG. Dans un premier temps, nous avons présenté une méthode permettant de recaler la projection de ces modèles avec des images leur correspondant, étape préalable incontournable pour le calcul des textures des façades (partie II).
Nous cherchons désormais à exploiter ce recalage, afin dans un premier temps d'extraire des images sus-citées les textures des différentes façades visualisées, et dans un deuxième temps de calculer la micro-structure de ces façades correspondant aux portes, fenêtres, etc. qui ne sont pas modélisées dans la base SIG originale.

7.2 Extraction de textures

Nous nous intéressons ici aux méthodes proposées dans la littérature pour calculer des textures de façades à partir d'un ensemble d'images, où les-dites images sont totalement ou partiellement visibles.

7.2.1 Problèmes à résoudre

Une fois que le recalage avec le modèle correspondant a été effectué, on peut extraire les textures des façades visibles dans les images recalées. L'extraction puis la fusion d'un ensemble de ces textures, acquises depuis des points de vue différents, permet de résoudre de nombreux problèmes, impossibles à traiter pour la plupart en se basant uniquement sur les informations d'une seule image.

Lors du processus d'acquisition, on ne peut garantir que chaque façade visée soit visible entièrement dans l'image. Il se peut par exemple qu'une partie se projette en dehors du plan image (façade trop grande), ou qu'un autre bâtiment la masque en partie. Ces zones occultées sont détectables en connaissant conjointement la pose de la caméra, ses paramètres intrinsèques, ainsi que le modèle 3D correspondant. On parle alors d'*occultations modélisables*.

Dans un cadre réel de reconstruction de zones urbaines, les bâtiments visualisés sont en général également partiellement occultés par des objets fixes ou mobiles de l'environnement. On peut citer à titre d'exemple les véhicules, les poteaux électriques ou de signalisation, les arbres, les piétons, etc. Tous ces objets se retrouvent au premier plan dans les images et masquent partiellement les bâtiments. Or ils ne sont pas modélisés dans les données 3D dont on dispose. On parle alors d'*occultations non modélisables*, et leur suppression des textures de façade ne peut se faire qu'en traitant les images seules.

La position et l'orientation de la caméra par rapport aux façades visualisées influent également sur les images acquises, et donc sur la qualité des textures reconstruites. En effet, la *résolution spatiale* - au sens du ratio entre les unités surfaciques de la façade et de l'image - de la façade projetée sur le plan image varie en fonction à la fois des dimensions de la façade elle-même et des paramètres extrinsèques de la caméra. Pour une même façade et une même orientation de la caméra, la résolution dans l'image sera inversement proportionnelle à la distance entre le centre optique et cette façade. De même, pour une seule image, si la façade n'est pas parallèle au plan image, alors les zones où les points de la façade sont plus éloignés du centre de la caméra correspondent aux zones de plus faible résolution, du fait de la projection perspective.

Que ce soit dans le cas des occultations modélisables ou non modélisables, il se peut qu'une partie de façade ne soit jamais visible dans les images considérées. Ces parties sont alors étiquetées comme *zones inconnues*, et seuls des algorithmes de synthèse de texture se basant généralement sur le voisinage de la-dite zone sont à même de la reconstruire correctement.

Enfin, on pourra noter les deux derniers problèmes, traités de façon plus marginale dans la littérature, que sont les variations d'illumination entre les différentes images acquises, ainsi que le traitement des spécularités présentes du fait des nombreuses fenêtres visibles en général en zone urbaine en conjonction avec le mouvement de la caméra.

7.2.2 Techniques existantes

La reconstruction réaliste de zones urbaines étant un sujet relativement étudié ces dernières années, de nombreuses techniques ont déjà été développées pour l'extraction, la fusion et l'estimation de textures de façades. Nous en décrivons quelques unes dans cette section.

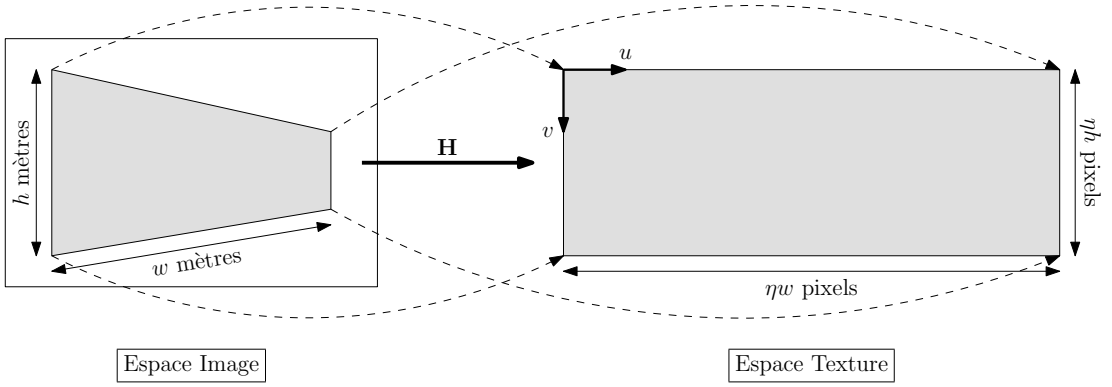


FIG. 7.1 – Recalage des images de façade dans l'espace texture

7.2.2.1 Recalage dans un même repère

Afin d'utiliser les informations redondantes de plusieurs images pour calculer la "meilleure" texture finale, les images utilisées comme source de données doivent être recalées dans un même repère. Deux solutions se dégagent de la littérature pour effectuer un tel recalage.
Une première approche consiste à choisir parmi toutes les images de façades une image de référence, qui sera inchangée alors que les autres seront recalées avec elle [OR05] [BÖ4] [KR05b, KR06, KR05a]. Sous l'hypothèse de façades planes, la transformation permettant d'aligner deux images de façades est définie par l'homographie $\mathbf{H}$, qui s'écrit sous la forme d'une matrice 3×3 de rang 2. Si $\mathbf{x}_{ref}$ représente un ensemble de points de la façade considérée dans l'image de référence I_{ref}, et $\mathbf{x}_k$ les points correspondants dans l'image I_k, alors l'ensemble des points $\mathbf{x}'_k$ correspondant aux points $\mathbf{x}_k$ recalés avec l'image de la façade dans I_{ref} se calculent de la façon suivante :

$$\forall i \in \{1..N\}, \quad \mathbf{x}'_{k_i} \sim \mathbf{H}_k \mathbf{x}_{k_i} \tag{7.1}$$

Si on connaît N correspondances ($N \geq 4$) entre des points appartenant à la façade dans I_{ref} et I_k, on peut recaler les deux images entre elles pour tous les pixels de la façade.
Une deuxième approche consiste à exprimer toutes les façades visibles des images dans le *repère texture* [POF98] [BKB+01, MBB+01, MKB01] [WTT+02, Tai00], qui correspond à une image fronto-parallèle de la façade, celle-ci définissant les dimensions de la texture en question (voir figure 7.1). Un tel recalage des images nécessite un paramètre de résolution η pour définir le ratio des dimensions de la texture et de la façade. Il s'exprime par exemple en pixels par mètre. De la même façon que pour le recalage par rapport à une image de référence, la transformation permettant de passer du repère image au repère texture est une homographie. Elle est calculée en mettant en correspondance les quatre coins de la façade projetés dans l'image avec les quatre coins de la texture.

Dans les deux cas, le recalage en lui-même s'effectue en échantillonnant la zone recalée. La couleur finale c_f d'un pixel d'une image recalée est calculée en appliquant la

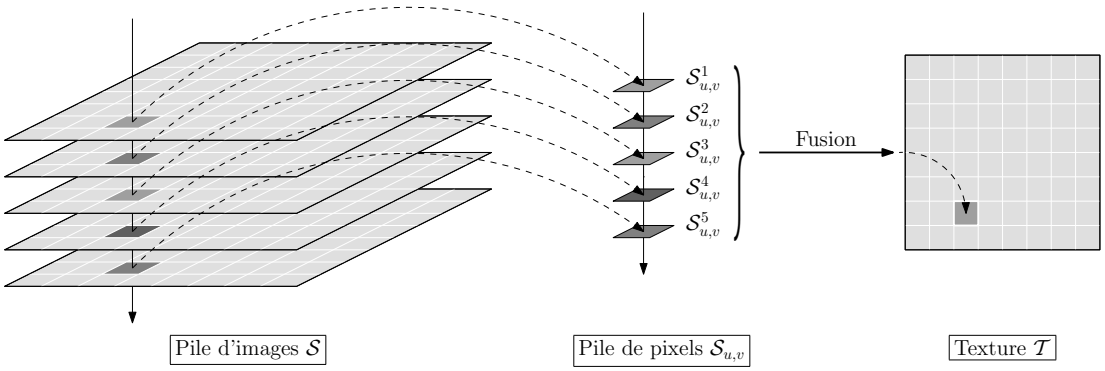


FIG. 7.2 – Principe de la fusion pixel à pixel

transformée inverse $\mathbf{H}_k^{-1}$ sur sa position p_f pour retrouver la position correspondante p_i dans l'image source. Comme p_i n'est en général pas une valeur entière de pixel, et que la résolution des deux images pour ces positions sont différentes, la couleur source c_i est calculée par interpolation des couleurs de l'image source :

$$c_f = I_k(p_i) = I_k\left(\mathbf{H}_k^{-1} p_f\right) \tag{7.2}$$

Une interpolation bicubique permet de préserver les détails des contours [OR05], mais des interpolations au plus proche voisin, bilinéaire ou basée splines peuvent également être utilisées.

7.2.2.2 Reconstruction de textures

Une fois les différentes images source recalées dans le même repère (on les notera I^k_{rec}), on peut construire une pile d'images $\mathcal{S}$ à partir de laquelle sera reconstruite la texture $\mathcal{T}$. La reconstruction en elle-même est effectuée pixel à pixel (voir figure 7.2). On notera $\mathcal{S}_{u,v}$ la pile de pixels de coordonnées (u, v) dans $\mathcal{S}$, et $\mathcal{T}_{u,v}$ le pixel correspondant dans la texture finale. $\mathcal{S}^k_{u,v}$ correspond au pixel de $\mathcal{S}_{u,v}$ issu de l'image recalée I^k_{rec}.

Fusion de textures Dans tous les cas, les algorithmes de la littérature visent à exploiter la redondance présente dans les différentes images I^k_{rec} pour calculer $\mathcal{T}$ pixel à pixel. Le but à atteindre étant de gérer les différents problèmes décrits dans la sous-section 7.2.1. Pour une même pile de pixels $\mathcal{S}_{u,v}$, le pixel $\mathcal{T}_{u,v}$ est défini soit comme le "meilleur" pixel de $\mathcal{S}_{u,v}$ [OR05] [BÖ4] [KR05b, KR06, KR05a], soit comme une somme pondérée de $\mathcal{S}_{u,v}$ [POF98] [BKB+01, MBB+01, MKB01] [WTT+02, Tai00].

Occultations modélisables Un des problèmes les plus simples concerne les occultations modélisables. On suppose à ce stade que le modèle 3D du bâtiment visualisé est connu, de même que la pose de la caméra correspondante, ou à défaut sa matrice de projection. Sous l'hypothèse de façades planes, les occultations modélisables sont générées soit par une projection du bâtiment en dehors du plan image, soit par un masquage de celui-ci par

un autre bâtiment ou une partie de lui-même. Les pixels $\mathcal{S}_{u,v}^k$ correspondant à de telles zones n'entrent pas en compte dans le calcul de $\mathcal{T}_{u,v}$. Leur détection peut se faire par calcul de l'intersection entre la ligne de vue et le modèle [BKB+01, MBB+01, MKB01] ou par triangulation [POF98], suivie d'un test de profondeur. Dans le cas du calcul de $\mathcal{T}_{u,v}$ comme somme pondérée des $\mathcal{S}_{u,v}$, le poids associé à ces pixels sera nul [WTT+02, Tai00].

Occultations non modélisables La suppression des occultations non modélisables représente un défi plus important. Ici la diversité des approches considérées est donc plus conséquente. Si on considère que les pixels d'*arrière plan* (*i.e.* ceux de la façade) sont représentés en majorité dans $\mathcal{S}_{u,v}$ par rapport aux pixels de *premier plan* (*i.e.* ceux des objets occultants non modélisés), la suppression de ces derniers se fait à l'aide de techniques d'estimation robuste.
Dans le cadre des méthodes de sélection du "meilleur" pixel de la pile $\mathcal{S}_{u,v}$, le calcul de $\mathcal{T}$ par filtrage médian donne de bons résultats pour la suppression des occultations [OR05] [KR05b, KR06, KR05a].
Les travaux présentés dans [Bö4] proposent une méthode de type RANSAC. Là encore le "meilleur" pixel est choisi. Tous les $\mathcal{S}_{u,v}^k$ sont projetés dans un espace de couleur tridimensionnel (RVB ou HSV), et pour chaque pixel une mesure de distance (Mahalanobis ou Euclidienne) permet de déterminer par seuillage combien sont suffisamment proches dans cet espace. Le pixel ayant le plus grand nombre de proches voisins est sélectionné.
Dans [BKB+01, MBB+01, MKB01], les pixels finals sont calculés comme étant une somme pondérée du sous-ensemble $\widehat{\mathcal{S}_{u,v}} \subseteq \mathcal{S}_{u,v}$ défini comme étant l'ensemble des pixels ayant une couleur suffisamment proche de la médiane de $\mathcal{S}_{u,v}$. Dans [POF98] le même principe est utilisé, sans pour autant préciser le mécanisme de rejet des *outlier*.
Dans [WTT+02, Tai00], une procédure itérative utilisant des masques de pondération est présentée. A partir d'une initialisation $\mathcal{T}^{init}$, un poids défini par un score de corrélation entre chaque image de la pile et la texture $\mathcal{T}$ calculée à l'itération courante est attribué à chaque pixel $\mathcal{S}_{u,v}^k$, puis $\mathcal{T}$ est recalculée en fonction de ces poids. Le processus est itéré jusqu'à convergence.

Résolution spatiale Certaines méthodes utilisant une somme pondérée des $\mathcal{S}_{u,v}$ visent à prendre également en compte la résolution spatiale des pixels source pour calculer $\mathcal{T}$. Un poids plus faible est attribué à des $\mathcal{S}_{u,v}$ plus éloignés de la caméra, ou plus déformés par la projection perspective.
Plus l'angle θ formé dans l'image originale I^k entre la ligne de vue de $\mathcal{S}_{u,v}^k$ et la ligne de vue du point $\mathbf{x}_\perp$ (projeté du point $\mathbf{X}_\perp$ de la façade étant le plus proche du centre de la caméra) est important, moins la résolution spatiale sera bonne du fait de la projection perspective (voir figure 7.3). [WTT+02, Tai00] propose donc d'attribuer un poids à chaque pixel de $\mathcal{S}_{u,v}$ comme étant le cosinus de cet angle θ.
Dans [BKB+01, MBB+01, MKB01], la texture finale $\mathcal{T}$ est construite sous la forme d'un quad-tree pour gérer les différences de résolution. L'aire du pixel $\mathcal{S}_{u,v}^k$ projeté dans l'image originale correspondante permet de déterminer à quelle profondeur, et donc à quelle résolution spatiale ce pixel appartient. Ce critère de mesure d'aire est également

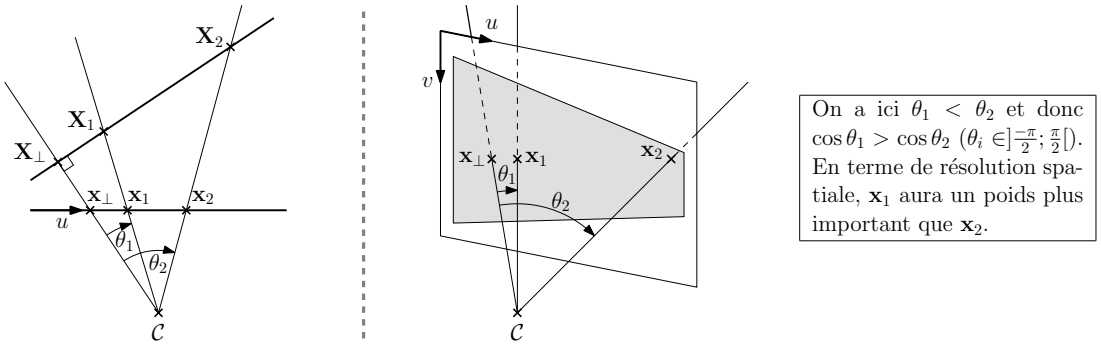


FIG. 7.3 – Influence de l'angle de la ligne de vue sur la résolution spatiale

utilisé dans [POF98] pour attribuer un poids à chaque pixel, mais les auteurs notent qu'à moins d'utiliser des textures à haute résolution, le coût de calcul supplémentaire associé aux pertes d'information générées par le sous-échantillonage entre les différents niveaux du quad-tree ne justifie pas vraiment l'utilisation d'une telle structure de données pour le calcul de $\mathcal{T}$.

Zones inconnues Peu de travaux cherchent à résoudre le problème des zones inconnues pour le calcul de $\mathcal{T}$. Dans [RTC+06] par exemple, une simple copie miroir des pixels manquants est effectuée. Une technique plus élaborée est présentée dans [KR05b, KR06, KR05a]. Les zones inconnues sont définies comme celles n'étant associées à aucune information source, ainsi que comme l'ensemble des pixels dont la pile $\mathcal{S}_{u,v}$ présente une valeur de *Median Absolute Deviation* (MAD) supérieure à un certain seuil. Elles sont ensuite remplies à l'aide d'une version adaptée de l'algorithme d'inpainting de Criminisi, Pérez et Toyama [CPT04].

Variations d'illumination Les images source étant extraites généralement de séquences vidéo, le problème de variations importantes d'intensité lumineuse n'est généralement pas traité. Dans [WTT+02, Tai00] cependant, les données en entrée peuvent être acquises depuis des points de vue et/ou à des moments très différents. Les images en entrée sont alors normalisées de telle sorte qu'elles aient une luminance moyenne identique.

7.2.2.3 Résumé

Un classement des travaux existants en fonction des problèmes résolus est présenté dans le tableau 7.1.

7.3 Calcul de la micro-structure des façades

On trouve un peu moins de travaux dans la littérature concernant la reconstruction réaliste des micro-structures de façades de bâtiments à partir des images, que de travaux

Problème	Méthode	Références
Fus	Choix meilleur pixel	[OR05] [Bö4] [KR05b] [KR06] [KR05a]
	Somme pondérée	[POF98] [BKB+01] [MBB+01] [MKB01] [WTT+02] [Tai00]
Rec	Image de référence	[OR05] [Bö4] [KR05b] [KR06] [KR05a]
	Espace Texture	[POF98] [BKB+01] [MBB+01] [MKB01] [WTT+02] [Tai00]
OcM	Triangulation	[POF98]
	Intersection	[BKB+01] [MBB+01] [MKB01] [WTT+02] [Tai00]
OcNM	Filtrage médian	[OR05] [KR05b] [KR06] [KR05a]
	Corrélation	[WTT+02] [Tai00]
	Clustering couleurs	[Bö4]
ReSp	Cosinus	[WTT+02] [Tai00]
	Quad-tree	[BKB+01] [MBB+01] [MKB01]
	Aire projection	[POF98] [BKB+01] [MBB+01] [MKB01]
VarIl	Normalisation	[WTT+02] [Tai00]

Légende

Fus : Fusion
Rec : Recalage
OcM : Occultations modélisables
OcNM : Occultations Non Modélisables
ReSp : Résolution Spatiale
VarIl : Variations d'Illumination

Tab. 7.1 – Résumé des méthodes présentées en fusion de textures de façades

concernant le calcul des textures de façades elle-mêmes. La majorité des travaux emploient des techniques supervisée ; les méthodes automatiques sont moins bien représentées. On notera également certains travaux récents visant à calculer ces micro-structures à partir d'une seule image.

7.3.1 Modélisation multi-vues supervisée

Dans leurs travaux, Debevec *et al.* [DTM96a, DTM96b] présentent un système de reconstruction de bâtiments nommé *Facade*, qui permet de retrouver les détails géométriques précis des bâtiments à partir d'un ensemble de vues prises depuis des points de vue assez différents. Dans un premier temps, un modèle simple composé de primitives géométriques élémentaires est défini par l'utilisateur à partir des images en entrée. Des

contraintes de parrallèlisme ou d'orthogonalité par exemple peuvent être rajoutées entre ces primitives. Dans un deuxième temps, les micro-structures sont calculées par *stéréovision basée modèle* (SBM). Deux images sont utilisées : une image *clé* et une image *offset*. L'image *offset* est rectifiée de telle sorte qu'elle se superpose avec l'image *clé*, et la SBM mesure sur ces images à quel point le modèle réel à estimer dévie du modèle généré à l'étape précédente. Une mise en correspondance de points basée sur une mesure de corrélation permet de déterminer quels pixels sont en mouvement entre ces deux images, et donc quelles parties des images correspondent à des zones qui ne sont pas correctement décrites par le modèle généré. Des cartes de profondeurs des façades sont alors déduites et le modèle est raffiné.
Cornou et al. [CDS03] proposent une méthode similaire de reconstruction supervisée. De nombreuses contraintes géométriques peuvent être définies entre un ensemble de points, puis le modèle est reconstruit en respectant ces contraintes au mieux grace à un ajustement de faisceaux.
L'utilisateur est également amené à intervenir dans les travaux de Sormann *et al.* [SRB+04], dont le but est la reconstruction précise de bâtiments historiques, en particulier pour les villes de Graz et Vienne. Une séquence d'images est utilisée en entrée. La contribution de l'utilisateur consiste à détecter, mettre en correspondance et segmenter les primitives d'intérêt dans les images, que ce soit des points, lignes ou surfaces. La reconstruction en elle-même utilise des techniques de stéréovision classiques. L'intervention manuelle est justifiée ici par le fait que le but visé est la conservation numérique de bâtiments historiques, et la reconstruction doit être précise et exacte.

7.3.2 Modélisation multi-vues automatique

Dans le projet *City Scanning*, Teller et al. [WTT+02] proposent de reconstruire automatiquement les micro-structures des façades à partir des images utilisées en entrée. Une texture de façade ayant été calculée (voir 7.2), un algorithme de croissance de région orientée — qui permet de détecter des classes d'objets réguliers dans des images — est utilisé pour déterminer la position des fenêtres (modélisées par des rectangles). Le bruit et les variations d'illumination globale dans les images empêchant parfois la détection de toutes les micro-structures, un algorithme de *pattern fitting* périodique est appliqué pour retrouver au sein des façades la périodicité des micro-structures, et rajouter des fenêtres candidates à celles déjà détectées. Puis une profondeur est assignée à ces zones pour raffiner le modèle plan de façade. Elle est estimée en utilisant une variation de l'algorithme de Fua et Leclerc [FL94].

7.3.3 Modélisation mono-vue

Des travaux récents sur la reconstruction de façades visent à estimer ls structure des bâtiments en n'utilisant qu'une seule image. Par exemple, dans [MZWG07], Müller *et al.* proposent une technique basée sur une analyse sémantique des images de façade (redressées pour être parallèles au plan image) pour une reconstruction photo-réaliste. L'image est subdivisée en plusieurs zones, comme les étages, fenêtres, portes, etc. L'in-

formation mutuelle est utilisée pour détecter cette structure de haut niveau de la façade, ainsi que les répétitions des motifs de micro-structures. Les répétitions sont exploitées pour définir la structure minimale de la façade et lisser le bruit dans l'image par la même occasion. Une recherche dans une banque de modèles est effectuée pour trouver ceux qui correspondent le mieux aux fenêtres, portes, etc., puis une profondeur est assignée manuellement aux différentes parties de ces modèles pour raffiner la structure de la façade. Dans le modèle 3D final, des ombres et des spécularités dans les zones correspondant aux vitres sont ajoutées pour donner un plus grand photo-réalisme.

7.4 Conclusion

Nous avons présenté dans ce chapitre un ensemble de méthodes de la littérature visant à calculer des textures de façades ainsi que leur micro-structure.
Connaissant la projection du plan principal d'une façade dans différentes images, les textures correspondantes sont recalées dans le même repère par homographie puis fusionnées pixel par pixel. Des méthodes statistiques permettent de supprimer les occultations non modélisables, et la connaissance de la pose de la caméra autorise une reconstruction de texture en ne conservant que les informations de plus haute résolution.
Les travaux concernant la reconstruction des micro-structures des façades sont moins bien représentés. Les techniques utilisées sont souvent supervisées, et les méthodes automatiques manquent encore de réalisme. Il est à noter la tendance actuelle sur le sujet, à savoir une reconstruction effectuée conjointement à une analyse sémantique de la façade, pour un rendu final moins bruité et plus convainquant d'un point de vue visuel.
Nous présentons dans le chapitre suivant nos contributions à l'estimation des textures de façades en présence d'objets occultants modélisés ou non modélisés.

Chapitre 8

Extraction et fusion de textures

Real stupidity beats artificial intelligence every time - **Terry Pratchett**

8.1 Introduction

Le problème que l'on cherche à résoudre ici consiste à exploiter le recalage entre les images source et la projection du modèle 3D correspondant (partie II) pour calculer la texture finale $\mathcal{T}$ de chaque façade visible du modèle. Pour cela, chaque façade f visible dans chaque image I_k permet de générer une texture $\mathcal{T}_k^f$, qui sera généralement incomplète, masquée par des objets du premier plan mais non modélisés dans la base SIG, et plus ou moins floue selon la méthode d'extraction et la configuration géométrique de la caméra (section 7.2.1). Le calcul de $\mathcal{T}$ se fait alors en deux étapes :

1. Extraction des textures $\mathcal{T}_k^f$ et construction de la pile d'images correspondante (section 8.2).
2. Calcul $\mathcal{T}$ par fusion texel à texel[i] de la pile d'images (section 8.3).

8.2 Extraction et recalage des images

Soit une image I_k ($k \in \{1..n\}$) de la séquence d'origine. On suppose qu'un nombre m de façades sont visibles dans I_k. On cherche alors à calculer — à partir de cette image, de la pose de la caméra et du modèle 3D SIG— les m textures $\mathcal{T}_k^f$ correspondant à une image fronto-parallèle des-dites façades (*i.e.* une texture de façade). Le ratio des dimensions de $\mathcal{T}_k^f$ respecte celui des dimensions de la façade dans le modèle. Nous utilisons donc un facteur d'échelle η (choisi par l'utilisateur) pour passer du domaine métrique au domaine texel.

[i] On différentie dans le texte les *pixels*, qui sont les unités de base des images, des *texels* qui sont les unités de base dans les textures, pour mieux différentier les deux représentations.

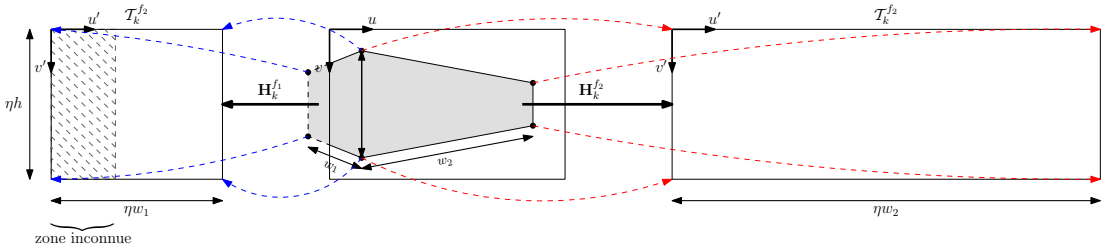


FIG. 8.1 – Utilisation des coordonnées de sommets de façades pour calculer les textures $\mathcal{T}_k^f$

8.2.1 Transformation image-texture

Les données connues, une fois le recalage image-modèle effectué, sont les coordonnées dans le repère image des quatre coins de chaque façade. On notera $\mathbf{x} = [u_i\ v_i]^\top, i \in \{1..4\}$ ces coordonnées. Pour calculer la transformation homographique permettant de passer dans le repère texture $\mathcal{T}_k^f$, on met ces points en correspondance avec les quatre coins de $\mathcal{T}_k^f$, de coordonnées $\mathbf{x}' = [0/u'_j\ 0/v'_j]^\top$, $j \in \{1..4\}$. Si w (resp. h) est la largeur (resp. la hauteur) de la façade f, on a alors :

$$\begin{cases} u'_j = \eta w \\ v'_j = \eta h \end{cases} \tag{8.1}$$

On notera également que la connaissance de la pose de la caméra par rapport au modèle 3D n'impose pas aux points de $\mathbf{x}$ de se situer dans les limites de l'image. Ces correspondances sont illustrées sur la figure 8.1.

La correspondance entre $\mathbf{x}$ et $\mathbf{x}'$ est formalisée par la relation suivante :

$$\mathbf{x}' \sim \mathbf{H}_k^f \mathbf{x} \tag{8.2}$$

On peut déduire de cette équation (en fixant arbitrairement $\mathbf{H}_{k33}^f = 1$, $\mathbf{H}_k^f$ étant définie à un facteur d'échelle près) un système de la forme $\mathbf{Ah} = \mathbf{x}'$, $\mathbf{h}$ étant un vecteur contenant les entrées de l'homographie à estimer :

$$\begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & 0 & 0 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2\eta h & -v_2\eta h \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -u_3\eta w & -v_3\eta w \\ 0 & 0 & 0 & u_3 & v_3 & 1 & 0 & 0 \\ u_4 & v_4 & 1 & 0 & 0 & 0 & -u_4\eta w & -v_4\eta w \\ 0 & 0 & 0 & u_4 & v_4 & 1 & -u_4\eta h & -v_4\eta h \end{bmatrix} \begin{bmatrix} \mathbf{H}_{k11}^f \\ \mathbf{H}_{k12}^f \\ \mathbf{H}_{k13}^f \\ \mathbf{H}_{k21}^f \\ \mathbf{H}_{k22}^f \\ \mathbf{H}_{k23}^f \\ \mathbf{H}_{k31}^f \\ \mathbf{H}_{k32}^f \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \eta h \\ \eta w \\ 0 \\ \eta w \\ \eta h \end{bmatrix} \tag{8.3}$$

Puisque ce système est construit à partir de quatre correspondances de points 2D, la matrice $\mathbf{A}$ est carrée et $\mathbf{h} = \mathbf{A}^{-1}\mathbf{x}'$.

(a) Interpolation bilinéaire (b) Interpolation bicubique

FIG. 8.2 – Différence entre interpolation bilinéaire et bicubique - détail d'une texture extraite

8.2.2 Construction des images de texture

Une fois les homographies $\mathbf{H}_k^f$ estimées, les textures $\mathcal{T}_k^f$ sont calculées en utilisant la transformée inverse $\mathbf{H}_k^{f\,-1}$. Pour chaque texel $[u'\ v']^\top$ de $\mathcal{T}_k^f$, sa position correspondante $[u\ v]^\top$ dans I_k est calculée :

$$[u\ v\ 1]^\top \sim \mathbf{H}_k^{f\,-1} \left[u'\ v'\ 1\right]^\top \tag{8.4}$$

Comme $[u\ v]^\top$ n'est en général pas une position en valeur entière de pixel, la couleur de $[u'\ v']^\top$ est calculée par interpolation des couleurs des pixels de I_k dans le voisinage de $[u\ v]^\top$. Nous utilisons dans notre cas une interpolation bicubique, qui préserve mieux les contours que l'interpolation bilinéaire, comme on peut le voir sur la figure 8.2.

8.3 Fusion de textures

Pour une façade f donnée, nous avons calculé k textures $\mathcal{T}_k^f$, k étant le nombre d'images dans lesquelles f est visible au moins partiellement. On dispose alors d'une pile de textures, et le calcul de la texture finale $\mathcal{T}^f$ se fait texel à texel. Pour chaque pile de texels, la couleur finale est calculée comme étant la somme pondérée des couleurs de la pile, le poids associe à chaque texel de la pile étant défini par $w^{f,u,v}$.

8.3.1 Occultations modélisables

C'est lors du calcul de la couleur de chaque texel $\mathcal{T}_{k\ u,v}^f$ par transformation inverse puis interpolation que sont gérées les occultations modélisables. A ce stade, chaque texel se voit attribuer un poids $w_{occ.mod.}^{f,u,v}$, qui vaut 1 si le texel est visible et 0 sinon. Si le pixel de coordonnées $[u\ v\ 1]^\top \sim \mathbf{H}_k^{f\,-1}\ [u'\ v'\ 1]^\top$ est en dehors de l'image I_k, alors le texel n'est pas visible. De plus, si le pixel est dans l'image, la pose de la caméra est utilisée pour déterminer par rétro-projection la façade f' à laquelle appartient ce pixel. Si $f \neq f'$,

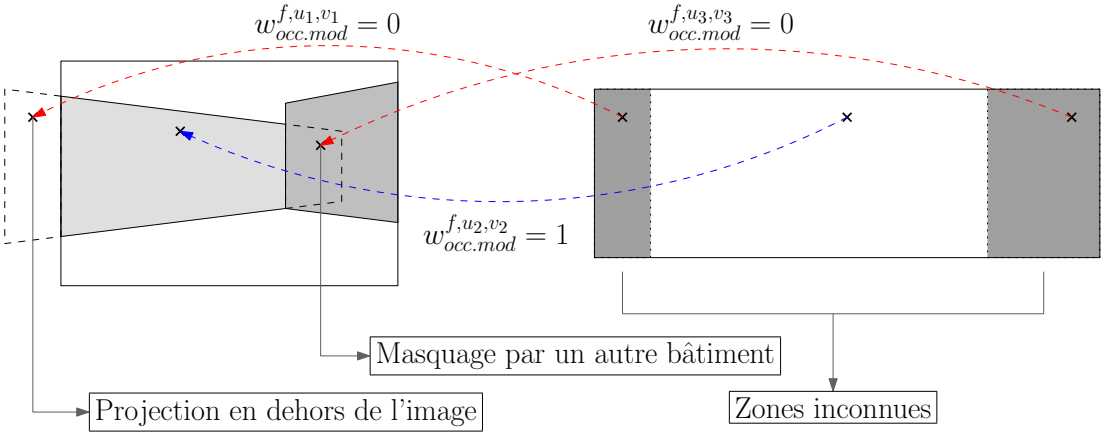


FIG. 8.3 – Gestion des occultations modélisables lors de la construction des textures

alors il n'est pas non plus visible (voir figure 8.3).

$$\begin{cases} w^{f,u,v}_{occ.mod} = 1 & \text{si le texel est visible} \\ w^{f,u,v}_{occ.mod} = 0 & \text{sinon} \end{cases} \tag{8.5}$$

8.3.2 Occultations non modélisables

Soit $\mathcal{T}^f_{u,v}$ la pile de texels de position $[u\ v]^\top$ dans $\mathcal{T}^f$. Certains de ces texels sont entachés d'erreur (au sens de la couleur) du fait des objets occultants non modélisés (*outliers*). Nous souhaitons à ce stade ne conserver que les texels correspondant à la façade (*inliers*) pour calculer la couleur finale du texel dans $\mathcal{T}$.

Si on suppose que pour une pile de texels donnée $\mathcal{T}^f_{u,v}$ les *outliers* représentent moins de 50% des échantillons, alors le texel $\mathcal{T}^f_{u,v_j}$ dont la couleur est la médiane des couleurs de la pile est un *inlier*. Ceux dont la couleur est suffisamment proche sont également considérés comme des *inliers*. Soit $C(\mathcal{T}^f_{u,v_j})$ la couleur du $j^{\text{ième}}$ texel de la pile. La couleur médiane est donnée par :

$$C(\mathcal{T}^f_{u,v})_{\text{med}} = \underset{j}{\text{median}} \left(C(\mathcal{T}^f_{u,v_j}) \right) \tag{8.6}$$

On peut calculer l'écart des *inliers* à cette couleur de façon robuste en prenant la médiane de la valeur absolue de l'écart des couleurs à $C(\mathcal{T}^f_{u,v})_{\text{med}}$:

$$\Delta C(\mathcal{T}^f_{u,v}) = \text{MAD} \left(C(\mathcal{T}^f_{u,v}) \right) = \underset{j}{\text{median}} \left(\left| C(\mathcal{T}^f_{u,v_j}) - \underset{k}{\text{median}} \left(C(\mathcal{T}^f_{u,v_k}) \right) \right| \right) \tag{8.7}$$

On peut alors considérer que les *inliers* conservés $\widehat{\mathcal{T}^f_{u,v}}$ seront tous ceux dont l'écart à $C(\mathcal{T}^f_{u,v})_{\text{med}}$ est inférieur à $\lambda\Delta C(\mathcal{T}^f_{u,v})$, λ étant un scalaire fixé à 2 dans notre cas :

$$\forall k, \quad \mathcal{T}^f_{u,v_k} \in \widehat{\mathcal{T}^f_{u,v}} \quad \Leftrightarrow \quad \left| C(\mathcal{T}^f_{u,v_k}) - C(\mathcal{T}^f_{u,v})_{\text{med}} \right| \leq \lambda\Delta C(\mathcal{T}^f_{u,v}) \tag{8.8}$$

La suppression des *outliers* se fait alors en leur attribuant un poids $w^{f,u,v}_{occ.n.mod.}$ nul :

$$\begin{cases} w^{f,u,v}_{occ.n.mod.} = 1 \quad \text{si} \left| C(\mathcal{T}^f_{u,v_k}) - C(\mathcal{T}^f_{u,v})_{\text{med}} \right| \leq \lambda\Delta C(\mathcal{T}^f_{u,v}) \\ w^{f,u,v}_{occ.n.mod.} = 0 \quad \text{sinon} \end{cases} \tag{8.9}$$

8.3.3 Résolution Spatiale

A partir de la liste d'*inliers* $\widehat{\mathcal{T}^f_{u,v}}$, on cherche désormais à calculer la couleur finale du texel $\mathcal{T}_{u,v}$. Nous attribuons un poids à chaque *inlier* de la pile de telle sorte que l'influence des texels de plus haute résolution soit plus importante que ceux de basse résolution. Plusieurs critères peuvent être utilisés pour mesurer cette résolution, soit en utilisant la configuration géométrique de la scène, soit en utilisant les images en entrée elles-mêmes.

8.3.3.1 Distance et angle

Dans [WTT+02, Tai00], l'angle θ entre la ligne de vue du pixel et celle du projeté du centre optique projeté sur la façade est utilisé. Plus cet angle est important, plus le poids $w^{f,u,v}_{angle}$ est faible. Si on considère que $\theta \in]\frac{-\pi}{2}; \frac{\pi}{2}[$, alors on peut poser :

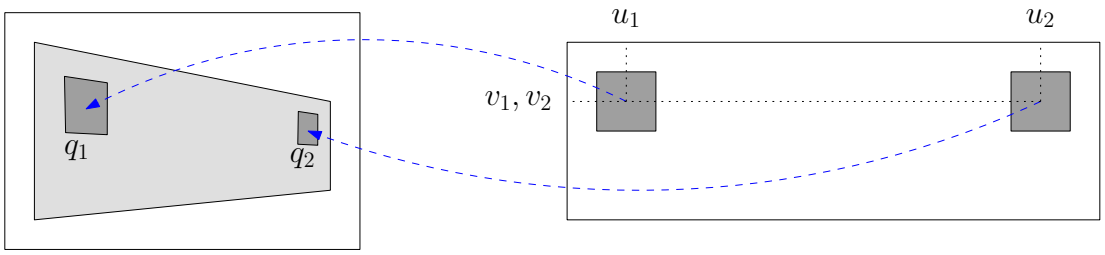
$$w^{f,u,v}_{angle} = \cos|\theta| \tag{8.10}$$

La distance entre la façade et la caméra étant également déterminante pour la résolution finale des texels, nous proposons de leur assigner également un poids $w^{f,u,v}_{dist}$, défini comme la distance entre le centre optique de la caméra et le pixel considéré. Cette distance est calculée en utilisant le z-buffer pour l'image k, au point de coordonnées $[u'\ v'\ 1]^\top \sim \mathbf{H}^{f\,-1}_k [u\ v\ 1]^\top$.

$$w^{f,u,v}_{dist} = 1/\text{z-buffer}(k, u', v') \tag{8.11}$$

8.3.3.2 Aire de projection

Un autre moyen de mesurer la résolution des texels en utilisant la géométrie de la scène est de calculer, pour chacun d'entre eux, l'aire du quadrilatère correspondant dans l'image d'origine (voir figure 8.4). Plus l'aire est grande, et plus le texel apporte une information visuelle pertinente. Si **a**, **b**, **c** et **d** sont les coordonnées des "sommets" du texel dans la texture $\mathcal{T}^f$, alors les coordonnées correspondantes $\mathbf{a}'$, $\mathbf{b}'$, $\mathbf{c}'$ et $\mathbf{d}'$ dans I^f_k sont calculées à l'aide de l'homographie $\mathbf{H}^{f\,-1}_k$. Le poids $w^{f,u,v}_{aire}$ est alors calculé comme



$$\text{Aire}(q_1) > \text{Aire}(q_2) \quad \Rightarrow \quad w_{aire}^{f,u_1,v_1} > w_{aire}^{f,u_2,v_2}$$

FIG. 8.4 – Mesure de l'aire des texels projetés

l'aire du quadrilatère $(\mathbf{a}', \mathbf{b}', \mathbf{c}', \mathbf{d}')$, c'est-à-dire comme la moitié de la norme du produit vectoriel de ses diagonales :

$$w_{aire}^{f,u,v} = \frac{1}{2}\|(\mathbf{a}' - \mathbf{c}') \times (\mathbf{b}' - \mathbf{d}')\| \tag{8.12}$$

8.4 Résultats

Nous présentons tout d'abord dans cette partie des résultats de fusion de texture sur une séquence de synthèse (pour laquelle le recalage images-modèle est connu, voir annexe A). Les différentes étapes de calcul de la texture finale sont les suivantes :

1. Extraction des textures (figure 8.5)
2. Suppression des occultations modélisables (figure 8.6(b))
3. Suppression des occultations non modélisables (figure 8.6(c))
4. Calcul de la texture finale par somme pondérée des texels valides de la pile :
   - Pondération *distance-angle* (figure 8.6(d))
   - Pondération *aire* (figure 8.6(e))

Des résultats de fusion sont également présentés pour des images réelles sur la figure 8.7. Dans ce cas, le recalage images-modèle utilisé pour l'extraction est calculé automatiquement. La mesure d'aire est utilisée pour la pondération de la résolution. La colonne de gauche illustre les images complètes, et celle de droite un détail de celles-ci.

Sur la figure 8.6, pour la séquence de synthèse *Synth-tex*, on remarque que la majeure partie des texels non valides sont écartés lors de la phase de suppression des zones d'occultation non modélisables. La phase de gestion de la résolution spatiale n'a que peu d'influence. On observe à titre d'exemple un PSNR (*Peak Signal to Noise Ratio*) de 35.316 dB entre l'image de la figure 8.6(c) et la texture finale calculée avec pondération par mesure distance-angle. Il est de 35.494 dB avec l'image calculée en utilisant la pondération par mesure d'aire.

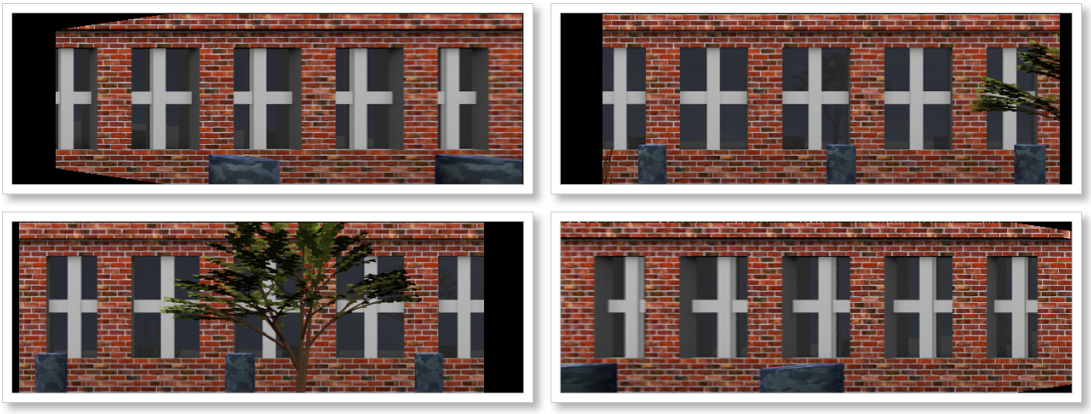


FIG. 8.5 – Exemples de textures extraites

On remarque également que la texture est bien reconstruite pour les zones appartenant au plan principal de la façade (celles qui correspondent au plan dans le modèle 3D). Par contre, elles sont soit floues à cause de la parallaxe (zone supérieure de la façade), soit bruitées sur les contours (fenêtres), pour les zones de micro-structure qu'il reste à estimer. Ce problème de coutours bruités n'apparaît cependant pas sur les séquences réelles, où la structure des détails dans les images est conservée (par exemple les poteaux sur la figure 8.7(a) ou la rembarde de l'escalier sur la figure 8.7(e)). La différence entre la séquence de synthèse et les séquences réelles vient du fait que dans la première, les occultations non modélisables ont une influence beaucoup plus importante.

On notera également que les objets occultants sont correctement supprimés. Pour les séquences réelles, ce résultat est particulièrement visible sur la figure 8.7(c), où les plots en béton disparaissent, ainsi qu'une partie de la rembarde en métal sur la partie droite. La voiture quant à elle reste en partie apparente. Ceci est dû au fait que sa partie gauche masque la façade dans toutes les images.

On notera enfin que par rapport aux images originales, les spécularités sont supprimées dans les textures finales (figures 8.6, 8.7(c) et 8.7(d) en particulier).

Sur la figure 8.8, nous comparons nos résultats avec l'algorithme de fusion de textures de Teller *et al.* [WTT+02, Tai00]. On peut alors mettre en avant plusieurs avantages de notre méthode :

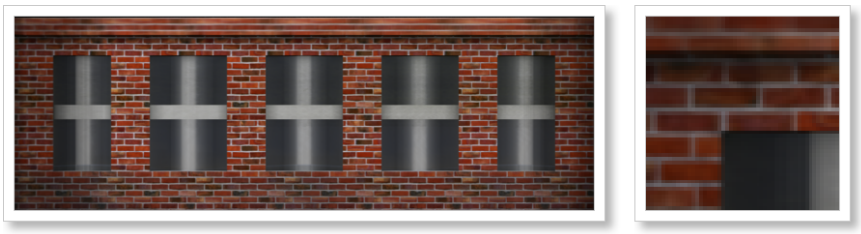
- Nous conservons mieux les contours, qui sont plus net avec notre algorithme (8.8(a) *vs* 8.6(e) ; 8.8(e) *vs* 8.7(e))
- Les objets occultants sont supprimés plus efficacement (8.8(a) *vs* 8.6(e) ; 8.8(c) *vs* 8.7(c))

8.5 Conclusion

Nous avons présenté dans ce chapitre un nouvel algorithme de calcul de textures de façades de bâtiments, qui correspond à l'intégration des différentes briques algorithmiques de base présentées dans la littérature sur le sujet. Il se base sur la fusion de textures extraites des différentes images de la vidéo à partir de la pose estimée de la caméra pour toute la séquence. La texture est construite sur un schéma de fusion texel à texel, et permet la suppression des occultations modélisables, la suppression des occultations non modélisables, ainsi que la conservation des texels de plus haute résolution spatiale.

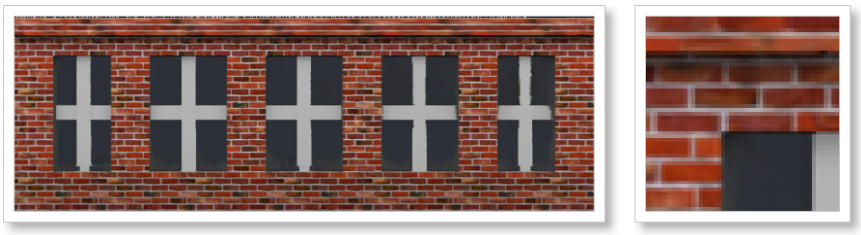
Les zones de micro-structure très texturées sont cependant floues après reconstruction du fait de la parallaxe dans les images de façades recalées utilisées en entrée. Si le calcul de la géométrie de ces zones en dehors du plan principal de la façade est faisable, alors il sera possible de reconstruire ces zones avec plus de précision ultérieurement.
De plus les zones totalement inconnues ne sont pas traitées. Il serait intéressant de tester l'algorithme d'inpainting temporel de [CPT04] pour remédier à ce problème.

Les textures de façades étant estimées, il reste pour être complet à estimer leur micro-structure. C'est ce que nous présentons dans le chapitre suivant.

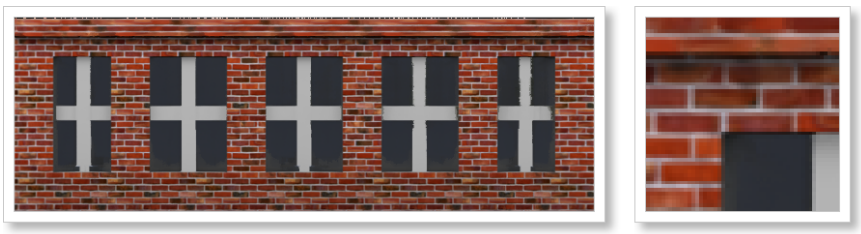


(a) Moyenne simple

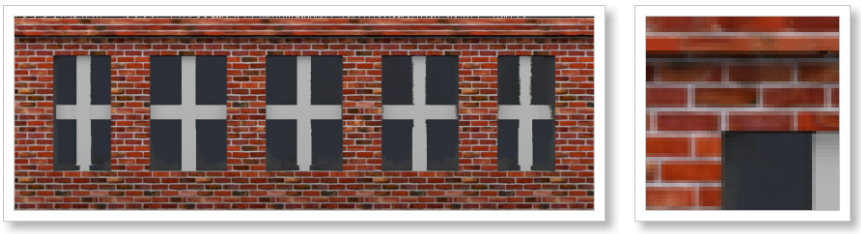
(b) Suppression des occultations modélisables



(c) Suppression des occultations non modélisables

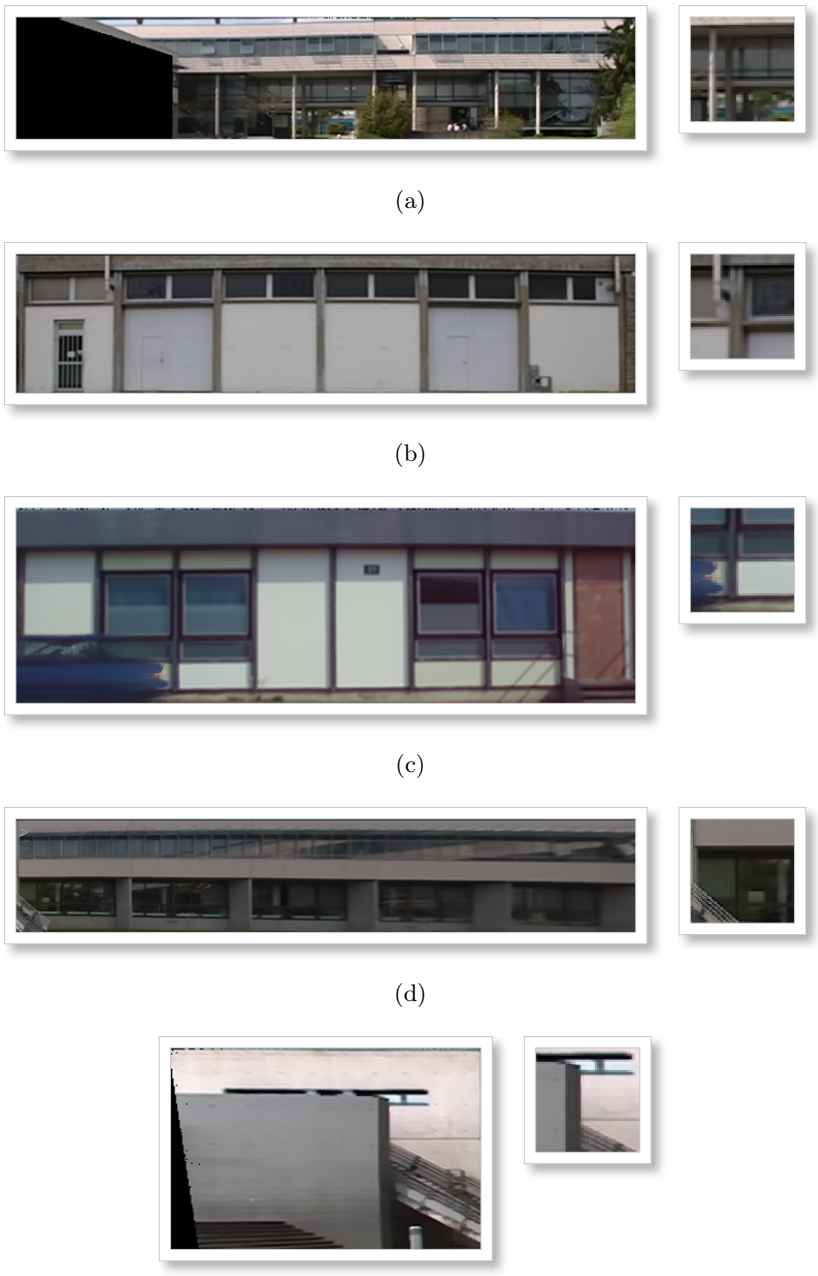


(d) Pondération par mesure distance-angle



(e) Pondération par mesure d'aire

FIG. 8.6 – Fusion de textures - Étapes intermédiaires et résultats finals



(a)

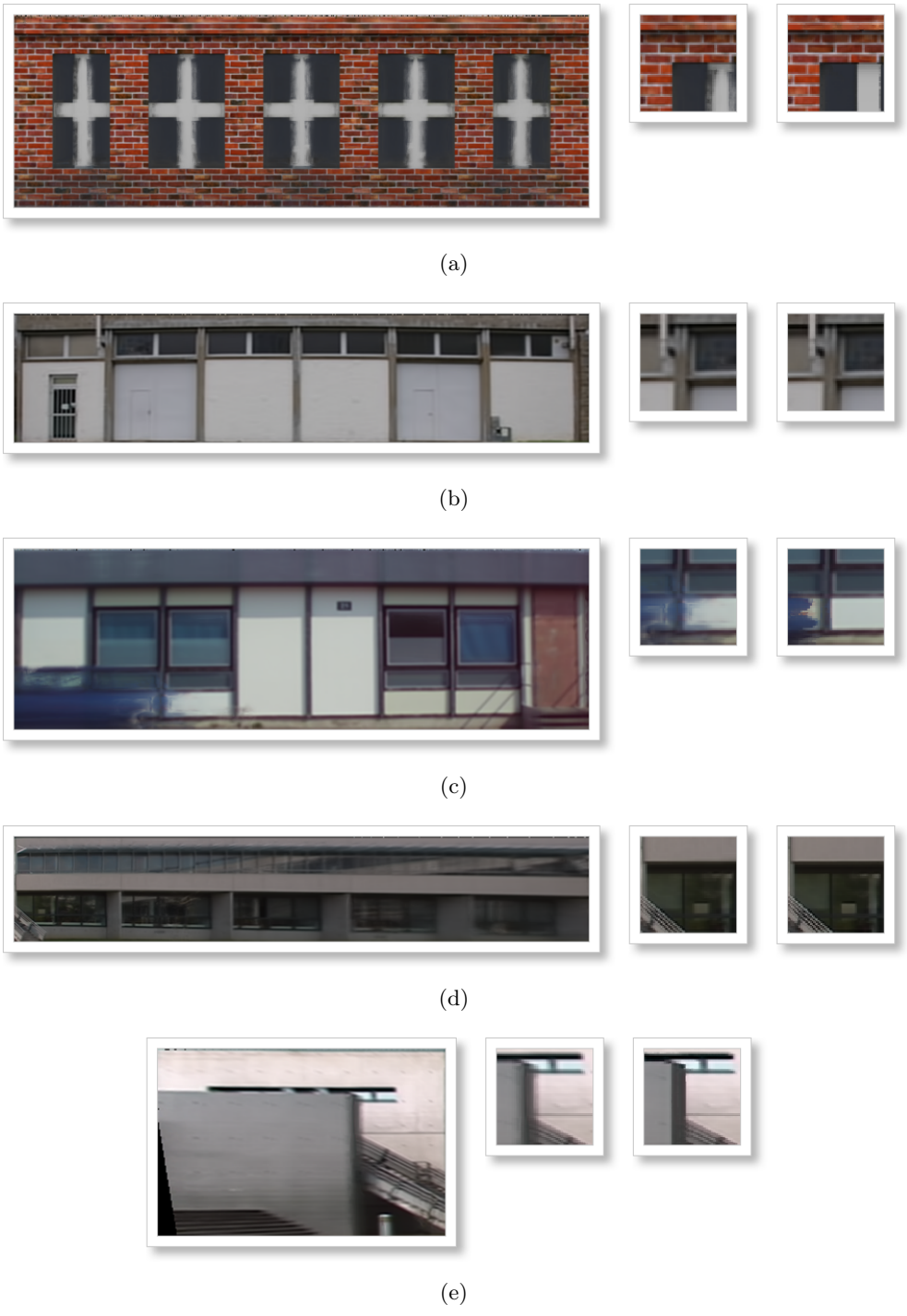
(b)

(c)

(d)

(e)

FIG. 8.7 – Fusion de textures - notre algorithme. La colonne de gauche présente les textures réelles calculées avec la pondération par mesure d'aire. La colonne de droite représente un détail de chacune de ces textures. On notera en particulier que les objets occultants sont correctement supprimés, sous l'hypothèse qu'ils représentent moins de 50% des cas pour un texel donné, et que les contours sont préservés en général.



(a)

(b)

(c)

(d)

(e)

FIG. 8.8 – Fusion de textures - algorithme de Teller *et al.* La colonne de gauche présente les textures calculées avec l'algorithme de Teller. La colonne centrale représente un détail de chacune de ces textures, et celle de droite ce même détail pour la texture calculée avec notre algorithme.

Chapitre 9

Calcul de la micro-structure

The real problem is not whether machines think but wheter men do - **BF Skinner**

9.1 Introduction

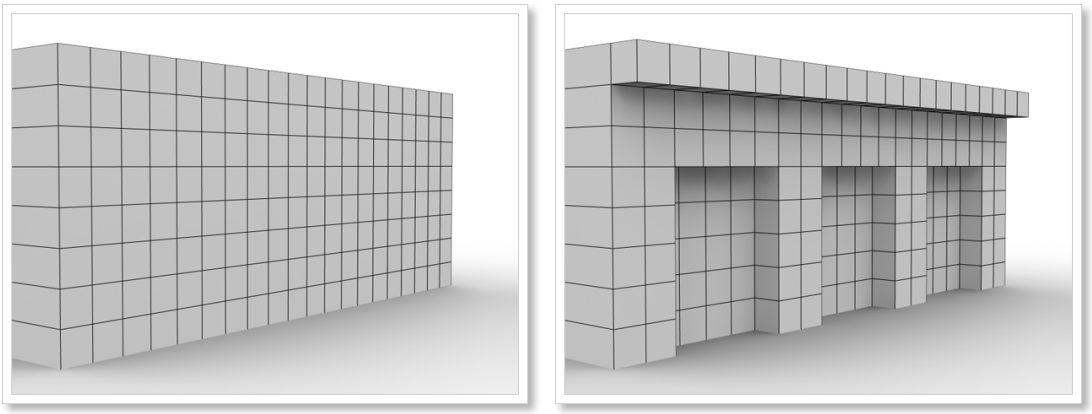
Le problème que l'on cherche à résoudre ici consiste à estimer la micro-structure des bâtiments. En particulier, on vise à remplacer le modèle 3D plan de chaque façade extrait de la base SIG par un modèle similaire auquel on rajoute la géométrie de ses portes, ses fenêtres, ou tout autre particularité correspondant à un renfoncement ou un dépassement du plan principal en question.

Une approche simplificatrice par rapport à la réalité consiste à considérer que ces zones que l'on cherche à estimer correspondent à une extrusion de la façade selon la direction de sa normale (voir figure 9.1). Dans ce cas, on peut par exemple représenter la micro-structure d'une façade par une carte de profondeur ou d'élévation que l'on peut modéliser sous la forme d'une image en niveaux de gris.

On va donc chercher dans ce chapitre à estimer de telles cartes de profondeur. Dans notre cadre d'étude, ce calcul ne peut s'effectuer qu'à partir des images extraites de la vidéo. On peut également considérer qu'il est plus simple de se baser sur les textures de façades extraites (section 8.2) car dans celles-ci, seules les parties en dehors du plan principal de la façade sont en mouvement d'une image à l'autre.

Cependant, comme l'indiquent Teller *et al.* dans [WTT+02], une telle reconstruction de la géométrie des façades à partir des images est difficile, car le ratio entre les profondeurs des structures (quelques centimètres) à estimer et la distance entre la caméra et le mur (quelques mètres ou dizaines de mètres) est trop petit.

Hypothèses de travail De ce fait, nous considérons que le ratio entre la profondeur des structures à estimer et la distance entre la caméra et les bâtiments visés est trop faible



(a) Façade plane (b) Façade extrudée

FIG. 9.1 – Le calcul de la micro-structure vu comme la recherche des extrusion du plan principal des façades le long de leur normale

pour qu'une estimation précise des profondeurs soit effectuée. Nous cherchons donc non pas les valeurs elles-mêmes des profondeurs, mais uniquement leur signe :

- Si la façade présente une extrusion vers l'intérieur du bâtiment (une fenêtre par exemple), on dit que la profondeur est positive.
- Si la façade présente une extrusion vers l'extérieur du bâtiment, on dit que la profondeur est négative.

Une fois les zones de micro-structure localisées et leur signe identifié, on choisit alors pour une première version de cette reconstruction d'assigner des valeurs arbitraires aux profondeurs en question.

On sait que les structures en dehors du plan principal de la façade sont animées dans les textures redressées d'un mouvement dû à la parallaxe (voir par exemple [KR05b]). On fait de plus l'hypothèse que le mouvement de la caméra est globalement horizontal (vis-à-vis du sol). C'est l'étude du signe de ce mouvement qui va permettre de déterminer le signe des extrusions. Soit D le sens du mouvement de la caméra par rapport à la façade (de gauche à droite ou de droite à gauche), et d le sens du mouvement d'une structure donnée dans les textures redressées :

- Si d est du même signe que D, alors la structure en question est une extrusion positive (vers l'intérieur).
- Si d est de signe contraire à celui de D, alors la structure en question est une extrusion négative (vers l'extérieur).

Ce principe est illustré sur la figure 9.2. Deux caméras $\mathcal{C}_1$ et $\mathcal{C}_2$ regardent une façade. Le point 3D P est vu comme étant P_1 depuis $\mathcal{C}_1$, et comme P_2 depuis $\mathcal{C}_2$, puisque la façade est modélisée comme un plan unique Π. On peut voir que le mouvement apparent dans les textures de ce point P (symbolisé par le vecteur bleu d) est du même signe (resp. de

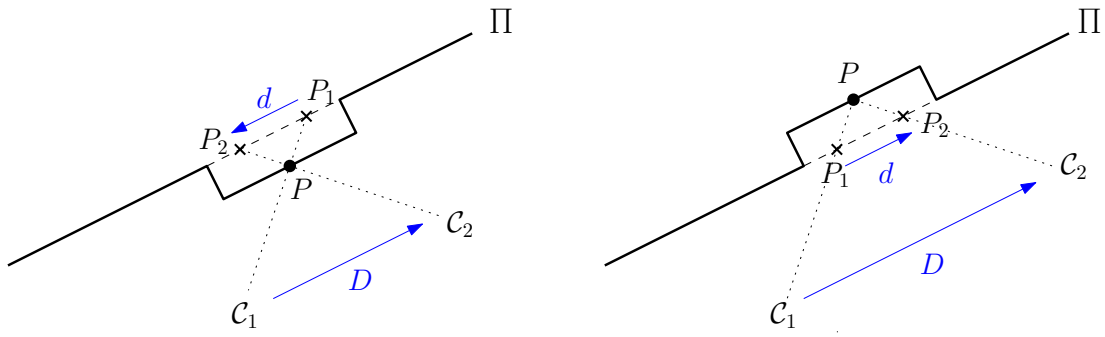


FIG. 9.2 – Le signe du mouvement apparent des pixels permet de déterminer le sens de l'extrusion

signe contraire) que le mouvement entre $\mathcal{C}_1$ et $\mathcal{C}_2$ — par rapport à Π — si l'extrusion de la façade est vers l'intérieur (resp. vers l'extérieur) du bâtiment. Plus formellement, si $P_1^\perp, P_2^\perp, \mathcal{C}_1^\perp$ et $\mathcal{C}_2^\perp$ sont les projetés respectifs de $P_1, P_2, \mathcal{C}_1$ et $\mathcal{C}_2$ sur Π, alors le signe de l'extrusion est le même que celui du produit scalaire $\overrightarrow{\mathcal{C}_1^\perp \mathcal{C}_2^\perp} \cdot \overrightarrow{P_1^\perp P_2^\perp}$.

Le problème de la localisation des micro-structures peut alors revenir à estimer un champ de mouvement dense entre les différentes textures redressées, et à déterminer les points ou zones de mouvement non nul. En effet, seules les micro-structures sont animées d'un mouvement dans cet espace. Plusieurs méthodes classiques dans la littérature peuvent être exploitées pour estimer de tels champs de mouvement, en se basant en entrée sur deux images ou plus.

Nous présentons dans ce chapitre une étude préliminaire de la faisabilité d'une telle reconstruction de la micro-structure des façades en se basant sur les images extraites de la vidéo. Une recherche des micro-structures entre deux images et basée sur le calcul de disparités est tout d'abord présentée (section 9.2). Cette recherche est ensuite étendue à un plus grand nombre d'images (section 9.3).

9.2 Estimation basée sur 2 images

Nous présentons dans cette section des méthodes visant à calculer des cartes de profondeur de façades en utilisant un couple de textures redressées (chapitre 8).

9.2.1 Calcul de disparités

Le calcul de disparités est une technique classique en vision, qui consiste à comparer deux images I_1 et I_2 pour estimer le mouvement entre les deux et en déduire la profondeur des différentes zones visualisées.
Généralement, la géométrie épipolaire ([HZ04], chapitre 9) a été estimée entre I_1 et I_2, et on sait alors que l'on peut rechercher le correspondant de tout point d'une des images

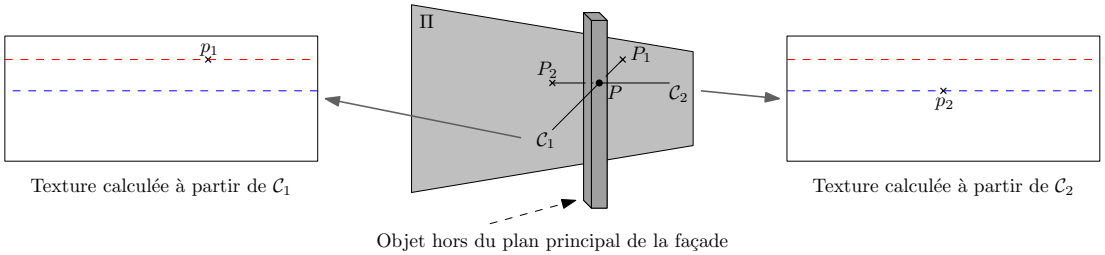


FIG. 9.3 – L'utilisation des textures redressées n'autorise pas la recherche de correspondances sur une même ligne. ici le point P, hors du plan principal de la façade, se projette en deux points p_1 et p_2 dans les textures redressées qui ne sont pas sur la même ligne, car $\mathcal{C}_1$ et $\mathcal{C}_2$ ne sont pas à la même hauteur.

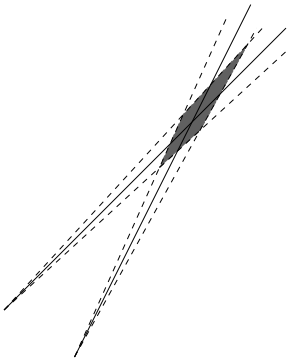
le long d'une ligne (épipolaire) dans l'autre image. On peut également rectifier I_1 et I_2 en fonction de cette géométrie épipolaire pour que les lignes de recherche en question correspondent aux lignes des images.

9.2.2 Adéquation entre redressement de textures et recalage épipolaire

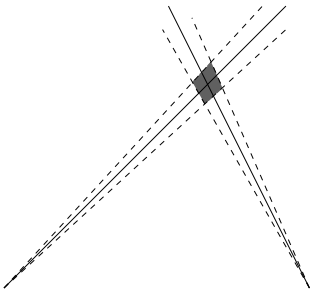
Soit I_1^e et I_2^e des images correspondant à I_1 et I_2, et rectifiées en fonction de la géométrie épipolaire, de telle sorte que les droites épipolaires des images soient parallèles aux lignes des images. Alors tout point p de coordonnées $[x\ y]^\top$ dans I_1^e aura pour coordonnées $[(x+d)\ y]^\top$ dans I_2^e, et réciproquement. Cette technique permet de limiter la complexité de la recherche de correspondances entre les images.
En utilisant les textures redressées, cette recherche sur une même ligne n'est pas toujours valide, mais elle en est assez proche en général. Cette contrainte est violée par exemple dans le cas où l'altitude de la caméra par rapport à la façade est différente pour les deux images considérées (voir figure 9.3). Nous recherchons donc les appariements de points non pas sur les lignes correspondantes d'une image à l'autre, mais sur un ensemble (réduit) de lignes.

9.2.3 Sélection des couples d'images

Le calcul de disparité est basé sur deux images. Connaissant les poses de la caméra par rapport aux façades que l'on veut analyser (partie II), on recherche alors à estimer quelles sont les meilleurs couples d'images à sélectionner pour estimer la disparité.
Plusieurs critères déterminent le bon conditionnement ou non de la configuration géométrique des caméras par rapport à la façade visée pour le calcul de disparités. On peut relier ce conditionnement à la qualité de reconstruction 3D atteignable en fonction de la position des caméras et du bruit dans les points mis en correspondance (figure 9.4). En premier lieu se trouve la *ligne de base*, ou *baseline*, qui correspond à l'écartement des deux caméras. On cherche à maximiser cette *baseline*. L'angle entre les lignes de visée des caméras influe également sur ce conditionnement. On cherche également à le maximiser.



L'angle entre les lignes de vue des caméras ainsi que la *baseline* sont faibles : la zone d'incertitude est importante.



L'angle entre les lignes de vue des caméras ainsi que la *baseline* sont plus importants : la zone d'incertitude devient plus faible.

FIG. 9.4 – Une bonne configuration géométrique des caméras pour la reconstruction 3D implique une *baseline* et un angle entre les caméras suffisamment importants pour minimiser la zone d'incertitude sur les profondeurs à estimer

Soit $b_{i,j}$ la *baseline* entre les caméras $\mathcal{C}_i$ et $\mathcal{C}_j$, et θ_i l'angle entre la normale à la façade visée et la ligne de vue de la caméra $\mathcal{C}_i$. On note également $C_{i,j}$ la couverture entre les deux images, c'est-à-dire le pourcentage de zone commune visible (les façades ne sont pas forcément entièrement visibles dans les images, et certaines parties des textures sont alors indéfinies).
Une mesure de "correlation géométrique" $\mathfrak{c}_{i,j}$ est estimée à partir de la couverture $C_{i,j}$, des angles θ_i, θ_j, et de la *baseline* $b_{i,j}$:

$$\mathfrak{c}_{i,j} = C_{i,j} * (\theta_i - \theta_j) * b_{i,j} \tag{9.1}$$

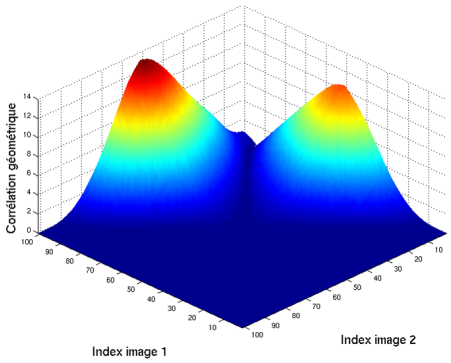
Le couple d'images sélectionné est celui qui maximise les valeurs $\mathfrak{c}_{i,j}$ calculées.

L'algorithme de sélection du couple d'images pour le calcul de disparités est résumé dans l'algorithme 9.1. La figure 9.5 illustre les scores de corrélation obtenus pour différentes façades. On indique en abscisse les indices i et j ; les scores de corrélation sont indiqués en ordonnée, et ne sont calculés que dans le cas $j > i$. Les images sélectionnées — celles dont les indices correspondent au pic maximal de chaque graphe — sont présentées sur les figures 9.6, 9.7, 9.8, 9.9 et 9.10.

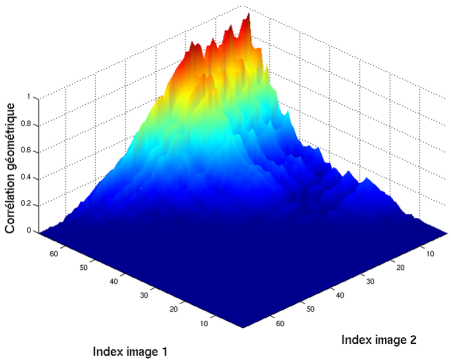
9.2.4 Coupes de graphes

De nombreuses techniques existent dans la littérature pour calculer la disparité entre deux images [SS02]. L'algorithme de Kolmogorov basé sur des coupes de graphes (*graph cuts* en anglais) fait partie des méthodes les plus performantes en la matière[i] [KZ01].

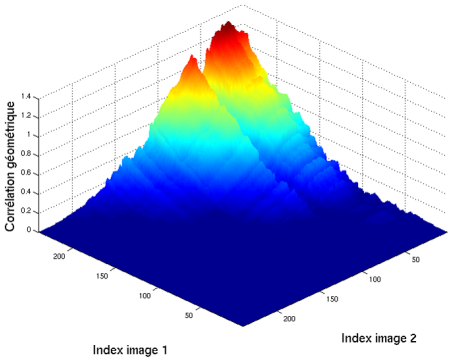
[i]Une implantation de la méthode est disponible en ligne : `http://www.adastral.ucl.ac.uk/~vladkolm/software.html`



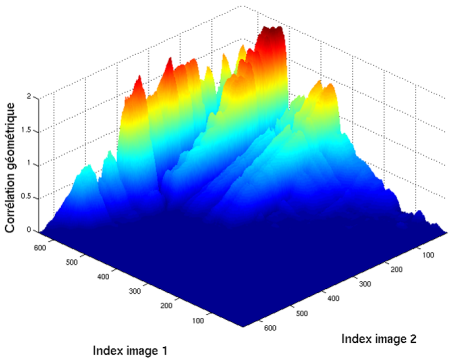
(a) Séquence *Synth-tex*, façade 001-00



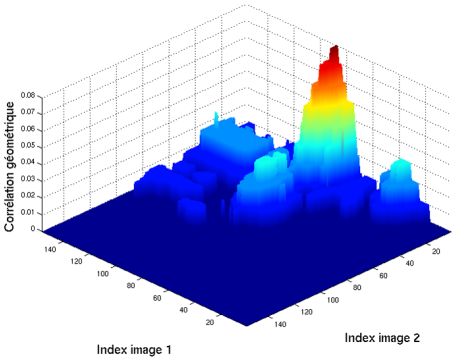
(b) Séquence *Beaulieu 1*, façade 153-08



(c) Séquence *Beaulieu 2*, façade 256-01



(d) Séquence *Ifsic*, façade 286-01



(e) Séquence *Ifsic*, façade 286-02

FIG. 9.5 – Cartes de *corrélation géométrique* permettant de déterminer quels couples d'images seront sélectionnés pour le calcul de disparités.

Objectif
Sélectionner automatiquement deux textures de façade redressées pour calculer les disparités entre elles.

Algorithme

1. Pour tout couple i, j de textures ($j > i$), calculer la couverture $C_{i,j}$
2. $\forall i, j$ si $C_{i,j} > C_{th}$ alors $\mathfrak{c}_{i,j} = C_{i,j} * (\theta_i - \theta_j) * b_{i,j}$
3. Sélectionner le couple i, j qui maximise la mesure $\mathfrak{c}_{i,j}$

ALG 9.1: Sélection d'un couple d'images pour le calcul de disparités

Cet algorithme a été appliqué sur des couples d'images sélectionnées automatiquement avec la méthode décrite dans la section 9.2.3.

Pour nos tests, la fenêtre de recherche de correspondance est centrée sur le pixel considéré, et a une taille de 31 pixels de large pour 3 pixels de haut. On a vu que la recherche pouvait être limitée en hauteur. Ici cette fenêtre de recherche réduite a également l'avantage de limiter grandement la complexité algorithmique. L'algorithme de Kolmogorov permet également de détecter les zones d'occultation. Il remplit alors ces zones automatiquement en leur assignant une valeur de disparité qui est fonction des disparités calculées dans leur voisinage non occulté.

Nous présentons sur les figures 9.6, 9.7, 9.8, 9.9 et 9.10 des résultats de calcul de disparité avec cet algorithme basé sur les coupes de graphes. On peut voir sur la première ligne des figures le couple d'images sélectionné automatiquement pour le calcul de disparités. Les cartes de disparité sont présentées sur la deuxième ligne, avec les occultations détectées en rouge[ii] (image de gauche), et avec les zones d'occultation remplies automatiquement par l'algorithme de Kolmogorov (image de droite). Les zones claires correspondent à une extrusion vers l'extérieur de la façade, et les zones sombres à une extrusion vers l'intérieur. L'intensité des niveaux de gris correspond aux valeurs de disparité estimées.

On peut remarquer que bien que très bruitées, on retrouve généralement dans ces cartes de disparité la forme globale de la façade, avec ses fenêtres, etc. Le sens des disparités — quand elles sont suffisamment marquées — est bien estimé. Par contre les valeurs de disparité elles-mêmes sont à ne considérer que de façon qualitative. Par exemple, sur la figure 9.7, on peut voir plusieurs zones avec des disparités différentes, alors qu'elles sont toutes à la même profondeur.
Les occultations non modélisables (par exemple l'arbre sur la figure 9.6) sont globalement bien détectées, mais les valeurs de disparité estimées dans ces zones sont parfois très bruités et peu pertinentes. Ces zones ayant déjà été détectées lors de la fusion de

[ii] Les occultations correspondent aux pixels visibles uniquement dans une des deux images. Un terme de visibilité est intégré à cette fin dans la phase de minimisation globale de l'énergie permettant de calculer les disparités.

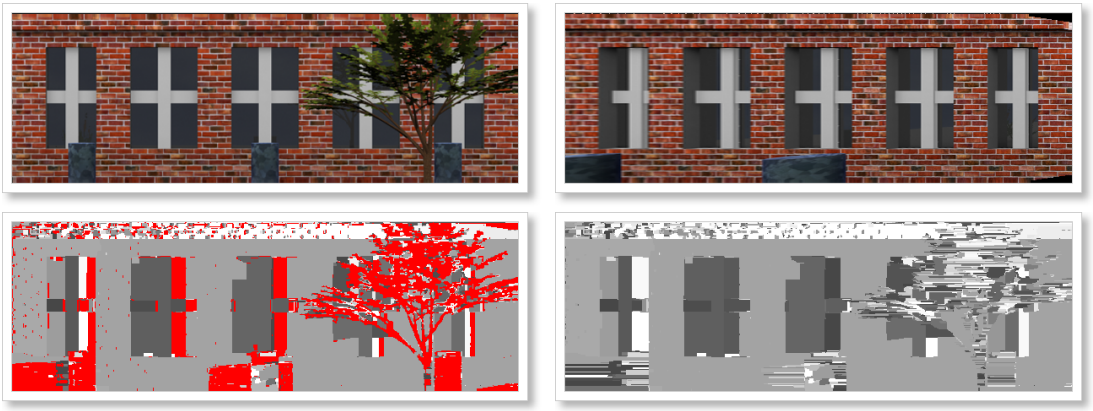


FIG. 9.6 – Calcul des disparités par coupes de graphes, séquence *Synth-tex*, façade 001-00

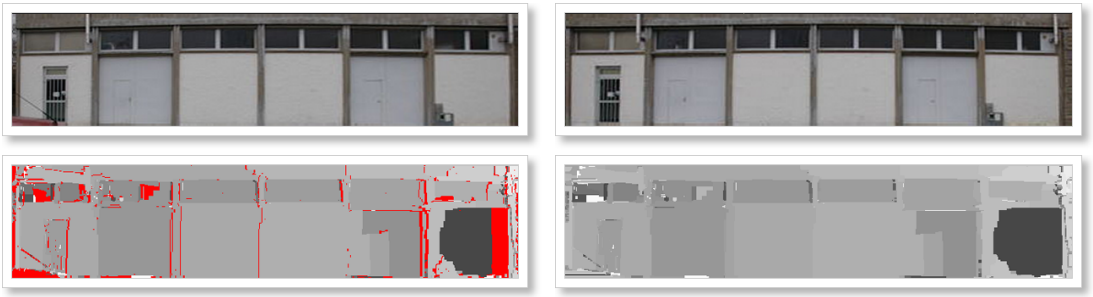


FIG. 9.7 – Calcul des disparités par coupes de graphes, séquence *Beaulieu 1*, façade 153-08

textures (chapitre 8), il serait intéressant d'étudier un éventuel calcul de disparité uniquement sur les zones valides. Le résultat du remplissage des zones de disparité inconnue serait peut-être moins bruité.

9.3 Estimation basée sur N images

L'estimation du mouvement au sein des textures redressées peut également se calculer en utilisant une estimation du flot optique. Ici, tous les couples de textures consécutives sont considérés, et un système de vote permet de déterminer le mouvement dominant des différentes zones de la texture. Le calcul du flot optique est basé sur la méthode décrite dans [HS81], et dont le code est disponible dans [Int].

Soit $\mathcal{T}_t$ et $\mathcal{T}_{t+1}$ deux textures de façades. Le flot optique nous donne pour chaque texel un vecteur de déplacement $\delta_{u,v}^{t,t+1}$ dont on ne conserve que la composante horizontale. Soit $d_{u,v}^{t,t+1}$ le signe de ce déplacement (au sens du signe de l'extrusion, cf section 9.1). On calcule alors pour chaque texel le sens du mouvement dominant $d_{u,v}$ comme la moyenne des $d_{u,v}^{t,t+1}$ estimés entre les images successives, sous la contrainte que le poids $w^{f,u,v}$

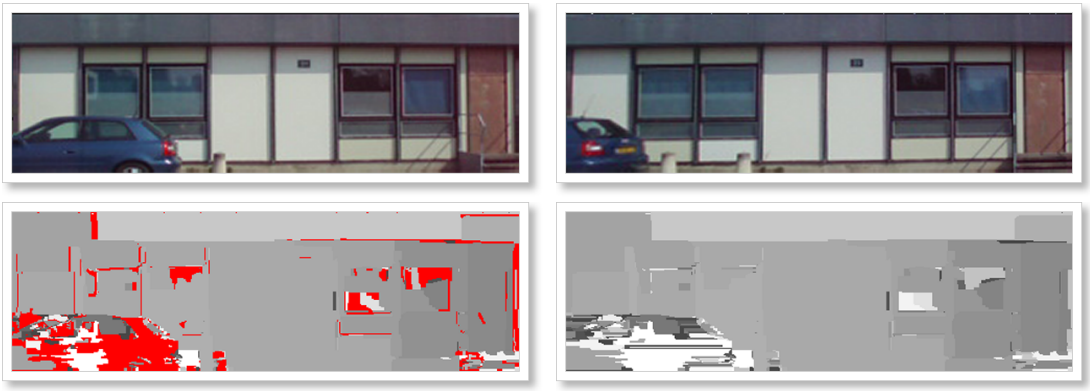


FIG. 9.8 – Calcul des disparités par coupes de graphes, séquence *Beaulieu 2*, façade 256-01

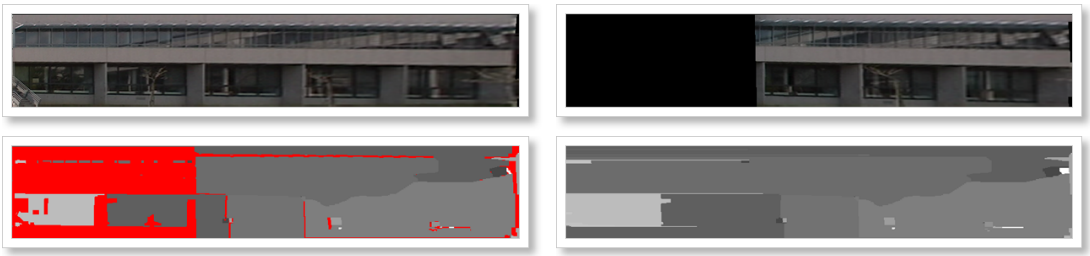


FIG. 9.9 – Calcul des disparités par coupes de graphes, séquence *Ifsic*, façade 286-01

associé à chaque texel et calculé lors de la fusion de textures (chapitre 8) soit non nul :

$$d_{u,v} = \frac{1}{N-1} \sum_{t=1}^{N} d_{u,v}^{t,t+1}$$

$$\text{avec } d_{u,v}^{t,t+1} \text{ la composante horizontale du vecteur estimé } \delta_{u,v}^{t,t+1} \text{ si } w^{f,u,v} > 0 \qquad (9.2)$$

$$\text{et } d_{u,v}^{t,t+1} = 0 \text{ sinon}$$

Etant donné que l'on cherche le signe uniquement du mouvement, et que $d_{u,v}$ est une valeur réelle dans $[-1, 1]$, on régularise ces valeurs par seuillage :

$$\begin{array}{ll} d_{u,v} = -1 & \text{si } d_{u,v} < -\lambda \\ d_{u,v} = 1 & \text{si } d_{u,v} > \lambda \\ d_{u,v} = 0 & \text{sinon} \end{array} \qquad (9.3)$$

Nous avons fixé expérimentalement la valeur de λ à $0,2$.

Ce calcul est généralement très bruité (figure 9.11(a)). Nous appliquons alors un filtre médian sur l'image résultante pour supprimer au moins une partie de ce bruit (figure 9.11(b)). L'inconvénient de cette méthode est que l'information structurelle de la façade, comme les bords de fenêtres, n'est pas conservée après application du filtre. On

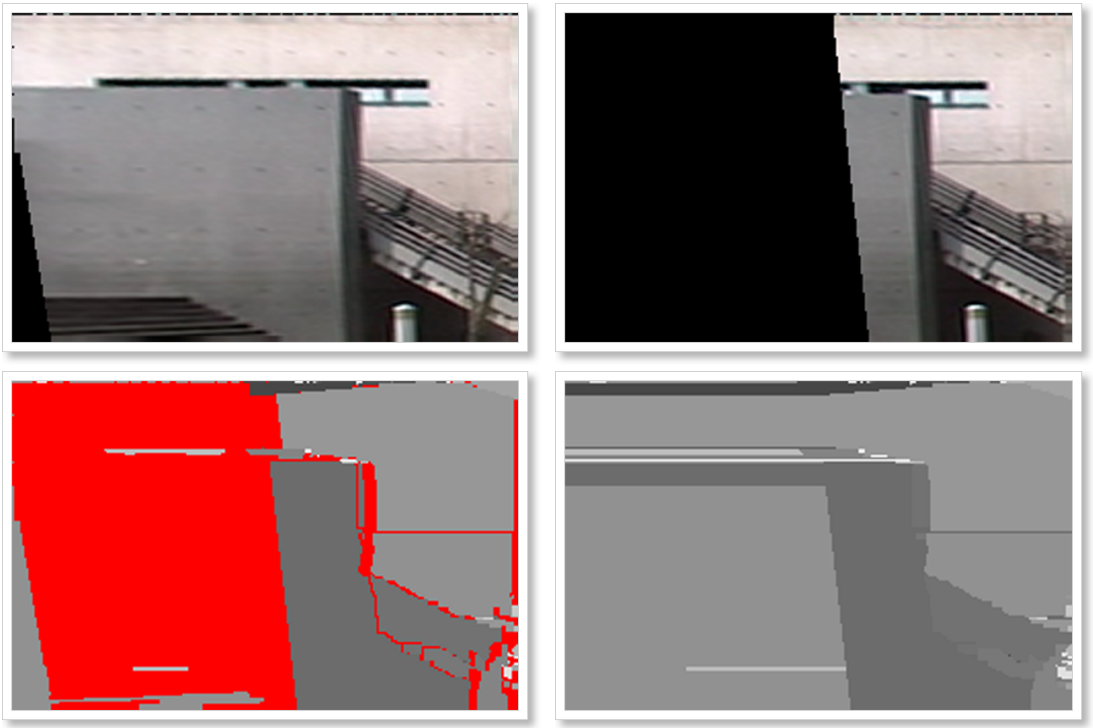
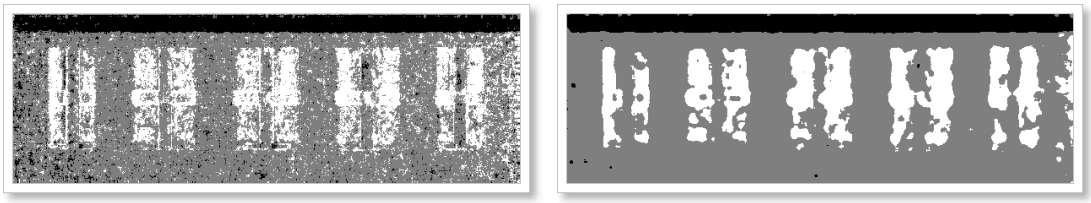


FIG. 9.10 – Calcul des disparités par coupes de graphes, séquence *Ifsic*, façade 286-02

pourrait cependant se baser éventuellement sur l'application de filtres bilatéraux pour conserver les contours [TM98], ou sur une segmentation de la texture de façade calculée précédemment, et appliquer à chaque zone la valeur majoritaire dans la carte du mouvement.
On notera également que si les résultats de la figure 9.11 semblent exploitables, c'est principalement parce que les zones de micro-structure sont bien marquées et que la caméra est proche de la façade. Avec des données réelles, cet algorithme échoue et les cartes estimées ne contiennent généralement que du bruit (voir figure 9.12).
On pourra également noter que d'autres types d'analyse de l'ensemble des champs de mouvements estimés ont été testés, mais sans succès. Par exemple, le calcul de la médiane du signe du mouvement pour l'estimation du signe du mouvement dominant discrimine assez bien les zones de microstructures, mais assigne un mouvement non nul à tout pixel de l'image. Les zones dans le plan principal de la façade sont de fait très bruitées (voir figure 9.13).

9.4 Conclusion

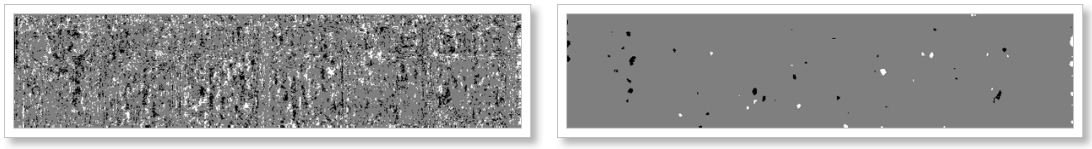
Nous avons présenté dans ce chapitre des travaux préliminaires sur la reconstruction des micro-structures de façades en estimant des cartes d'élévation correspondantes, en utilisant les textures de façades redressées. Comme attendu, une telle estimation basée



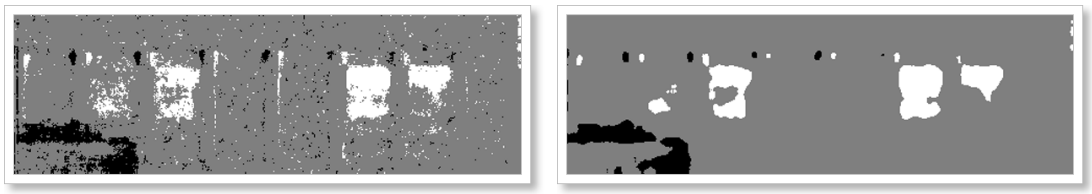
(a) Signe du mouvement seuillé (b) Application d'un filtre médian (de rayon 3)

FIG. 9.11 – Micro-structures détectées en utilisant le flot optique. les zones blanches correspondent à une extrusion positive de la façade, les zones noires à une extrusion négative, et les zones grises aux parties dans le plan principal de la façade. 9.11(a) : le seuillage du sens de mouvement donne une image très bruitée. 9.11(b) : application d'un filtre médian pour limiter ce bruit.

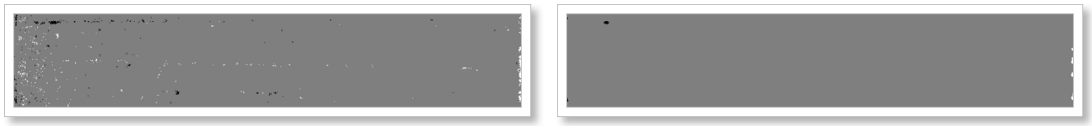
uniquement sur la mise en correspondance de points entre les textures s'avère extrêmement difficile et imprécise.
Il nous paraît de ce fait beaucoup plus approprié d'envisager des méthodes basées sur une analyse sémantique de la façade, comme la détection préliminaire des portes, fenêtres, etc. Les outils présentés dans ce chapitre resteraient cependant utilisables, mais sous la contrainte de ne les exploiter que dans certaines zones segmentées des façades. Une telle combinaison d'approches a montré de bon résultats par exemple dans les travaux de [KSK06] pour le calcul de disparités, ou ceux de Muller [MZWG07] pour le calcul de micro-structures de façades en utilisant une seule image (voir également section 7.3).
On pourrait également envisager une approche orientée sur une mise en correpondance avec des modèles. La géométrie de la façade étant estimée par une technique de bas-niveau (comme la coupe de graphe ou l'estimation du flot optique), on pourrait détecter dans cette géométrie estimée la présence d'un modèle particulier de fenêtre, porte, etc. disponible dans une base de modèle connus, le modèle choisi étant celui qui minimise une distance donnée entre deux modèles 3D. Les zones de micro-structures détectées seraient alors directement remplacées par les modèles reconnus.



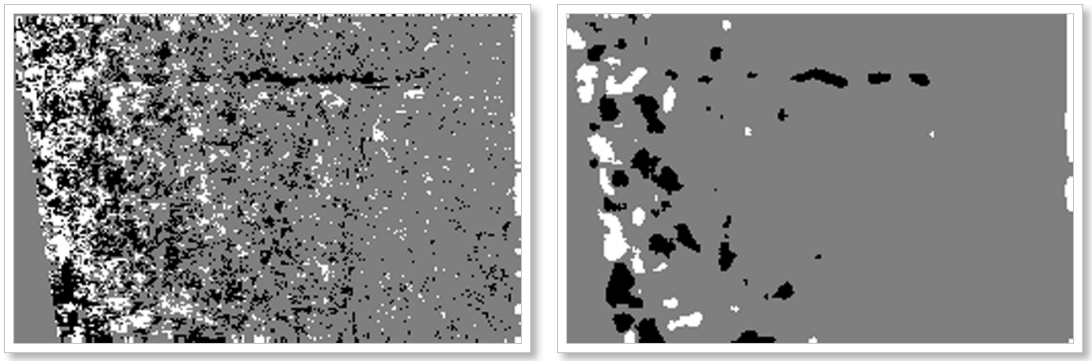
(a) Façade 153-08



(b) Façade 256-01



(c) Façade 286-01



(d) Façade 286-02

FIG. 9.12 – L'algorithme échoue à détecter les micro-structures sur des images réelles (colonne de gauche : signe du mouvement seuillé ; colonne de droite : application d'un filtre médian)

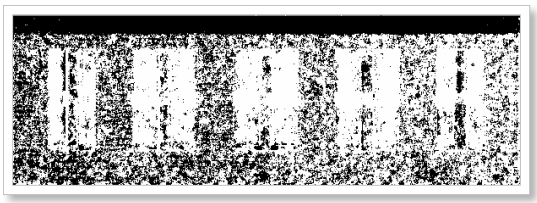


FIG. 9.13 – La sélection du signe du mouvement dominant en utilisant la médiane assigne un mouvement à tout pixel de la façade

Conclusion

The art of quotation is the art of those who don't know how to think by themselves - **Voltaire**

Synthèse

L'étude présentée se situe au croisement de la vision par ordinateur, du traitement d'images et de la synthèse d'images. Nous avons présenté un schéma de recalage entre des modèles 3D simples et non texturés de bâtiments avec des vidéos. Une méthode automatique exploitant ce recalage pour calculer les textures des bâtiments, ainsi que leur micro-structure, a été également présentée. Les différentes parties de ce schéma de recalage et de reconstruction ont été développées, puis testées sur un ensemble de séquences de synthèse et réelles. Nous faisons maintenant la synthèse de l'étude.

Recalage entre modèles 3D et vidéos

La première partie s'attache tout d'abord au traitement des données en entrée (modèles 3D issus d'une base SIG, mesures GPS, et vidéos), puis au recalage entre les modèles 3D et les vidéos correspondantes.

- La première étape consiste à prétraiter les données fournies en entrée. Les mesures GPS sont converties dans le repère de la base SIG (et donc dans le repère des modèles 3D), puis sont interpolées pour être échantillonnées à la même fréquence que la vidéo. La caméra virtuelle modélisant la caméra d'acquisition des vidéos est supposée non calibrée. Nous proposons donc soit d'utiliser les paramètres constructeur, soit d'exploiter un recalage manuel entre un bâtiment 3D et une image de la vidéo pour calculer les paramètres internes de la caméra.
- Une méthode de recalage automatique entre les modèles 3D et la première image est ensuite proposée. Elle se décompose en deux étapes. Dans un premier temps, un recalage approximatif est calculé en estimant les positions relatives de la caméra entre la première image et une image clé sélectionnée dans la vidéo, puis en identifiant ces positions aux mesures GPS. Dans un deuxième temps, une mise en correspondance

robuste de lignes (basée RANSAC) entre la projection du modèle et la première image permet de calculer précisément le recalage souhaité par asservissement visuel virtuel.
- Une troisième étape permet de calculer le recalage pour toutes les images restantes. Cette étape se base sur la mise en correspondance entre des points du modèle 3D et les points 2D correspondants suivis dans les différentes images de la vidéo. Un algorithme d'asservissement visuel virtuel robuste basé sur une M-estimation assure le recalage entre la projection du modèle et les images correspondantes.

Calcul des textures des façades

La deuxième étape de notre schéma de reconstruction de bâtiments est ensuite présentée. Elle consiste à exploiter le recalage images-modèle pour calculer les textures des façades visibles de la vidéo.

- Le recalage entre les images et la projection du modèle est tout d'abord exploitée pour extraite de celles-ci toutes les textures des façades visibles au moins partiellement. Les textures sont calculées dans le repère du plan principal de chaque façade. Lors de cette étape, on extrait également les information de visibilité et de résolution spatiale de tous les pixels des textures.
- Un algorithme de fusion de textures est ensuite présenté. Le calcul de la texture finale de chaque façade consiste à combiner tous les mêmes pixels de l'ensemble de textures extraites. La méthode proposée permet la gestion des occultations modélisables, des occultations non modélisables, ainsi que des différences de résolution spatiale entre les différentes textures utilisées en entrée. Cette fusion a été évaluée et validée sur des images de synthèse et réelles.

Calcul de la micro-structure des façades

Pour la dernière étape de nos travaux, nous présentons une étude préliminaire sur la reconstruction des micro-structures des bâtiments à partir des images acquises, les calculs étant effectués sur les textures redressées utilisées pour la fusion à l'étape précédente.

- Une première approche présentée consiste à sélectionner un couple d'images automatiquement en fonction de la configuration géométrique des caméras par rapport à la façade visée pour estimer des disparités à l'aide de coupes de graphes. Le sens des zones de disparité est globalement bien estimé bien que bruité, mais parfois incohérent : deux zones ayant en théorie une disparité identique sont mesurées comme ayant des disparités différentes.
- Une deuxième approche utilise toutes les textures redressées. Un champ de mouvement dense est estimé entre chaque couple de textures successives à l'aide d'une estimation du flot optique. Seul le signe du mouvement de chaque pixel est conservé, et le signe final du mouvement du chaque pixel est déterminé par seuillage de la moyenne des mouvements élémentaires. Les cartes résultantes sont assez bruitées. Les résultats sont assez encourageants sur des images de synthèse où la façade est vue d'assez près, mais presque inexploitables dans le cas où la façade considérée se trouve trop loin de la caméra.

Discussion

Nous avons vu au cours de cette étude l'intérêt que peut apporter l'ajout d'informations photométriques et géométriques réalistes à des modèles 3D de bâtiment simples pour des applications de visualisation, navigation virtuelle ou réalité augmentée. Cependant, la simplicité du système d'acquisition de données en entrée complexifie la phase de calcul des textures et de la micro-structure des façades de bâtiments. A des fins purement de reconstruction 3D précise — ou de raffinement de la reconstruction — il serait plus pertinent d'acquérir des données plus précises et calibrées. Par exemple, des mesures laser permettraient d'obtenir aisément la micro-structure des façades à reconstruire. Il faudrait donc bien séparer deux applications possibles. Nous avons d'un côté le recalage des données SIG avec une vidéo dont l'acquisition est non contrainte, pour des applications de réalité virtuelle et augmentée. Une simple vidéo acquise conjointement avec des données GPS est ici suffisante. D'un autre côté, si l'on vise explicitement la reconstruction réaliste des bâtiments, alors un système d'acquisition plus perfectionné et contraint est plus pertinent.
Cependant, notre système d'acquisition simple peut être avantageux pour la reconstruction si l'on imagine que celle-ci se fait sur un modèle communautaire (comme c'est actuellement le cas pour le système Google SketchUp®, ou les encyclopédies en ligne de type *wiki* dans un domaine tout à fait différent). Ainsi, avec l'apparition de récepteurs GPS sur des PDA ou smartphones de dernière génération, et l'amélioration des capteurs CCD embarqués sur de tels appareils, on peut imaginer une reconstruction à grande échelle de zones urbaines par de simples utilisateurs lambda.

Perspectives

Ces développements et les problèmes posés au cours de l'étude nous amènent à envisager les développements suivants à court et long termes afin d'améliorer le schéma :

- Lors de la phase du calcul de pose approximatif pour la première image, l'image clé utilisée pour calculer la pose relative des caméras est déterminer arbitrairement. Dans l'idéal, elle devrait être sélectionnée automatiquement. Une mesure d'erreur basée sur la mesure du résidu épipolaire devrait permettre de déterminer quelle image sélectionner pour calculer la pose relative des caméras de façon suffisamment pertinente.
- Lors le calcul de la pose précise de la caméra pour la première image, deux degrés de liberté sont fixés pour l'estimation de la rotation, alors qu'un seul n'est pas déterminable en utilisant la mise en correspondance entre les position caméra exprimées d'un côté dans le repère UTM et de l'autre dans le repère de la première image. Un choix plus pertinent de l'image clé devrait permettre de lever le dernier degré de liberté que l'on contraint (rotation selon l'axe de visée de la caméra).
- Une des volontés fortes à l'origine de ces travaux a été de développer des algorithmes qui combinent au mieux simplicité et robustesse. C'est dans cette optique que les primitives utilisées dans la phase du suivi du recalage sont des points. Or, connaissant

la pose initiale de la caméra pour la première image, on pourrait également utiliser des lignes, que ce soient celles correspondant aux frontières de façades ou des lignes dans les images dont on calcule les paramètres dans le modèle 3D. Le surcoût induit par le suivi et le calcul de pose utilisant des lignes en plus des points ne semble pas si important, et l'algorithme gagnerait certainement beaucoup en robustesse.

- Etant donné que les différentes textures de façades fusionnées entre elles proviennent d'images acquises depuis des points de vue différents, on a vu que la résolution spatiale de celles-ci est plus ou moins bonne selon en fonction de la projection perspective considérée. Or l'affaiblissement de l'influence de ces pixels de basse résolution pour la fusion de textures semble n'avoir que peu d'impact dans le calcul de la texture finale quand il est utilisé après la phase de suppression des pixels correspondant aux occultations non modélisables. Il serait intéressant d'étudier les causes de ce phénomène, et le cas échéant de conclure sur l'utilité ou non de cette phase lors du calcul des textures.
- Le principal défaut de l'algorithme de fusion de textures est que les zones de micro-structure très texturées sont parfois floues après reconstruction du fait de la parallaxe dans les textures de façades utilisées en entrée. Il serait alors intéressant d'exploiter le calcul de la micro-structure sur les zones en question pour recalculer la texture finale.
- Le calcul des micro-structures des façades de bâtiments en se basant uniquement sur le mouvement apparent des pixels dans les images nous paraît assez peu réalisable dans des conditions d'acquisitions génériques (caméra loin de la façade par exemple). Ils nous paraît beaucoup plus réaliste d'envisager en amont des méthodes de plus haut niveau, basées sur une segmentation des façades et une reconnaissance sémantique et structurelle de leur géométrie locale. Des travaux récents [MZWG07] on d'ailleurs démontré la viabilité de telles approches, même en utilisant une unique image.

Quatrième partie

Annexes

Annexe A

Séquences de test

Nous présentons dans cette annexe les différentes séquences de test — réelles ou de synthèse — utilisées pour tester les différents algorithmes présentés dans ce manuscrit.

- A.1 : Séquence *Synth-rec*, 600 × 400 pixels, 30 Hz, 101 images, couleur. Cette séquence de synthèse représente un traveling le long d'une façade, visible dans toutes les images. La rotation est assez faible. Cette séquence est utilisée pour tester l'algorithme de suivi du recalage. Elle existe avec un trajet de caméra fluide ou bruité, et avec la présence ou non d'objets occultants non modélisés.
- A.2 : Séquence *Synth-tex*, 600×400 pixels, 30 Hz, 101 images, couleur. Cette séquence de synthèse représente un traveling associé à une forte rotation le long d'une façade. Celle-ci n'est pas entièrement visible dans toutes les image, et de nombreux objets occultants non modélisés sont présents. Elle est utilisée pour tester l'algorithme de fusion de textures.
- A.3 : Séquence *Beaulieu 1*, 400×300 pixels, 3 Hz, 69 images, couleur. Cette séquence a été acquise à l'aide d'un appareil photo numérique en mode rafale. Le mouvement global est une translation le long d'une unique façade, visible dans toutes les images. Il y a peu de spécularités et pas d'objets occultants.
- A.4 : Séquence *Ifsic*, 400 × 300 pixels, 25 Hz, 650 images, couleur. Cette séquence a été acquise à main levée à l'aide d'un caméscope numérique. Le mouvement, qui ne vise aucune façade en particulier, est général et assez bruité. Trois façades sont visibles au cours de la séquences, elles ne sont visibles que dans certaines images.
- A.5 : Séquence *Beaulieu 2*, 320 × 240 pixels, 15 Hz, 240 images, couleur. Cette séquence, acquise à l'aide d'un appareil photo numérique compact en mode vidéo, représente un traveling le long d'une façade en présence de nombreuses spécularités et objets occultants. Elle est visible dans toutes les images.

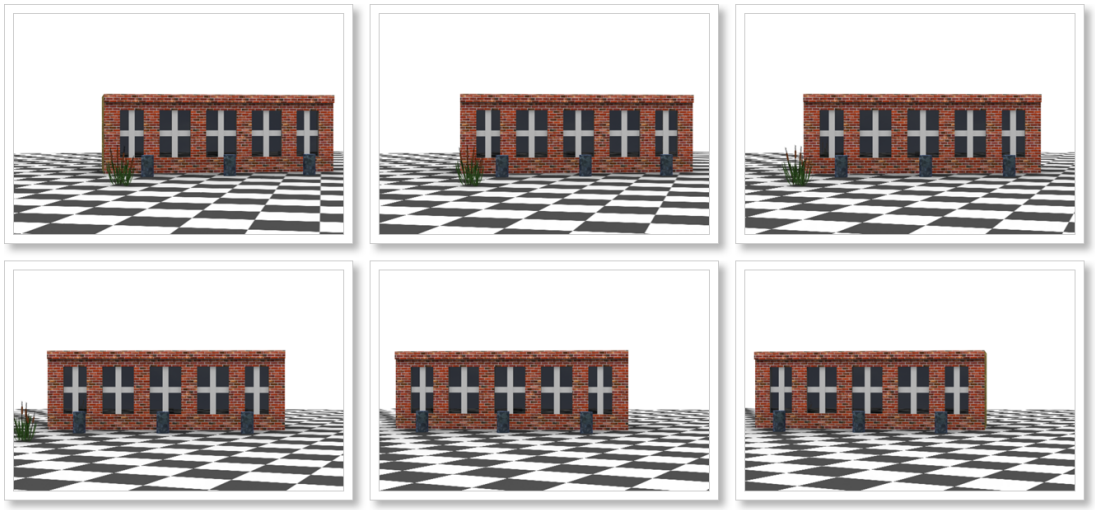


FIG. A.1 – Séquence *Synth-rec*

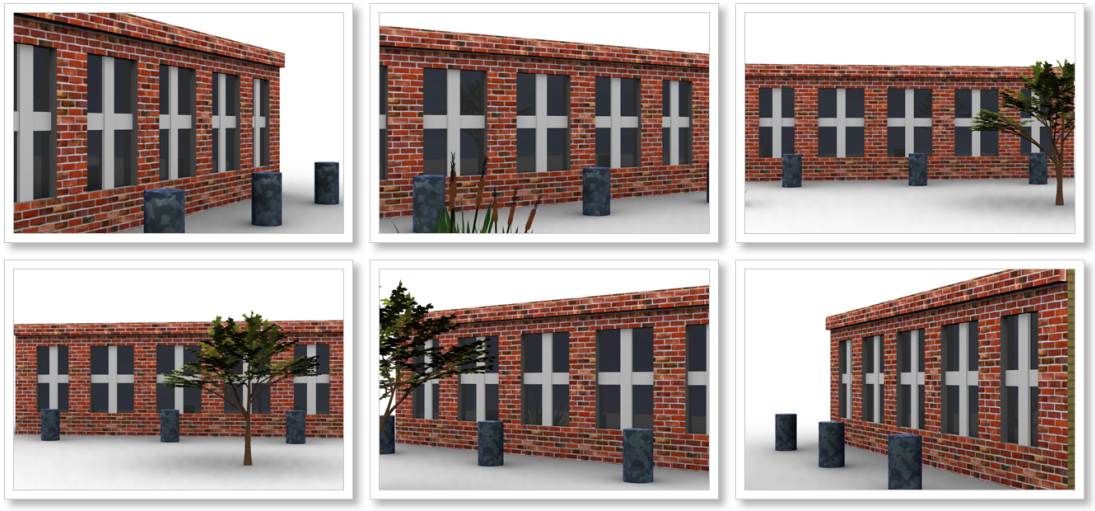


FIG. A.2 – Séquence *Synth-tex*

FIG. A.3 – Séquence *Beaulieu 1*

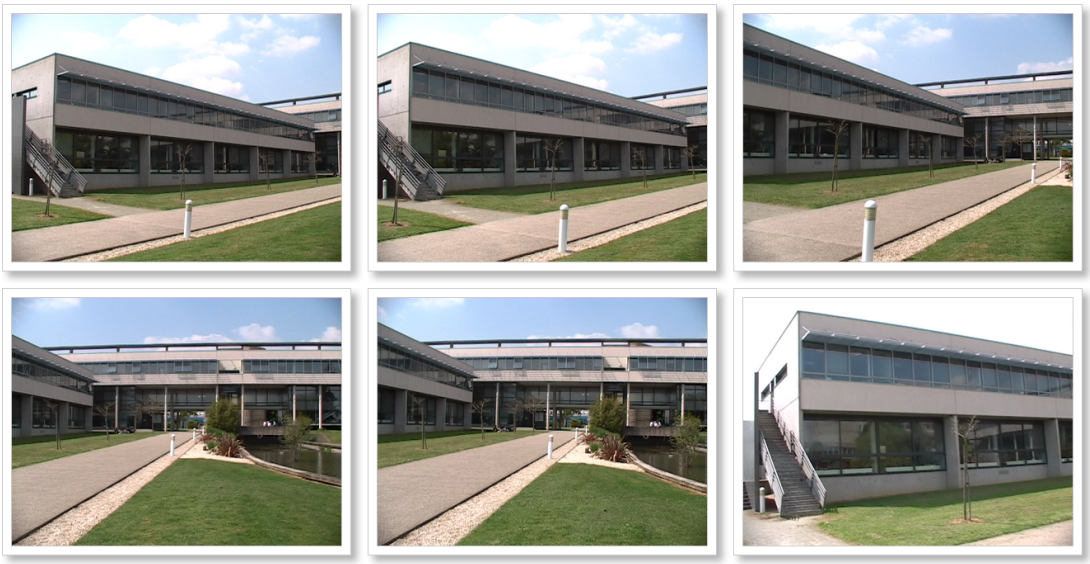


FIG. A.4 – Séquence *Ifsic*

FIG. A.5 – Séquence *Beaulieu 2*

Annexe B

Conversion des coordonnées polaires dans le repère UTM

Les mesures de positionnement effectuées par GPS sont données en coordonnées polaires, sous la forme $(latitude, longitude, altitude)$. Les modèles 3D de bâtiments extraits de la base SIG sont quand à eux décrits dans le repère UTM (*Universal Transverse Mercator*), qui correspond à une projection cylindrique de la surface terrestre. Ce repère est métrique. Les mesures GPS sont donc converties dans ce repère pour donner une position des différentes caméras dans le repère des modèles 3D. Le système géodésique WGS34 (qui décrit certaines données sur la surface non sphérique de la Terre) est utilisé dans la conversion. Les différents symboles utilisés pour la conversion sont donnés dans le tableau B.1.

Nous cherchons tout d'abord à estimer S, qui est la longueur de l'arc méridional passant par le point mesuré (*i.e.* la distance le long de la surface terrestre depuis l'équateur jusqu'à ce point).

$$\begin{cases} S &= A'lat - B'\sin(2lat) + C'\sin(4lat) - D'\sin(6lat) + E'\sin(8lat) \\ A' &= a\left[1 - n + \frac{5}{4}(n^2 - n^3) + \frac{81}{64}(n^4 - n^5)\ldots\right] \\ B' &= \frac{3}{2}an\left[1 - n + \frac{7}{8}(n^2 - n^3) + \frac{55}{64}(n^4 - n^5)\ldots\right] \\ C' &= \frac{15}{16}an^2\left[1 - n + \frac{3}{4}(n^2 - n^3)\ldots\right] \\ D' &= \frac{35}{48}an^3\left[1 - n + \frac{11}{16}(n^2 - n^3)\ldots\right] \\ E' &= \frac{315}{51}an^4\left[1 - n\ldots\right] \end{cases} \tag{B.1}$$

Les coordonnées UTM X (*easting*) et Y (*northing*) sont alors calculées de la façon suivante :

Symbole	Signification
lat	latitude en radians
lon	longitude en radians
lon_0	méridien central
$p = lon - lon_0$	écart de longitude au méridien central
$a = 6378137$	rayon équatorial en mètres
$b = 6356752.3142$	rayon polaire en mètres
$k_0 = 0.9996$	facteur d'échelle le long de $long_0$
$e = \sqrt{1 - (\frac{b}{a})^2}$	excentricité de la coupe elliptique de la Terre
$e'^2 = (\frac{ea}{b})^2 = \frac{e^2}{1-e^2}$	utilisé pour simplifier les formules
$n = \frac{a-b}{a+b}$	utilisé pour simplifier les formules
$\nu = \frac{a}{\sqrt{1-e^2 \sin^2(lat)}}$	distance du point mesuré à l'axe polaire

TAB. B.1 – Symboles utilisés pour la conversion GPS → UTM

$$
\begin{aligned}
&\bullet\ Y = K_1 + K_2 p^2 + K_3 p^4 \\
&\qquad \text{où} \quad \circ\ K_1 = S k_0 \\
&\qquad\qquad \circ\ K_2 = \frac{\nu k_0 \sin^2 1'' \sin(lat)\cos(lat)}{2} \\
&\qquad\qquad \circ\ K_3 = \frac{\left(\nu k_0 \sin^4 1'' \sin(lat)\cos^3(lat)\right)\left(5 - \tan^2(lat) + 9e'^2\cos^2(lat) + 4e'^4\cos^4(lat)\right)}{24} \\
&\bullet\ X = K_4 p + K_5 p^3 \\
&\qquad \text{où} \quad \circ\ K_4 = \nu k_0 \sin 1'' \cos(lat) \\
&\qquad\qquad \circ\ K_5 = \frac{\left(\nu k_0 \sin^3 1'' \cos^3(lat)\right)\left(1 - \tan^2(lat) + e'^2\cos^2(lat)\right)}{6}
\end{aligned}
\tag{B.2}
$$

Bibliographie

[BÖ4] Jan Böm. Multi-image fusion for occlusion-free façade texturing. In *International Society for Photogrammetry and Remote Sensing Congress*, 2004.

[BKB+01] Alexander Bornik, Konrad F. Karner, Joachim Bauer, Franz Leberl, and Heinz Mayer. High-quality texture reconstruction from multiple views. *Journal of Visualization and Computer Animation*, 12(5) :263–276, 2001.

[Can86] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6) :679–698, 1986.

[CDS03] S. Cornou, M. Dhome, and P. Sayd. Architectural reconstruction with multiple views and geometric constraints. In *British Machine Vision Conference*, 2003.

[CMC06] A.I. Comport, E. Marchand, and F. Chaumette. Statistically robust 2d visual servoing. *IEEE Trans. on Robotics*, 22(2) :415–421, April 2006.

[CMPC06] A.I. Comport, E. Marchand, M. Pressigout, and F. Chaumette. Real-time markerless tracking for augmented reality : the virtual visual servoing framework. *IEEE Trans. on Visualization and Computer Graphics*, 12(4) :615–628, July 2006.

[Com05a] A.I. Comport. *Robust real-time 3D tracking of rigid and articulated objects for augmented reality and robotics.* PhD thesis, Université de Rennes 1, Mention informatique, September 2005.

[Com05b] A.I. Comport. *Robust real-time 3D tracking of rigid and articulated objects for augmented reality and robotics.* PhD thesis, Université de Rennes 1, Mention informatique, September 2005.

[CPT04] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based inpainting. *IEEE Trans. Image Processing*, 13(9) :1200–1212, 2004.

[DD95] Daniel F. Dementhon and Larry S. Davis. Model-based object pose in 25 lines of code. *Int. J. Comput. Vision*, 15(1-2) :123–141, 1995.

[DDDS04] Philip David, Daniel Dementhon, Ramani Duraiswami, and Hanan Samet. Softposit : Simultaneous pose and correspondence determination. *Int. J. Comput. Vision*, 59(3) :259–284, 2004.

[Del34] Boris Delaunay. Sur la sphère du vide. *Bulletin of Academy of Sciences of the USSR*, pages 793–800, 1934.

[Der90] R. Deriche. Fast algorithms for low-level vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(1) :78–87, 1990.

[DH72] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1) :11–15, 1972.

[DTM96a] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs : a hybrid geometry- and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20. ACM Press, 1996.

[DTM96b] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs : a hybrid geometry- and image-based approach. Technical report, University of California at Berkeley, 1996.

[EBM02] Ahmed F. Elaksher, James S. Bethel, and Edward M. Mikhail. Reconstructing 3d building wireframes from multiple images. In *Proceedings of the ISPRS Commission III Symposium on Photogrammetric Computer Vision*, pages A–091 ff, September 2002.

[Fau93] Olivier Faugeras. *Three-dimensional computer vision : a geometric viewpoint.* MIT Press, Cambridge, MA, USA, 1993.

[FB81] Martin A. Fischler and Robert C. Bolles. Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6) :381–395, 1981.

[FL94] P. Fua and Y. G. Leclerc. Using 3-dimensional meshes to combine image-based and geometry-based constraints. In *ECCV '94 : Proceedings of the third European conference on Computer Vision (Vol. II)*, pages 281–291, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[FZ03a] Christian Frueh and Avideh Zakhor. Automated reconstruction of building facades for virtual walk-thrus. In *Proceedings of the SIGGRAPH 2003 Conference on Sketches & Applications*, pages 1–1. ACM Press, 2003.

[FZ03b] Christian Frueh and Avideh Zakhor. Constructing 3d city models by merging ground-based and airborne views. In *CVPR03*, pages II : 562–569, 2003.

[FZ04] Christian Frueh and Avideh Zakhor. An automated method for large-scale, ground-based city model acquisition. *IJCV*, 60(1) :5–24, October 2004.

[Gal02] Franck Galpin. *Représentation 3D de séquences vidéo ; Schéma d'extraction automatique d'un flux de modèles 3D, applications à la compression et à la réalité virtuelle.* PhD thesis, Thèse de doctorat en Informatique, Université de Rennes 1, France, jan 2002.

[Gen92] Donald B. Gennery. Visual tracking of known three-dimensional objects. *Int. J. Comput. Vision*, 7(3) :243–270, 1992.

[GSB05] J.-F. Vigueras Gomez, G. Simon, and M.-O. Berger. Calibration errors in augmented reality : A practical study. In *Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2005.

[HEH05] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Geometric context from a single image. In *International Conference of Computer Vision (ICCV)*, volume 1, pages 654 – 661. IEEE, October 2005.

[HMS02] Alexandra D. Hofman, Hans-Gerd Maas, and André Streilein. Knowledge-based building detection based on laser scanner data and topographic map information. In *Proceedings of the ISPRS Technical Commission III Syposium on Photogrammetric Computer Vision*, September 2002.

[HS81] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3) :185–203, August 1981.

[HS88] Chris Harris and Mike Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[HS97] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In *CVPR '97 : Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, page 1106, Washington, DC, USA, 1997. IEEE Computer Society.

[Hub81] P.J. Huber. *Robust Statistics.* Wiley, New York, 1981.

[HYN06] Jinhui Hu, Suya You, and Ulrich Neumann. Integrating lidar, aerial image and ground images for complete urban building modeling. In *3DPVT '06 : Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06)*, pages 184–191, Washington, DC, USA, 2006. IEEE Computer Society.

[HZ04] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN : 0521540518, second edition, 2004.

[Int] Intel. Open source computer vision library.

[KBKS02] K. Karner, J. Bauer, A. Klaus, and K. Schindler. Metropogis : a city information system. In *ICIP02*, pages III : 533–536, 2002.

[KDN93] David Koller, Kostas Danilidis, and Hans-Hellmut Nagel. Model-based object tracking in monocular image sequences of road traffic scenes. *Int. J. Comput. Vision*, 10(3) :257–281, 1993.

[KN95] A. Kosaka and G. Nakazawa. Vision-based motion tracking of rigid objects using prediction of uncertainties. In *Proceedings of the International Conference on Robotics and Automation*, pages 2637–2644, Nagoya, Japan, May 1995.

[KR05a] Thommen Korah and Christopher Rasmussen. Randomized view planning and occlusion removal for mosaicing building facades. In *IEEE International Conference on Intelligent Robots and Systems*, 2005.

[KR05b] Thommen Korah and Christopher Rasmussen. Spatiotemporal inpainting for recovering texture maps of partially occluded building facades. In *International Conference on Image Processing*, 2005.

[KR06] Thommen Korah and Christopher Rasmussen. Improving spatiotemporal inpainting with layer appearance models. In *International Symposium on Visual Computing*, 2006.

[KSK06] Andreas Klaus, Mario Sormann, and Konrad Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *ICPR '06 : Proceedings of the 18th International Conference on Pattern Recognition*, pages 15–18, Washington, DC, USA, 2006. IEEE Computer Society.

[KZ01] Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions via graph cuts. In *ICCV*, Vancouver, Canada, July 2001.

[LF06] Vincent Lepetit and Pascal Fua. Monocular model-based 3d tracking of rigid objects. *Found. Trends. Comput. Graph. Vis.*, 1(1) :1–89, 2006.

[Low92] David G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Int. J. Comput. Vision*, 8(2) :113–122, 1992.

[LS05] Lingyun Liu and Ioannis Stamos. Automatic 3d to 2d registration for the photorealistic rendering of urban scenes. In *CVPR '05 - Volume 2*, pages 137–143, Washington, DC, USA, 2005. IEEE Computer Society.

[Mac02] John H. MacNicol. *A Study of Satellite Navigation, Dilution of Precision, and Positioning Techniques for Use On and Around the Moon.* PhD thesis, Airforce Institute of Technology, 2002.

[MBB+01] Heinz Mayer, Alexander Bornik, Joachim Bauer, Konrad F. Karner, and Franz Leberl. Multiresolution texture for photorealistic rendering. In *Spring Conference on Computer Graphics*, 2001.

[MC02] E. Marchand and F. Chaumette. Virtual visual servoing : a framework for real-time augmented reality. *Computer Graphics Forum*, 21(3) :289–298, September 2002.

[MKB01] Heinz Mayer, Konrad F. Karner, and Alexander Bornik. Reconstruction of textured models from multiple views. Technical report, VRVis, 2001.

[MZWG07] Pascal Müller, Gang Zeng, Peter Wonka, and Luc Van Gool. Image-based procedural modeling of facades. *Proceedings of ACM SIGGRAPH 2007 / ACM Transactions on Graphics*, 26(3), 2007.

[ODD96] Denis Oberkampf, Daniel DeMenthon, and Larry S. Davis. Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, 63(3) :495–511, 1996.

[OR05] Diego Ortin and Fabio Remondino. Occlusion-free image generation for realistic texture mapping. In *International Society for Photogrammetry and Remote Sensing International Workshop '3D-ARCH 2005 : Virtual Reconstruction and Visualization of Complex Architectures'*, Venice, Italy, August 2005.

[POF98] Pierre Poulin, Mathieu Ouimet, and Marie-Claude Frasson. Interactively modeling with photogrammetry. In *Proceedings of Eurographics Workshop on Rendering 98*, pages 93–104, 1998.

[RD06] Gerhard Reitmayr and Tom Drummond. Going out : robust model-based tracking for outdoor augmented reality. In *ISMAR*, pages 109–118, 2006.

[RTC+06] J.Y. Rau, T.A. Teo, L.C. Chen, F. Tsai, K.H. Hsiao, and W.C. Hsu. Integration of gps, gis and photogrammetry for texture mapping in photo-realistic city modeling. In *Pacific-Rim Symposium on Image and Video Technology*, pages 1283–1292, 2006.

[RTHN97] Andreas Ruf, M. Tonko, Radu P. Horaud, and Hans-Hellmut Nagel. Visual tracking of an end-effector by adaptive kinematic prediction. In *Proceeding of the International Conference on Intelligent Robots and Systems, IROS 97*, volume 2, pages 893–898, Grenoble, France, September 1997. IEEE/RSJ.

[SB02] Gilles Simon and Marie-Odile Berger. Pose estimation for planar structures. *IEEE Comput. Graph. Appl.*, 22(6) :46–53, 2002.

[SB03] Konrad Schindler and Johachim Bauer. A model-based method for building reconstruction. In *Proceedings of the ICCV Workshop on Higher-Level Knowledge in 3D Modeling and Motion*, 2003.

[SFZ00] Gilles Simon, Andrew Fitzgibbon, and Andrew Zisserman. Markerless tracking using planar structures in the scene. In *Proc. International Symposium on Augmented Reality*, pages 120–128, October 2000.

[SRB+04] M. Sormann, B. Reitinger, J. Bauer, A. Klaus, and K. Karner. Fast and detailed 3d reconstruction of cultural heritage. In *International Workshop on Vision Techniques applied to Rehabilitation of City Centres*, 2004.

[SS02] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal on Computer Vision*, 47(1-3) :7–42, 2002.

[ST94] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, 1994.

[SV00] Ildiko Suveg and George Vosselman. 3d reconstruction of building models. In *Proceedings of the XIXth ISPRS Congress, volume XXXIII-B3 of International Archives of Photogrammetry and Remote Sensing*, 2000.

[SV02] Ildiko Suveg and George Vosselman. Localization and generation of building models. In *Proceedings of the ISPRS Technical Commission III Syposium on Photogrammetric Computer Vision*, September 2002.

[TAB+03] Seth Teller, Matthew Antone, Zachary Bodnar, Michael Bosse, Satyan Coorg, Manish Jethwa, and Neel Master. Calibrated, registered images of an extended urban area. *Int. J. Comput. Vision*, 53(1) :93–107, 2003.

[Tai00] Franck Taillandier. Texture and relief estimation from multiple georeferenced images. Master's thesis, Ecole Polytechnique, 2000.

[TC98] Seth Teller and Satyan Coorg. Automatic extraction of textured vertical facades from pose imagery. Technical Report MIT LCS TR-729, Massachusetts Institute of Technology, January 1998.

[TC99] Seth Teller and Satyan Coorg. Extracting textured vertical facades from controlled close-range imagery. In *CVPR99*, pages I : 625–632, 1999.

[TDC00] Philip H.S. Torr, Anthony Dick, and Roberto Cipolla. Layer extraction with a bayesian model of shapes. In *ECCV00*, pages II : 273–289, 2000.

[Tel97] Seth Teller. Automatic acquisition of hierarchical, textured 3d geometric models of urban environments : Project plan. In *DARPA97*, pages 767–770, 1997.

[Tel98] Seth Teller. Automated urban model acquisition : Project rationale and status. In *DARPA98*, pages 455–462, 1998.

[TK91] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report, Carnegie Mellon University, 1991.

[TM98] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV '98 : Proceedings of the Sixth International Conference on Computer Vision*, page 839, Washington, DC, USA, 1998. IEEE Computer Society.

[TR03] F. Tupin and M. Roux. Detection of building outlines based on the fusion of sar and optical features. *PandRS*, 58(1) :71–82, June 2003.

[Tsa02] Jaan-Rong Tsay. A concept and algorithm for 3d city surface modeling. In *PCV02*, page B : 283, 2002.

[WSB05] Marta Wilczkowiak, Peter Sturm, and Edmond Boyer. Using geometric constraints through parallelepipeds for calibration and 3d modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2) :194–207, 2005.

[WTT+02] Xiaoguang Wang, Stefano Totaro, Franck Taillandier, Allen Hanson, and Seth Teller. Recovering facade texture and microstructure from real-world images. In *European Conference on Computer Vision*, 2002.

[WZ02a] Tomáš Werner and Andrew Zisserman. Model selection for automated architectural reconstruction from multiple views. In *Proceedings of the British Machine Vision Conference*, pages 53–62, 2002.

[WZ02b] Tomáš Werner and Andrew Zisserman. New techniques for automated architectural reconstruction from photographs. In *Proceedings of the 7th European Conference on Computer Vision-Part II*, pages 541–555. Springer-Verlag, 2002.

[Zha00] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11) :1330–1334, 2000.

Publications

If you don't find it in the index, look very carefully through the entire catalogue - **Sears, Roebuck, and Co., Consumer's Guide, 1897**

Congrès internationaux

[1] Gaël Sourimant, Luce Morin, Kadi Bouatouch - **Gps, Gis and Video Registration for Building Reconstruction** - *ICIP 2007, 14th IEEE International Conference on Image Processing* - San Antonio, Texas, USA, September, 2007.

[2] Gaël Sourimant, Luce Morin, Kadi Bouatouch - **Gps, Gis and Video fusion for urban modeling** - *CGI 2007, 25th Computer Graphics International Conference* - Petropolis, RJ, Brazil, May, 2007.

[3] Gaël Sourimant, Luce Morin - **Model-Based Video Coding with Generic 2D and 3D Models** - *Proceedings of Mirage 2005 (Computer Vision / Computer Graphics Collaboration Techniques and Applications)* - INRIA Rocquencourt, France, March, 2005.

Congrès nationaux

[4] Thomas Colleu, Gaël Sourimant, Luce Morin - **Une méthode d'initialisation automatique pour le recalage de données SIG et vidéo** - *COmpression et REpresentation des Signaux Visuels (CORESA 2007)* - Montpellier, France, Novembre, 2007.

[5] Gaël Sourimant, Luce Morin, Kadi Bouatouch - **Vers une reconstruction 3D de zones urbaines : mise en correspondance de données Gps, Sig et Vidéo** - *GRETSI 2007, 21e Colloque en Traitement su Signal et des Images* - Troyes, France, Septembre, 2007.

[6] Gaël Sourimant, Luce Morin, Kadi Bouatouch - **Fusion de données Gps, Sig et Vidéo pour la reconstruction d'environnements urbains** - *Orasis 2007, onzième congrès francophone des jeunes chercheurs en vision par ordinateur* - Obernai, France, Juin, 2007.

[7] Gaël Sourimant, Luce Morin - **Codage vidéo basé sur des modèles génériques 2D et 3D** - *Proceedings of Orasis 2005* - Fournols, France, Mai, 2005.

Table des figures

Résumé

Cette thèse s'inscrit dans le cadre de la reconstruction tridimensionnelle de zones urbaines, à l'aide de données GPS, SIG et Vidéo. L'objectif est de raffiner des modèles 3D simples et géo-référencés de bâtiments, extraits d'une base SIG (Systèmes d'Information Géographique) en les mettant en correspondance avec des séquences d'images acquises au sol, la position de la caméra étant estimée par GPS.
La mise en correspondance des données vidéo et des modèles 3D correspondants par asservissement visuel virtuel robuste est tout d'abord présentée. Le but est ici de retrouver, pour chaque image de la vidéo, la pose géo-référencée précise de la caméra d'acquisition (position et orientation), de telle façon que les modèles 3D se projettent exactement sur les images de bâtiments correspondantes.
Les textures des bâtiments visibles dans la vidéo sont alors extraites. Un nouvel algorithme pour la fusion de textures de façades de bâtiments basé sur une analyse statistique de la couleur des texel est présenté. Il permet entre autres la suppression dans la texture finale de tous les objets occultants n'appartenant pas à la façade elle-même.
Enfin, nous présentons une étude préliminaire sur l'extraction des détails géométriques de chaque façade. Connaissant les poses de la caméra pour toutes les images, un calcul de disparité par coupe de graphe ou flot optique est effectuée dans l'espace texture. Les micro-structures des façades peuvent alors être retrouvées en utilisant ces cartes de disparité.

Abstract

This thesis presents a new scheme for 3D buildings reconstruction, using GPS, GIS and Video datasets. The goal is to refine simple and geo-referenced 3D models of buildings, extracted from a GIS database (Geographic Information System). This refinement is performed thanks to a registration between these models and the video. The GPS provides rough information about the camera location.
First, the registration between the video and the 3D models using robust virtual visual servoing is presented. The aim is to find, for each image of the video, the geo-referenced pose of the camera (position and orientation), such as the rendered 3D models projects exactly on the building images in the video.
Next, textures of visible buildings are extracted from the video images. A new algorithm for façade texture fusion based on statistical analysis of the texels color is presented. It allows to remove from the final textures all occluding objects in front of the viewed building façades.
Finally, a preliminary study on façades geometric details extraction is presented. Knowing the pose of the camera for each image of the video, a disparity computation using either graph-cuts or optical flow is performed in texture space. The micro-structures of the viewed façades can then be recovered using these disparity maps.