Reconstruction 3D de scènes d'intérieurs à partir de photographies

Yohann Salaün

UNIVERSITÉ PARIS-EST

École Doctorale Paris-Est
Mathématiques & Sciences et Technologies
de l'Information et de la Communication

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PARIS EST
Domaine : Traitement du Signal et des Images

présentée par **Yohann SALAÜN**
pour obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ PARIS EST

Reconstruction 3D de scènes d'intérieur à partir de photographies

---

Soutenue publiquement le 6 juillet 2017 devant le jury composé de :

Agnès DESOLNEUX	École Normale Supérieure de Cachan	Rapporteur
Renaud MARLET	École des Ponts ParisTech	Directeur de Thèse
Pascal MONASSE	École des Ponts ParisTech	Co-Directeur de Thèse
Luc ROBERT	Bentley Systems	Examinateur
Peter STURM	Inria Grenoble	Rapporteur
Bruno VALLET	Institut national de l'information Géographique et Forestière	Examinateur

École des Ponts ParisTech
LIGM-IMAGINE
6, Av Blaise Pascal - Cité Descartes
Champs-sur-Marne
77455 Marne-la-Vallée cedex 2
France

Université Paris-Est Marne-la-Vallée
École Doctorale Paris-Est MSTIC
Département Études Doctorales
6, Av Blaise Pascal - Cité Descartes
Champs-sur-Marne
77454 Marne-la-Vallée cedex 2
France

"De toute façon, ça ne fonctionnera pas..."
Thésard anonyme (2013-2017)

Résumé

Les méthodes actuelles de photogrammétrie permettent de reconstruire en 3D de nombreux objets et/ou scènes à partir de leurs photographies. Pour ce faire, les méthodes classiques détectent des points saillants dans les images et les mettent en correspondance entre plusieurs images. Ces correspondances permettent d'obtenir une information de calibration entre les différentes positions d'où la scène a été photographié. Une fois ces positions déterminées, il est alors possible d'obtenir une reconstruction dense de la scène en triangulant les parties de la scène vues dans plusieurs images.

La détection et la mise en correspondance de points saillants jouent un rôle crucial dans le procédé de reconstruction 3D. C'est pourquoi certaines scènes ou objets sont encore difficiles à reconstruire à partir de méthode de photogrammétrie. C'est notamment le cas des scènes d'intérieur, souvent constituées de larges pans de mur peu texturés où la détection et la mise en correspondance de points sont souvent défaillantes. De plus, la très grande présence de motifs planaires, cas dégénérés des méthodes de calibration usuelles, rend ces scènes très difficiles à calibrer. Dans cette thèse, nous nous intéressons à l'utilisation de segments de droites pour compenser la faible efficacité des points dans le cas des scènes d'intérieur.

Dans un premier temps, nous introduisons une méthode de détection de segments plus robuste au manque de contraste des scènes d'intérieur. C'est une méthode multi-échelle qui permet également d'obtenir de bon résultats sur des images de haute et basse résolution. Nous utilisons pour cela des critères inspirés des méthodes *a contrario* pour éviter la spécification de nombreux paramètres.

Nous présentons ensuite une méthode de calibration bifocale utilisant à la fois les segments et les points pour obtenir une méthode robuste au manque de texture et à la planarité de la scène tout en conservant la précision des méthodes de points. Nous introduisons aussi une variante du RANSAC *a contrario* pour déterminer les cas où vaut mieux utiliser les segments plutôt que les points pour calibrer.

Enfin, pour compenser le faible recouvrement entre photographies fréquent lors des prises de vues de scènes d'intérieur, nous introduisons une méthode de calibration multi-vue utilisant des contraintes de coplanarité entre segments sans avoir besoin de contraintes trifocales. Nous modifions les contraintes trifocales usuelles pour les ajouter, quand elles sont disponibles, aux contraintes de coplanarité et ainsi obtenir une méthode plus robuste mais aussi précise que les méthodes usuelles.

Mots-clefs

reconstruction 3D ; calibration bifocale ; calibration multi-vue ; points saillants ; segments ; estimation robuste ; détection de ligne ; a contrario ; AC-RANSAC ; coplanarité ; parallélisme.

Abstract

The 3D reconstruction of many objects and/or scenes from their photographies is made possible by current photogrammetry methods. To do so, usual methods detect salient points in every pictures and then match them between each pictures. These matches then give information on the position of every camera that took a picture of the scene. Once these positions are obtained, a dense reconstruction of the scene can be obtained by triangulating the features seen in different pictures.

Point detection and matching are crucial parts of these 3D reconstruction methods. That is why some scenes or objects are still hard to reconstruct in 3D with photogrammetry methods. Indoor scenes belong to these difficult cases, with their lack of texture that causes point detection and matching to give poor results. Moreover, the planarity of these scenes is a degenerate case for usual calibration methods. Combined, these drawbacks explain the difficulty to calibrate such scenes. In this thesis, we explain how to use line segments to compensate for the lack of robustness of point methods in the case of indoor scenes.

First, we introduce a segment detection method that is more robust to the lack of contrast in indoor scenes. This multi-scale method also gives good results on high and low reolution images. We use criterion inspired from *a contrario* methods to avoid the use of many parameters.

We then present a bifocal calibration method that uses both line segments and points. Segments allow the method to still work in low-texture and/or planar scenes, and points allow the method to be as accurate as other point methods. To do so, we introduce an *a contrario* RANSAC variant to choose, for each scene, whether points or line segments should be used for calibration.

Finally, to deal with the lack of overlap between consecutive pictures, which is common in indoor scene datasets, we introduce a multi-view calibration method that uses coplanarity constraints between segments when there are no trifocal constraints. Moreover, we explain how to modify usual trifocal constraints to combine them, when available, with coplanarity constrains in order to obtain a method as accurate as usual methods but more robust in wide-baseline scenarios.

Keywords

3D reconstruction ; bifocal calibration ; multi-view calibration ; feature points ; feature lines ; robust estimation ; line detection ; a contrario ; AC-RANSAC ; coplanaroty ; parallelism.

Sommaire

Chapitre 1

Préambule

La numérisation ou la reproduction d'un modèle 3D d'un objet réel, voire d'un bâtiment ou même d'une ville entière permet de nombreuses applications. Un tel modèle 3D peut être utilisé pour créer des effets spéciaux dans les films ou pour rendre les jeux videos plus réalistes. De plus, avec l'avènement des imprimantes 3D, cela permet de reproduire à l'identique n'importe quel objet, de ceux utilisés dans la vie de tous les jours aux sculptures célèbres des musées. Il est également possible d'utiliser ces modèles 3D pour faire des simulations dans le cadre de l'aménagement des villes et des territoires ou de la rénovation de bâtiments. Enfin ces modèles peuvent également servir à la surveillance de certaines zones (*e.g.* érosion des bords de fleuves ou de mer,...).

FIGURE 1.1 – Reconstructions 3D d'objet réalisées à l'aide de différentes méthodes (scanner laser pour le vase, photogrammétrie pour les autres). Crédits : *Google Images*

L'un des principaux outils utilisés pour reproduire des modèles 3D est le scanner LIDAR. Le principe de cet outil est d'utiliser la réfléxivité des objets pour mesurer leur distance au scanner

à l'aide d'impulsions laser. Une telle méthode permet généralement d'obtenir un résultat très précis (*i.e.* de l'ordre du dixième de *mm* pour de petits objets) et très dense (*i.e.* pouvant dépasser les 5000 points par cm^2) mais présentent un certain nombre de défauts :

- le coût de l'appareil (souvent largement supérieur à 10 k€),
- la nécessité d'une formation pour pouvoir utiliser l'appareil,
- la complexité de mise en oeuvre pour les grandes scènes (*e.g.* recouvrement nécessitant l'utilisation de marqueurs),
- le manque de flexibilité qui le rend inutilisable pour certaines scènes difficilement accessibles et qui génère des "trous" de reconstruction dans certaines zones de la scène (*e.g.* toits, escaliers).

FIGURE 1.2 – Scanner LIDAR à gauche et dispositif Kinect à droite. Crédits : *Google Images*

Des méthodes plus récentes combinent appareils photo et sources de lumière contrôlée pour obtenir une reconstruction 3D de la scène en observant la déformation de la texture projetée. La Kinect de Microsoft a ainsi été beaucoup utilisée dans le cadre de la reconstruction 3D car son prix abordable et sa grande flexibilité font d'elle une sorte de scanner réduit. Cependant, si la plupart des inconvénients du scanner laser disparaissent, la précision de ce type de méthode est généralement très faible et l'utilisation d'une lumière contrôlée restreint son utilisation aux scènes intérieures.

Enfin la méthode que nous explorerons plus en détail dans cette thèse est celle de la photogrammétrie. L'idée de cette méthode est de prendre différentes photographies d'une même scène ou d'un même objet puis d'utiliser les recouvrements et corrélations entre images pour obtenir une reconstruction 3D de la scène. Cette méthode n'a pas les défauts du scanner LIDAR (*e.g.* faible coût, grande flexibilité, simplicité d'utilisation...) tout en conservant une précision importante (*e.g.* parfois de l'ordre du mm pour des bâtiments de plusieurs mètres).

Cependant, si les méthodes de photogrammétrie sont très précises dans de nombreux cas, il existe encore des cas de figure particulièrement difficiles qui font alors chuter la précision de la méthode parfois au point de la rendre inutilisable. Pour comprendre ces cas difficiles, il faut rentrer dans les détails de la méthode.

L'idée de la photogrammétrie est de combiner plusieurs images (deux ou plus) pour obtenir une information 3D de la scène. Pour cela les méthodes cherchent généralement des *points caractéristiques* (*e.g.* coin d'une fenêtre, sommet d'un clocher...) entre les images. En repérant suffisamment de ces points, il est alors possible d'estimer, via des calculs reposant sur la *géométrie projective*, la position des appareils photos lors des différentes prises de vues. Une fois ces positions obtenues, il est alors possible de *trianguler* la scène en utilisant la position des

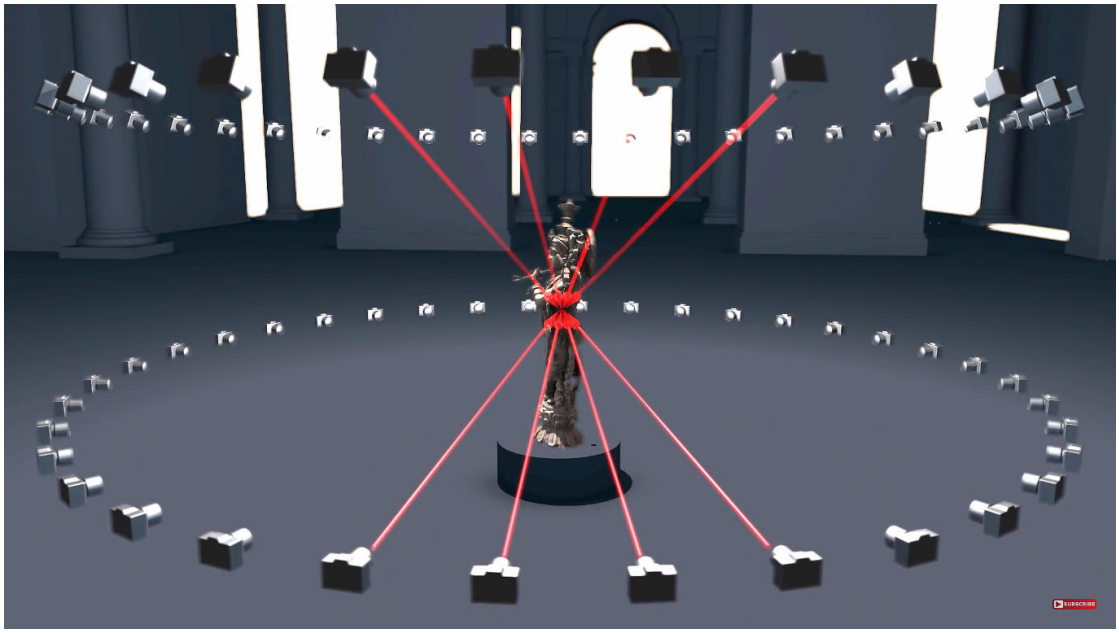


FIGURE 1.3 – Le principe de la photogrammétrie est de prendre plusieurs photographies d'une même scène ou d'un même objet avec différents points de vue. En combinant toutes ces photos, il est alors possible d'obtenir un modèle 3D de la scène. Crédits : *Google Images*

appareils photos et les observations d'un même point de la scène 3D sur différentes photographies. L'autre avantage de cette méthode est de fournir une reconstruction dense (*i.e.* maillage) et en couleur contrairement au LIDAR qui ne fournit qu'un nuage de point et une intensité de réflexion.

Même si cette présentation de la photogrammétrie est très simplifiée, un élément essentiel est que cette méthode repose sur la détection de points communs entre images. Dans le cadre de la reconstruction d'intérieurs de bâtiments, les scènes sont pauvres en points caractéristiques. En effet, ces scènes sont généralement peu texturées (*e.g.* pans de murs de couleur uniforme, peu d'objets) et les positions et angles de vues sont tels que les zones de recouvrement entre les différentes images sont limitées et peuvent subir des déformations projectives importantes. Ces cas deviennent alors très difficiles à traiter par les méthodes classiques de photogrammétrie et c'est sur ce point que nous nous concentrerons dans cette thèse.

Chapitre 2

Introduction

Les maquettes numériques 3D de bâtiments, qui rassemblent informations géométriques et informations sémantiques, sont de plus en plus utilisées dans l'industrie du bâtiment comme modèle numérique unifié pour toutes les phases du cycle de vie de la construction : conception, évaluation de la performance et simulations, planification de la construction ou de la rénovation, maintenance, etc. Pouvoir reconstruire de façon automatique la maquette numérique d'un bâtiment existant est un réel besoin pour le marché de la rénovation. Dans ce but l'acquisition de données peut se faire à l'aide d'un scanner. Cela fournit des mesures précises mais un scanner est onéreux et est inadapté pour les zones difficiles d'accès. Une alternative est d'utiliser des photographies d'intérieurs et d'extérieurs. Elle peut s'avérer utile dans deux scénarios :

- Dans une phase commerciale, avec un petit nombre de photos prises dans des conditions non nécessairement maîtrisées, une maquette approchée et partielle pourrait être reconstruite. Ce serait néanmoins suffisant pour effectuer un diagnostic rapide ainsi que les premiers métrés indicatifs.
- Davantage dans une phase d'exécution, avec un plus grand nombre de photos et dans un cadre mieux maîtrisé, une maquette de meilleure qualité et plus complète pourrait être reconstruite. Selon les besoins et les coûts induits, la photo pourrait être utilisée comme unique moyen d'acquisition, économique et facile d'usage, ou bien comme simple complément du laser pour les zones difficiles d'accès (y compris avec l'usage de photos aériennes pour la reconstruction de toits et d'éléments architecturaux invisibles du sol).

Le problème est que la reconstruction de modèles 3D à partir de photographies peut parfois échouer dans des cas difficiles et, lorsqu'elle produit un résultat, elle n'atteint pas la précision du laser. L'erreur est en pratique, et suivant les circonstances, de l'ordre de quelques millimètres, voire du centimètres, quand celle du laser est autour du dixième de millimètres.

Les questions scientifiques soulevées par ces tâches sont bien connues, mais n'ont toujours pas trouvé de réponse adéquate :

- **Absence de texture** : L'intérieur des bâtiments est généralement constitué de nombreux murs dont la texture est uniforme ou trop répétitive. Dans le premier cas, très peu de points sont détectés, dans le deuxième, la trop grande répétitivité du motif rend l'appariement ambigu un appariement de qualité. Dans tous les cas, peu d'appariements de points corrects sont trouvés et les méthodes usuelles de calibration ne peuvent pas être appliquées.
- **Changements de points de vue brutaux** : Les déformations dues aux changements de points de vue brutaux lors de franchissements de seuils quand on passe d'une pièce à une autre ne sont pas bien gérées aujourd'hui. S'appuyer sur des indices plus stables, tels que des segments de droites, permettrait des recalages jusqu'alors impossibles et améliorerait la robustesse des calibration.

- **Utilisation d'appareils grand ou très grand angle** : Pour les prises de vues intérieures, le recul nécessaire avec des appareils photo ordinaires forcerait la prise de plusieurs photos quasiment du même point de vue. L'utilisation d'appareil grand angle ou très grand angle (fisheye ou même panoptique) simplifierait alors la tâche du photographe. Cependant, l'utilisation de telles optiques nécessite une calibration interne spécifique. En particulier, la distorsion doit être corrigée, surtout dans le cas où l'on utilise des segments de droites.
- **Présence de surfaces planes** : Des problèmes de robustesse peuvent survenir dans la calibration et la reconstruction de façades et autre surfaces planes. Ces structures géométriques conduisent en effet à des équations de calibration dégénérées pour les points, classiquement utilisés dans les méthodes de calibration. Leur prise en compte nécessite donc un traitement spécifique.
- **Prises de vues dans des conditions peu maîtrisées** : Dans un scénario où les conditions de prises de vues sont plus contraintes, le nombre réduit de photos, leur recouvrement moindre, et leur plus faible qualité (cadrage, angles de vue et qualité d'image médiocres) seront autant d'obstacles supplémentaires à la production d'un résultat exploitable. Il s'agit donc de développer des méthodes robustes permettant une reconstruction proche de la réalité à partir d'un minimum d'images.

2.1 Contributions de cette thèse

Dans cette thèse, nous nous sommes intéressés à la calibration de deux ou plusieurs caméras dans le cadre des scènes d'intérieur. Partant de l'observation que les segments de droites sont généralement plus riches en information que les points dans ce type de scène, nous avons utilisé ceux-ci pour rendre nos méthodes robustes aux cas difficiles. Cependant, pour garder une méthode utilisable sur n'importe quelle scène, nous avons combiné l'utilisation de points et de segments. Les méthodes détaillées dans la suite associent donc la robustesse des segments à la précision des points tout en restant utilisables dans de nombreux cas (urbains ou non). Pour cela, nous avons utilisé des méthodes *a contrario* qui choisissent automatiquement quel *feature* (point ou segment) est le plus adapté pour une scène donnée et avec quelle paramétrisation.

Les contributions de cette thèse sont ainsi associées aux différentes étapes de la calibration multi-vue dans un cadre d'intérieur de bâtiments :

Détection de segments

Les méthodes état-de-l'art donnent généralement des résultats de bonne qualité en un temps très raisonnable (*i.e.* jusqu'au temps réel sur des images de moins de 1Mpx). Cependant, dans le cas de photographies haute-résolution (*i.e.* supérieure à 10Mpx), ces méthodes donnent souvent des résultats de qualité médiocre voire mauvaise. Dans le cas des scènes d'intérieur où le contraste est généralement faible, la qualité des résultats chute encore. Nous avons donc développé une méthode multi-échelle permettant de conserver à haute résolution la qualité des résultats obtenus sur des images de plus faible résolution. De plus, pour éviter l'utilisation de nombreux seuils lors de la détection et la fusion des segments, nous avons utilisé une approche *a contrario*.

Calibration bifocale

- Les méthodes à base de lignes sont généralement limitées car la correspondance de segments de droites entre deux vues successives ne donne pas suffisamment de contraintes sur la géométrie 3D de la scène. Ainsi, ces méthodes utilisent soit des contraintes trifocales et donc la présence de nombreux segments communs à au moins 3 vues, soit des contraintes

supplémentaires nécessitant l'utilisation d'hypothèse *Manhattan-world* (*i.e.* deux directions quelconques de la scène sont soit parallèles soit perpendiculaires). Nous avons développé une méthode de calibration à base de lignes se passant de la plupart des hypothèses superflues et ne nécessitant que 4 segments communs à deux images tout en conservant des résultats très précis.

- Les méthodes de calibration à base de points échouent généralement à cause du manque de détection faute de texture ainsi que de la planarité de la scène (cas dégénéré). Malgré ce manque de robustesse, les points permettent d'obtenir un résultat précis dans de nombreux cas et nous avons donc développé une méthode hybride points-lignes obtenant des résultats précis et robustes sur tous types de scènes.
- Pour éviter l'utilisation de nombreux seuils dans cette méthode, nous avons défini un cadre *a contrario* qui permet de choisir automatiquement lesquels, parmi lignes et points, privilégier.

Calibration multi-vue

- Pour calibrer un ensemble de n caméras, quelle que soit la méthode utilisée, il est toujours nécessaire d'utiliser des contraintes trifocales. Dans les scènes d'intérieur, il arrive que les points de vue des photographies soient trop différents et qu'il n'y ait donc pas de triplets de *features* pour contraindre la calibration. Cette absence de triplet génère alors des "trous" de calibration qui réduisent la précision et peuvent aussi aboutir à des scènes partiellement reconstruites. Nous avons donc développé une méthode utilisant des hypothèses de coplanarité entre segments pour se passer de la nécessité des triplets.
- Nous avons également reformulé les contraintes trifocales de points et de lignes pour pouvoir les combiner avec nos contraintes de coplanarité et obtenir un maximum de précision avec notre méthode.
- De même que dans le cadre de la calibration bifocale, nous avons défini un cadre *a contrario* permettant de combiner au mieux les différents types de contraintes.

2.1.1 Contributions

Les chapitres 4, 5 et 6 qui suivent ont donné lieu à des publications (soumission en cours pour le chapitre 6) :

- Conférence internationale : Robust and Accurate Line- and/or Point-Based Pose Estimation without Manhattan Assumptions, Yohann Salaun, Renaud Marlet, and Pascal Monasse, ECCV 2016 Multiscale line segment detector for robust and accurate SfM, Yohann Salaun, Renaud Marlet, and Pascal Monasse, ICPR 2016
- *Workshop* : The multiscale line segment detector, Yohann Salaun, Renaud Marlet, and Pascal Monasse, RRPR 2016
- Soumission à une conférence internationale : Robust SfM with Little Image Overlap, Yohann Salaun, Renaud Marlet, and Pascal Monasse, arXiv.org 2017 (1703.07957)

Chacune des publications est accompagnée d'un code C++ disponible sur Github

- Détecteur de segments sur image haute-résolution : MLSD, https://github.com/ySalaun/MLSD
- Calibration multi-vue à partir de points et de lignes : LineSfM, https://github.com/ySalaun/LineSfM

2.2 Organisation du manuscrit

Le sujet de cette thèse portant sur les différentes étapes de la calibration multi-vue, nous avons conservé cet ordre pour le manuscrit :

- Le chapitre 3 présente les différentes notations et méthodes existantes pour faire de la calibration multi-vue,
- le chapitre 4 présente la méthode de détection de segments multi-échelle,
- le chapitre 5 présente la méthode de calibration bifocale à l'aide de points et/ou de segments,
- le chapitre 6 présente la méthode de calibration multi-vue dans le cas d'un recouvrement trifocal limité voire absent.

Chapitre 3

Introduction à la calibration multi-vue

Nous présentons ici un certain nombre de notions de la calibration multi-vue ainsi que certains outils associés. Nous utiliserons les notions présentées plus en détail dans les chapitres suivants.

Contents

3.1 Notations

Nous résumons ici les notations utilisées dans la suite. Celles-ci seront introduites dans les sections suivantes et utilisées dans les chapitres suivants.

$\mathbb{R}^2$ / $\mathbb{P}^2$	plan (Euclidien / projectif)	section 3.2
$\mathbb{R}^3$ / $\mathbb{P}^3$	espace (Euclidien / projectif)	section 3.2
x / $\mathbf{x}$	un vecteur de $\mathbb{R}^2$ / $\mathbb{P}^2$	section 3.2
X / $\mathbf{X}$	un vecteur de $\mathbb{R}^3$ / $\mathbb{P}^3$	section 3.2
$[u]_\times$	matrice *produit-vectoriel* de $u \in \mathbb{R}^3$	section 3.2
p, q	lettres généralement utilisées pour les points	section 3.7.1
l, m	lettres généralement utilisées pour les lignes/segments	section 3.7.2
u, v	lettres généralement utilisées pour les points de fuite	section 3.7.3
(d, m)	notations de Plücker pour une ligne	section 3.3.1
(s, ω)	notations de Cayley pour une ligne	section 3.3.2
P	matrice de projection	section 3.4
K	matrice de calibration ou matrice des paramètres intrinsèques d'une caméra	section 3.4
R	matrice de rotation entre deux caméras	section 3.4
t	vecteur de translation entre deux caméras	section 3.4
$\mathcal{C}$	centre optique d'une caméra	section 3.4
E	matrice essentielle	section 3.5.1
F	matrice fondamentale	section 3.5.1
$\mathcal{T}$	tenseur trifocal	section 3.5.2
X_j	*feature* (*i.e.* point, segment de droite ou ligne) d'indice j	section 3.6
x_j^i	projection sur la caméra i du *feature* X_j	section 3.6
NFA	Nombre de Fausses Alarmes	section 3.9
$\mathcal{H}_0$	Modèle de fond	section 3.9
$\mathbb{P}_{\mathcal{H}_0}$	probabilité d'un évènement en suivant l'hypothèse $\mathcal{H}_0$	section 3.9

3.2 Espace projectif et euclidien

Par la suite, nous travaillerons en géométrie euclidienne ($\mathbb{R}^n$) ou en géométrie projective ($\mathbb{P}^n$). Nous travaillerons uniquement dans le plan ($\mathbb{R}^2$ ou $\mathbb{P}^2$) ou dans l'espace ($\mathbb{R}^3$ ou $\mathbb{P}^3$). Nous noterons les éléments de l'espace projectif en gras ($\mathbf{x}$) pour les différencier des éléments de l'espace Euclidien (x).

Le passage d'un élément de l'espace euclidien à un élément de l'espace projectif se fait de façon injective, en ajoutant 1 en dernière coordonnée. Ainsi, un vecteur $x = (X,Y)$ de $\mathbb{R}^2$ correspond au vecteur $\mathbf{x} = (X,Y,1)$ de $\mathbb{P}^2$. Réciproquement, le passage d'un élément de l'espace projectif à un élément de l'espace Euclidien se fait avec une division. Ainsi, un vecteur $\mathbf{x} = (X,Y,Z)$ de $\mathbb{P}^2$ correspond à un vecteur $x = (X/Z, Y/Z)$ de $\mathbb{R}^2$. Dans le cas où $Z = 0$ le point est alors dit *à l'infini* et n'a pas de correspondant dans l'espace Euclidien.

Notons également qu'un élément $\mathbf{x} = (X,Y,Z)$ de $\mathbb{P}^3$ a le même correspondant dans $\mathbb{R}^2$ que tous les éléments $\mathbf{y} = \lambda\mathbf{x}$ pour $\lambda \neq 0$. Pour prendre en compte cette ambiguïté, nous considérerons par la suite que deux éléments de l'espace projectif sont égaux si et seulement si leurs correspondants dans l'espace Euclidien associé sont les mêmes. Ainsi :

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{P}^n,\ \mathbf{x} = \mathbf{y} \Leftrightarrow \exists \lambda \in \mathbb{R}^\star \text{ t.q. } \mathbf{x} = \lambda\mathbf{y} \tag{3.1}$$

Par la suite, les coordonnées homogènes concerneront les points et les lignes, que nous noterons généralement p ou q, pour les points, et l ou m, pour les lignes. Nous rappelons ici certaines propriétés de $\mathbb{P}^2$ concernant les points et les lignes et qui seront fortement utilisées dans la suite :

- Le produit vectoriel ($\times$) fait correspondre à 2 points l'unique ligne passant par ces 2 points. Ce produit donne $\mathbf{0}$ ssi les 2 éléments sont proportionnels (donc égaux en géométrie projective).
- Le produit vectoriel ($\times$) fait correspondre à 2 lignes leur intersection (même à l'infini dans le cas des lignes parallèles). Ce produit donne $\mathbf{0}$ ssi les 2 éléments sont proportionnels (donc égaux en géométrie projective).
- Le produit scalaire (noté $a.b$ ou $a^\top b$) d'un point $\mathbf{p}$ et d'une ligne $\mathbf{l}$ donne accès à la distance point-ligne :

$$\delta = \|\mathbf{p}^\top \mathbf{l}\| / (|\mathbf{p}_z| \sqrt{\mathbf{l}_x^2 + \mathbf{l}_y^2}) \tag{3.2}$$

  où $\mathbf{p} = (p_x, p_y, p_z)$ et $\mathbf{l} = (l_x, l_y, l_z)$. En particulier $\mathbf{p}^\top \mathbf{l} = 0$ ssi $\mathbf{p} \in \mathbf{l}$. Notons de plus que $\mathbf{p}^\top \mathbf{l} = \infty$ lorsque $p_z = 0$ ce qui reste cohérent puisque $\mathbf{p}$ est alors *à l'infini*.

Pour des raisons de commodité nous utiliserons parfois la notation de *matrice produit-vectoriel* d'un vecteur $u \in \mathbb{R}^3$ définie par :

$$[u]_\times = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \tag{3.3}$$

où $u = (u_x, u_y, u_z)$.

Cette matrice est antisymétrique et vérifie, pour tout vecteur $u, v \in \mathbb{R}^3$, la relation :

$$[u]_\times v = u \times v \tag{3.4}$$

3.3 Représentation spatiale d'une ligne

Les points sont facilement représentés dans le plan ou dans l'espace en utilisant les coordonnées projectives ou euclidiennes et des vecteurs aux tailles correspondantes. Pour les lignes, par contre, si la représentation est aussi simple dans le plan, elle est bien plus complexe dans l'espace.

Dans l'espace, une ligne est généralement vue comme l'intersection de deux plans, soit :

$$\begin{cases} a_1x+a_2y+a_3z+a_4=0 \\ b_1x+b_2y+b_3z+b_4=0 \end{cases} \tag{3.5}$$

Si cette notation est facile à comprendre, elle est difficile à utiliser dans le cadre des relations entre une ligne et sa projection dans un plan et utilise un trop grand nombre de variables (une ligne dans l'espace n'ayant que 4 degrés de liberté).

3.3.1 Coordonnées de Plücker

Pour une ligne dans l'espace passant par un point p et de direction u, on définit les coordonnées de Plücker à l'aide du vecteur $(m,d) \in \mathbb{R}^6$:

$$\begin{cases} m= p \times d \\ d=u/||u|| \end{cases} \tag{3.6}$$

m est appelé le moment de la ligne et d représente sa direction normalisée.

Avec cette définition, on peut associer une ligne de l'espace à n'importe quel vecteur non nul de $\mathbb{R}^6$ vérifiant les conditions de Plücker :

$$||d|| = 1,\ d^\top m = 0 \tag{3.7}$$

Les coordonnées de Plücker n'ont pas l'avantage d'être minimaliste puisqu'on utilise 6 variables pour décrire 4 degrés de liberté. Cependant, elle permettent une utilisation bien plus facile lors de la projection d'une ligne sur un plan comme nous le verrons dans la section suivante.

3.3.2 Coordonnées de Cayley

Dans certains cas, il est utile d'avoir une notation minimaliste pour décrire une ligne dans l'espace (soit un vecteur de $\mathbb{R}^4$). Pour cela, nous utiliserons les coordonnées de Cayley définies à partir des coordonnées de Plücker. Le passage d'un système de coordonnées à l'autre se fait en utilisant la matrice orthogonale Q définie par :

$$Q = [d, \frac{m}{||m||}, \frac{d \times m}{||d \times m||}] \tag{3.8}$$

Les coordonnées de Cayley sont alors définies par le vecteur $(s,\omega) \in \mathbb{R}^4$:

$$\begin{cases} [s]_\times=(Q-I)(Q+I)^{-1} \\ \omega = ||m|| \end{cases} \tag{3.9}$$

Il est possible de repasser aux coordonnées de Plücker en utilisant la définition de Q (Eq. 3.8) et en la calculant à l'aide de la formule suivante :

$$Q = \frac{(1-||s||^2)I+2[s]_\times+2ss^\top}{1+||s||^2} \tag{3.10}$$

Dans le cas où $m = 0$, les formules précédentes ne peuvent plus être utilisées. En pratique, on utilise un seuil τ très faible (*e.g.* $\tau = 10^{-7}$) et on change la formule (3.8) en :

$$Q = [d, e_1, e_2] \tag{3.11}$$

où (e_1, e_2) est une base orthonormale du plan défini par $d^\top x = 0$.

Notons que la conversion retour conserve l'idée de $m = 0$ puisqu'on obtient $m = \omega e_1$ avec $\|m\| \leq \tau \approx 0$

3.4 Modélisation d'un appareil photographique

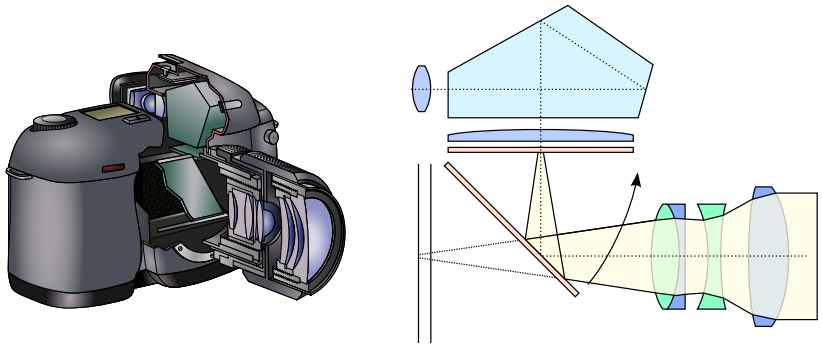


FIGURE 3.1 – Un appareil photographique numérique est un assemblage complexe de lentilles permettant la convergence d'un faisceau lumineux vers un capteur numérique. Ce modèle complexe peut être fortement simplifié dans un cadre classique d'utilisation de l'appareil. *Crédits Image : Wikipedia*

Un appareil photographique capture, via un capteur plan, la lumière réfléchie par les objets d'une scène réelle et traversant une ou plusieurs lentilles. Le modèle le plus couramment utilisé pour le représenter est le modèle *pinhole camera* (sténopé) qui approxime l'ensemble de lentilles comme un point infiniment petit laissant passer la lumière sur le capteur. Cette approximation peut être utilisée pour la plupart des appareils photographiques de la vie courante et permet, entre autre, de faire de la reconstruction 3D. Ce modèle réduit ainsi l'appareil à un simple point (situé au *centre optique* C de l'appareil) et la prise de photographie à une simple projection centrale de l'espace vers le plan du capteur (voir Figure 3.2). Par la suite nous utiliserons le terme *caméra* pour parler d'un appareil photographique.

Pour un point Q observé dans l'espace, sa projection q dans le plan du capteur (ou *plan focal*) est obtenue par passage aux coordonnées projectives via la formule suivante :

$$\mathbf{q} = K[R|t]\mathbf{Q} \tag{3.12}$$

La matrice K permet de passer du repère de la scène (ou *repère monde*) au repère de la caméra. La matrice $[R|t] \in \mathbb{R}^{3\times4}$ se décompose en une rotation $R \in \mathbb{R}^{3\times3}$ et une translation $t \in \mathbb{R}^3$. Cette multiplication en géométrie projective correspond, en géométrie Euclidienne, au changement de repère $RQ + t$. On appelle le couple (R,t) les *paramètres extrinsèques* de la caméra. On utilise parfois le couple (R, C) qui est équivalent au premier via la relation :

$$t = -RC \tag{3.13}$$

La matrice $[R|t]$ correspond à la projection effective du point Q sur le plan du capteur et fait passer des coordonnées métriques aux coordonnées pixelliques. $K \in \mathbb{R}^{3\times3}$ est appelée la

matrice de calibration et sa valeur s'exprime en fonction des *paramètres intrinsèques* de la caméra (focale, taille du capteur...) :

$$K = \begin{bmatrix} fk_u & s & c_u \\ & fk_v & c_v \\ & & 1 \end{bmatrix} \tag{3.14}$$

avec :

- f la focale de la caméra (en m), correspondant à la distance du plan focal au centre optique $\mathcal{C}$ de la caméra,
- k_u et k_v les facteurs d'agrandissement (en pixels.m^{-1}) liés à la taille du capteur de la caméra,
- s un paramètre traduisant la non orthogonalité des directions des cellules du capteur de la caméra,
- c_u et c_v sont les coordonnées du *point principal* (*i.e.* le point correspondant à la projection du centre $\mathcal{C}$ de la caméra).

Ces deux opérations sont généralement rassemblées sous la forme d'une *matrice de projection* $\mathrm{P} \in \mathbb{R}^{3\times4}$ et l'opération de projection d'un point Q de la scène sur le capteur de la caméra s'exprime alors par :

$$\mathbf{q} = \mathrm{P}\mathbf{Q} \tag{3.15}$$

Cette formule a également son équivalent pour les lignes. Pour l'obtenir, il est plus facile de passer par les coordonnées de Plücker. Pour une ligne $\mathbf{L}$ décrite en coordonnées de Plücker par (m,d) et de projection $\mathbf{l}$ on obtient :

$$\mathbf{l} = \mathrm{cof}(K)R[m - \mathcal{C} \times d] \tag{3.16}$$

où $\mathrm{cof}(K)$ est la matrice des cofacteurs de K. Par commodité, on notera :

$$\mathbf{l} = \mathrm{cof}(K)R[m - \mathcal{C} \times d] = \tilde{\mathrm{P}}\mathbf{L} \tag{3.17}$$

Le but de l'étape de calibration est alors, dans un premier temps, d'obtenir les matrices de projection de chacune des caméras observant la scène en utilisant les photographies fournies par les caméras.

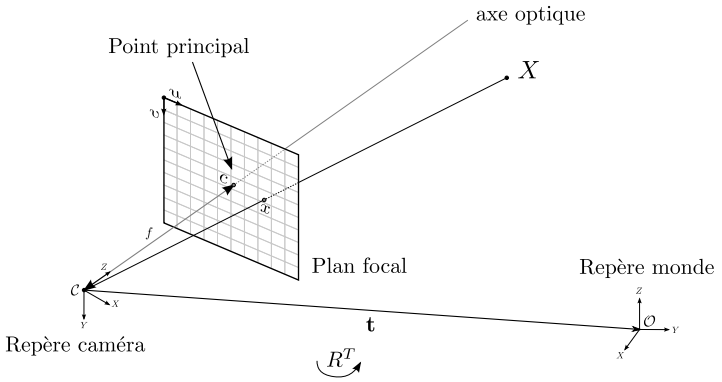


FIGURE 3.2 – Présentation du modèle *pinhole* pour un appareil photographique. *Crédits Image : P. Moulon.*

3.5 Géométrie épipolaire

Des correspondances de points ou de lignes entre plusieurs images permettent d'obtenir des informations sur les rotations et les translations relatives entre les différentes caméras. C'est à partir de cette information relative qu'une reconstruction globale pourra être effectuée comme nous le verrons dans la section suivante.

3.5.1 2 vues : le cas bifocal

Dans le cas d'un système à 2 caméras, il s'agit d'utiliser des correspondances de points et/ou de lignes pour obtenir la rotation relative et la translation relative entre les deux caméras. Soulignons que cette dernière information sera toujours obtenue à une échelle près, c'est à dire que seule la direction de la translation relative peut être calculée mais pas sa norme. Ce facteur d'échelle peut être fixé si on connaît la mesure réelle d'une distance dans la scène mais ne peut être trouvé sans information extérieure.

Deux relations équivalentes qui mettent en correspondance les observations p_1 et p_2 des caméras 1 et 2. La première utilise les observations en coordonnées homogènes des points 3D, $\mathbf{p}_1$ et $\mathbf{p}_2$:

$$\mathbf{p}_2^\top F \mathbf{p}_1 = 0 \tag{3.18}$$

où $F \in \mathbb{R}^{3\times 3}$ est appelée la *matrice fondamentale*. Cette matrice est définie à un facteur multiplicatif près, est de rang 2 et possède donc 7 degrés de libertés. Elle peut être estimée à partir de 8 correspondances [25] ou à partir de 7 correspondances uniquement, en utilisant la contrainte de rang [77].

Cette relation, également appelée contrainte épipolaire, permet de définir une distance entre deux points en coordonnées homogènes via une matrice fondamentale. En effet, le vecteur $F\mathbf{p}_1$ correspond, en coordonnées homogènes, à une droite appelée droite épipolaire. Lorsque la contrainte épipolaire est vérifiée, la droite épipolaire créée à partir de $\mathbf{p}_1$ passe par le point $\mathbf{p}_2$ et réciproquement. On peut aussi définir une distance entre deux points et une matrice fondamentale permettant de quantifier à quel point la contrainte épipolaire est vérifiée :

$$d_{\mathcal{F}}(\mathbf{p}_1, \mathbf{p}_2, F) = d_\perp(\mathbf{p}_2, F\mathbf{p}_1) \tag{3.19}$$

où $d_\perp$ correspond à la distance Euclidienne entre une droite et un point (voir Eq. 3.2). En pratique, comme cette distance est utilisée sur des paires de points ne vérifiant pas *strictement* la contrainte épipolaire, nous la symétrisons :

$$d_{\mathcal{F}}(\mathbf{p}_1, \mathbf{p}_2, F) = 0.5(d_\perp(\mathbf{p}_2, F\mathbf{p}_1) + d_\perp(\mathbf{p}_1, F^\top \mathbf{p}_2)) \tag{3.20}$$

Si on fait intervenir la matrice de calibration K ainsi que les coordonnées normalisées $\hat{\mathbf{p}}_i = K^{-1}\mathbf{p}_i$, l'équation 3.18 devient :

$$(K\hat{\mathbf{p}}_2)^\top F (K\hat{\mathbf{p}}_1) = 0 \tag{3.21}$$

$$\hat{\mathbf{p}}_2^\top (K^\top F K)\hat{\mathbf{p}}_1 = 0 \tag{3.22}$$

$$\hat{\mathbf{p}}_2^\top E \hat{\mathbf{p}}_1 = 0 \tag{3.23}$$

La matrice $E = K^\top F K$ est appelée *matrice essentielle*. De même que la matrice fondamentale, elle peut être estimée à partir de 8 correspondances mais d'autres méthodes ne nécessitent que 5 correspondances [61]. Cependant, contrairement aux méthodes d'estimation de la matrice fondamentale, elle requiert d'avoir accès à la matrice de calibration K pour obtenir les coordonnées normalisées.

De même que pour la matrice fondamentale, la distance épipolaire peut aussi être définie :

$$d_{\mathcal{E}}(\hat{\mathbf{p}}_1,\hat{\mathbf{p}}_2,E) = 0.5(d_{\perp}(\hat{\mathbf{p}}_2,E\hat{\mathbf{p}}_1)+d_{\perp}(\hat{\mathbf{p}}_1,E^{\top}\hat{\mathbf{p}}_2)) \tag{3.24}$$

En pratique, il est possible d'estimer la matrice de calibration à partir de la formule suivante :

$$K = \begin{bmatrix} f & 0 & w/2 \\ & f & h/2 \\ & & 1 \end{bmatrix} \tag{3.25}$$

où w (resp. h) représente la largeur (resp. longueur) en pixel de l'image. Cette estimation est en général suffisante pour obtenir un modèle 3D convenable qui sera ensuite raffiné. Il est également possible de calibrer l'appareil photographique à l'aide de mires pour obtenir une estimation plus fiable de la matrice de calibration. Ce procédé est plus long mais n'a besoin d'être fait qu'une seule fois pour un appareil donné utilisé à une focale donnée. Il a l'avantage de pouvoir également estimer les paramètres de distorsion permettant une meilleure correction de ce défaut (indispensable dans le cas d'utilisation de lignes). Dans la suite, nous supposerons que nous avons accès à la matrice de calibration et utiliserons donc les coordonnées normalisées plutôt que les observations directes.

Une fois la matrice essentielle estimée, il est possible d'obtenir le couple (R,t) à l'aide d'une décomposition en valeurs singulières. Cette méthode (voir [24]) donne 4 possibilités pour (R,t) cependant il est également démontré qu'une seule d'entre elles génère une scène où les points sont situés devant les caméras. Inversement, il est possible d'obtenir la matrice essentielle à partir d'un couple (R,t) en utilisant la formule :

$$E = [t]_{\times}R \tag{3.26}$$

Notons également que l'estimation de pose relative à l'aide de lignes n'est pas réalisable sans contraintes supplémentaires. Pour le comprendre, il est plus aisé de se concentrer sur l'aspect géométrique du problème. Chercher la pose relative, c'est en fait chercher la position du centre de la seconde caméra en gardant fixe la première caméra ainsi que les positions des observations par rapport aux centres des caméras correspondants.

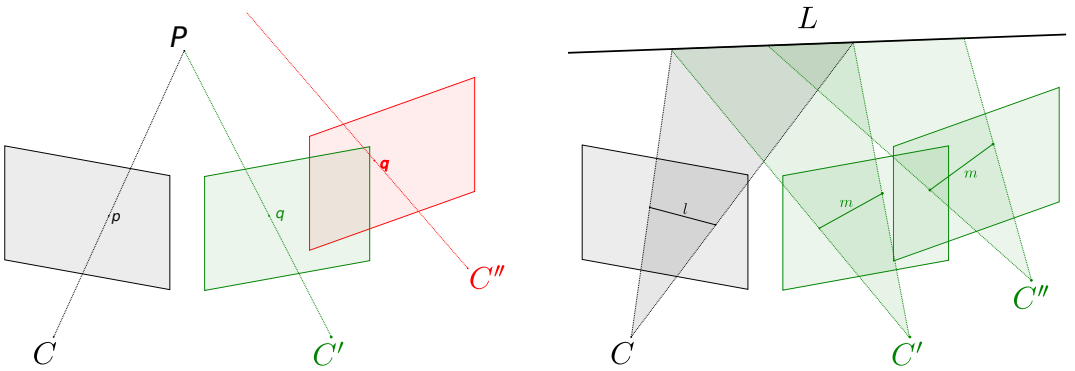
Lorsque 2 observations d'un même point 3D sont mises en correspondance, on sait que les lignes 3D passant par le centre de chacune des caméras et l'observation correspondante doivent s'intersecter (par construction, les observations sont les projections du point 3D). La contrainte épipolaire traduite par l'équation 3.18 peut être vue de manière géométrique comme le fait que 2 lignes de $\mathbb{R}^3$ déterminées par les positions des caméras sont sécantes (voir Fig. 3.3). L'existence d'une intersection est une contrainte forte puisque 2 lignes quelconques de $\mathbb{R}^3$ sont en général non sécantes.

Lorsque 2 observations d'une même ligne 3D sont mises en correspondance, on sait que les plans 3D passant par le centre de chacune des caméras et l'observation correspondante doivent s'intersecter. Or, deux plans non-parallèles de $\mathbb{R}^3$ s'intersectent nécessairement. L'existence d'une intersection n'engendre qu'une contrainte faible puisque seules les positions relatives générant des plans parallèles ne sont pas compatibles avec le problème posé.

Malgré cette difficulté, il existe des méthodes permettant l'estimation de pose à partir de lignes et de 2 vues uniquement. Ces méthodes utilisent des contraintes supplémentaires (extrémités de segments de droite, parallélisme, orthogonalité...) pour estimer la pose relative et nous les décrirons plus en détail dans les chapitres suivants.

3.5.2 3 vues : le cas trifocal

Dans le cas d'un système à 3 caméras, on utilise des triplets de correspondances de points et/ou de lignes pour obtenir les rotations et translations relatives entre les 3 caméras. Cette fois



Géométrie épipolaire bifocale avec des points — Géométrie épipolaire bifocale avec des lignes

FIGURE 3.3 – Dans le cas des points, il existe des positions relatives de caméras (en rouge) qui n'engendrent pas l'intersection des lignes centre-observations. Dans le cas des lignes, la plupart des positions relatives de caméras engendrent l'intersection des plans centre-observations.

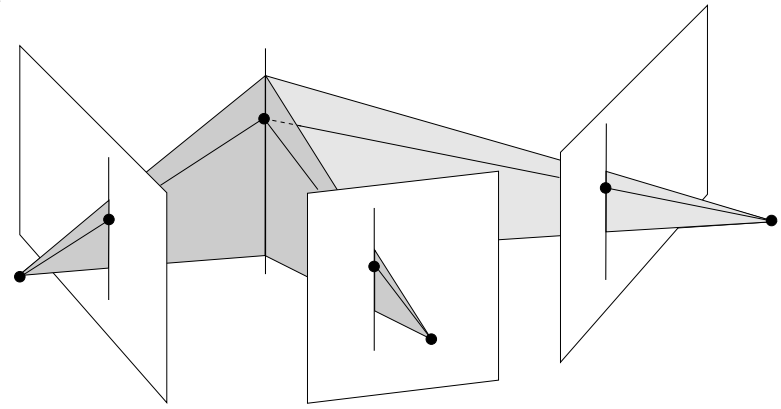


FIGURE 3.4 – Le tenseur trifocal peut être estimé à partir de triplets de lignes, points ou mixte point-ligne. *Crédits Pierre Moulon*

encore, les informations sont obtenues à un facteur d'échelle près par rapport à la scène réelle. Cependant ce facteur d'échelle est le même pour les trois translations obtenues.

Shashua *et al.* [**shashua**] introduisent le tenseur trifocal $\mathcal{T} \in \mathbb{R}^{3\times3\times3}$ à partir des matrices de projections P_i de chacune des caméras :

$$\mathcal{T}_{i,j,k} = a_{j,i}b_{k,4} - a_{j,4}b_{k,i} \tag{3.27}$$

pour $i,j,k = 1,2,3$ et avec $\mathrm{P}_0 = [Id|0]$, $\mathrm{P}_1 = [a_{i,j}]$ et $\mathrm{P}_2 = [b_{i,j}]$.

Ce tenseur met en correspondance des triplets de points uniquement, de lignes uniquement ou des combinaisons des deux et permet d'obtenir les rotations et translations relatives entre chaque caméras. Cependant son estimation requiert au moins 7 triplets de points ou 13 triplets de lignes ce qui est difficile à obtenir dans certaines scènes (notamment les intérieurs de bâtiments peu texturés).

Pour reprendre l'interprétation géométrique de la partie précédente, les points engendrent toujours des contraintes fortes puisqu'il s'agit de chercher l'intersection de 3 droites dans l'espace. Cette fois, les lignes engendrent également une contrainte forte puisqu'il s'agit de chercher l'intersection de 3 plans dans l'espace. Cependant cette contrainte est moins forte que celle créée par les points et se traduit par la nécessité d'avoir 13 triplets de lignes tandis que 7 triplets de points suffisent pour estimer le tenseur trifocal.

3.6 Reconstruction multi-vues

Les méthodes présentées précédemment permettent d'estimer la pose de 2 ou 3 caméras à un facteur d'échelle près (avec une échelle qui diffère d'une paire/triplet à un(e) autre). Pour obtenir la pose de n caméras dans un même repère monde, il existe différents types de méthodes qui utilisent l'une, l'autre ou les deux méthodes présentées précédemment.

- Les méthodes *incrémentales* : la structure de la scène est initialisée avec 2 ou 3 vues à l'aide des méthodes précédentes. Ensuite, les positions/orientations de nouvelles caméras sont ajoutées successivement en utilisant les correspondances entre la reconstruction courante et les observations de la nouvelle caméra. Les méthodes associées diffèrent si on utilise des correspondances de points (les méthodes *Perspective-n-Point* ou *PnP* [39, 27, 43]) ou de lignes (les méthodes *Perspective-n-Line* ou *PnL* [50, 87, 37]). A chaque ajout de caméra, un raffinement est calculé pour éviter de trop grosses erreurs de dérive.
- Les méthodes *globales* : dans un premier temps, les rotations et translations relatives entre chaque caméra sont estimées à l'aide de méthodes bifocales et/ou trifocales en décomposant le système de n caméras en plusieurs sous-systèmes à 3 vues maximum. Ensuite, la structure globale est estimée en une unique étape en utilisant comme contraintes les transformations relatives trouvées précédemment.

Les méthodes incrémentales sont généralement moins précises que les méthodes globales (*e.g.* phénomène de dérive pas toujours bien compensé, système moins bien contraint) mais sont également plus rapides car elles requièrent moins de calculs. Dans le cadre de la reconstruction en intérieur, nous avons privilégié les méthodes globales car la précision est importantes pour les applications pratiques.

3.6.1 Triangulation

Une fois les positions et orientations des caméras connues, une reconstruction partielle de la structure peut alors être réalisée. Il s'agit de trianguler les points et lignes ayant servi à la calibration pour obtenir leur position dans l'espace.

Trianguler un points (resp. une ligne) correspond au calcul de l'intersection des rayons (resp. faisceaux) passant par le centre de chacune des caméras dans lesquelles il/elle apparaît et l'observation de la caméra correspondante. Pour les points, on cherche l'intersection de plusieurs droites, tandis que pour les lignes on cherche l'intersection de plusieurs plans.

Dans un cadre purement théorique et sans bruit, cette tâche est aisée puisqu'il existe toujours une unique solution. Cependant, nous obtenons en général des résultats légèrement bruités. Il n'existe alors pas d'intersection exacte et celle-ci doit être approximée.

Dans le cas des points, on cherche à minimiser l'erreur de reprojection entre le projeté du point triangulé $\mathbf{Q}$ et ses observations $\mathbf{q}_i$:

$$\arg\min_{Q\in\mathbb{P}^4} \sum_{i=0}^{N_{CAMERAS}} ||\mathrm{P}_i\mathbf{Q} - \mathbf{q}^i||_2 \tag{3.28}$$

Cette équation étant difficile à résoudre de manière optimale, on procède plus souvent en cherchant une solution approchée puis on la raffine, notamment lors du procédé d'ajustement de faisceaux décrit ci-après.

Plutôt que d'exprimer le problème sous la forme de l'Eq. 3.28, on cherche plutôt à utiliser le fait que, dans le cas non-bruité, les rayons $\mathrm{P}_i\mathbf{Q}$ et $\mathbf{q}^i$ sont colinéaires et donc :

$$[\mathbf{q}^i]_\times \mathrm{P}_i\mathbf{Q} = 0 \tag{3.29}$$

qui est une équation linéaire matricielle en $\mathbf{Q}$ solvable par une méthode aux moindres carrés. C'est la méthode DLT [**hartley_sturm_cviu**, 24] (pour *Direct Linear Transform*), qui est utilisée

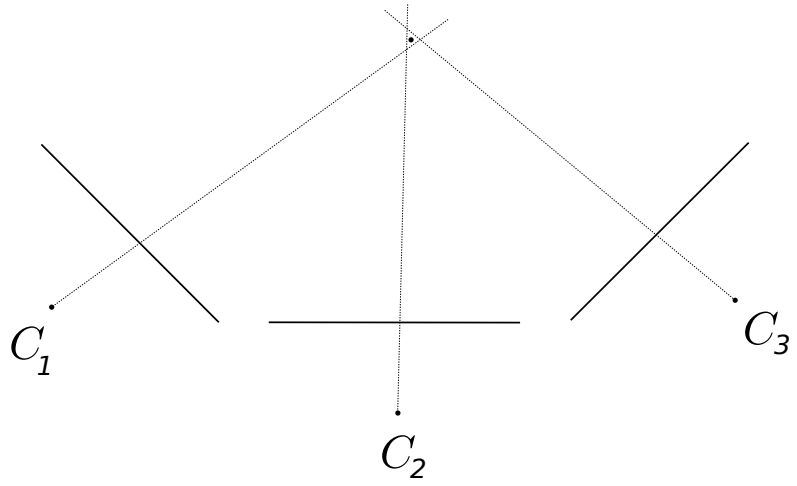


FIGURE 3.5 – Dans le cas des points, la triangulation consiste à chercher le point 3D le plus proche des n rayons reliant chacunes des observations au centre de la caméra associée.

généralement pour la triangulation. Elle donne une approximation suffisamment bonne pour être utilisée dans la suite.

Dans le cas des lignes, il n'existe pas de fonction d'erreur de reprojection parfaite. On cherche à mesurer une erreur entre une ligne 3D triangulée et des segments observés. La distance généralement utilisée ([85], [28]) correspond à la distance moyenne des extrémités du segment observé à la reprojection de la ligne 3D (voir Fig. 3.6). Cependant cette distance est arbitraire (forte dépendance des extrémités) même si ses bons résultats expérimentaux confirment ce choix.

Bartoli *et al.* [6] ont développé une méthode de triangulation de n lignes proche de la méthode DLT utilisée pour les points. Une méthode linéaire est utilisée pour minimiser les distances de reprojection aux extrémités (d_1 et d_2 dans la Fig. 3.6) en obtenant un vecteur de $\mathbb{R}^6$. Une correction de Plücker est alors appliquée pour vérifier les conditions de l'Eq. 3.7. Cette méthode est ensuite itérée jusqu'à convergence. Cette méthode est lourde en calcul et le résultat est souvent raffiné par la suite dans l'étape d'ajustement de faisceaux. Dans la suite, nous avons donc utilisé des approximations dans les cas où la précision requise n'était pas jugée trop forte. Notons également que, contrairement au point, le cas d'une ligne à 2 vues est très simple à traiter puisqu'il ne s'agit que de l'intersection de deux plans.

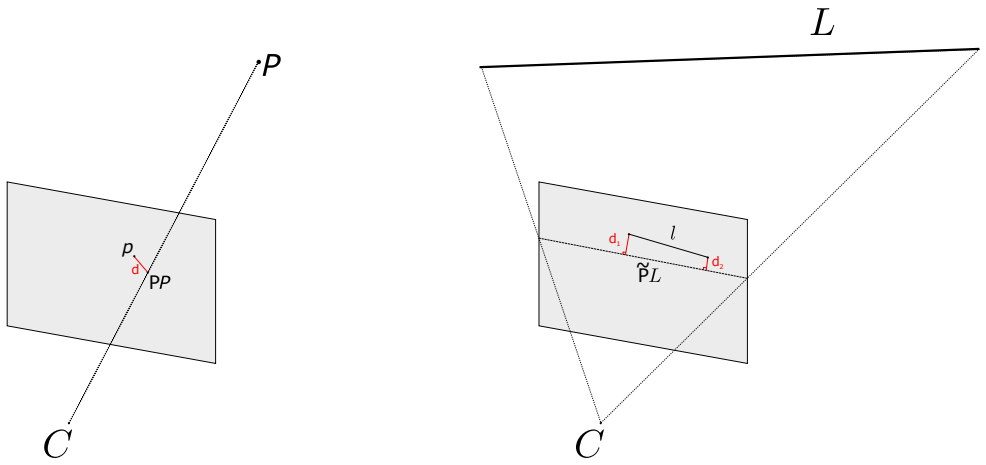
3.6.2 Ajustement de faisceaux

L'ajustement de faisceaux ou *Bundle Adjustment* est un processus de correction des positions des caméras et des points et lignes triangulées pour les rendre plus en adéquation avec les observations faites sur chacune des caméras. Cette optimisation, qui permet de gagner en précision, est utilisée à la fin des méthodes globales et à chaque itération des méthodes incrémentales pour éviter un phénomène de dérive trop important.

Ce processus consiste à minimiser la distance de reprojection des points et lignes sur chacune des caméras où ils apparaissent :

$$\min_{\mathrm{P}_i,\mathbf{P}_j,\mathbf{L}_k} \sum_{i=1}^{N_{CAMERAS}} \Big(\sum_{j=1}^{N_{POINTS}} ||\mathrm{P}_i\mathbf{P}_j - \mathbf{p}_j^i||_2 + \sum_{k=1}^{N_{LIGNES}} ||d(\mathrm{P}_i,\mathbf{L}_k,\mathbf{l}_k^i)||_2 \Big) \tag{3.30}$$

où la distance $d(\mathrm{P}_i,\mathbf{L}_k,\mathbf{l}_k^i)$ correspond à la moyenne des distances aux extrémités (d_1 et d_2 dans la Fig. 3.6).



Points Lignes

FIGURE 3.6 – Illustration des erreur de reprojection pour le cas des points (à gauche) et celui des lignes (à droite). Si la distance utilisée pour les points est peu discutable, celle des lignes résulte d'un choix arbitraire que seule l'expérience tend à confirmer.

Minimiser l'équation 3.30 nous ramène à un problème de moindres carrés non-linéaire. Il est possible de le résoudre en utilisant l'algorithme de *Levenberg-Marquardt* [**LMalgo**] qui permet une convergence vers le minimum optimal à condition de se trouver initialement proche de celui-ci.

Dans [28], Hofer *et al.* pondèrent la distance de reprojection des lignes avec la différence d'angle entre la direction du segment observé et la reprojection de la ligne reconstruite. Si cette formule a du sens, elle n'en a pas pour autant plus que la précédente et pour des questions d'homogénéité avec la distance pixellique des points, il n'est pas possible de l'utiliser dans un cadre mixte point-ligne.

Notons une fois encore que les lignes observées par 2 caméras uniquement n'apportent pas de contraintes sur la structure globale des caméras. Elles sont donc ignorées lors de cette phase pour éviter des dérives qui seraient générées en pratique par l'algorithme.

3.7 Détections & mise en correspondance de *features*

Comme vu dans les parties précédentes, la reconstruction 3D nécessite d'avoir, au préalable, détecté puis mis en correspondance des *features* (*e.g.* points, lignes ...) d'une image à une autre. Pour obtenir ces correspondances, le schéma suivi peut être séparé en trois parties :

1 **Détection** : les *features* sont détectés comme des caractéristiques saillantes dans l'image, notamment à l'aide du gradient. En effet, le gradient est un indicateur des zones de fortes variations (bords potentiels) et permet donc une détection efficace des lignes et de certains types de points saillants.

2 **Description** : une fois détectés, les *features* sont décrit à l'aide d'un *descripteur*. Ce descripteur est généralement un vecteur de faible dimension qui résume l'information du *feature* et de son voisinage.

3 **Appariement** : les *features* sont enfin mis en correspondance à l'aide de leurs descripteurs (la distance entre deux descripteurs donne une information de similarité entre deux *features*). Cette distance peut être utilisée telle quelle, mais pour obtenir de meilleurs résultats notamment pour les lignes, elle est souvent combinée à d'autres critères (*e.g.* comparaison des voisins aux descripteurs les plus proches, critères géométriques...).

3.7.1 Les points saillants

FIGURE 3.7 – Exemple de détection et d'appariement SIFT [46] de points saillants dans une scène d'intérieur. On remarque au passage l'un des problèmes des scènes d'intérieur puisque, si de nombreux points sont détectés, très peu sont mis en correspondance.

Les points ont été longuement étudiés car ils sont au centre de nombreux algorithmes de vision par ordinateur. L'une des premières approches proposées est un détecteur de coin [23] invariant à l'orientation de la scène. L'utilisation de différence de gaussiennes et d'espace échelles a ensuite permis une meilleure invariance aux changements d'échelles [46]. Ces invariances engendrant des calculs plus longs, certaines méthodes ont été approximées pour permettre une utilisation plus rapide sans trop perdre en précision [7].

Les descripteurs utilisés sont généralement ceux de SIFT [46], composés d'un histogramme d'orientations de gradients résumés dans un vecteur de 128 valeurs pour chaque points d'intérêt. Ce descripteur a l'avantage d'être robuste au bruit, à la compression JPEG, au changement d'éclairement, à la rotation et au changement d'échelles (voir [49] pour une étude plus détaillée).

L'appariement des points d'intérêt se fait ensuite à l'aide d'une distance entre les descripteurs. Cette distance est souvent accompagné d'autres critères comme celui du *Distance Ratio* qui compare les 2 meilleurs appariements possibles pour chaque points. Dans le cas où ces 2 appariements ont des distances de descripteurs trop proches, on rejette les 2 paires pour éviter des erreurs d'appariement. D'autres méthodes faisant intervenir des critères géométriques peuvent également être utilisés pour permettre un meilleur appariement. Nous avons notamment utilisé la méthode KVLD [44] pour réduire le ratio de mauvaises correspondances.

3.7.2 Les segments

Notons tout d'abord que nous parlons de ligne pour l'aspect géométrique mais de segment pour l'aspect détection et mise en correspondance. En effet, nous détectons des segments dans les images (*i.e.* des lignes de longueur finie). Cependant leurs extrémités sont rarement fiables et nous ne pouvons considérer, en général, qu'elles seront les mêmes d'une image à une autre (même lorsque le segment observés est vu entièrement dans les deux images). Ainsi, lors de la mise en correspondance, nous cherchons à mettre en relation des segments qui correspondent à peu près au même segment dans la scène réelle (*i.e.* un bord de tableau, de fenêtre ...). Enfin, lors de la phase de reconstruction 3D nous utilisons les lignes infinies, générées par chacun des segments détectés dans chaque image.

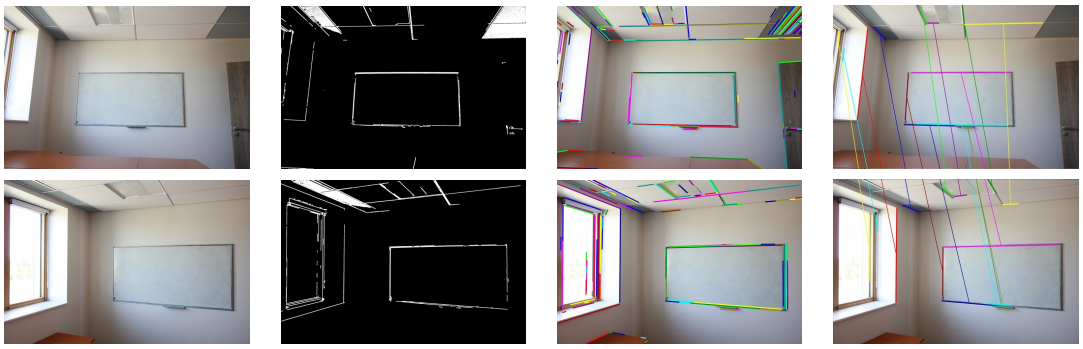


Image originale | Détection de bords | Détection de segments | Appariement des segments

FIGURE 3.8 – Les segments sont détectés en deux étapes : un détecteur de bords comme celui de Canny [9] suivi d'une sélection des segments [19]. Enfin, ils sont mis en correspondance à l'aide de descripteurs et de critères géométriques [84].

La détection de segments se fait généralement en deux temps. Tout d'abord une détection de bord comme celle de Canny [9] réalisée à l'aide d'un calcul de la carte de gradients et suivi d'un seuillage grossier. Les pixels sélectionnés comme étant des bords sont alors regroupés en fonction de l'alignement de leur gradient. Des critères sont alors appliqués pour valider ou non certains groupe de pixels comme étant des segments (*e.g.* nombre de pixels alignés [32], critère *a contrario* [19, 2], ...). La détection de segments faisant l'objet d'un chapitre, nous la présenterons plus en détail dans les prochains chapitres.

Les descripteurs de segments ont presque toujours en commun l'idée de moyenner le long du segment. En effet, les segments étant de longueur variée, pour obtenir un descripteur de même taille quelque soit la longueur du segment et donc comparable, la moyenne semble avoir été le moyen le plus efficace. Bay *et al.* [8] utilisent des histogrammes de couleur pour décrire le voisinage du segment tandis que [79, 84] utilisent des descripteurs dérivant de SIFT [46] moyennés selon la longueur et l'épaisseur du segment. Cependant, le manque de précision des extrémités rend ces descripteurs moins fiables que pour les points. Des critères géométriques sont alors utilisés à la place [78] ou en complément [84] de ces descripteurs pour la mise en correspondance. Ces critères géométriques font intervenir les segments voisins et cherchent à favoriser les appariements conservant, d'une image à l'autre, les positions relatives, les angles entre segments ou des rapports de longueur.

3.7.3 Les points de fuite

FIGURE 3.9 – Les lignes rouges suivent la direction des rails et sont parallèles dans la scène réelle mais concourantes dans le plan image (point rouge). Les lignes bleues suivent la direction orthogonale à celle des rails et sont toujours parallèles dans le plan image. Le point de fuite bleu est *à l'infini*.

La notion de *point de fuite* provient de l'art et de l'architecture où elle est une notion commune. Ils correspondent à la projection, parfois finie, d'un point situé à l'infini. Deux droites parallèles dans l'espace vont, en général, engendrer deux projections sécantes sur le plan image de l'appareil photographique. Cette observation se visualise aisément lorsqu'on prend en photo une ligne de chemin de fer en regardant dans la direction des rails (voir Fig. 3.9). En fait, un ensemble de droites de l'espace de même direction $d \in \mathbb{R}^3$ va générer un ensemble de droites projetées concourantes en un même point $p \in \mathbb{R}^2$ appelé point de fuite.

Si on utilise la géométrie projective, on peut exprimer une relation simple entre d et p :

$$\mathbf{p} \propto KRd \tag{3.31}$$

où K et R sont les matrices de calibration et de rotation de la caméra. Notons que cette relation reste vraie dans le cas où les droites projetées restent strictement parallèles. Cependant le point $\mathbf{p}$ est alors *à l'infini* et son correspondant p dans le plan image n'existe plus.

Les points de fuite étant, par construction, des intersections particulières de lignes, leur détection se fait à l'aide des lignes. Pour les détecter, il faut donc repérer quelles sont les zones dans lesquelles un grand nombre de lignes s'intersectent. La difficulté de cette détection est similaire à celle de la triangulation des points et des lignes. En effet, les lignes de même direction s'intersectent théoriquement en un même point. Cependant la pratique veut que des erreurs de détection rendent ces intersections uniquement approximatives. Un point de fuite ne correspond donc pas à une intersection de plusieurs lignes mais plutôt à une zone de faible taille dans laquelle passe de nombreuses lignes. Une des difficultés est alors de déterminer ce qu'est une zone de faible taille. De nombreuses méthodes existantes utilisent des seuils fixes [75] tandis que d'autres utilisent des critères *a contrario* [3, 41] permettant de se passer de seuil en se basant à la place sur un modèle statistique. D'autres hypothèses peuvent également être utilisées : Nieto *et al.* [60] imposent de fixer au préalable le nombre de points de fuite détectés tandis que de nombreux auteurs [30, 52, 88] utilisent des hypothèses de type *Manhattan world* (*i.e.* on suppose que dans les scènes étudiées, deux lignes quelconques sont soit orthogonales soit parallèles).

Concernant l'appariement de point de fuite, il n'existe pas de méthode standard de mise en correspondance. Lorsque nous en avons besoin, nous utilisons les correspondances des segments et sélectionnons l'appariement de points de fuite le plus cohérent avec celui des segments (*i.e.* deux points de fuite se correspondent ssi une majorité des segments qui leurs sont associés se correspondent).

3.8 RANSAC

La reconstruction 3D requiert d'estimer un certain nombre de paramètres (*e.g.* matrice essentielle ou fondamentale) à partir de données généralement bruitées (*e.g.* des correspondances de points), c'est-à-dire peu précises pour certaines, voire même fausses pour d'autres. Pour obtenir une estimation robuste et précise à partir de telles données, nous devons donc être capables de différencier les données suffisamment précises (que l'on nomme *inliers*) des données trop peu précises ou fausses (que l'on nomme *outliers* ou données aberrantes). Pour cela la méthode la plus couramment utilisée est la méthode du *RANSAC* [**ransac**] (pour *RANdom SAmpling Consensus*).

L'algorithme de RANSAC ne se cantonne pas à des applications en reconstruction 3D, il est générique et s'applique à n'importe quel type de problème possédant les critères suivants :

- des données d'entrée I contenant une proportion p d'*inliers* (*e.g.* des correspondances de points),
- une méthode permettant d'estimer un modèle $m \in \mathcal{M}$ à partir de k élements de I,
- une distance $d(e,m)$ d'une donnée $e \in I$ à un modèle $m \in \mathcal{M}$ (*e.g.* la distance épipolaire) telle que :

$$d(e,m) < d(e',m) \Leftrightarrow e \text{ est plus en accord que } e' \text{ avec le modèle } m \tag{3.32}$$

L'algorithme de *RANSAC* fonctionne de manière itérative. Tout d'abord, il sélectionne aléatoirement k éléments parmi les données en entrée et calcule le modèle m correspondant. Il estime alors le nombre d'*inliers* pour ce modèle en cherchant les éléments à une distance suffisamment faible de m. Pour cela, un seuil δ est utilisé. Ce tirage est répété un certain nombre de fois (généralement grand) et le modèle avec le plus d'*inliers* est considéré comme la meilleure estimation (la méthode est illustrée à la Fig. 3.10).

Données : I
Paramètres : seuil δ, nombre d'itérations N
Output : modèle $m^\star$
$N^\star_{INLIERS} \leftarrow 0$
pour N itérations **faire**
  Sélectionner aléatoirement k éléments de I
  Estimer un modèle m à partir de ces k éléments
  $N_{INLIERS} \leftarrow 0$
  pour $e \in I$ **faire**
    si $d(e,m) < \delta$ **alors**
      e est un inlier : $N_{INLIERS} \leftarrow N_{INLIERS} + 1$
    fin si
  fin pour
  si $N^\star_{INLIERS} < N_{INLIERS}$ **alors**
    Mettre à jour le modèle final : $m^\star \leftarrow m$, $N^\star_{INLIERS} \leftarrow N_{INLIERS}$
  fin si
fin pour
retourner $m^\star$

FIGURE 3.10 – Algorithme de *RANSAC*

Dans cette méthode deux paramètres sont utilisés et leur choix affecte fortement le résultat :

- le nombre d'itération N
- la distance δ séparant les *inliers* des *outliers*

Le nombre d'itération N est souvent relié à une estimation de la proportion d'inliers p à travers la formule :

$$N = \frac{\log(1-s)}{\log(1-p^k)} \tag{3.33}$$

où s représente la probabilité que l'algorithme ait choisi au moins une fois k inliers pour estimer le modèle.

En effet, étudier tous les modèles m possibles est en général infaisable du fait de leur trop grand nombre (*e.g.* généralement supérieur à $\binom{1000}{5} \sim 10^{12}$ dans le cadre de la calibration). Il apparait d'ailleurs qu'il est nullement nécessaire de tout essayer et qu'un nombre d'itération calculé selon la formule 3.33 suffit en pratique à trouver un modèle précis. Les paramètres p et s restent alors à fixer et nécessitent un compromis entre la précision du résultat souhaité et le temps d'exécution de la méthode.

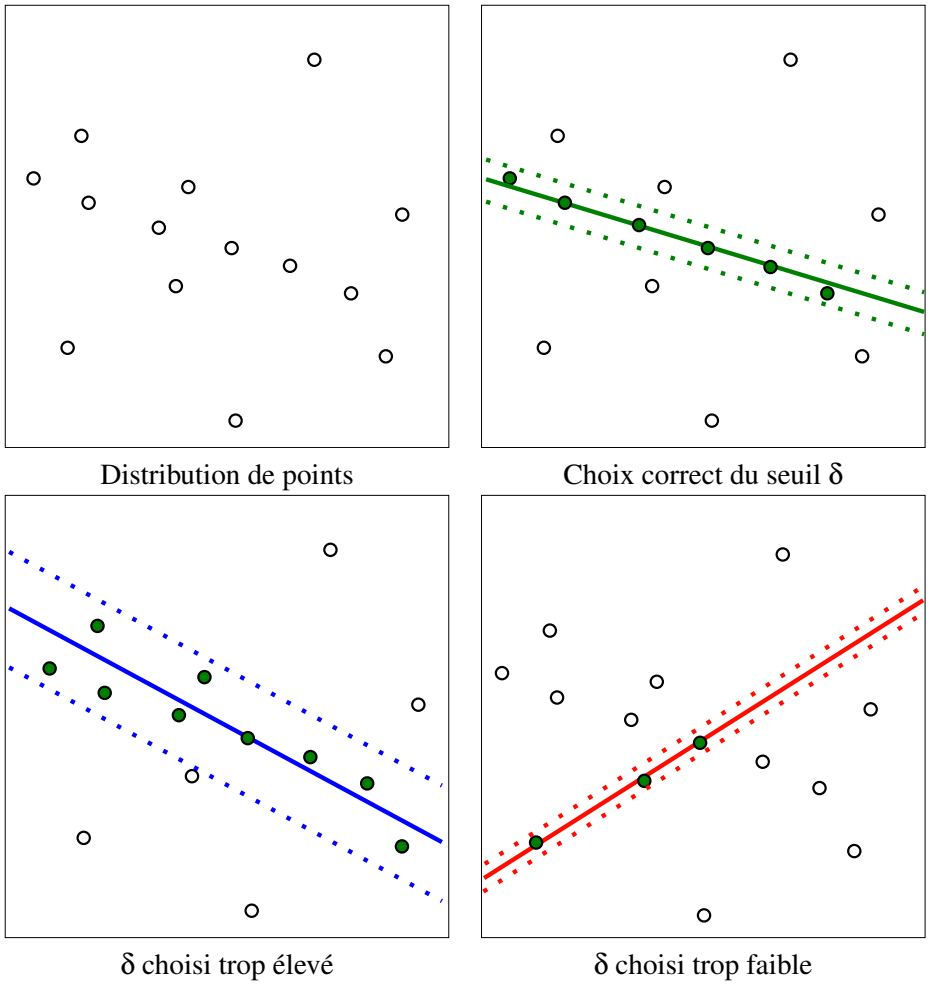


FIGURE 3.11 – Exemple de choix du seuil δ séparant les *inliers* des *outliers* dans le cadre d'une détection de ligne par *RANSAC*. Un choix trop mauvais du seuil peut générer une détection peu précise (choix d'un δ trop grand) voire totalement incorrect (choix d'un δ trop petit). *Crédits : Pierre Moulon*

Le choix du second paramètre δ est souvent plus crucial pour obtenir un résultat correct et précis. En effet, le résultat trouvé peut perdre en précision voire même devenir incorrect avec un trop mauvais choix du seuil δ (voir Fig. 3.11).

Malgré la présence de ces deux paramètres importants dans la méthode, elle reste généralement très efficace et est utilisée dans de nombreuses applications notamment dans le cadre de la

reconstruction 3D. La section suivante présente une alternative au classique *RANSAC* permettant de se passer du choix du paramètre δ. C'est ce type de variante que nous utiliserons dans les chapitres suivants.

3.9 Méthodes *a contrario*

Le principe général d'une méthode *a contrario* est de remplacer l'utilisation d'un seuil fixe arbitraire par une comparaison statistique lors de l'évaluation d'un type de modèle $\mathcal{M}$. Ces méthodes s'adaptent mieux aux différents environnements et le changement de scène pose alors moins de problème (voir Fig. 3.12). Ce type de méthode peut s'appliquer à de nombreux contextes, notamment lors d'évaluations type *RANSAC* [53, 58, 69] et pour de la détection de segments [19, 2, 68]. Ainsi, le type de modèle évalué peut être une matrice essentielle qui relie deux groupes de *features*, ou le segment qui correspond à un groupe de pixels. Dans cette section nous conserverons l'aspect générique du type de modèle $\mathcal{M}$ mais ces questions seront davantage détaillées lors de la description de méthodes *a contrario* dans les chapitres suivants.

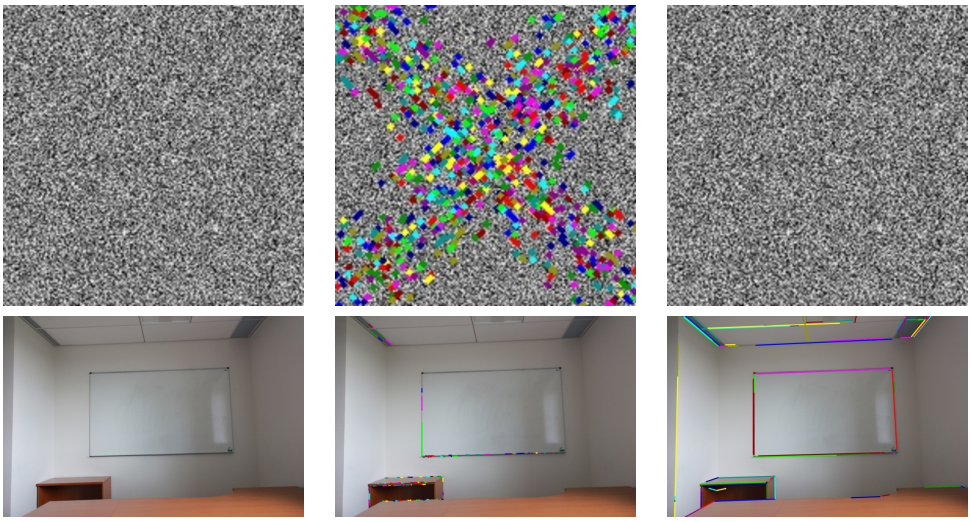


Image originale | Méthode avec seuil [32] | Méthode *a contrario* [68]

FIGURE 3.12 – Nous affichons ici les segments détectés pour une méthode avec seuil (Transformée de Hough [32]) et pour une méthode *a contrario* [68]. Pour la méthode avec seuil [32], nous avons utilisé le même seuil pour les 2 images. Contrairement aux méthodes *a contrario* qui s'adaptent à l'image, les méthodes avec seuil sont très sensibles au bruit et au manque de contraste.

Décrivons tout d'abord la classe de problèmes dans lesquels une méthode *a contrario* peut être appliquée. Un tel problème définit :

- des données d'entrée I dans un ensemble $\mathcal{E}$ (*e.g.* des correspondances de points),
- un ensemble de modèles $\mathcal{M}$ que peuvent suivre ces données (*e.g.* une liste de matrice essentielle),
- une distance $d(e,m)$ d'une donnée $e \in \mathcal{E}$ à un modèle donné $m \in \mathcal{M}$ (*e.g.* la distance épipolaire) telle que :

$$d(e,m) < d(e',m) \Leftrightarrow e \text{ est plus en accord que } e' \text{ avec le modèle } m \tag{3.34}$$

Dans les méthodes *classiques*, on fixe un seuil δ au préalable et on dira qu'une donnée $e \in \mathcal{E}$ est en accord avec un modèle m lorsque $d(e,m) < \delta$. On valide alors le modèle avec le plus

d'élément en accord (*e.g. RANSAC*), ou les modèles avec suffisamment d'éléments en accord (*e.g.* accumulateur de Hough).

Dans les méthodes *a contrario*, on définit une variable aléatoire X qui génère $n_I = \text{card}(I)$ éléments dans $\mathcal{E}$ en suivant un modèle de fond $\mathcal{H}_0$. Le modèle de fond étant généralement une loi de répartition uniforme sur l'ensemble $\mathcal{E}$, I est alors vu comme un tirage particulier de X.

Pour un modèle $m \in \mathcal{M}$ et un seuil δ, nous notons $k(X,m,\delta)$ le nombre de données d'un ensemble X en accord avec m à une précision de δ :

$$k(X,m,\delta) = \text{card}(\{x \in X \text{ t.q. } d(x,m) < \delta\}) \tag{3.35}$$

En particulier, $k(I,m,\delta)$ correspond au nombre de données d'entrée compatibles avec le modèle m à une précision de δ.

L'idée est ensuite de tester dans quelle mesure un modèle m obtiendrait un score $k(X,m,\delta)$ sur des données aléatoires. Le fonctionnement est similaire à celui des tests en statistiques mais au lieu de savoir si l'hypothèse $\mathcal{H}_0$ est correcte (ce qui n'est généralement pas exactement le cas), on cherche à trouver les modèles qui maximisent l'écart entre l'observation et le modèle de fond.

Plutôt que d'utiliser la p-valeur des statistiques, Desolneux *et al.* [12] définissent le *Nombre de Fausses Alarmes* (NFA) pour un modèle m comme étant :

$$\text{NFA}(m,\delta,I) = N_{TEST}\,\mathbb{P}_{\mathcal{H}_0}[k(X,m,\delta) \geq k(I,m,\delta)] \tag{3.36}$$

où $N_{TEST} = \text{card}(\mathcal{M})$ est le nombre de modèles potentiellement testés.

On dit alors qu'un modèle m est ε-significatif pour les données I lorsque :

$$\exists \delta > 0, \text{ t.q. } \text{NFA}(m,\delta,I) < \varepsilon \tag{3.37}$$

On peut alors définir le NFA d'un modèle m pour des données I par :

$$\text{NFA}(m) = \arg\min\{\varepsilon > 0, \text{ t.q. } \exists \delta > 0,\ \text{NFA}(m,\delta,I) < \varepsilon\} \tag{3.38}$$

Les méthodes *a contrario* fonctionnent alors de la même façon que les méthodes classiques mais utilise cette fois le score donné par le NFA au lieu de la distance d. On conserve soit le modèle avec le plus faible NFA, soit les modèles dont le NFA est en dessous d'un certain seuil ε.

Notons que les méthodes *a contrario* se disent *sans paramètre* et en toute rigueur cet aspect est contredit lorsqu'on conserve les modèles dont le NFA est en dessous d'un seuil ε. Cependant, Desolneux *et al.* [12] expliquent que cette dépendance est logarithmique, une valeur très précise n'a donc pas besoin d'être trouvée. De fait, dans [19], les auteurs expérimentent plusieurs seuils et montrent empiriquement que la valeur de ε ne joue pas un rôle aussi important que dans les méthodes classiques et notamment qu'un réglage grossier est suffisant tandis que les méthodes classiques requièrent un réglage fin qui diffère chaque fois que les données en entrée sont modifiées.

Notons enfin que les NFA ne sont jamais calculés de manière exacte (*i.e.* en suivant l'Eq.3.38) car cela serait trop coûteux en temps de calcul. Il est calculé en pratique de manière approchée et le choix de l'hypothèse $\mathcal{H}_0$ devient alors crucial puisqu'il permet ou non de faire certaines approximations permettant une estimation fiable du NFA.

Chapitre 4

Détection multi-échelle de segments de droites

Les scènes d'intérieur contiennent peu de texture ce qui rend la détection et la mise en correspondance de points saillants plus difficile, et la calibration parfois impossible. Pour résoudre ce problème, une approche consisterait à adapter les détecteurs de points à ce type de scène. Cependant, même l'œil humain n'en détecte pas toujours et la trop grande présence de cas planaires (cas dégénéré de calibration avec points [63]) rendraient la calibration peu précise. Une autre approche consiste à utiliser un autre type de *feature* comme les segments de droites, présents en grande quantité dans les scènes d'intérieur et plus généralement dans les scènes urbaines.

Il existe déjà un grand nombre de détecteurs de segments de droite et certains d'entre eux obtiennent de très bons résultats sur la plupart des images. Cependant, les détections obtenues donnent de moins bons résultats pour les images haute-résolution et en particulier dans les scènes d'intérieur.

Nous présentons, dans ce chapitre, un détecteur de segments multi-échelle qui généralise l'une des méthodes état-de-l'art pour la faire fonctionner dans le cadre des images haute résolution ainsi que pour les scènes d'intérieur. Nous comparons ensuite les détections de cette méthode avec celles de l'état de l'art tant qualitativement que quantitativement (dans le cadre de la calibration).

Contents

4.1 Méthodes existantes

Les segments étant des bords de régions particuliers des images, ceux-ci sont généralement détectés après une détection de bord. On procède à un calcul de la carte de gradient suivi d'un seuillage grossier pour séparer les pixels appartenant aux bords des autres.

L'une des premières méthodes de détection de segments [5] correspond à un seuillage de gradient suivie d'une transformée de Hough pour détecter les segments rectilignes parmi les bords. De nombreux seuils sont ensuite utilisés pour séparer ces lignes en segments (*e.g.* différence angulaires entre gradients, longueur du segment, ...). Si cette méthode fonctionne correctement une fois les seuils bien paramétrés, ils doivent généralement être changés d'une scène, voire d'une image, à l'autre. Enfin, ce type de méthode se comporte généralement très mal dans les zones de bruit ou à la texture très variée (*e.g.* buisson, zones texturées, ...) où des segments aberrants sont alors détectés (voir Figure 4.1).

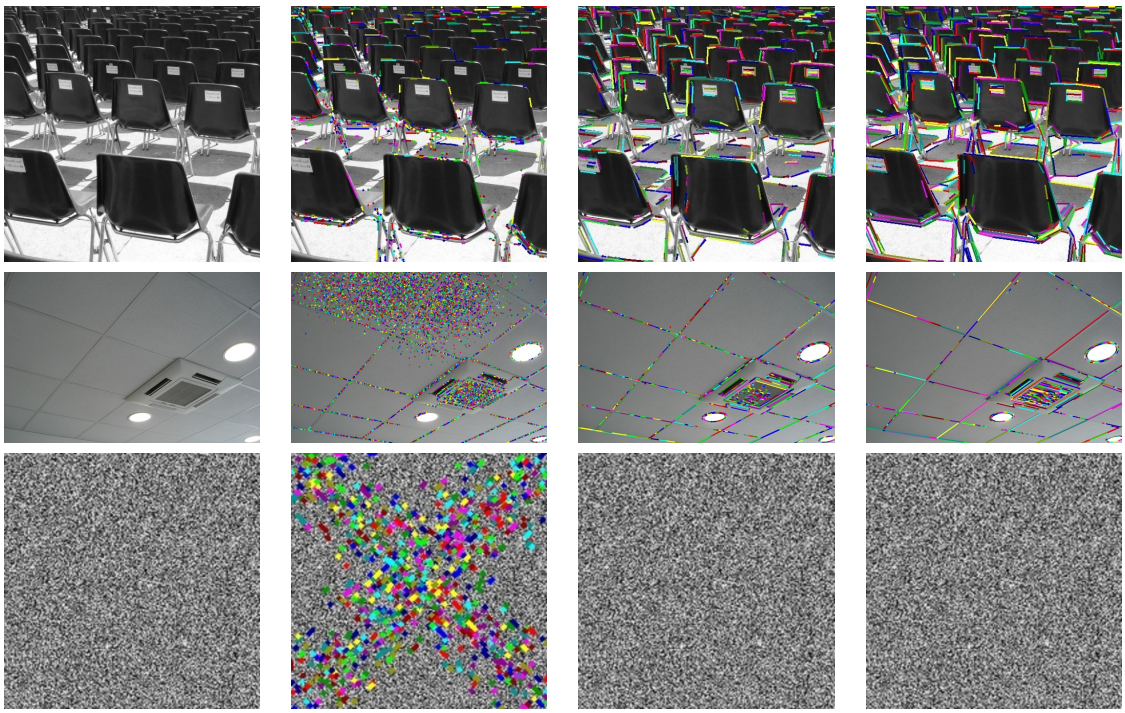


Image originale | HT [32] | LSD [19] | ED-Lines [2]

FIGURE 4.1 – Comparaison de différents détecteurs sur 3 images (*Chaises*, *Plafond* et *Bruit*). Les détections via transformée de Hough sont très sensibles aux seuils choisis (*e.g. Bruit*, coin supérieur gauche de *Plafond*. Les méthodes *a contrario* (LSD, ED-Lines) ne détectent ni trop ni trop peu de segments. Elle ont cependant des problèmes de sur-segmentation sur les images de haute résolution (*e.g. Plafond*).

Des méthodes plus sophistiquées ont alors été développées. Faugeras *et al.* [15] relient les points appartenant aux bords avec des critères de rectitude tandis que Burns *et al.* [22] choisissent de s'intéresser uniquement aux orientations des gradients en ignorant leur magnitude. Si la plupart de ces méthodes tendent à améliorer les résultats de leur prédécesseurs, elles se heurtent toujours au problème principal du choix du ou des seuils pour la détection de segments.

Desolneux *et al.* [12] introduisent les premiers le principe de Helmholtz et les méthodes *a contrario* pour la détection de segments (on notera leur méthode DMM dans la suite). Ces méthodes ont l'avantage de se fonder sur un modèle statistique non paramétrique pour éviter l'utilisation de seuils. Cependant, la méthode développée a tendance à trop fusionner certains

segments. Cette méthode est ensuite corrigée avec un critère permettant de séparer les segments [20] mais elle reste très lente (complexité en $O(n^4)$). En combinant la méthode de détection de Burns *et al.* [22] avec un critère de validation *a contrario*, la méthode LSD [19] permet une détection rapide et de très bonne qualité tout en évitant l'utilisation de seuils. Elle reste encore aujourd'hui, avec ED-Lines [2], l'une des meilleures méthodes de détection de segments.

Image originale LSD [19] ED-Lines [2]

FIGURE 4.2 – On observe de nombreux phénomènes de sur-segmentation sur des images de haute résolution (ici plus de 15 Mpixels). Les zones localement peu contrastées génèrent également peu de détections (*e.g.* carrelage et bord droit du tableau).

La Figure 4.1 montre des exemples de détection avec quelques unes des nombreuses méthodes de détection de segments :

- HT, pour *Hough Transform*, implémentation d'openCV [32] avec les paramètres par défaut,
- LSD [19], pour *Line Segment Detection*, méthode *a contrario*,
- ED-Lines [2], pour *Edge Drawing Lines*, méthode *a contrario*.

L'exemple typique des méthodes *a contrario* est celui d'une image de bruit. En effet, les méthodes plus classiques ont tendance à détecter de nombreuses lignes (en fonction du seuil choisi) tandis que les méthodes *a contrario* ne détectent rien, à juste titre. Nous pouvons également remarquer que la présence de seuil pour la méthode HT entraîne la détection de segments due au bruit dans le coin supérieur gauche de *Plafond* mais empêche la détection dans le coin supérieur droit de *Chaises*.

Même si les méthodes à base de transformée de Hough sont en général plus rapides (temps réel), elles sont également moins précises et nécessitent de fixer un seuil pour chaque image. Le temps de calcul n'étant pas un réel handicap dans le cadre de la reconstruction 3D, nous avons privilégié l'utilisation de méthodes *a contrario*.

Les deux principales méthodes sont LSD [19] et ED-Lines [2]. Elles fournissent de très bons résultats dans la plupart des cas et peuvent être utilisées telles quelles en reconstruction 3D. Cependant leur qualité chute lorsqu'on les utilise sur des images de haute résolution et pour les scènes d'intérieur. En particulier, les long segments ont tendance à être découpés en de nombreux petits morceaux (effet de sur-segmentation) qui empêcheraient une reconstruction 3D de bonne qualité. De plus les zones de faible contraste sont souvent vides de toute détection. Les phénomènes sont partiellement observables, en particulier pour LSD, dans l'image *Plafond* de la Figure 4.1 car l'image d'origine a une taille de 5 Mpixels. Cependant, l'effet s'accentue sur des images de 15 Mpixels comme on peut le voir dans la Figure 4.2.

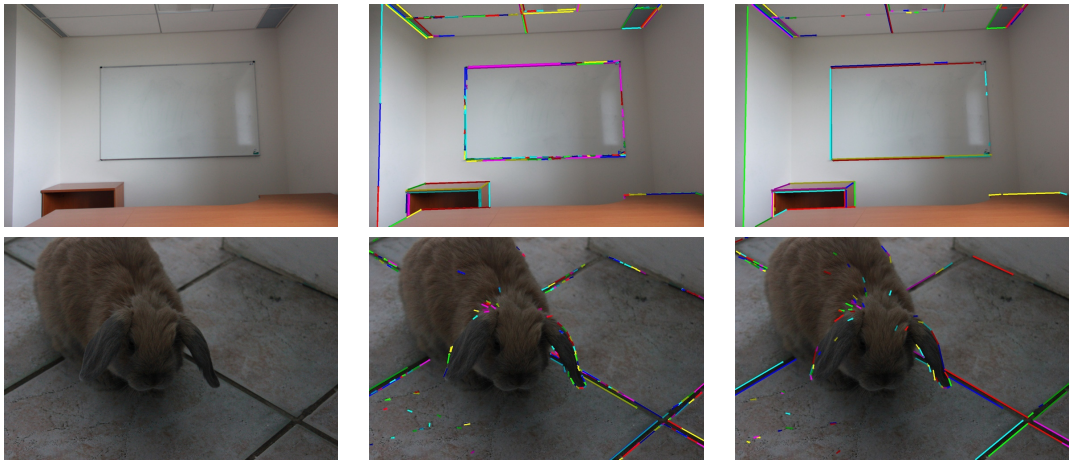


Image originale LSD [19] ED-Lines [2]

FIGURE 4.3 – Les phénomènes de sur-segmentation se réduisent si on réduit suffisamment la taille de l'image (ici d'un rapport 4 en largeur et en hauteur). Les zones localement peu contrastées à haute résolution génèrent ici quelques détections.

Ce phénomène de sur-segmentation affecte tous les détecteurs et HT n'y échappe pas non plus comme on peut le voir dans la Figure 4.1. Il apparaît cependant qu'en réduisant la résolution des images, les détecteurs ne sur-segmentent plus autant leurs détections (voir Figure 4.3). Motivés par cette observation, Lopez *et al.* [45] proposent une méthode multi-échelle. Cependant celle-ci utilise de nombreux critères (parallélisme, proximité...) accompagnés de paramètres pour valider la fusion de certains segments. L'utilisation d'autant de paramètres la rend alors peu robuste au changement de scènes.

4.2 Validation *a contrario* d'un segment

Plusieurs méthodes (DMM[12], LSD [19], ED-Lines [2]) utilisent des critères *a contrario* dans le cadre de la détection de segments. Si ces critères diffèrent légèrement, la méthode suivie est très similaire. Nous présentons dans cette section les détails de chacune de ces méthodes en reprenant les termes présentés dans la section 3.9.

Données d'entrée :

Pour chaque méthode, les données correspondent à des groupes de pixels dans une image I donnée de largeur w et hauteur h. Cependant si DMM et ED-Lines imposent à ces groupes d'avoir une épaisseur de 1 pixel uniquement, LSD les autorise à avoir une épaisseur variable. De plus ED-Lines et LSD construisent leurs groupes avec des méthodes gloutonnes initialisées n'importe où dans l'image. La méthode DMM n'étudie que les lignes passant par 2 points du bord de l'image ; les groupes de pixels correspondent alors aux pixels présents sur ces lignes.

Modèle estimé :

Chaque méthode propose un critère permettant de faire correspondre un segment à un groupe de pixels. Les méthodes diffèrent légèrement :

- DMM : comme vu précédemment, le groupe de pixel appartient à une ligne support. Le segment associé est alors défini par les deux extrémités du groupe de pixels.

- LSD : le segment est calculé comme boite englobante du groupe de pixels considéré. Le centre de masse c pondéré par la magnitude des gradients des pixels ainsi que le premier axe d'inertie d sont d'abord calculés. Une ligne est alors définie à partir du point c et avec pour direction le vecteur d. La longueur et la largeur du segment sont enfin déterminés par la taille de la boîte englobante.
- ED-Lines : le segment est calculé à l'aide d'une régression aux moindres carrés du groupe de pixels.

Les méthodes LSD et ED-Lines obtiennent ainsi des coordonnées sub-pixeliques pour les segments tandis que la précision de DMM est moins bonne puisque limitée à la discrétisation pixellique de l'image.

Dans le cas de ED-Lines et DMM, le nombre de segments pouvant être estimés est de $n_{SEG} = (wh)^2$ car 2 points de l'image définissent un segment. Pour LSD, comme les segments peuvent avoir une épaisseur variable, le nombre de segments possible est plus élevé ; $n_{SEG} = (wh)^{5/2}$.

Distance données-modèle :

Dans chacune des méthodes, on utilise la distance angulaire entre la direction du gradient d'un pixel et la direction du segment candidat. Un paramètre de précision angulaire τ est introduit qui rend la distance d binaire :

- $d = 0$ lorsque la direction du pixel considéré et celle du segment sont les mêmes à τ près,
- $d = \infty$ sinon

Modèle de fond :

Le modèle de fond $\mathcal{H}_0$ utilisé est le même pour les trois méthodes. Il correspond à des images de même taille que I mais dont les directions de gradient en chaque pixel sont indépendamment et identiquement distribuées selon une loi uniforme. Notons que c'est le fait d'utiliser un tel modèle $\mathcal{H}_0$ qui rend les détections des méthodes *a contrario* peu sensibles aux textures très bruitées (*e.g.* buisson, pelage, image de bruit, ...).

Nombre de Fausses Alarmes :

Le Nombre de Fausses Alarmes (NFA) peut alors être défini pour évaluer l'adéquation d'un groupe de pixel $\mathcal{P}$ au segment s qui lui est associé. Sa formule suit l'équation présentée dans la section 3.9 :

$$\mathrm{NFA}(I,s,p) = N_{TEST}\,\mathbb{P}_{\mathcal{H}_0}[k(X,s,p) \geq k(I,s,p)] \tag{4.1}$$

où $k(I,s,p) = \sum_{q\in\mathcal{P}} \mathbb{1}_{d(q,s,p)=0}$ représente le nombre de pixel de $\mathcal{P}$ alignés avec la direction de s à une précision angulaire $\tau = \pi p$ près. La formule $\mathbb{P}_{\mathcal{H}_0}[k(X,s,p) \geq k(I,s,p)]$ représente la probabilité d'obtenir au moins $k(I,s,p)$ pixels alignés à p près dans une image aléatoire X suivant le modèle de fond $\mathcal{H}_0$. Même si les segments étudiés diffèrent selon les méthodes, cette probabilité suit la même formule :

$$\mathbb{P}_{\mathcal{H}_0}[k(X,s,p) \geq k(I,s,p)] = \mathcal{B}(|s|, k(I,s,p), p) \tag{4.2}$$

où $|s|$ correspond aux nombre de pixels dans le rectangle englobant du segment, soit :

- $|s|$ est la longueur du segment pour DMM et ED-Lines,
- $|s|$ est l'aire du segment-rectangle pour LSD.

D'autre part, $\mathcal{B}(n,k,p)$ désigne la queue de la loi binomiale, c'est-à-dire :

$$\mathcal{B}(n,k,p) = \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i}. \tag{4.3}$$

Enfin, le nombre N_{TEST} de modèles potentiellement testés correspond au nombre de segments n_{SEG} pouvant être observés dans l'image I. Dans le cas de LSD, comme plusieurs précisions angulaires p sont potentiellement testées, le nombre de modèle augmente d'un facteur γ égal au nombre de valeur différentes de p.

On dit alors qu'un segment s d'une image I est ε-significatif lorsque, pour un $\varepsilon > 0$ donné :

$$\exists p \in \mathbb{R}^{+,\star} \text{ t.q. } \mathrm{NFA}(I,s,p) < \varepsilon \tag{4.4}$$

4.3 Nombre de Fausses Alarmes pour multi-segments

Dans le but de corriger l'une des restrictions de la méthode DMM, Grompone von Gioi *et al.* [20] définissent également le NFA d'un groupe de n segments ou n-segment (que nous notons $\mathrm{NFA}_{\mathcal{M}}$). Ce $\mathrm{NFA}_{\mathcal{M}}$ permet de définir s'il est plus significatif de fusionner n segments en un seul ou de les laisser séparés. Cependant, ils se basent sur la méthode DMM et ne considèrent donc que le cas des segments d'épaisseur 1 pixel. Pour un n-segment $S_n = \{s_1, ..., s_n\}$ sur une ligne support L, le $\mathrm{NFA}_{\mathcal{M}}$ est défini par :

$$\mathrm{NFA}_{\mathcal{M}}(S_n,p) = \#\mathcal{L} \binom{|L|}{2n} \prod_{i=1}^{n} (|s_i|+1)\mathcal{B}(|s_i|,k_{s_i},p/2\pi) \tag{4.5}$$

où $\#\mathcal{L} \sim (w+h)^2$ correspond au nombre de lignes support dans l'image I, $|L|$ à la longueur de la ligne support et k_{s_i} au nombre de pixels de s_i alignés à p près avec L.

Cette formule fait apparaître un facteur $\#\mathcal{L}\binom{|L|}{2n}$ qui correspond au nombre de tests. Le second facteur est choisi de telle sorte que le nombre de n-segments ε-significatif soit en moyenne inférieur à ε dans une image suivant le modèle de fond $\mathcal{H}_0$ (voir [20] pour plus de détails).

Cette formule possède quelques propriétés intéressantes :

- $\forall n > 0,\ \forall \varepsilon > 0,\ \mathbb{E}_{\mathcal{H}_0}[\text{nombre de } n\text{-segments } \varepsilon\text{-significatif}] \leq \varepsilon$
- Soit $n > 1$, si $\forall i \in [1,n],\ s_i$ est 1-significatif, alors $\forall i \in [1,n],\ \mathrm{NFA}_{\mathcal{M}}(S_n,p) < \mathrm{NFA}_{\mathcal{M}}(s_i,p)$

La première propriété permet de justifier cette définition du NFA. Elle conserve le sens usuel des formules *a contrario* et peut donc être utilisée pour évaluer si un segment n'a pas été détecté *par hasard*.

La seconde propriété indique, au sens du $\mathrm{NFA}_{\mathcal{M}}$, qu'un multi-segment est toujours plus significatif que chacun de ses sous-segments pris séparément dès lors qu'ils sont tous au moins 1-significatif. Ainsi, ce $\mathrm{NFA}_{\mathcal{M}}$ n'a de sens que pour la comparaison entre un n-segment $S_n = \{s_1, ..., s_n\}$ et le segment généré par l'ensemble des sous-segments $\tilde{S}_n = \mathcal{S}eg(\cup_{i=1}^{n} s_i)$ (où $\mathcal{S}eg(\cup_{i=1}^{n} s_i)$ représente le rectangle minimal contenant tous les s_i). Si le $\mathrm{NFA}_{\mathcal{M}}$ de S_n est plus faible que celui de $\tilde{S}_n$ alors les segments doivent être conservés séparés, sinon ils doivent être fusionnés.

Nous avons cherché à généraliser ce $\mathrm{NFA}_{\mathcal{M}}$ au cas d'un n-segment avec la structure utilisée dans LSD (*i.e.* des segments d'épaisseur variable) tout en conservant les propriétés présentées précédemment.

Définition : Pour un n-segment S_n dans une image I, nous définissons son $\mathrm{NFA}_{\mathcal{M}}$ par :

$$\mathrm{NFA}_{\mathcal{M}}(S_n,p) = \gamma(wh)^5 \binom{|\tilde{S}_n|^{5/2}}{n} \prod_{i=1}^{n} (|s_i|+1)\mathcal{B}(|s_i|,k_{s_i},p/2\pi) \tag{4.6}$$

où $|\tilde{S}_n|$ est l'aire du segment englobant $\tilde{S}_n = \mathcal{S}eg(\cup_{i=1}^{n} si)$. On suppose de plus que les s_i sont disjoints. Si les ensembles de pixels définissant les segments sont toujours disjoints par construction, leurs rectangles englobants (les s_i) ne sont pas nécessairement parfaitement disjoint, d'où la nécessité de l'hypothèse.

La définition 4.6 est à mettre en parallèle avec la précédente (4.5). Le premier facteur correspond au nombre de tests possibles. Cependant, ici la notion de ligne support est remplacée par celle de segment englobant faisant apparaître le terme $(wh)^5$ au lieu de $\#\mathcal{L}$. Ce terme de $(wh)^5$ correspond au produit du nombre $(wh)^{5/2}$ de segments possibles dans une image de taille $w \times h$ avec un terme nécessaire aux propriétés suivantes. Le nombre de segment est calculé de la même manière que dans LSD [19], il y a wh choix pour les deux extrémités et l'épaisseur peut s'approximer comme un choix parmi $(wh)^{1/2}$ valeurs en moyenne. Le nombre de n-segments potentiellement étudiés à partir d'un segment englobant devient $\binom{|\tilde{S}_n|^{5/2}}{n}$ au lieu de $\binom{|L|}{2n}$. Enfin, comme dans LSD [19], le terme γ représente les tests de différentes précisions angulaires. Le second facteur est par contre exactement le même, seul le calcul de $|s_i|$ diffère puisqu'il correspond à une aire et non plus à une longueur.

4.3.1 Preuve des propriétés

Nous avons redémontré les propriétés vérifiées par le $\text{NFA}_{\mathcal{M}}$ dans le cadre des segments de LSD. Pour cela nous avons eu besoin du lemme suivant :
Lemme : $\forall \alpha > 0,\ \mathbb{P}_{\mathcal{H}_0}\left[\prod_{i=1}^{n} \mathcal{B}(|s_i|, k_{s_i}, p) \leq \alpha\right] \leq \prod_{i=1}^{n}(|s_i|+1)\alpha$

Démonstration. On note $U_i = \mathcal{B}(|s_i|, k_{s_i}, p)$. Soit $\alpha > 0$:

$$\mathbb{P}_{\mathcal{H}_0}\left[\prod_{i=1}^{n} U_i \leq \alpha\right] = \sum_{(u_2,\ldots,u_n)} \mathbb{P}_{\mathcal{H}_0}\left[\prod_{i=1}^{n} U_i \leq \alpha | U_2 = u_2, \ldots, U_n = u_n\right] \mathbb{P}_{\mathcal{H}_0}\left[U_2 = u_2, \ldots, U_n = u_n\right] \tag{4.7}$$

En supposant les s_i disjoints et donc les U_i indépendants, l'équation devient :

$$\mathbb{P}_{\mathcal{H}_0}\left[\prod_{i=1}^{n} U_i \leq \alpha\right] = \sum_{(u_2,\ldots,u_n)} \mathbb{P}_{\mathcal{H}_0}\left[U_1 \leq \frac{\alpha}{\prod_{i=2}^{n} u_i}\right] \mathbb{P}_{\mathcal{H}_0}\left[U_2 = u_2, \ldots, U_n = u_n\right] \tag{4.8}$$

De plus, comme $\forall \alpha > 0,\ \mathbb{P}[U_i \leq \alpha] \leq \alpha$ et $\mathbb{P}_{\mathcal{H}_0}\left[U_2 = u_2, \ldots, U_n = u_n\right] \leq \mathbb{P}_{\mathcal{H}_0}\left[U_2 \leq u_2, \ldots, U_n \leq u_n\right]$:

$$\mathbb{P}_{\mathcal{H}_0}\left[\prod_{i=1}^{n} U_i \leq \alpha\right] \leq \sum_{(u_2,\ldots,u_n)} \frac{\alpha}{\prod_{i=2}^{n} u_i} \prod_{i=2}^{n} u_i \tag{4.9}$$

Enfin, le nombre de valeurs différentes pour u_i étant déterminé par les $(1+|s_i|)$ valeurs de k_{s_i}, on obtient finalement :

$$\mathbb{P}_{\mathcal{H}_0}\left[\prod_{i=1}^{n} U_i \leq \alpha\right] \leq \prod_{i=2}^{n}(|s_i|+1)\alpha \leq \prod_{i=1}^{n}(|s_i|+1)\alpha \tag{4.10}$$

La deuxième inégalité est triviale mais elle permet une formule symétrique en s_i. □

La première propriété découle du lemme précédent :
Propriété 1 : $\forall n > 0,\ \mathbb{E}_{\mathcal{H}_0}[\text{nombre de } n\text{-segments } \varepsilon\text{-significatifs}] \leq \varepsilon$

Démonstration. Soit $n > 0$. Par définition :

$$\mathbb{E}_{\mathcal{H}_0}[\varepsilon] = \mathbb{E}_{\mathcal{H}_0}[\text{nombre de } n\text{-segments } \varepsilon\text{-significatifs}] = \sum_{S \in I} \sum_{p=p_1}^{p_\gamma} \sum_{S_n \in S} \mathbb{P}_{\mathcal{H}_0}[\text{NFA}_{\mathcal{M}}(S_n, p) \leq \varepsilon] \tag{4.11}$$

Or $\mathrm{NFA}_{\mathcal{M}}(S_n,p) \leq \varepsilon$ est équivalent à :

$$\prod_{i=1}^{n} \mathcal{B}(|s_i|,k_{s_i},p/2\pi) \leq \frac{\varepsilon}{\gamma(wh)^5 \binom{|\tilde{S}_n|^{5/2}}{n} \prod_{i=1}^{n}(|s_i|+1)} \tag{4.12}$$

En utilisant le lemme, on obtient :

$$\mathbb{P}_{\mathcal{H}_0}\left[\prod_{i=1}^{n} \mathcal{B}(|s_i|,k_{s_i},p/2\pi) \leq \frac{\varepsilon}{\gamma(wh)^5 \binom{|\tilde{S}_n|^{5/2}}{n} \prod_{i=1}^{n}(|s_i|+1)}\right] \leq \frac{\varepsilon}{\gamma(wh)^5 \binom{|\tilde{S}_n|^{5/2}}{n}} \tag{4.13}$$

En combinant avec 4.11, on obtient :

$$\mathbb{E}_{\mathcal{H}_0}[\varepsilon] \leq \sum_{S\in I} \sum_{p=p_1}^{p_\gamma} \sum_{S_n\in S} \frac{\varepsilon}{\gamma(wh)^5 \binom{|\tilde{S}_n|^{5/2}}{n}} = (wh)^{5/2}\gamma \binom{|\tilde{S}_n|^{5/2}}{n} \frac{\varepsilon}{\gamma(wh)^5 \binom{|\tilde{S}_n|^{5/2}}{n}} \leq \varepsilon \tag{4.14}$$

□

Propriété 2 : Soit $n>1$, si $\forall i \in [1,n]$, s_i est 1-significatif, alors $\forall i \in [1,n]$, $\mathrm{NFA}_{\mathcal{M}}(S_n,p) < \mathrm{NFA}_{\mathcal{M}}(s_i,p)$

Démonstration.

$$\begin{aligned}
\mathrm{NFA}_{\mathcal{M}}(S_n,p) &= \gamma(wh)^5 \binom{|\tilde{S}_n|^{5/2}}{n} \prod_{i=1}^{n}(|s_i|+1)\mathcal{B}(|s_i|,k_{s_i},p/2\pi) \\
&= \frac{\binom{|\tilde{S}_n|^{5/2}}{n}}{(\gamma(wh)^5)^{n-1}\prod_{i=1}^{n}\binom{|s_i|^{5/2}}{1}} \prod_{i=1}^{n} \gamma(wh)^5 \binom{|s_i|^{5/2}}{1}(|s_i|+1)\mathcal{B}(|s_i|,k_{s_i},p/2\pi) \\
&= \frac{\binom{|\tilde{S}_n|^{5/2}}{n}}{(\gamma(wh)^5)^{n-1}\prod_{i=1}^{n}\binom{|s_i|^{5}}{1}} \prod_{i=1}^{n} \mathrm{NFA}_{\mathcal{M}}(s_i,p) \\
&\leq \frac{\binom{(wh)^{5/2}}{n}}{(wh)^{5(n-1)}} \prod_{i=1}^{n} \mathrm{NFA}_{\mathcal{M}}(s_i,p) \text{ car } |\tilde{S}_n| \leq wh,\ \gamma \geq 1 \text{ et } \prod_{i=1}^{n}\binom{|s_i|^5}{1} \geq 1 \\
&< \frac{(wh)^{5n/2}}{(wh)^{5(n-1)}} \prod_{i=1}^{n} \mathrm{NFA}_{\mathcal{M}}(s_i,p) = (wh)^{5(1-n/2)} \prod_{i=1}^{n} \mathrm{NFA}_{\mathcal{M}}(s_i,p) \\
&< \prod_{i=1}^{n} \mathrm{NFA}_{\mathcal{M}}(s_i,p)
\end{aligned}$$

Comme les segments s_i sont 1-significatifs, le produit des $\mathrm{NFA}_{\mathcal{M}}$ est inférieur au $\mathrm{NFA}_{\mathcal{M}}$ d'un s_i en particulier et la propriété est bien démontrée. □

4.3.2 Score de Fusion

Définition : Pour un n-segment S_n dans une image I, nous définissons le score de fusion par :

$$\mathcal{F}(S_n) = \log\left(\frac{\mathrm{NFA}_{\mathcal{M}}(S_n,p)}{\mathrm{NFA}_{\mathcal{M}}(\tilde{S}_n p)}\right) = \log\left(\frac{\mathrm{NFA}_{\mathcal{M}}(s_1,\ldots,s_n,p)}{\mathrm{NFA}_{\mathcal{M}}(Seg(\cup_{i=1}^{n} s_i),p)}\right) \tag{4.15}$$

où le terme $Seg(\cup_{i=1}^{n} s_i)$ représente le plus petit segment-rectangle englobant l'union des segments-rectangles s_i.

Cette définition permet de relier directement le fait de devoir fusionner un n-segment à un score positif de fusion. En pratique, l'utilisation du logarithme permet également de contenir les valeurs des NFA qui tendent à être extrêmement élevées ou extrêmement faibles.

4.4 Approche multi-échelle

Nous avons vu que les méthodes *a contrario* obtenaient des résultats de qualité médiocre pour des images de haute résolution (voir Fig. 4.2). Or ces mêmes algorithmes obtiennent de bons résultats sur ces mêmes images une fois que la résolution à été réduite (voir Fig. 4.3). Partant de ce constat, nous avons cherché à utiliser une approche multi-échelle en utilisant les résultats obtenus à l'aide de LSD aux échelles plus grossières puis en les raffinant aux échelles plus fines. Nous avons choisi d'utiliser LSD plutôt qu'ED-Lines car leurs résultats sont assez comparables, car l'algorithme de LSD se prête plus facilement à une approche multi-échelle et pour des raisons plus pratiques (le code source LSD est disponible contrairement à celui d'ED-Lines). Cependant, nous pensons qu'une approche similaire à celle présentée pourrait être appliquée à de nombreux autres détecteurs de segments.

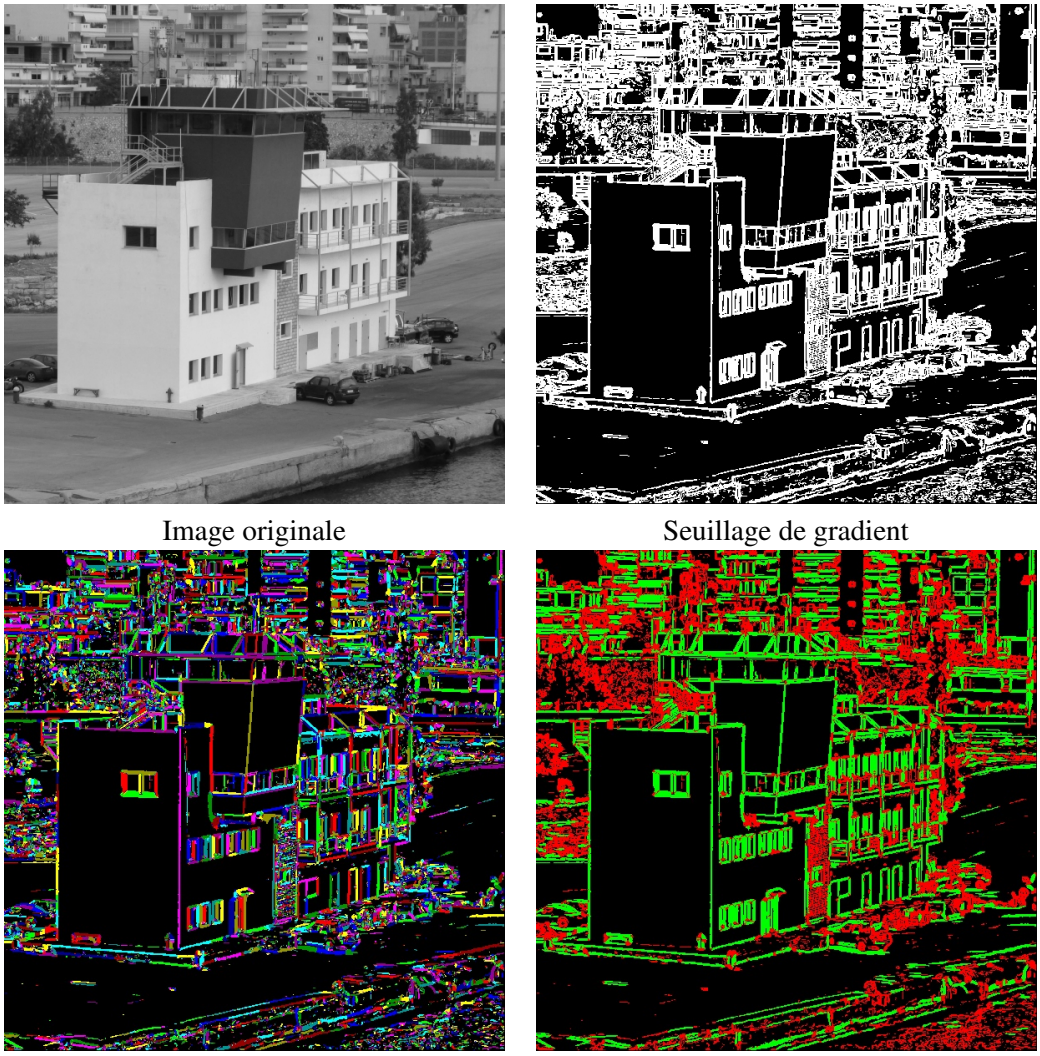


Image originale | Seuillage de gradient

Croissance de Région | Validation *a contrario*

FIGURE 4.4 – Illustration des différentes étapes de l'algorithme de LSD. Une détection de bord suivie d'une agrégation en régions de même direction et une validation *a contrario* des segments corrects. Les bords sont affichés en blanc, puis les différents segments agrégés par LSD (une couleur correspond à un segment). Enfin, les segments validés par le NFA sont affichés en vert tandis que ceux rejetés sont affichés en rouge.

4.4.1 LSD : Line Segment Detection

Nous décrivons ici les détails de la méthode de détection LSD [19]. En effet, nous en avons repris les grandes étapes pour notre méthode multi-échelle. La méthode LSD tient en trois étapes principales :

- **Détection des contours** de l'image,
- **Agrégation** des pixels de bords en régions d'orientation similaires,
- **Validation** des segments à l'aide du NFA.

Les étapes successives sont illustrées dans la Figure 4.4.

L'étape de détection de contours est en fait un seuillage grossier du gradient. Ce seuil est choisi suffisamment bas pour ne pas empêcher la détection de certains segments dont le contraste serait trop faible. Il est également choisi suffisamment élevé pour éviter de mauvaises détections générées par du bruit ou certaines textures (*e.g.* rainures dans le bois). Seuls les pixels dont le gradient dépasse ce seuil seront étudiés dans la suite de l'algorithme.

L'agrégation des pixels se fait à l'aide d'une méthode de croissance de région. Pour cela, une région est initialisée avec le pixel dont le gradient a la plus grande magnitude, puis est potentiellement étendue aux 8 voisins. Un voisin est ajouté si, d'une part, son gradient est suffisamment élevé (*i.e.* appartient aux bords détectés dans l'étape 1) et, d'autre part, si sa direction est suffisamment proche de celle des pixels de la région (*i.e.* la direction de son gradient et celle du segment associé à la région sont les mêmes à une précision p près). Cette région est alors étendue de manière itérative jusqu'à ne plus avoir de candidats à ajouter. La région est enfin conservée pour l'étape suivante et la croissance de région reprend sur le pixel non visité avec la plus grande magnitude de gradient.

Les régions de pixels sont alors assimilées au segment correspondant en suivant la méthode présentée dans la section 4.2. Enfin, ces segments sont conservés en fonction du score fourni par le NFA. Avant d'être définitivement rejeté, les segments sont raffinés pour essayer de contrebalancer les effets négatifs de la croissance de région et un nouveau calcul de NFA est réalisé (*e.g.* deux segments d'angle similaires et en contact "en V" vont généralement ne créer qu'une région, à tort).

C'est lors de l'étape d'agrégation des pixels que le phénomène de sur-segmentation apparaît. En effet, dans les images haute-résolution et particulièrement en intérieur, le manque de contraste et la présence de bruit dans certaines zones a tendance à énormément affecter le gradient. Ainsi, le bruit et le faible contraste tendent à découper les segments en petites régions connexes et l'algorithme de croissance de région ne détecte que ces petites régions au lieu du segment entier. Ces segments sont soit validés par le NFA et engendrent le phénomène de sur-segmentation, soit rejetés par le NFA et produisent des absences de détections (voir Figure 4.2).

4.4.2 MLSD : Multiscale Line Segment Detection

Nous avons cherché à compenser la mauvaise agrégation des pixels en région de LSD à l'aide d'une méthode multi-échelle (MLSD). Notre méthode peut se diviser en 3 étapes :

- Une **transition multi-échelle** où l'information obtenue à l'échelle précédente est raffinée pour calculer les segments à l'échelle courante,
- Une **détection classique** à l'aide de LSD sur les zones non raffinées,
- Une **fusion post-détection** qui fusionne les segments voisins si nécessaire.

L'ensemble des étapes est illustré dans l'Algorithme 1.

Calcul de la pyramide d'images

La pyramide d'image correspond à un ensemble de K images, version réduites de l'image originale avec un rapport 2^k. Le nombre d'échelles considérées dépend de la taille de l'image originale. Nous avons utilisé la formule suivante :

$$K = \min\{k \in \mathbb{N} \text{ s.t. } \max(w,h) \leq 2^k s_{max}\} \tag{4.16}$$

où w et h sont la largeur et la hauteur de l'image.

Cette formule cherche le nombre minimal d'images pour qu'à l'échelle la plus grossière, l'image ait une taille inférieure à $s_{max} \times s_{max}$. Remarquant que LSD donne généralement de bons résultats sur des images de taille inférieure à 1000×1000 pixels, nous avons fixé la valeur de $s_{max} = 1000$ (voir figure 4.5 ou 4.7, seule la première ligne correspond à des images de moins de 1 Mpixel). Par ailleurs, réduire trop l'image génère des détections artefacts qui se répercuteront aux échelles supérieures.

Enfin, chaque image de la pyramide est calculée après une convolution avec un noyau gaussien, comme l'algorithme original.

Entrée : Image I;
Sortie : Ensemble de segments $\mathcal{S}$;

pour $k = 0$ to K **faire**
  Calculer l'image réduite I^k (I^K est l'image originale);

  // 1. Initialisation des segments à partir de l'information obtenue à l'échelle précédente, quand elle existe
  si $k = 0$ **alors**
    $\mathcal{S}^k \leftarrow \emptyset$;
    Optionnellement, la détection est désactivée dans les régions de l'image où la densité de gradient élevé est trop importante (Filtre de gradient dense, voir page 54);
  sinon
    $\mathcal{S}^k \leftarrow \text{RAFFINEMENT}(\mathcal{S}^{k-1})$;
  fin si

  // 2. Ajout des segments détectés par LSD [19] :;
  $\mathcal{S}^k \leftarrow \mathcal{S}^k \cup \text{LSD}(I^k)$;

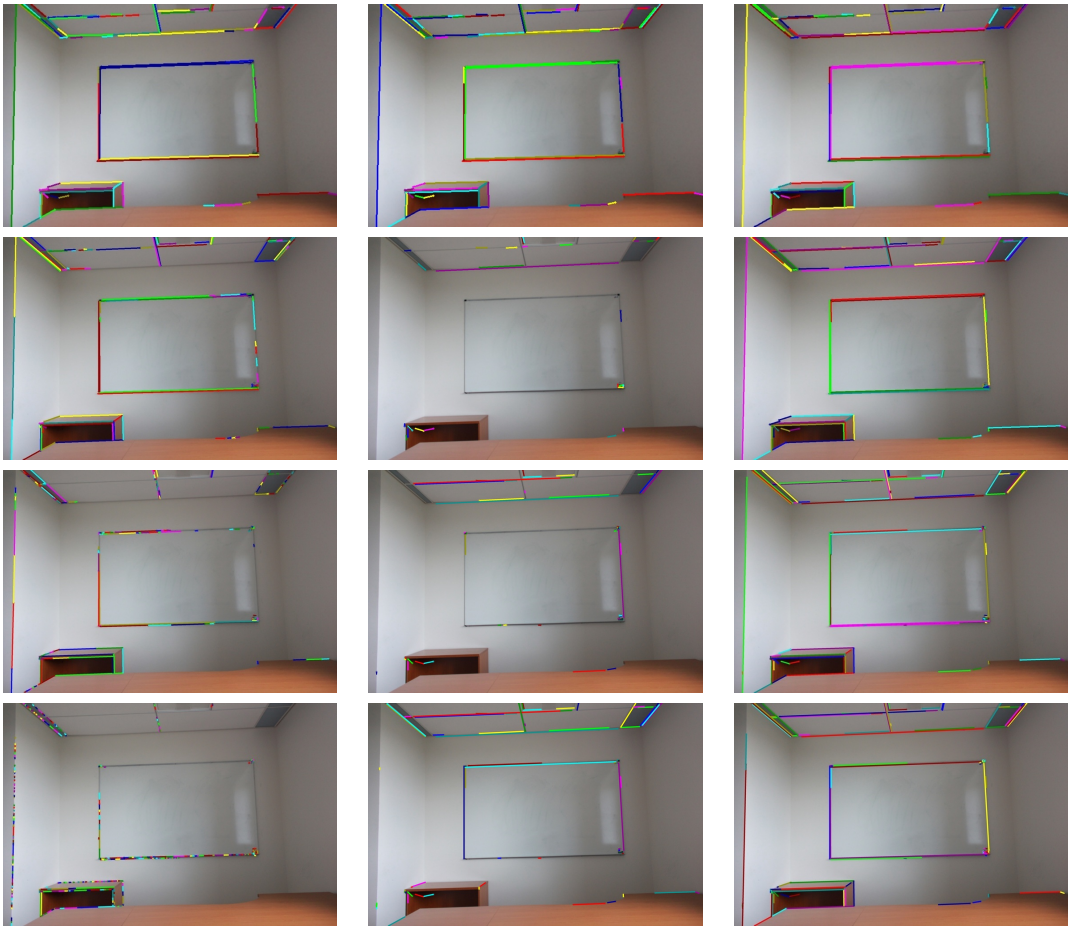
  // 3. Fusion des segments proches
  $\mathcal{S}^k \leftarrow \text{FUSION}(\mathcal{S}^k)$;
fin pour

retourner $\mathcal{S}^K$;

Algorithm 1: Algorithme en pseudo code de la méthode multi-échelle de détection de segments, Multiscale Line Segment Detection (MLSD).

Raffinement multi-échelle

A l'échelle la plus grossière, nous détectons les segments de façon classique avec LSD. Puis à chaque échelle k nous raffinons séparément tous les segments s^{k-1} détectés à l'échelle précédente. Le procédé est illustré dans la Fig. 4.8, le pseudo-code associé est fourni dans l'Algorithme 2 et des exemples de résultats sont illustrés dans les figures 4.5 et 4.7.



LSD MLSD (étapes avant fusion) MLSD (étapes après fusion)

FIGURE 4.5 – Illustration des détections successives à chaque étape de la pyramide pour MLSD avec la version LSD en comparaison. De gauche à droite, LSD, les étapes de MLSD avant puis après fusion post-détection. De haut en bas, de l'échelle la plus grossière (1Mpixel) à la plus fine (15Mpixel). Notons qu'à la première ligne, les détections LSD et avant fusion sont identiques.

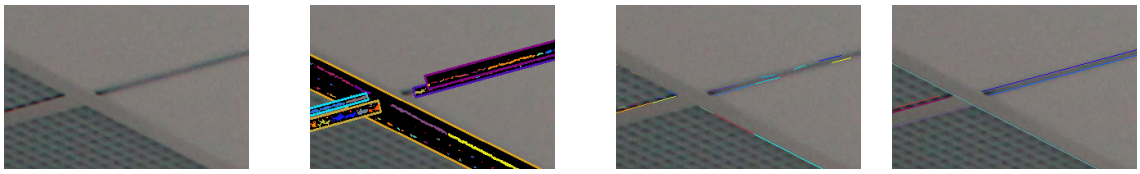
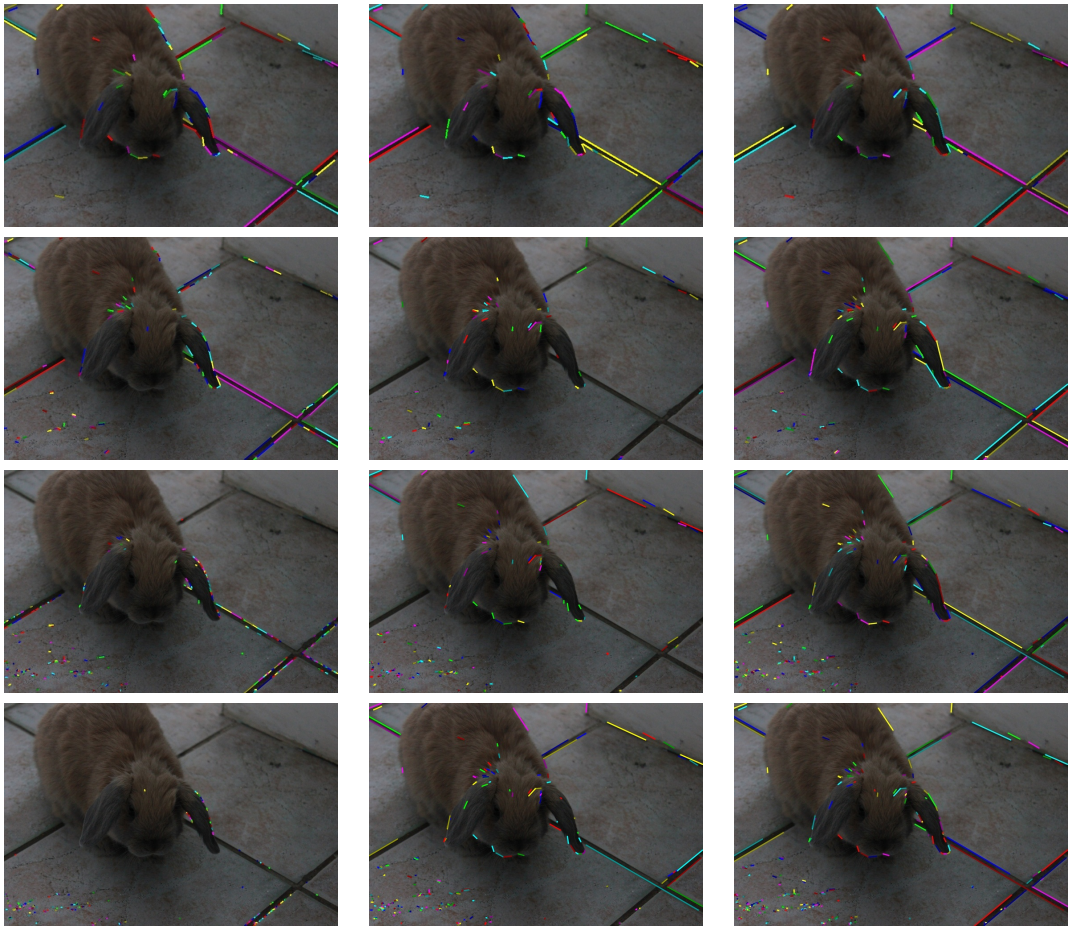


Image originale Raffinement multi-échelle LSD MLSD

FIGURE 4.6 – Illustration du raffinement multi-échelle. La deuxième image correspond au raffinement des segments mis à l'échelle. Chacune des composantes connexes trouvées est représentée d'une couleur différente. Après fusion de ces composantes, on obtient un résultat proche de celui de MLSD (quatrième image) où le phénomène de sur-segmentation présent dans LSD (troisième image) a disparu.

Pour un segment s^{k-1} d'angle θ^{k-1} détecté avec une tolérance angulaire de p^{k-1}, nous définissons tout d'abord le rectangle englobant $\mathcal{A}^k$ mis à l'échelle de l'image I^k. A l'intérieur de ce rectangle, nous considérons l'ensemble $\mathcal{P}^k$ des pixels dont la direction est proche de celle du



LSD MLSD (étapes avant fusion) MLSD (étapes après fusion)

FIGURE 4.7 – Illustration des détections successives à chaque étape de la pyramide pour MLSD avec la version LSD en comparaison. De gauche à droite, LSD, les étapes de MLSD avant puis après fusion post-détection. De haut en bas, de l'échelle la plus grossière (1Mpixel) à la plus fine (15Mpixel). Notons qu'à la première ligne, les détections LSD et avant fusion sont identiques.

Entrée : Ensemble des segments $\mathcal{S}^{k-1}$ détectés à l'échelle précédente;
Sortie : $\mathcal{S}^k$, version raffinée et mise à l'échelle de $\mathcal{S}^{k-1}$;
;
$\mathcal{S}^k \leftarrow \emptyset$
pour tout $s_i \in \mathcal{S}^{k-1}$ **faire**
1. Mettre les coordonnées de s_i à l'échelle et sélectionner les pixels de cette zone dont la direction du gradient est orthogonale à la direction de s_i (*i.e.* , calculer $\mathcal{P}_i^k$);
2. Agréger les pixels de $\mathcal{P}_i^k$ en composantes connexes à l'aide d'une méthode de croissance de région avec un voisinage de 8.;
3. Fusionner les composantes connexes selon le résultat du score de fusion (4.15);
4. Ajouter les segments dans $\mathcal{S}^k$ si leur NFA les rend suffisamment significatifs;

fin pour
;
retourner $\mathcal{S}^k$;

Algorithm 2: Pseudo-code du raffinement multi-échelle.

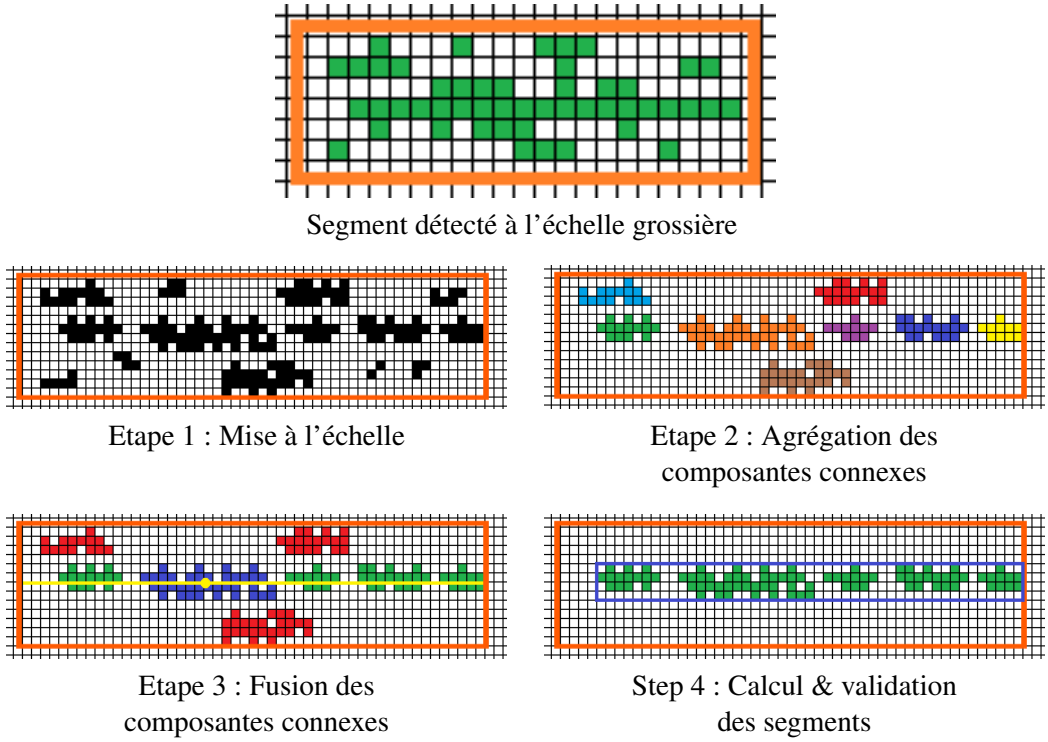


FIGURE 4.8 – Détails de l'étape de raffinement des segments d'une échelle grossière à une échelle plus fine.

segment s^{k-1} :

$$\mathcal{P}^k = \left\{ q \in \mathcal{A}^k \text{ s.t. } |\theta(q) - \theta(s^{k-1})|_{(\text{mod } \pi)} < \pi\, p^{k-1} \right\}. \tag{4.17}$$

Nous utilisons alors un procédé proche de celui de LSD puisque nous utilisons une méthode de croissance de région pour regrouper les pixels de $\mathcal{P}^k$ en composantes connexes dont l'ensemble est noté $\mathcal{C}^k$. Chacune de ces composantes connexes est un segment potentiel qui aurait été détecté par LSD et pour éviter le phénomène de sur-segmentation observé précédemment, nous cherchons à en fusionner certains. Cependant, plusieurs cas peuvent se produire ; le segment détecté à l'échelle $k-1$ peut correspondre à plusieurs segments proches et parallèles à l'échelle k ou à seulement un segment voire encore à plusieurs segments de directions proches mais différentes et fusionnés à une échelle plus grossière. Nous devons donc déterminer quelles composantes connexes devraient être fusionnées entre elles mais pour des raisons d'efficacité, nous ne pouvons considérer toute la combinatoire des fusions possibles. Pour traiter ce problème, nous considérons, pour une composant $c \in \mathcal{C}_i^k$, la droite $l(c)$ passant par le centre de c et d'angle $\theta(c)$. Nous cherchons alors l'ensemble des composantes connexes qui intersectent cette droite :

$$I(c) = \{c' \in \mathcal{C}_i^k \backslash \{c\} \text{ s.t. } l(c) \cap c' \neq \emptyset\} \tag{4.18}$$

Le procédé de fusion est glouton ; nous commençons avec la composante connexe c dont le NFA est le plus faible (*i.e.* la composante la plus significative), nous cherchons son ensemble $I(c)$ correspondant et fusionnons cet ensemble en un nouvel élément selon le résultat du score de fusion. Ce nouvel élément est ajouté à l'ensemble $\mathcal{C}_i^k$ puis le procédé est itéré.

Une fois que le raffinement est terminé, nous vérifions que les segments obtenus sont bien significatifs. En effet, il arrive que certaines composantes soient détectés mais ne génèrent pas de segments significatifs par la suite ou même qu'un segment puisse être détecté à une échelle $k-1$ mais qu'aucun pixel avec un gradient significatif ne soit présent à l'échelle k. Dans le

premier cas, nous rejetons les segments obtenus considérés comme du bruit de détection. Dans le second cas, nous considérons que le segment détecté à l'échelle $k-1$ ne pourra plus être raffiné par manque de contraste. Nous le conservons donc, ainsi que l'indice $k-1$ correspondant à sa meilleure échelle de détection, mais il ne sera plus raffiné par la suite. Ce facteur d'échelle peut ensuite être utilisé pour la mise en correspondance des segments.

Fusion post-détection

Une fois le raffinement appliqué, une détection à l'aide de LSD est appliquée aux autres parties de l'image. Cependant, il apparait que l'approche multi-échelle n'est pas suffisante pour supprimer toute sur-segmentation (voir les figures 4.5 et 4.7). En particulier, l'échelle la plus grossière divise parfois certains segments ou ne détecte pas le segment entièrement et les extrémités sont uniquement détectées aux échelles plus fines. Pour permettre une détection plus complète, nous avons donc ajouté une étape de fusion post-détection. Enfin, à partir de l'ensemble des lignes détectées par LSD et raffinées à l'échelle courante, nous cherchons à fusionner les segments proches et alignés. Le pseudo-code de ce procédé est fourni dans l'Algorithme 3.

Entrée : Ensemble de segments $\mathcal{S}^k$ détectés à l'échelle courante;
Sortie : Ensemble de segments fusionnés $\tilde{\mathcal{S}}^k$;

$\tilde{\mathcal{S}}^k \leftarrow \mathcal{S}^k$;
pour tout $s_i \in \tilde{\mathcal{S}}^k$ **faire**
1. Trouver les deux segments alignés avec s_i les plus proches de s_i;
2. Fusionner les segments selon le résultat du score de fusion (4.15);
3. Si nouveau segment, l'ajouter à $\tilde{\mathcal{S}}^k$ et supprimer la version sur-segmentée;

fin pour

retourner $\tilde{\mathcal{S}}^k$;

Algorithm 3: Pseudo-code de l'étape de fusion post-détection

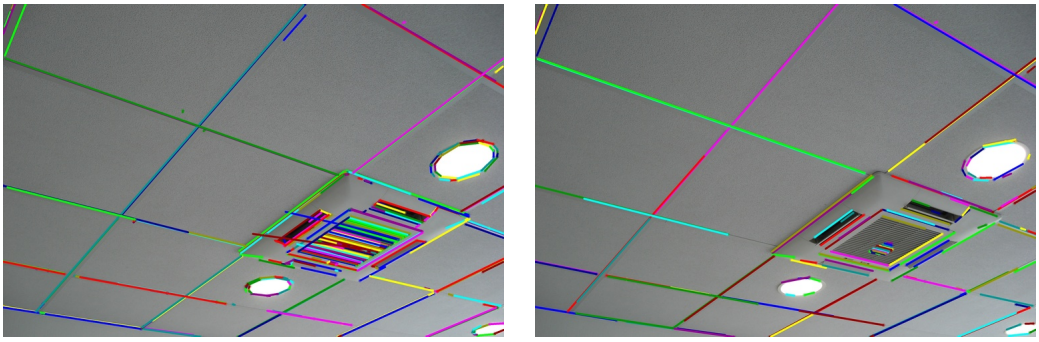
Cette étape de fusion est très proche de l'étape de raffinement puisqu'elle applique un procédé similaire à l'image entière au lieu d'une zone restreinte au segment englobant d'une détection à l'échelle précédente. L'étape de fusion commence avec un tri des segments par NFA croissant (*i.e.* du plus significatif au moins significatif). Un procédé glouton est alors appliqué sur chaque segment s ; on cherche les segments avec un angle proche de $\theta(s)$ qui intersectent la ligne prolongeant s et on teste la fusion avec les 2 segments les plus proches. Si le segment est fusionné, on l'ajoute à la pile de segments en supprimant les segments fusionnés. Le procédé est ensuite itéré jusqu'à avoir parcouru tous les segments.

Filtre de gradient dense (DGF)

Les expériences montrent que MLSD obtient de bonnes détections dans les images haute-résolution (voir par exemple la figure 4.10). Ces résultats sont à la fois dus à la méthode multi-échelle qui permet une meilleure détection dans les zones de faible-contraste et/ou bruitées ainsi qu'à l'étape de fusion post-détection qui permet d'obtenir des segments longs là où le phénomène de sur-segmentation est particulièrement présent. Par ailleurs, la complexité de l'algorithme est proche de celle de LSD ce qui se voit au niveau des temps de calcul des 2 méthodes (voir Tableau 4.1). Cependant la complexité de l'étape de fusion est particulièrement affectée par la nature de la scène. Ainsi, une scène générant beaucoup de sur-segmentation nécessitera une phase de fusion d'autant plus longue qu'il y a de morceaux à fusionner.

Image originale

LSD (2.38 s)



MLSD (5.53 s)
sans DGF

MLSD (3.20 s)

FIGURE 4.9 – Comparaison des temps de calcul et des résultats avec et sans filtrage des zones de gradient dense.

Cet aspect est particulièrement contraignant dans les scènes où ce phénomène est présent dans une partie conséquente de l'image. Par exemple, les motifs de grille fine ralentissent énormément l'algorithme en morcelant tous les segments de la grille. Outre l'aspect temps de calcul, ce phénomène engendre de nombreuses détections de segments très similaires. Dans une optique de reconstruction 3D, ces segments similaires et présents en forte proportion dans la scène ont tendance à détériorer le résultat de la calibration. En effet, leur proportion les rend beaucoup plus important dans des algorithmes type RANSAC et la calibration est alors entièrement déterminée par leur qualité. L'appariement est également détérioré car il se base sur des relations entre segments voisins qui se retrouvent faussées par la trop forte densité de segments similaires dans certaines régions.

Dans le but de conserver un temps de calcul comparable à celui de LSD et d'éviter la présence d'artefact pour la mise en correspondance et la reconstruction 3D, nous avons mis en place un filtre réduisant les détections dans des zones de ce type. L'idée est de préférer obtenir une détection de segments moins proche de la réalité mais plus utile dans un cadre de reconstruction 3D. Le pseudo-code de ce procédé est fourni dans l'Algorithme 4.

Cette méthode utilise la carte de gradient calculée à partir du seuillage de gradient pour détecter les zones où la densité de bord est trop importante. Pour cela nous calculons la densité de bord dans des fenêtres de taille 5×5. Si cette densité dépasse un seuil de $\tau_{\text{DENSE}} = 75\%$, alors la fenêtre est considérée comme appartenant à une zone dense de gradient. Cette zone est ensuite étendue à une fenêtre 21×21 où la détection de segment est désactivée. L'utilisation de 2 fenêtres permet de ne sélectionner que les zones extrêmement denses en bords et elle est étendue dans le but de supprimer les frontières de ces zones. Notons également que les seuils ne sont pas fixés par rapport à la taille de l'image. En effet, le masque n'étant calculé qu'à l'échelle

Entrée : Image à l'échelle la plus grossière I^0;
Sortie : Image masquée $\tilde{I}^0$ sans les pixels filtrés;
Paramètres : taille de voisinage v_1 et v_2, seuil de densité de gradient τ_{DENSE};
;
Calculer les images intégrales des gradients de I^0;
$\tilde{I}^0 \leftarrow \emptyset$;
pour tout pixel $p \in I^0$ **faire**
  Calculer la densité τ de pixels avec un gradient significatif (*i.e.* au dessus du seuil ρ) dans le voisinage v_1 de p;
  si $\tau > \tau_{DENSE}$ **alors**
    Ajouter p à $\tilde{I}^0$;
  fin si
fin pour
Étendre les zones invalidées de $\tilde{I}^0$ par dilatation dans un voisinage v_2;
retourner $\tilde{I}^0$;

Algorithm 4: Pseudo-code du filtre à gradient dense (DGF).

la plus grossière, la taille de l'image est toujours de l'ordre de 1 Mpixel.

L'utilisation d'images intégrales permet un calcul très rapide de la densité de gradient. De plus ce filtrage n'est calculé qu'à l'échelle la plus grossière, il est uniquement agrandi à chaque changement d'échelle.

Cette méthode a quelques défauts :

- elle n'est pas optimale : elle conserve de nombreuses zones gênantes pour la détection notamment les bords
- elle fait apparaître 2 seuils dans une méthode qui se voulait justement sans paramètre
- elle n'est pas assez efficace sur certaines images où la présence de motifs de grille est très importante

Malgré ces défauts, elle fonctionne correctement et très rapidement sur de nombreuses images et permet donc parfois une meilleure utilisation de MLSD. Notons également que les cas d'échec du filtre donnent des résultats particulièrement sur-segmentés (et donc mauvais) pour LSD.

4.5 Expériences sur la détection de segments

Nous avons tout d'abord cherché à mesurer l'impact des différentes étapes introduites dans MLSD en comparant les résultats de LSD et MLSD en utilisant uniquement certaines étapes. Nous avons ensuite étudié les comportements de ces deux algorithmes face à des changements de résolution ou au manque de contraste pour montrer empiriquement l'intérêt de MLSD dans le cadre de la reconstruction en intérieur. Enfin, nous avons comparé les temps de calcul de chacune des méthodes pour évaluer le ralentissement.

4.5.1 Utilité des différentes étapes

Comme on peut le voir dans la Figure 4.10, chacune des étapes à son importance. La partie multi-échelle permet de détecter des segments là où LSD n'en détecte pas ou peu tandis que la partie fusion post-détection permet de rattraper certaines détections encore trop segmentées malgré l'utilisation du procédé multi-échelle. Selon le type d'image chaque étape peut avoir une importance plus marquée que l'autre. Ainsi dans les images très haute résolution, à faible

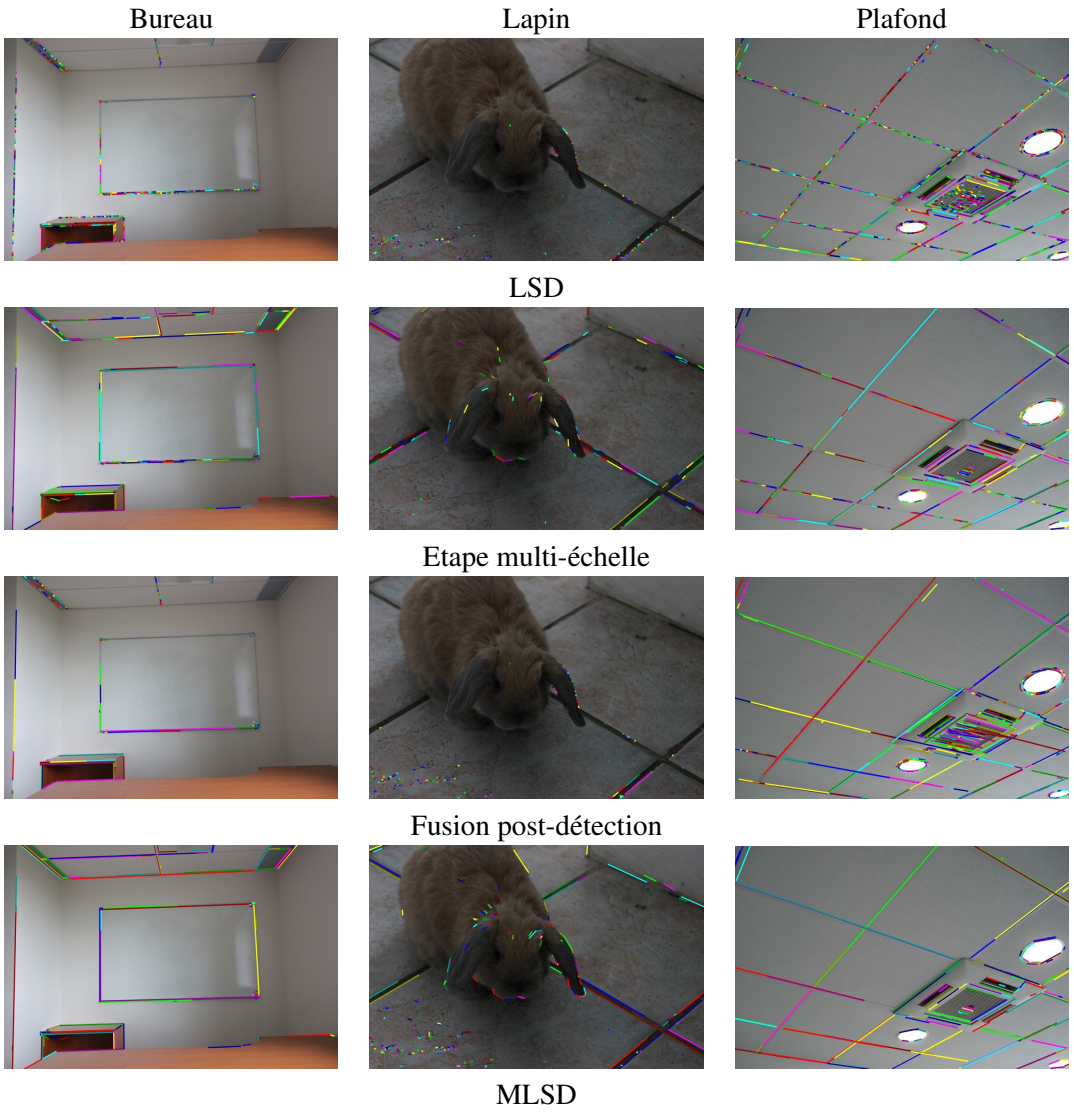


FIGURE 4.10 – De haut en bas ; LSD original [19], l'étape multi-échelle uniquement, l'étape de fusion post-détection uniquement et MLSD [68]. Chaque étape possède son utilité propre. L'aspect multi-échelle est très important pour pallier la présence de bruit et l'absence de contraste mais l'étape de fusion permet d'obtenir un résultat final de meilleure qualité.

contraste et en présence de bruit (*i.e.* les scènes *Bureau* et *Lapin*), c'est la partie multi-échelle qui sera plus importante car les détections de LSD sont trop peu nombreuses. A l'inverse dans les images de moins haute résolution et en présence de plus de contraste et/ou moins de bruit (*i.e.* la scène *Plafond*), c'est la partie fusion qui permettra de meilleurs résultats.

4.5.2 Manque de contraste

Le manque de contraste est fréquent dans les scènes d'intérieur de bâtiments. Il a pour conséquence une carte de gradient moins nette. Combiné au bruit, fréquent dans les zones sombres, la détection de segment devient particulièrement difficile pour LSD. L'approche multi-échelle permet d'atténuer le bruit (les réductions d'échelles correspondent à des moyennages locaux et suppriment donc les hautes-fréquences) et obtient donc de meilleurs résultats. La figure 4.11 permet d'illustrer ce propos avec deux zooms successifs sur une zone peu contrastée.

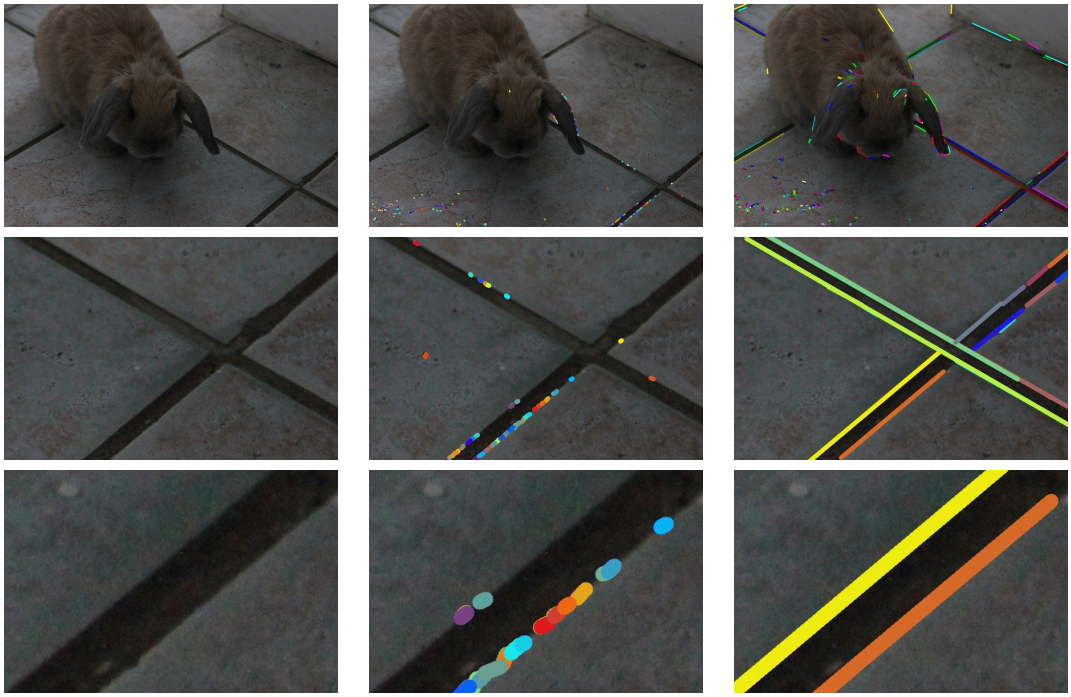
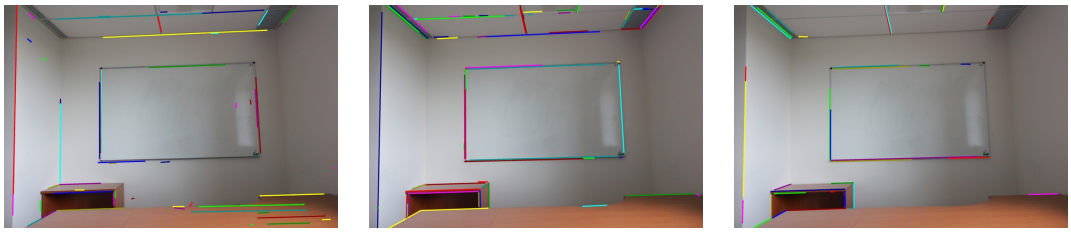


Image originale LSD MLSD

FIGURE 4.11 – Comparaison entre les détections de LSD et MLSD sur une image avec un très faible contraste. Le zoom sur le carrelage permet de mieux visualiser le manque de contraste et la présence de bruit dans cette image.

4.5.3 Choix du seuil de gradient

Que ce soit pour LSD ou MLSD, la méthode débute avec un seuillage de gradient. Ce filtrage fait intervenir un seuil ρ dont la valeur influence le résultat final. En effet, si le prendre très bas permet des détections correctes dans des zones à très faible contraste, cela génère également des détections parasites dans les zones bruitées ou dans les zones avec une texture spécifique (*e.g.* texture boisée). De plus, le trop grand nombre de bord gène la partie croissance de région et empêche la détection de nombreux segments.

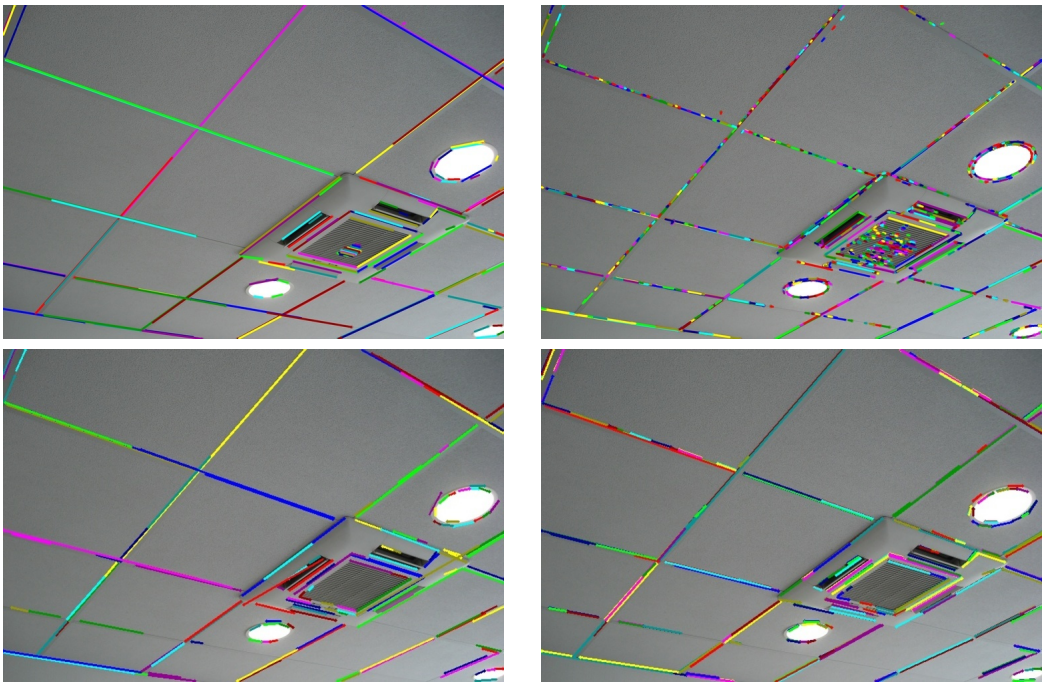


$\rho = 1$ $\rho \approx 5.2$ $\rho = 10$

FIGURE 4.12 – La détection diffère avec le choix du seuil de gradient ρ. Choisir ρ trop faible permet des détection dans les zones à faible contraste (coin du mur à gauche) mais également des détections de bruit (bureau). Choisir ρ trop élevé empêche une détection complète de la scène. Le choix fait dans LSD [19] permet un bon compromis.

A l'inverse, le prendre trop élevé empêche de nombreuses détections spécifiquement là où le contraste n'est pas assez fort. Dans LSD [19], Grompone von Gioi *et al.* justifient ce choix en

utilisant une formule faisant intervenir la discrétisation des images sur l'intervalle $[0,255]$ ainsi que la précision angulaire utilisée dans l'étape de croissance de région. Nous avons conservé le même seuil mais il apparaît qu'un seuil adapté localement pourrait permettre de meilleurs détections dans les zones de faible contraste (*e.g.* coin gauche du mur détecté uniquement lorsque $\rho = 1$).



MLSD LSD

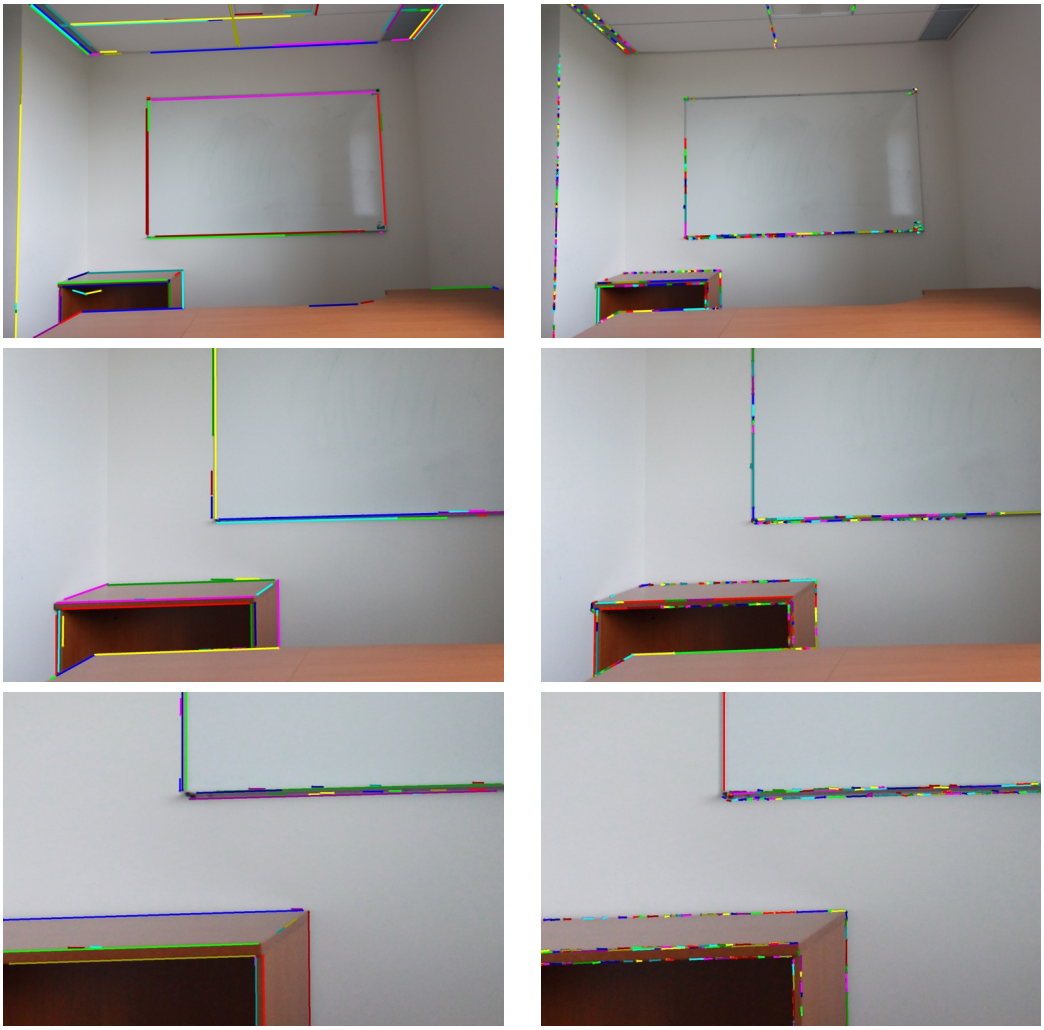
FIGURE 4.13 – Comparaison de LSD et MLSD en diminuant d'une rapport 4 la résolution d'une image très résolue. Bien que les deux méthodes obtiennent des détections différentes, LSD obtient de bien meilleurs résultats dans la nouvelle scène tandis que MLSD reste à peu près stable. C'est d'ailleurs cette observation qui nous a conduit à utiliser une méthode multi-échelle.

4.5.4 Invariance au changement d'échelle

Comme nous l'avons vu précédemment, l'algorithme de LSD dépend de la taille de l'image à travers deux facteurs :

- la formule du NFA fait intervenir directement la taille de l'image. Ainsi, un découpage de l'image facilitera les détections en diminuant le seuil de validation du NFA. Cette diminution s'explique à l'aide de la théorie *a contrario*. En effet, il est plus probable de trouver un segment de taille 5 pixel dans une image suivant le modèle $\mathcal{H}_0$ de taille 5000×5000 que dans une image 5×5.
- la méthode de croissance de région. En effet, dans les images de haute résolution, les segments étant beaucoup plus longs (en terme de pixels), il est plus fréquent que les composantes connexes soient séparées par des pixels bruités. De plus un pixel correspond couvre une aire de la scène plus faible et il est donc plus sensible à des phénomènes comme le flou de bougé. Le cumul de ces différents phénomènes diminue fortement l'efficacité de la croissance de région.

Ainsi, LSD voit ses résultats changer fortement dans le cas d'un changement de résolution ou dans le cas d'un découpage d'une portion de l'image. Comme on peut le voir dans les figures



MLSD LSD

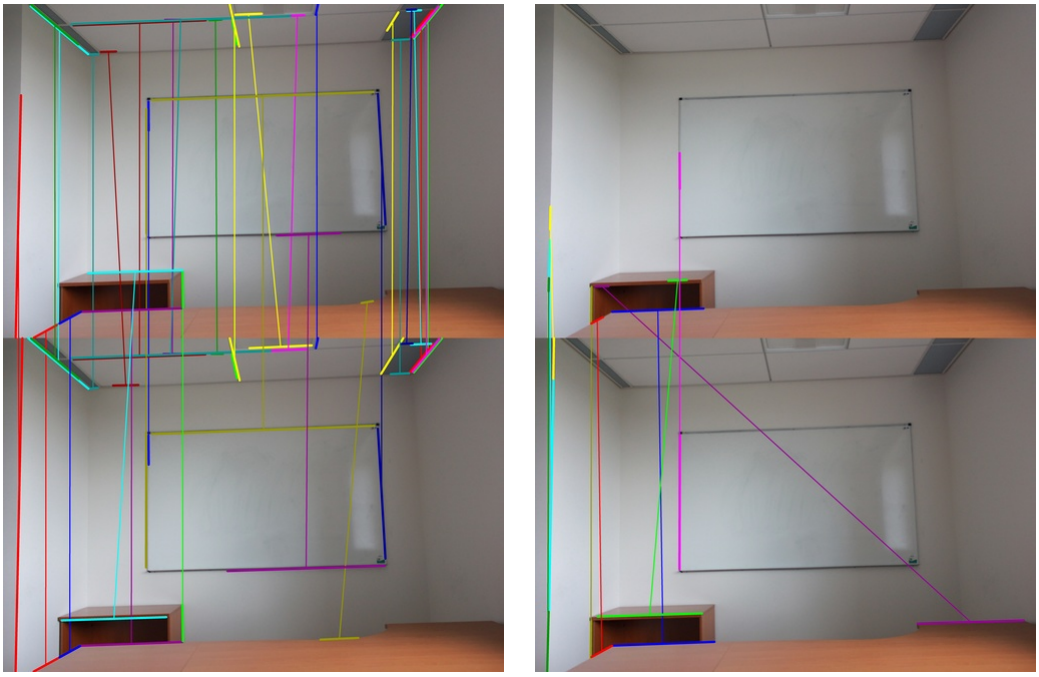
FIGURE 4.14 – Comparaison de LSD et MLSD sur des images de haute-résolution (en haut, 18 Mpixels) et leurs versions découpées (pas de changement de résolution) de respectivement 5.6 Mpixels (milieu) et 1.5 Mpixels (bas).

4.14 et 4.13 , MLSD est plus stable à ce type de transformations. Ce critère le rend également plus intéressant dans une optique de reconstruction 3D car l'appariement est plus robuste face aux changements d'échelle (voir Figure 4.15).

4.5.5 Temps de calcul

Les méthodes MLSD et LSD ont des complexités algorithmiques similaires et on le retrouve avec des temps de calculs assez proches. Cependant, l'étape de fusion post-détection dépend de la présence de sur-segmentation dans l'image. Ainsi, dans la plupart des images, son temps de calcul n'aura pas un impact énorme. Cependant, en présence de certains motifs comme des grilles fines, le phénomène de sur-segmentation est extrêmement présent ce qui ralentit fortement l'étape de fusion. Dans ce cas là, l'utilisation du DGF permet une accéleration importante du calcul.

Nous avons comparé les temps de calcul de différentes méthodes sur un ensemble de 4 images haute-résolution (voir Figure 4.16) pour avoir un aperçu de l'efficacité de chacunes (voir



MLSD LSD

FIGURE 4.15 – Comparaison de la robustesse de l'appariement en fonction des détecteurs. L'image du dessus à une résolution de 15 Mpixels tandis que celle du dessous a été réduite à moins de 1 Mpixel. Les détections de LSD étant très dépendantes de la résolution de l'image, l'appariement est de mauvaise qualité tandis que pour MLSD, l'appariement n'est pas affecté.

Bureau 5184 × 3456 | Lapin 5184 × 3456 | Plafond 2816 × 1872 | Colonnes 2816 × 1872

FIGURE 4.16 – Images utilisées pour les comparaisons de temps de calcul. Les comparaisons se font sur des images de haute-résolution car c'est avec elles qu'apparaissent les réelles différences.

Tableau 4.1). ED-Lines est nettement plus performant que les autres méthodes. Cependant sa complexité théorique est similaire et son code n'est pas disponible contrairement à ceux de LSD et MLSD qui reprennent des parties communes. De plus, LSD et MLSD n'étant ni optimisés ni parallélisés, il semble logique que ED-Lines soit bien plus performant.

En comparant LSD à MLSD, on s'aperçoit que le temps de calcul est augmenté de 40% pour les images de 15 Mpixels (*i.e. Bureau* et *Lapin*) et de 30% pour les images de 5 Mpixels (*i.e. Plafond* et *Colonnes*). Il apparaît donc que l'augmentation du temps de calcul est lié à la taille de l'image ce qui semble cohérent avec le fait que le nombre d'échelles utilisées augmente également. Cependant, cette augmentation reste raisonnable et n'est pas gênante dans un cadre de reconstruction 3D.

De plus, la détection de segments n'est pas uniquement liée à la taille de l'image mais également à la nature de la scène qui présente plus ou moins de segments.

Méthode / *Dataset*	ED-Lines	LSD	MLSD	MLSD (sans DGF)
Bureau	1.4 (0.08)	3.8 (0.21)	5.2 (0.29)	5.4 (0.30)
Lapin	1.2 (0.07)	4.0 (0.22)	5.7 (0.32)	6.7 (0.37)
Plafond	0.4 (0.08)	2.4 (0.46)	3.2 (0.61)	5.5 (1.04)
Colonnes	0.9 (0.17)	5.7 (1.08)	7.2 (1.37)	×

TABLE 4.1 – Temps de calcul (en secondes) des différentes méthodes sur les images présentées à la Fig. 4.16. Nous avons indiqué entre parenthèse le temps de calcul par Mpixel. Notons que le temps de calcul de MLSD sans filtrage dépasse les 10 minutes sur l'image *Colonnes*.

Enfin, nous pouvons remarquer que l'utilisation du filtre à gradient dense (DGF) permet d'éviter les cas limites où l'étape de fusion post-détection prend trop de temps (en particulier sur *Colonnes* où le temps de calcul dépasse les 10 min).

4.6 Application à la reconstruction 3D

Nous avons comparé les 3 détecteurs de segment (ED-Lines, LSD et MLSD) dans une optique de calibration. En effet, même si la comparaison peut sembler parfois évidente, il est généralement difficile de juger si tel détecteur est meilleur qu'un autre sans introduire un biais subjectif. Comparer de telles méthodes dans une optique de calibration permet d'introduire une mesure plus objective car quantitative. Cependant, les résultats sont alors dépendants des méthodes d'appariement et de calibration utilisées ce qui empêche d'obtenir un classement parfait des détecteurs. Notons que nous avons utilisé les implémentations originales des différents auteurs pour chacune des méthodes.

4.6.1 Calibration bifocale

Pour la calibration, nous avons utilisé des méthode état-de-l'art pour l'appariement des segments [84] et pour la calibration bifocale à partir de segments [69]. Nous avons alors comparé les erreurs de calibration sur deux jeux de données. Notons que seule l'erreur de rotation est fournie puisque les segments ne permettent pas de calculer la translation relative dans le cas bifocal.

Les données utilisées pour la comparaison proviennent de deux jeux de données :

- *CastleP19* [73] : une série de 19 photos de la cour intérieure d'un château. Les images de la scène ont une résolution de 3072×2048 soit un peu plus de 5 Mpixels.
- *Office* [69] : une série de 8 photos d'un bureau. Les images de la scène ont une résolution de 5184×3456 soit un peu plus de 15 Mpixels.

Certaines images de ces jeux de données sont illustrées dans la Figure 4.17.

Les résultats de calibration sont résumés sur les figures 4.18 et 4.19 et leurs moyennes dans le tableau 4.2. Comme on peut le voir, les résultats sont assez similaires pour chacun des détecteurs en utilisant le *dataset* Castle P19. En effet, la scène présente de nombreux segments bien contrastés et relativement courts. De plus la résolution de la scène n'est que de 5 Mpixel. Les résultats obtenus sur le *dataset Office* sont un peu plus intéressants. Dans le cas du détecteur ED-Lines, les mauvaises détections peuvent entrainer un appariement particulièrement mauvais et donc une calibration incorrecte (paire 5-6). Le graphique des résultats montrent des résultats similaires pour chacun des détecteurs sur une la moitié des paires. Sur l'autre moitié par contre, LSD et ED-Lines montrent quelques écarts de précisions.

Castle P19 *Office*

FIGURE 4.17 – Nous utilisons deux jeu de données pour la comparaison de calibration en fonction des détecteurs de segments. La série *CastleP19* a une résolution de 5 Mpixel tandis que celle d'*Office* est de 15 Mpixel. La série *Office* a également moins de contraste et plus de bruit.

Méthode / Dataset	LSD [19]	ED-Lines [2]	MLSD [68]
Castle P19 [73]	**0.79**	1.06	0.91
Office [69]	1.77	8.00	**1.27**

TABLE 4.2 – Erreur angulaire moyenne de rotation (en °) pour les différents *datasets*. La différence se fait plus sur les images de haute-résolution qu'en moyenne résolution où les résultats des différentes méthodes sont plus comparables.

Finalement MLSD ne permet pas d'obtenir une calibration vraiment plus précise que les autres méthodes mais obtient des résultats plus réguliers en particulier sur les images de haute-résolution. En détectant plus correctement les segments dans les images d'intérieur, l'appariement est alors plus complet et correct permettant une calibration plus stable. MLSD permet donc un gain de robustesse dans une optique de calibration sans diminuer significativement la précision. Ses atouts font donc de MLSD une méthode particulièrement adaptée à la calibration en intérieur.

4.6.2 Reconstruction 3D

Pour tester la fiabilité de la reconstruction indépendamment du processus de calibration, nous avons utilisé la vérité terrain de l'ensemble de données de Strecha *et al.* [73]. Nous avons reconstruit les lignes en 3D à partir des différents appariements et des positions réelles des caméras. Nous avons ensuite comparé la reconstruction 3D obtenue à celle fournie par le scanner laser. Pour cela, nous avons calculé le nombre de segments 3D en dessous d'une certaine distance

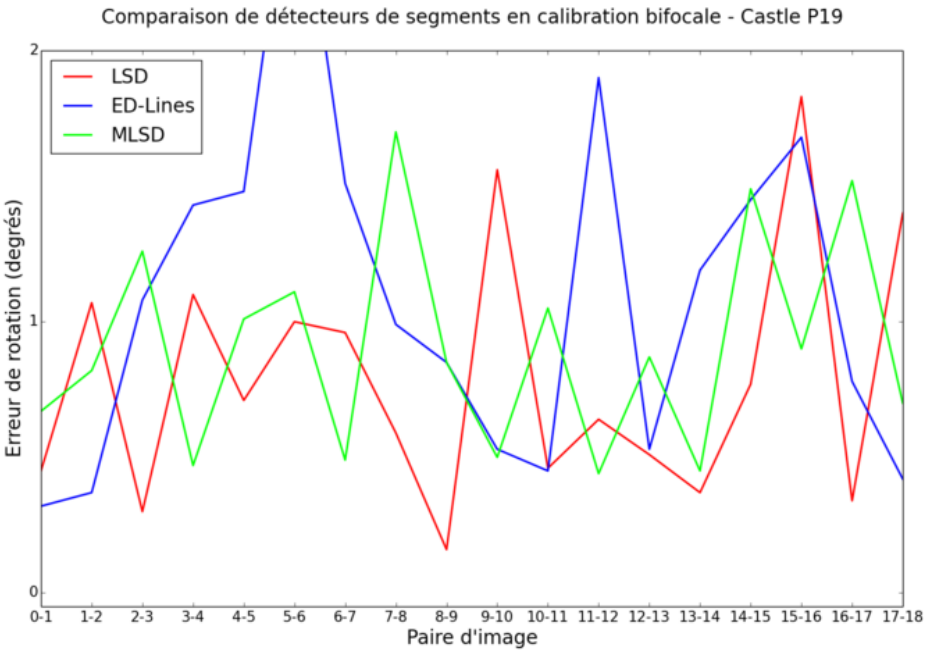


FIGURE 4.18 – Comparaison des qualités des détecteurs du point de vue de la calibration bifocale sur la scène *Castle*.

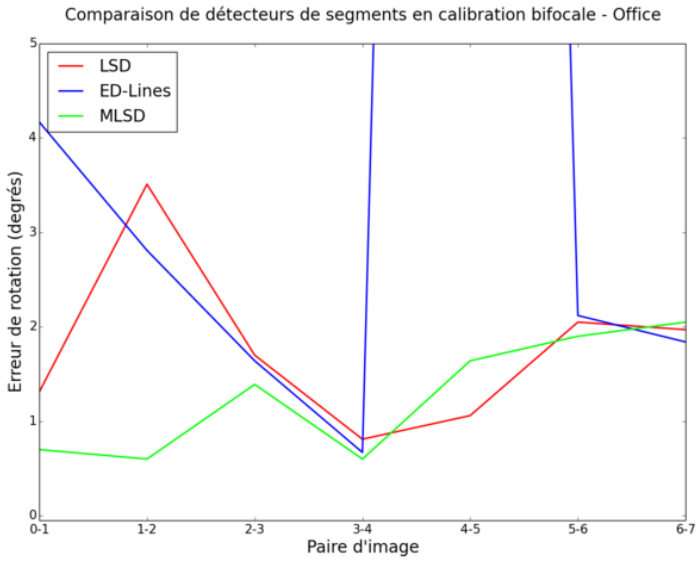
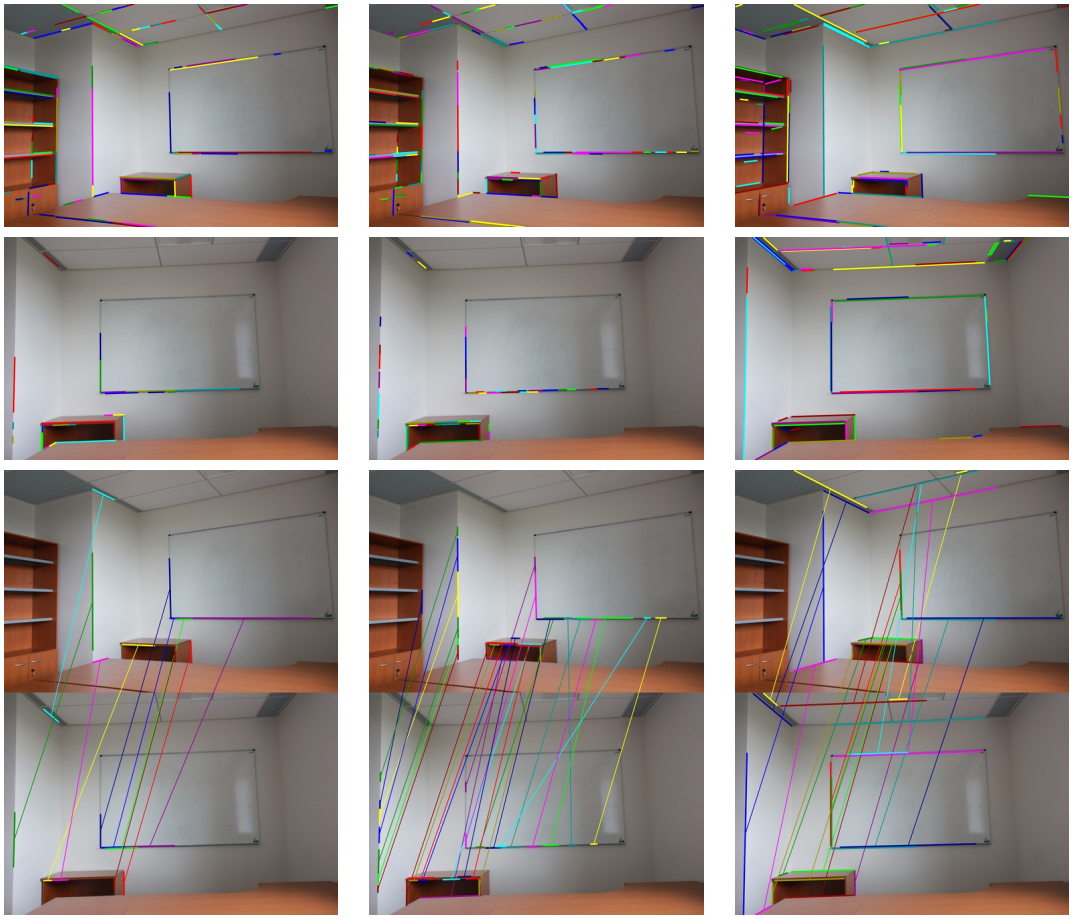


FIGURE 4.19 – Comparaison des qualités des détecteurs du point de vue de la calibration bifocale sur la scène *Office*.

du nuage de points. Nous avons ainsi accès au nombre d'appariements corrects

Les résultats sont affichés dans le tableau 4.3 et les différentes reconstructions obtenues sont affichées dans les figures 4.21 et 4.22. Ces résultats combinent la précision des détections et la robustesse de l'appariement. En effet, une détection précise permettra une reconstruction du segment plus proche de la réalité à condition que l'appariement soit correct (dans le cas où



LSD ED-Lines MLSD

FIGURE 4.20 – Comparaison des détections et appariements sur une scène d'*Office*.

l'appariement est faux, le segment reconstruit est généralement loin du nuage de points). On voit alors que les détections de MLSD, avec des extrémités plus stables d'une image à l'autre, permettent à la fois une détection plus précises et des appariements plus corrects ce qui rend la reconstruction de meilleure qualité.

Dataset \ Distance (m)		∞	1	0.1	0.01	0.001
Herz-Jesu-P8 [73]	LSD	752	246 (33 %)	214 (28 %)	174 (23 %)	118 (16 %)
	ED-Lines	**856**	309 (36 %)	244 (29 %)	172 (20 %)	97 (11 %)
	MLSD	505	**357 (71 %)**	**287 (57 %)**	**230 (46 %)**	**137 (27 %)**
Fountain-P11 [73]	LSD	1205	268 (22 %)	230 (19 %)	180 (15 %)	124 (10 %)
	ED-Lines	**1225**	319 (26 %)	267 (22 %)	177 (14 %)	97 (8 %)
	MLSD	879	**372 (42 %)**	**297 (34 %)**	**214 (24 %)**	**140 (16 %)**

TABLE 4.3 – Affichage du nombre d'appariements corrects en fonction de la distance du segment reconstruit au nuage de points. Les proportions d'appariements corrects sont indiqués entre parenthèses et les valeurs en gras représentent le nombre d'appariements le plus élevé pour une catégorie donnée. On remarque alors que MLSD fourni des segments beaucoup plus utiles dans une optique de reconstruction 3D.

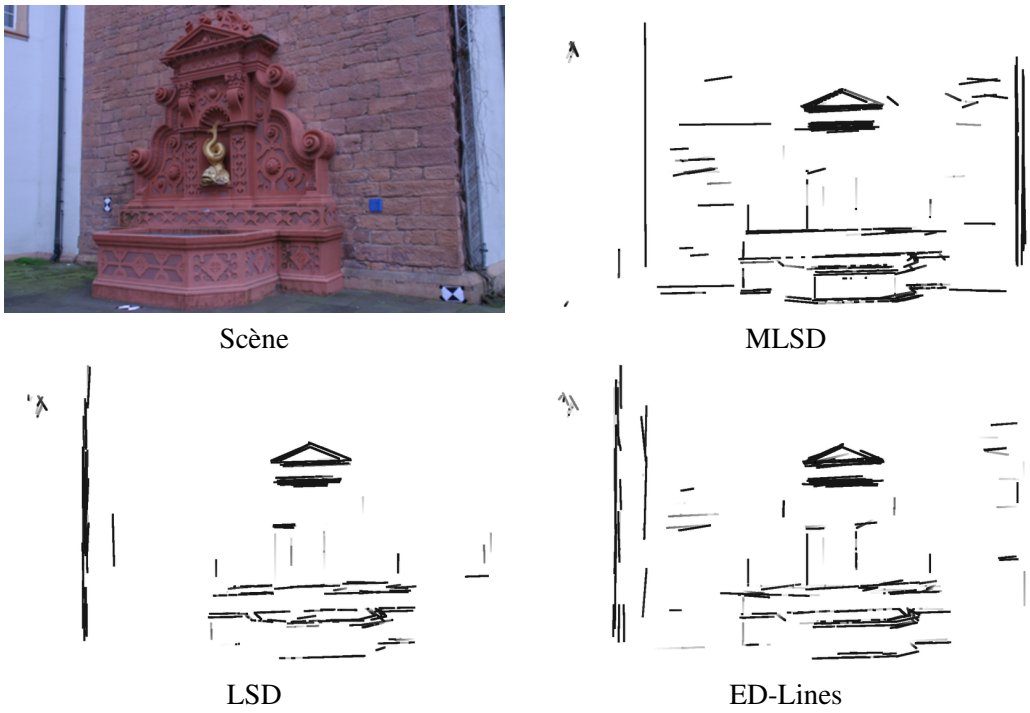


FIGURE 4.21 – Reconstruction de la scène *Fountain-P11* à partir de la vérité terrain et en fonction des différents détecteurs.

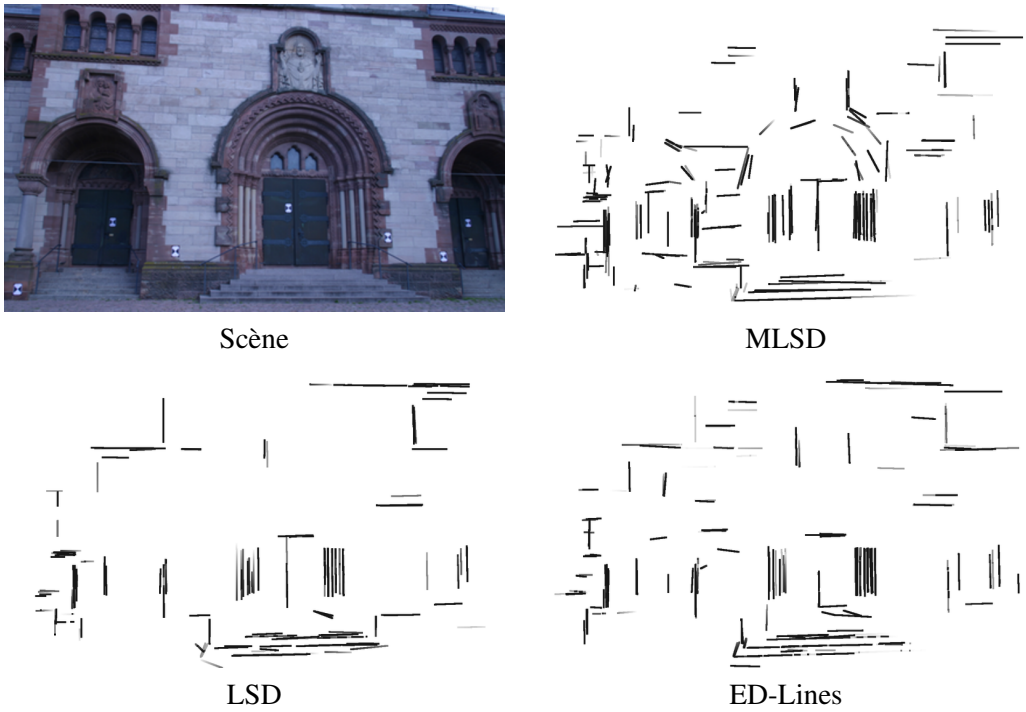


FIGURE 4.22 – Reconstruction de la scène *HerzJesu-P8* à partir de la vérité terrain et en fonction des différents détecteurs.

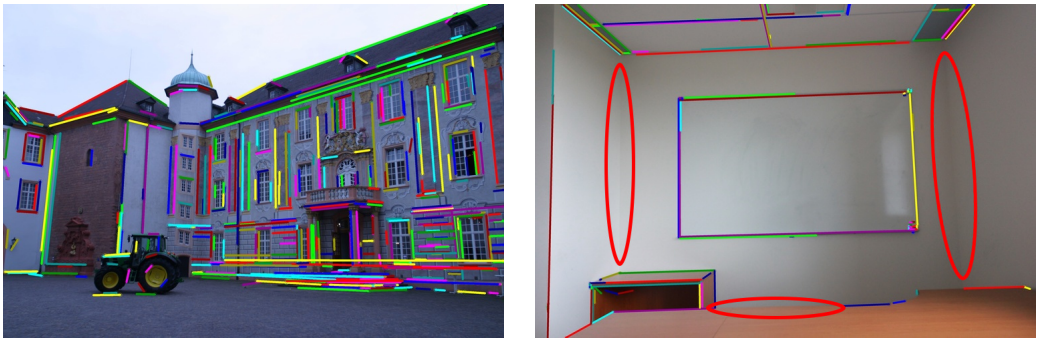
4.7 Limites de la méthode et perspectives

L'une des limites principales de la méthode est mise en avant par le filtre à gradient dense. La trop forte présence du phénomène de sur-segmentation dans les images entraîne une détection trop lente et peu utile dans une optique de reconstruction 3D. Le filtre permet de pallier partiellement ce défaut mais il n'est pas assez efficace et utilise des seuils dans une méthode se voulant sans paramètres. Une piste d'amélioration serait donc de perfectionner le filtre ou de chercher les candidats potentiels à la fusion de façon moins naïve que dans la méthode actuelle.

Image originale | MLSD sans DGF (1380s) | MLSD avec DGF (202s)

FIGURE 4.23 – Exemple typique d'une scène où MLSD donne de très mauvais résultats. La scène fait 15 Mpixels et est constituée d'une multitude de petits segments. La version filtrée par DGF génère une détection lente et avec pratiquement aucune information tandis que la version non-filtrée donne un résultat exploitable malgré quelques phénomènes de sous-segmentation. Cependant, c'est au prix d'un temps de calcul extrêmement long. Ce type de scène n'est pas trop fréquent en intérieur mais met en avant les principaux défauts de la méthode.

De plus, le score de fusion ne fonctionne pas toujours parfaitement puisque certains segments sont fusionnés à tort (voir Fig. 4.24). Cependant ce phénomène de *sous-segmentation* concerne principalement des segments de petite taille proches les uns des autres (*e.g.* une grille, des fenêtres de bâtiments vu de loin) et il est donc moins gênant dans les scènes d'intérieur.



sous-segmentation | segments non détectés

FIGURE 4.24 – A gauche, un exemple typique de *sous-segmentation*. Certains segments courts et alignés sont fusionnés à tort. A droite, certaines zones mises en avant par des ovales rouge montrent une absence de détection là où l'œil reconnaît des segments.

Enfin, même si le défaut a été en grande partie corrigée par notre méthode, la détection des segments dans les zones de trop faible contraste ne fonctionne pas dans tous les cas (voir Fig. 4.24). Il reste encore de nombreux segments que l'œil voit mais que les méthodes automatiques ne détectent pas. Ce manque de détection est en partie dû à l'utilisation d'un seuil pour la sélection des gradients utilisés. L'utilisation d'un seuil s'adaptant aux zones de l'image permettrait une détection encore plus robuste.

4.8 Contributions de ce chapitre

Nous avons présenté, dans ce chapitre, quelques unes des méthodes état-de-l'art de détection de segment et les défauts qu'elles présentaient dans les scènes d'intérieur et de haute-résolution.

Remarquant que les défauts principaux disparaissaient en réduisant simplement la résolution de l'image, nous avons proposé une méthode multi-échelle permettant d'obtenir des détections de bonne qualité sur les scènes d'intérieur. Cette amélioration s'est faite au prix d'un temps de calcul légèrement plus long. Cependant, cet aspect est négligeable dans une optique de reconstruction 3D.

Nous avons comparé les différentes méthodes de façon qualitative à l'aide d'image représentatives de problèmes courants dans les scènes d'intérieur notamment. Puis, nous avons comparé ces méthodes sur des données de calibration pour obtenir une comparaison plus objective. Si la méthode proposée ne permet pas toujours de gagner en précision des résultats, elle permet un gain de robustesse en particulier dans les scènes d'intérieur.

Ce travail a été présenté à la conférence *ICPR* [68] et son implémentation détaillée au workshop *RRPR* (pour *Reproducible Research in Pattern Recognition*). Une implémentation libre est disponible sur GitHub (`https://github.com/ySalaun/MLSD`).

Chapitre 5

Calibration bifocale mixte point-ligne

Il existe déjà de nombreuses méthodes de calibrations et celles-ci sont suffisamment matures pour permettre la reconstruction de nombreux bâtiments ou objets. Cependant ces méthodes sont, pour la plupart, basées sur l'utilisation de points remarquables entre les images. De plus, les méthodes de calibration à partir de points requièrent généralement la présence de points sur un espace 3D, les surfaces planes conduisant à des cas dégénérés.

Ce sont ces deux principaux aspects qui rendent la calibration de scènes d'intérieur particulièrement difficile. D'ailleurs, les méthodes actuelles échouent généralement à calibrer ce type de scène.

Nous présentons, dans ce chapitre, une méthode de calibration à partir de droites et de points permettant de calibrer les scènes d'intérieur autrefois impossible à calibrer. Nous montrons que l'utilisation conjointe des points et des lignes permet d'obtenir des résultats qui allient la précision des points à la robustesse des lignes et ainsi obtenir d'excellents résultats en intérieur et en extérieur. Nous comparons à ce propos notre méthode avec celles de l'état de l'art sur différentes scènes représentatives de nombreux cas pratiques.

Contents

5.1 Méthodes existantes

Comme vu précédemment (voir section 3.5.1), l'estimation de pose entre deux caméras à partir de lignes uniquement n'est pas possible [80, 90]. C'est pourquoi les méthodes à base de lignes utilisent généralement le tenseur trifocal [24] pour estimer la pose. Un procédé de calibration complet a récemment été développé [85] utilisant les contraintes trifocales dans le cadre du SLAM. Cependant, ce type de méthode est difficile à utiliser dans le cadre des scènes d'intérieur car il nécessite un trop grand nombre de lignes vues dans des triplets d'image (voir section 3.5.2). Une autre façon de faire, est d'utiliser un dispositif conçu spécialement pour la reconstruction à base de lignes. Ainsi, certains auteurs [10, 64] ont développé un dispositif permettant d'obtenir deux images stéréo et calibrent ainsi à l'aide de lignes et de contraintes trifocales. Une telle méthode nécessite donc toujours un grand nombre d'appariements corrects de lignes et les exigences de la méthode empêche son utilisation sur un jeu d'images quelconque.

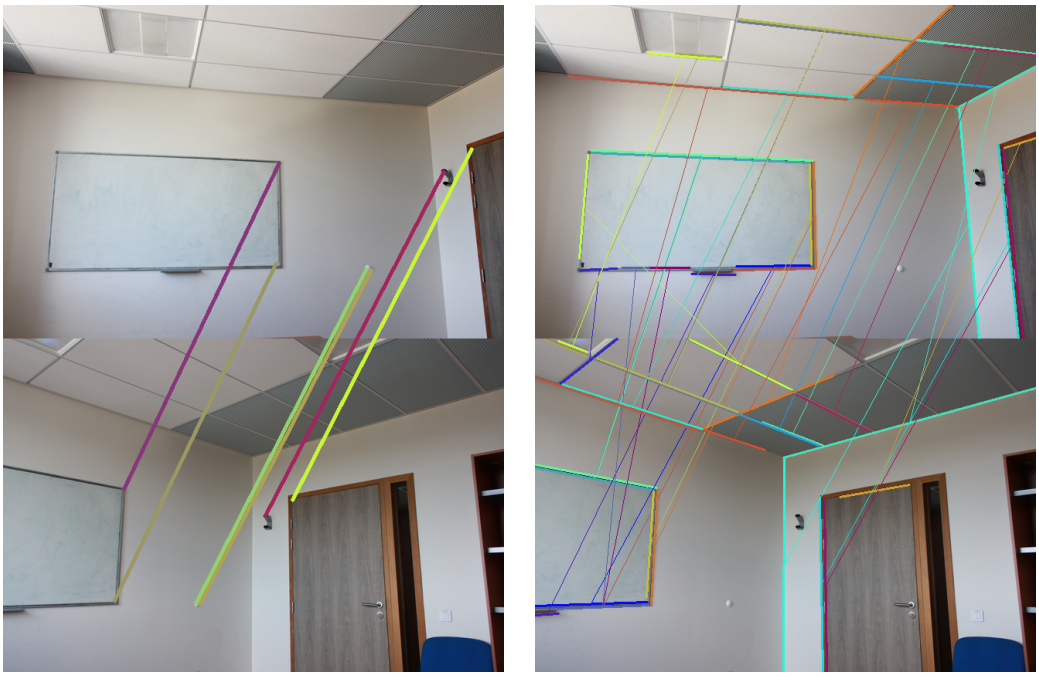


FIGURE 5.1 – Exemple typique de photographies d'une scène d'intérieur. L'appariement des points est assez pauvre et principalement localisé sur un plan tandis que l'appariement de lignes est mieux et plus densément réparti dans l'image.

Malgré l'impossibilité de calibrer à partir de lignes dans un cadre bifocal, l'utilisation d'hypothèses supplémentaires permet d'obtenir une estimation de la pose. Ainsi, certaines méthodes [89, 57] utilisent des hypothèses de similarité des segments d'une image à l'autre. Zhang *et al.* [89] cherchent à maximiser le chevauchement des segments mis en correspondances tandis que Montiel *et al.* [57] mettent en correspondance les milieux des segments. Mais ces méthodes sont sensibles au manque de robustesse dans la détection des extrémités des segments, et à la sur-segmentation. D'autres auteurs [8, 13] utilisent les intersections des segments lorsque ceux-ci correspondent à des droites coplanaires. Cependant, dans tous ces cas, ces méthodes se ramènent finalement à des méthodes de calibration à base de points en utilisant des *features* généralement peu précis et dont la proportion d'*outliers* est plus élevée que la normale (*e.g.* segments qui se chevauchent très peu d'une image à l'autre, sur-segmentation rendant l'appariement incorrect en terme de segments, intersection de segments non coplanaires ...). De plus, un problème important de la reconstruction 3D en intérieur n'est toujours pas résolu : la présence de la majorité des

points sur un plan commun, qui est une configuration critique pour l'estimation de pose relative [63].

Une autre façon d'estimer partiellement la pose relative à partir de lignes est d'utiliser les points de fuite. En calculant les points de fuite sur chaque images et en les mettant en correspondance, il est possible d'obtenir la rotation relative. Cependant ces méthodes ne sont généralement pas très précises et peu robustes aux changements importants d'angles de vue, fréquents en intérieur. De plus, la précision est rarement la priorité des méthodes de détection de points de fuite car ceux-ci ne sont pas évaluées sur des critères de calibration mais plutôt sur des critères de regroupement de lignes ainsi que sur des estimations des directions du zénith et de l'horizon, aux vérités terrains arbitraires [83, 41]. De plus les points de fuite ne sont que des constructions théoriques. Affirmer que deux lignes sont parallèles parce que leur direction est similaire engendre nécessairement des erreurs. En pratique, les points de fuite sont différents pour la plupart des lignes (même si ils sont proches pour une grande partie d'entre eux). Construire des points de fuite sur la base de nombreuses approximations de ce type réduit alors la meilleure précision atteignable pour une estimation de rotation. En pratique, les méthodes à base de points de fuite sont surtout utilisées dans le cadre du SLAM [38] où la différence d'une image à l'autre est faible ce qui rend le procédé plus stable et plus précis. De plus la rapidité de cette estimation la rend intéressante dans ce cadre particulier.

Critère \ Méthode		Points			Lignes	
		Essentielle [61]	Homographie [24]	Trifocal [24]	Bifocal [14]	Trifocal [85]
Scène planaire		✗	✓	✓	✓	✓
Peu de texture		✗	✗	✗	✓	✓
Recouvr. limité		✗	✗	✗	✓	✗
M-W	✓	✓	✓	✓	✓	✓
	✗	✓	✓	✓	✗	✓

TABLE 5.1 – Récapitulatif de l'efficacité des méthodes d'estimation de pose relative existantes. La comparaison se fait avec des critères fréquents dans les scènes d'intérieur (planarité, manque de texture, manque de recouvrement entre images) ainsi qu'avec un critère de vérification des hypothèses de type *Manhattan-world* (noté *M-W* dans le tableau).

Enfin, il est possible d'utiliser des contraintes de type *Manhattan-world* (*i.e.* deux directions quelconques de la scène sont soit parallèles soit orthogonales). Cependant de telles méthodes [36] sont alors restreintes aux scènes vérifiant pleinement ces hypothèses. Plus récemment, Elqursh *et al.* [14] proposent une méthodes utilisant les lignes avec des contraintes d'orthogonalité et de parallélisme. Ils définissent ainsi une structure localement de type *Manhattan-world* pour estimer la rotation relative entre les deux caméras. Cette structure nécessite la présence de 3 lignes l_1, l_2 et l_3 telles que l_1 soit parallèle à l_2 et la direction l_3 soit orthogonale à celles de l_1 et l_2. Ce triplet de ligne permet de définir 2 points de fuite dans chaque image dont les directions correspondantes sont orthogonales. Leur correspondance permet alors d'obtenir la rotation relative. Comme nous le verrons dans la suite, cette méthode se trouve être plus précise et plus robuste qu'une méthode utilisant directement les points de fuite pour estimer la rotation relative. Pour estimer la direction de translation entre les deux caméras, Elqursh *et al.* utilisent des correspondances de points. En effet, les contraintes de parallélisme et d'orthogonalités ne font intervenir que les directions 3D des lignes et la rotation relative entre les deux caméras. Elles ne permettent donc pas d'obtenir d'informations sur la direction de la translation. Elqursh *et al.* [14] utilisent des intersections de lignes, supposées coplanaires, pour obtenir la direction de la translation. Ils peuvent donc justifier d'une méthode à partir de lignes uniquement. Nous

verrons cependant qu'il est plus précis d'utiliser les correspondances de points détectés à partir de méthodes comme SIFT [46] ou SURF [7] quand il y en a dans la scène.

Nous avons rassemblé dans le Tableau 5.1 l'efficacité des méthodes existantes en fonction de certains critères relatif à la scène que l'on veut calibrer. Dans le cadre de cette thèse, nous avons sélectionné des critères correspondant à des problèmes fréquents dans la reconstruction d'intérieur (planarité, manque de texture, manque de recouvrement entre images). Nous avons également ajouté l'utilisation ou non d'hypothèse de type *Manhattan-World* qui est au centre des prochaines sections. Comme nous pouvons le voir, les méthodes à base de points sont fortement gênées par les critères d'intérieur mais sont indifférents aux critères d'orthogonalité ou de parallélisme dans la scène. A l'inverse, les méthodes à base de lignes sont moins affectées par les critères d'intérieur mais peuvent l'être par l'orthogonalité ou le parallélisme de la scène. Ainsi, les méthodes actuelles permettent la reconstruction en intérieur sous certaines hypothèses (*e.g.* présence de suffisamment de *features* pour les méthodes trifocales ou scène de type *Manhattan-world* pour la méthode d'Elqursh *et al.* [14]).

5.2 Estimation de pose à partir de lignes avec des hypothèses type *Manhattan-world*

Elqursh *et al.* [14], présentent une méthode pour estimer la pose relative entre deux caméras, à partir de segments et d'hypothèses de structures localement *Manhattan-world*. Cette méthode divise l'estimation de pose en deux étapes :

- un calcul de la rotation à partir de correspondances de lignes et des points de fuite associés
- un calcul de la direction de translation à rotation connue et à partir de correspondances de points

5.2.1 Estimation de rotation relative à partir de segments

Pour cela, les auteurs utilisent des triplets de lignes $\mathcal{L}$ définis par :

$$\mathcal{L} = \{(L_1, L_2, L_3) \text{ lignes 3D, telle que } L_1 \parallel L_2 \text{ et } u_{L_1} \perp u_{L_3}\} \tag{5.1}$$

où u_{L_i} correspond à la direction de la ligne L_i.

A l'aide des projections de ces lignes dans les deux images, ils peuvent estimer les coordonnées de deux points de fuite dans chacune des deux images. Comme nous l'avons vu précédemment (voir section 3.7.3), un point de fuite correspond, en coordonnées normalisées, à la direction de la ligne 3D associée. On notera $(\mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3)$ les projections de la caméra 1 et $(\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3)$ les projections de la caméra 2. Les lignes L_1 et L_2 sont associées aux points de fuite $\mathbf{u}_1$ pour la caméra 1 et $\mathbf{v}_1$ pour la caméra 2. On peut définir leurs coordonnées à partir de la définition d'un point de fuite, soit :

$$\mathbf{u}_1 = \mathbf{l}_1 \times \mathbf{l}_2 \tag{5.2}$$

$$\mathbf{v}_1 = \mathbf{m}_1 \times \mathbf{m}_2 \tag{5.3}$$

La ligne L_3 est associée aux points de fuite $\mathbf{u}_2$ pour la caméra 1 et $\mathbf{v}_2$ pour la caméra 2. Pour obtenir leurs coordonnées, on utilise les deux propriétés suivantes :

- Un point de fuite $\mathbf{u}$ appartient à la ligne $\mathbf{l}$ à laquelle il est associé et donc $\mathbf{u}_2^T \mathbf{l}_3 = 0$ et $\mathbf{v}_2^T \mathbf{m}_3 = 0$
- $L_1 \perp L_3$ donc leurs directions 3D (*i.e.* les point de fuite en coordonnées normalisées) sont orthogonales : $\mathbf{u}_1^T \mathbf{u}_2 = 0$ et $\mathbf{v}_1^T \mathbf{v}_2 = 0$.

Ainsi, on obtient les coordonnées des points de fuite $\mathbf{u}_2$ et $\mathbf{v}_2$ à l'aide des équations suivantes :

$$\mathbf{u}_2 = \mathbf{u}_1 \times \mathbf{l}_3 \tag{5.4}$$

$$\mathbf{v}_2 = \mathbf{v}_1 \times \mathbf{m}_3 \tag{5.5}$$

En notant R la rotation relative de la caméra 1 à la caméra 2, comme les points de fuite correspondent aux direction 3D des lignes, on obtient les relations suivantes :

$$s_1\mathbf{v}_1 = R\mathbf{u}_1 \tag{5.6}$$

$$s_2\mathbf{v}_2 = R\mathbf{u}_2 \tag{5.7}$$

Où $s_1 = \pm 1$ et $s_2 = \pm 1$ correspondent à des potentiels changement de signes des points de fuite d'une image à l'autre. En effet, les points de fuite correspondent à des directions 3D mais sont également signés par l'intersection des lignes projetées. Ainsi, une observation parfaitement fronto-parallèle générera des points de fuite des deux signes (points *à l'infini*). En se décalant d'un côté ou de l'autre, l'observation génèrera un point de fuite de signe positif ou négatif (voir Figure 5.2).

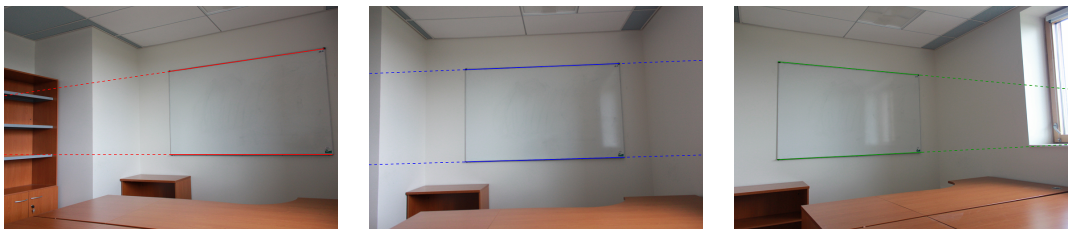


FIGURE 5.2 – Exemple d'un même objet générant des points de fuite de signes différents selon la position de la caméra qui l'observe. Dans l'image centrale, l'observation du tableau est presque fronto-parallèle, le point de fuite bleu n'a alors pas de côté privilégié. Dans les images de gauche et droite le point de fuite change alors de côté en passant du côté droit du tableau (rouge) au côté gauche (vert). Ce changement de côté correspond à un changement de signe du point de fuite en coordonnées homogènes.

Comme les points de fuite sont calculés en coordonnées homogènes, on peut leur imposer d'avoir une norme de 1. Avec cette contrainte, on peut définir les matrices de rotations $R_1 = [\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_1 \times \mathbf{u}_2]$ et $R_2 = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_1 \times \mathbf{v}_2]$. Ces matrices sont bien dans $SO_3(\mathbb{R})$ car leurs vecteurs colonnes forment une base orthonormale directe. La rotation relative entre les deux caméras est alors donnée par :

$$R = R_2 \Sigma R_1^T \tag{5.8}$$

où Σ est la matrice de rotation qui décrit les changements de signes des points de fuite :

$$\Sigma = \begin{bmatrix} s_1 & & \\ & s_2 & \\ & & s_1 s_2 \end{bmatrix} \tag{5.9}$$

En notant $e_i \in \mathbb{R}^3$ les vecteurs dont la i-ième coordonnée est 1 et les autres sont nulles, on obtient :

$$\forall i \in [1,2],\ R\mathbf{u}_i = R_2 \Sigma e_i = s_i R_2 e_i = s_i \mathbf{v}_i \tag{5.10}$$

Ainsi, cette rotation fait bien passer de $\mathbf{u}_i$ à $\mathbf{v}_i$ en utilisant s_i pour un potentiel changement de signe. La rotation R décrite dans l'équation 5.8 correspond bien à la rotation relative entre la caméra 1 et la caméra 2.

Notons que le changement de signe décrit par Σ peut être fixé en imposant que l'angle de rotation autour de l'axe soit de moins que 90°. Il est également possible de fixer cette indéterminée en utilisant les mêmes méthodes que pour sélectionner la *bonne* décomposition en rotation-translation d'une matrice essentielle à partir de 5 points. Pour cela, on calcule les positions des lignes dans la scène 3D et on impose que celles-ci soient en majorité devant les caméras.

5.2.2 Estimation robuste de la rotation relative

La méthode décrite précédemment permet d'estimer la rotation relative entre deux caméras à partir d'une correspondance d'un triplet de segments suivant une configuration particulière.

Cependant, détecter de tels triplets peut se révéler être un problème difficile. Elqursh *et al.* [14] proposent, plutôt que de détecter les triplets particuliers, de combiner ce calcul à une méthode de type RANSAC. Pour cela, ils calculent l'ensemble des triplets générés à partir de toutes les lignes. Ils en éliminent une partie à l'aide de critères géométriques mais en conservent une majorité contenant une portion importante d'*outliers*.

Pour définir l'adéquation d'une rotation R à un triplet de segments $\mathcal{L}$, ils définissent la distance suivante :

$$d(\mathcal{L},R) = \frac{1}{2}\sum_{i=1}^{2} \cos^{-1}(\mathbf{v}_i^\top R\mathbf{u}_i) \tag{5.11}$$

où $\mathbf{u}_i$ et $\mathbf{v}_i$ sont les points de fuite normalisés définis comme dans la section précédente. Cette distance correspond à l'erreur angulaire de positionnement du point de fuite de la seconde caméra avec le point de fuite de la première caméra auquel on a appliqué la rotation relative R.

Une première estimation de la rotation relative se fait à l'aide d'un RANSAC sur l'ensemble des triplets de segments, puis un raffinement est réalisé à partir des *inliers* sélectionnés.

La rotation relative finale $R^\star$ doit vérifier l'équation suivante :

$$R^\star = \underset{R\in SO_3(\mathbb{R})}{\arg\min} \sum_{\mathcal{L}_i\ inliers} d(\mathcal{L}_i,R) \tag{5.12}$$

Cette équation nécessitant une optimisation non linéaire, Elqursh *et al.* résolvent une équation similaire :

$$R^\star = \underset{R\in SO_3(\mathbb{R})}{\arg\min} \sum_{\mathcal{L}_i\ inliers} \sum_{j=1}^{2} \|\mathbf{v}_{ij} - R\mathbf{u}_{ij}\|_2^2 \tag{5.13}$$

Cette équation peut se réecrire sous la forme :

$$R^\star = \underset{R\in SO_3(\mathbb{R})}{\arg\min} \; ||D - R^\top D'||_F^2 \tag{5.14}$$

où D et D' sont des matrices $3\times 2N$ regroupant en colonnes les points de fuite des triplets *inliers*. Ce problème est connu sous le nom de *problème de Procruste orthogonal* et sa solution est donnée à l'aide d'une décomposition en valeur singulière de la matrice $D'D^T$ [70].

5.2.3 Estimation de la direction de translation à rotation connue

L'estimation de translation à partir d'une rotation connue ou, de manière équivalente, dans le cas d'un mouvement de pure translation a déjà été étudiée par de nombreux auteurs [74, 16]. Elqursh *et al.* proposent d'ailleurs d'utiliser la méthode linéaire de Sun *et al.* [74]. Pour un appariement de points $(\mathbf{p}_i,\mathbf{q}_i)$, nous avons la relation suivante :

$$(R\mathbf{p}_i \times \mathbf{q}_i)^\top t = 0 \tag{5.15}$$

Cette relation découle du fait que pour un point quelconque de la scène P_i, $(\mathcal{C}, \mathcal{C}', P_i)$ forment un plan qui contient, par construction, les projetés de P_i soit $\mathbf{p}_i$ et $\mathbf{q}_i$. Comme la translation ne peut

être obtenue qu'à un facteur d'échelle près, seules deux équations de ce type suffisent à estimer t. La direction de t correspond alors au vecteur associé à la plus petite valeur singulière de la décomposition en valeurs singulières de la matrice 3×3 suivante :

$$\sum_{i=1}^{2} (R\mathbf{p}_i \times \mathbf{q}_i)(R\mathbf{p}_i \times \mathbf{q}_i)^\top \tag{5.16}$$

L'estimation de la translation se fait à l'aide d'un RANSAC en utilisant la distance épipolaire pour déterminer *outliers* et *inliers*. De plus, Elqursh *et al.* [14], pour conserver une méthode à base de lignes uniquement, utilisent les intersections des segments détectés comme *inliers* lors de l'estimation de rotation. Comme nous le verrons dans la suite, l'utilisation d'intersections n'est pas très adaptée même dans le cas des scènes d'intérieur et nous utiliserons les détections et appariements SIFT [46] à la place quand ils sont disponibles.

5.2.4 Raffinement non linéaire

Les lignes n'apportant pas de contraintes bifocales, elles ne peuvent pas être utilisées telles quelles pour un raffinement supplémentaire. Elqursh *et al.* décident donc d'utiliser une formulation classique de raffinement non linéaire pour les points en l'appliquant aux points et aux points de fuite *inliers* (pour rappel, chaque triplet de segments fournit deux correspondances de points de fuite). L'équation est sous la forme :

$$(R^\star, t^\star) = \arg\min_{R,t} \sum_{(\mathbf{p}_i, \mathbf{q}_i) \in \mathcal{P}} d(\mathbf{p}_i, \mathbf{q}_i) \tag{5.17}$$

où $\mathcal{P}$ représente l'ensemble des appariement *inliers* de points et points de fuite et d la distance épipolaire (voir Eq. 3.24).

Arguant du fait que la distance épipolaire *sans bruit* pour une paire de points de fuite est nulle, ils utilisent cette distance comme contrainte douce sur la rotation. En effet, la nullité de cette distance n'est due qu'à la rotation et n'apporte aucune contrainte sur la translation. Pour deux points de fuite correspondants u et v, la distance épipolaire est bien nulle :

$$v^\top E u = v^\top [t]_\times R u = v^\top (t \times v) = 0 \tag{5.18}$$

Cependant, cette approche présente deux défauts :

- les points de fuite pouvant être situés loin de l'image (voire *à l'infini*) comme en plein milieu de l'image, la distance épipolaire entre deux points de fuite dépend de la position du point de fuite considéré. Additionner des distances dont les ordres de grandeur peuvent varier impose alors des poids arbitraires sur certains *inliers* sans justification théorique réelle.
- sur les scènes testées, cette optimisation tend en pratique à ignorer l'apport des points de fuite. Nous montrerons empiriquement comment la minimisation a un effet positif lorsque les points sont plus précis que les lignes et négatif dans le cas inverse.

5.3 Estimation de rotation sans hypothèses de type *Manhattan-world*

Le principal défaut de la méthode décrite précédemment est l'utilisation d'hypothèses de type *Manhattan-world*. Même si les scènes urbaines vérifient généralement ces critères, il existe de nombreuses scènes que nous appellerons *quasi-Manhattan* avec trois directions principales d_1, d_2, d_3 telles que :

- la direction verticale d_1 est orthogonale aux deux autres,
- les deux autres directions ne sont pas orthogonales.

Une telle scène n'empêche pas de trouver des triplets vérifiant les hypothèses fixées par Elqursh *et al.* Cependant, les triplets étant énumérés de manière exhaustive, de nombreux *outliers* sont inclus dans l'ensemble des triplets. Nous verrons dans la suite (section 5.5) comment ces triplets *outliers* détériorent la calibration.

Plutôt que de chercher à détecter de manière plus robuste les triplets *inliers*, nous avons préféré changer quelques hypothèses pour obtenir une méthode utilisable dans pratiquement n'importe quelle scène urbaine. Pour estimer une rotation entre deux caméras, nous n'avons besoin que de deux directions distinctes de la scène (*i.e.* deux points de fuite distincts). De plus ces deux directions n'ont pas besoin d'être orthogonales. En effet, en supposant que nous ayons accès à deux appariements de points de fuite distincts, notés $(\mathbf{u}_1,\mathbf{u}_2)$ pour la caméra 1 et $(\mathbf{v}_1,\mathbf{v}_2)$ pour la caméra 2, nous pouvons déterminer la rotation relative entre les deux caméras [48] :

$$R = AB^T \tag{5.19}$$

où A et B sont définis par la décomposition en valeur singulière de la matrice M suivante :

$$M = s_1\mathbf{v}_1\mathbf{u}_1^\top + s_2\mathbf{v}_2\,\mathbf{u}_2^\top \tag{5.20}$$

$$= A\Sigma B^\top \tag{5.21}$$

avec s_1 et s_2 représentant les potentiels changements de signe des points de fuite d'une caméra à l'autre. Tout comme la méthode précédente, nous obtenons donc 4 rotations possibles correspondant aux différentes combinaisons de signes. Nous sélectionnons alors, selon les mêmes critères, la rotation qui nous semble la plus envisageable.

L'obtention des appariements de point de fuite se fait en utilisant un appariement de segments couplés à une détection de point de fuite. Nous sélectionnons deux appariements de segments liés au même point de fuite pour obtenir les valeurs de u_1 et v_1 puis nous procédons de même avec une autre paire de segment. Comme notre méthode nécessite deux directions distinctes, pour éviter les cas dégénérés, nous vérifions que les angles $\angle(\mathbf{u}_1,\mathbf{u}_2)$ et $\angle(\mathbf{v}_1,\mathbf{v}_2)$ dépassent un certain seuil (nous utilisons 5° dans notre implémentation). Dans le cas où ce seuil n'est pas dépassé, nous n'utilisons pas cet appariement de point de fuite et en énumérons un autre.

Notons également que, contrairement à la méthode de Elqursh *et al.* [14], les points de fuite utilisés pour le calcul d'une rotation ne vérifient pas les relations de l'équation 5.7 de façon exacte. Un moyen simple de s'en convaincre est de remarquer que l'angle $\angle(\mathbf{u}_1,\mathbf{u}_2)$ est généralement différent de l'angle $\angle(\mathbf{v}_1,\mathbf{v}_2)$ (*e.g.* bruit de mesure, distorsion, appariement incorrect...) ce qui impose donc qu'il n'existe pas de rotation vérifiant les relations 5.7. Cependant, si cette méthode ne fournit qu'une rotation approximative, en la couplant à une méthode type RANSAC, il est possible d'obtenir une bonne estimation de la rotation relative entre les deux caméras.

De la même façon que la méthode précédente, nous utilisons un RANSAC pour filtrer les *outliers* avec une distance similaire à la précédente. Pour une paire $(\mathbf{l}_i,\mathbf{l}_j)$ mise en correspondance avec une paire $(\mathbf{m}_i,\mathbf{m}_j)$, la distance d'adéquation à la rotation est donnée par :

$$d_{\text{lignes}}(\mathbf{l}_i,\mathbf{l}_j,\mathbf{m}_i,\mathbf{m}_j,R) = \cos^{-1}|\mathbf{v}_{i,j}^\top R\mathbf{u}_{i,j}| \tag{5.22}$$

où $\mathbf{u}_{i,j}$ et $\mathbf{v}_{i,j}$ sont les points de fuite des paires de segments $(\mathbf{l}_i,\mathbf{l}_j)$ et $(\mathbf{m}_i,\mathbf{m}_j)$ (voir Fig. 5.3). Notons également que cette distance ne prend pas en compte les signes des points de fuite et ne cherche qu'une égalité de directions. En effet, il n'est pas possible à ce stade de savoir quel est le signe associé à cet appariement de points de fuite. Cependant, une fois la partie RANSAC terminée, le signe des *inliers* est alors connu et peut être utilisé dans le raffinement.

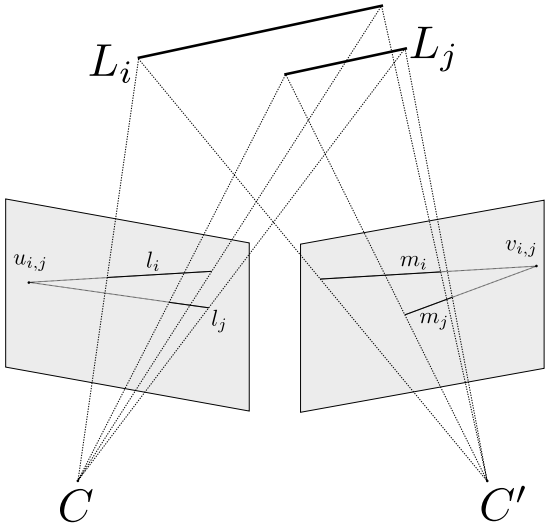


FIGURE 5.3 – Deux segments parallèles $\mathbf{l}_i$ et $\mathbf{l}_j$ (resp. $\mathbf{m}_i$ et $\mathbf{m}_j$) génèrent un point de fuite $\mathbf{u}_{i,j}$ (resp. $\mathbf{v}_{i,j}$). Si l'appariement est correct, la rotation relative R doit changer $\mathbf{u}_{i,j}$ en $\mathbf{v}_{i,j}$. En pratique, on cherche à minimiser cet angle.

Nous raffinons ensuite la solution à l'aide des *inliers* restants en suivant la méthode de résolution du *Procruste orthogonal* [70] puis la translation relative est estimée de la même façon que la méthode d'Elqursh *et al.* [14].

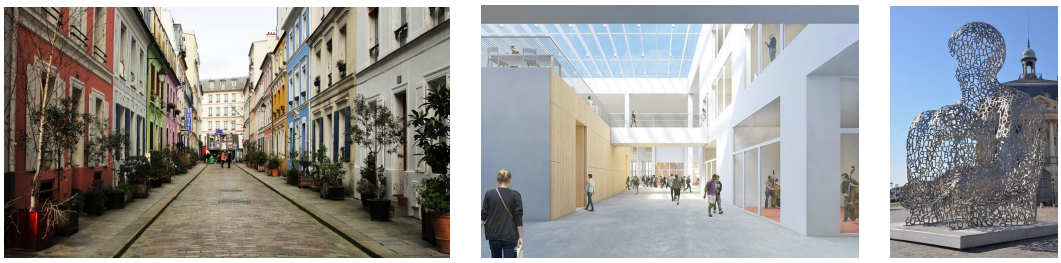


FIGURE 5.4 – Exemples de scènes urbaines et d'intérieur. On y observe toujours des segments parallèles selon deux directions différentes (même sur la troisième grâce au socle). Seule la troisième scène représente un cas gênant (trop nombreuses directions présentes qui risqueraient de bruiter les détections de point de fuite) pour notre algorithme mais ce cas est supposé peu fréquent.

Notons que cette méthode a plusieurs avantages sur la précédente :

- le nombre de *features* est réduit, passant de $O(n^3)$ à $O(n^2)$ puisque nous passons de triplets à des paires de segments (n étant le nombre de segments détectés dans la scène). Si le nombre de modèle à tester augmente (en passant de $O(\binom{n^3}{1})$ à $O(\binom{n^2}{2})$), le calcul du NFA passe de $O(n^3)$ à $O(n^2)$. Le nombre de modèle étant généralement seuillé au nombre d'itération, cela permet une accélération du RANSAC en réduisant le temps de calcul du nombre d'*inliers* pour chaque rotation trouvée.
- les nouvelles hypothèses sont en pratique vérifiées dans la quasi-totalité des scènes urbaines ou d'intérieur, la présence de ligne impliquant presque toujours la présence d'au moins deux groupes de lignes parallèles (voir Figure 5.4).

5.4 Estimation mixte point-ligne

Comme nous le verrons dans la suite, les expériences montrent que des calibrations calculées à partir de points [61] sont généralement plus précises que celles obtenues à l'aide des méthodes à base de segments. Cependant les méthodes utilisant les lignes sont généralement plus robustes dans le sens où elles obtiennent un résultat dans certaines scènes d'intérieur où les points ne suffisent pas à calibrer. En effet, comme le montrent Philip *et al.* [63], les surfaces planaires sont des cas critiques pour les méthodes d'estimation à partir de points.

Nous avons donc cherché à développer une méthode utilisant à la fois les lignes et les points pour obtenir des résultat plus robustes que les méthodes à base de points mais tout aussi précises.

5.4.1 Estimation robuste

Une approche naïve aurait été de combiner une méthode de calibration à partir de points avec l'une des méthodes précédentes et un RANSAC. Cependant, une telle méthode aurait fait apparaître deux seuils d'*inliers* ; un pour les points (pixellique) et un pour les segments (angulaire). Pour éviter l'utilisation de deux seuils distincts qui auraient rendu cette méthode trop dépendante du choix des paramètres, nous avons préféré une méthode homogène mesurant des écarts angulaires uniquement, et pour cela définir une distance angulaire entre un appariement de points et une matrice essentielle. Pour un appariement de point $(\mathbf{p},\mathbf{q})$ et une pose relative (R,t) entre les deux caméras, nous savons que :

$$R\mathbf{p}+t=\mathbf{q} \tag{5.23}$$

N'ayant pas accès à la norme de t, cette relation nous permet de dire néanmoins que :

$$R\mathbf{p}\times t=\mathbf{q}\times t \tag{5.24}$$

Cette relation étant vraie à un facteur d'échelle près, nous pouvons alors définir la distance angulaire épipolaire par :

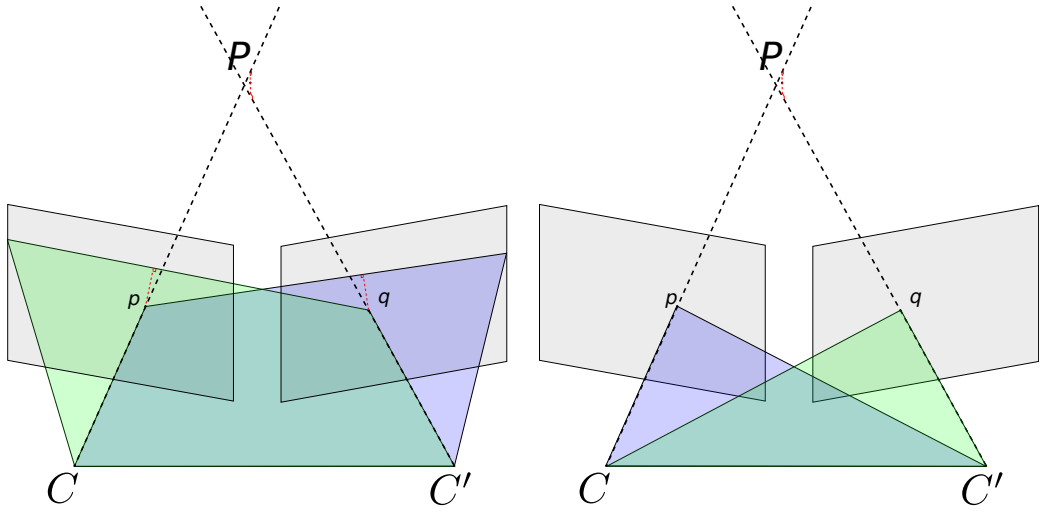
$$d_{\mathsf{points}}(\mathbf{p},\mathbf{q},R,t)=\cos^{-1}\left|\frac{(R\mathbf{p}\times t)^{\top}}{\|R\mathbf{p}\times t\|_2}\frac{\mathbf{q}\times t}{\|\mathbf{q}\times t\|_2}\right| \tag{5.25}$$

Cette distance peut également être vue comme l'écart des directions des normales des plans épipolaires $CC'p$ et $CC'q$ (voir Fig. 5.5).

Elle est similaire à la relation épipolaire usuelle mais elle fournit une mesure angulaire au lieu d'une mesure pixellique. En utilisant la distance d_{lignes} définie précédemment pour un appariement d'une paire de ligne parallèles (voir Eq. 5.22), il est alors possible d'utiliser un seuil commun pour déterminer les *inliers*, que ce soit des points ou des segments. L'estimation robuste suit alors le schéma usuel du RANSAC et l'algorithme est résumé sur la figure 5.6.

5.4.2 Méthode *a contrario*

Pour éviter d'avoir à sélectionner explicitement un seuil d'*inliers* pour chaque scène, nous avons également développé une version *a contrario* de la méthode. Comme l'ont montré Moulon *et al.* [58], AC-RANSAC (pour *a contrario* RANSAC) se comporte mieux que RANSAC car il sélectionne le seuil optimal pour chaque paire d'images. Il s'adapte ainsi à la fois aux différences entre les scènes mais également à la variabilité des paires d'images appartenant à une même scène. Nous reprenons ici les termes présentés dans la section 3.9 en les explicitant dans le cadre de cette méthode.



Distance épipolaire pixellique Distance épipolaire angulaire

FIGURE 5.5 – A gauche la distance épipolaire usuelle. Le plan épipolaire $CC'p$ (en bleu) s'intersecte avec le plan image de la seconde caméra pour former la droite épipolaire de p. C'est la distance entre q et cette droite qui permet de calculer la distance épipolaire pixellique usuelle. Le procédé est symétrisé avec le plan $CC'q$ (en vert) pour plus de stabilité. Dans notre cas, nous utilisons la distance angulaire entre les normales des deux plans épipolaires (bleu et vert) formés par $CC'p$ et $CC'q$.

Données d'entrée :

Les données en entrée correspondent à des appariements de points et de segments. Comme nous le verrons dans la suite, pour une question d'indépendance, les appariements ne concernent que les segments et non les paires de segments (ou de façon équivalente, les points de fuite associés).

Modèle estimé :

Le modèle estimé correspond à la matrice essentielle E pour les points et à la rotation relative pour les lignes. En effet, les lignes n'apportent aucune contrainte sur la translation. Notons également qu'en pratique nous estimons soit la matrice essentielle à l'aide de points, soit la rotation à partir de lignes puis la direction de translation à partir de points. Cependant, ces deux informations sont équivalentes (voir section 3.5.1, p.25) puisqu'il est possible de passer de l'un à l'autre en utilisant la relation :

$$E = [t]_\times R \tag{5.26}$$

Le nombre de modèles estimés varie en fonction de la méthode. Dans le cas des points, nous utilisons la méthode des 5 points [61] qui calcule 10 matrices essentielles différentes. Dans le cas des lignes nous estimons la rotation à partir de 4 matrices différentes (changement de signe potentiel des points de fuite).

Distance données-modèle :

Pour les points, la distance utilisée est la même que précédemment, soit d_{points} (Eq. 5.25). Dans le cas des segments, de même qu'un appariement de segments n'apporte aucune contrainte à la pose relative entre deux caméras (voir section 3.5.1), il n'existe pas de distance permettant de mesurer l'adéquation d'un appariement de segments à une pose relative donnée. En notant $\mathbf{l}$

Données : ensemble d'appariements de points $\mathcal{P}$, ensemble d'appariements de segments $\mathcal{S}$.
Paramètres : seuil ε, nombre d'itérations N
Output : pose relative $(R^\star, t^\star)$
Regrouper les segments en groupe de segments parallèles $\mathcal{S}_\parallel$ à l'aide des points de fuite.
$N^\star_{INLIERS} \leftarrow 0$
pour N itérations **faire**
  Sélectionner aléatoirement 5 éléments de $\mathcal{P}$, ou 2 de $\mathcal{S}_\parallel$ et 2 de $\mathcal{P}$
  (on alterne une fois sur deux)
  Estimer la pose relative associée (R,t)
  $N_{INLIERS} \leftarrow 0$
  # Décompte des *inliers* points
  pour $(p,q) \in \mathcal{P}$ **faire**
    si $d_{\mathsf{points}}(p,q,R,t) < \varepsilon$ **alors**
      (p,q) est un *inlier* : $N_{INLIERS} \leftarrow N_{INLIERS} + 1$
    fin si
  fin pour
  # Décompte des *inliers* segments
  pour $(l_i, m_i, l_j, m_j) \in \mathcal{S}_\parallel$ **faire**
    si $d_{\mathsf{lignes}}(l_i, m_i, l_j, m_j, R) < \varepsilon$ **alors**
      (l_i, m_i, l_j, m_j) est un *inlier* : $N_{INLIERS} \leftarrow N_{INLIERS} + 1$
    fin si
  fin pour
  # Mise à jour de la solution
  si $N_{INLIERS} < N^\star_{INLIERS}$ **alors**
    Mettre à jour le modèle final : $(R^\star, t^\star) \leftarrow (R,t)$, $N^\star_{INLIERS} \leftarrow N_{INLIERS}$
  fin si
fin pour
retourner $(R^\star, t^\star)$

FIGURE 5.6 – Estimation robuste de pose relative à partir de points et de lignes.

et $\mathbf{m}$ les projetés d'un segment de la scène dans les deux caméras et R une matrice de rotation, nous pouvons définir la direction du segment par :

$$u(R, \mathbf{l}, \mathbf{m}) = (R\,\mathbf{l}) \times \mathbf{m} \tag{5.27}$$

La direction du segment n'est correcte que lorsque R correspond à la rotation relative entre les deux caméras. En notant $u^\star$ cette direction, nous pouvons alors définir une distance d'adéquation entre une rotation et un appariement de segments $(\mathbf{l}, \mathbf{m})$ par :

$$d^\star_{\mathsf{ligne}}(R, \mathbf{l}, \mathbf{m}, u^\star) = \angle(u(R), u^\star) \tag{5.28}$$

Cependant, cette distance ne peut généralement pas être calculée puisque la direction $u^\star$ n'est pas connue. Une façon de déterminer la valeur de $u^\star$ serait d'utiliser les directions des points de fuite calculés sur les images. Cependant cette méthode reviendrait à considérer que les valeurs des points de fuite sont parfaites. Nous verrons par la suite que ces valeurs ne sont pas si précises que ça. Nous avons donc choisi de définir une distance qui cherche à approximer cette valeur :

$$d_{\mathsf{ligne}}(\mathbf{l}_i, \mathbf{m}_i, R) = \min_{L_j \| L_i} d_{\mathsf{lignes}}(\mathbf{l}_i, \mathbf{l}_j, \mathbf{m}_i, \mathbf{m}_j) = \min_{L_j \| L_i} d^\star_{\mathsf{ligne}}(R, \mathbf{l}_i, \mathbf{m}_i, (R\,\mathbf{l}_j) \times \mathbf{m}_j) \tag{5.29}$$

La distance d_{ligne} est en général différente de $d^\star_{\mathsf{ligne}}$. Cependant la distance $d^\star_{\mathsf{ligne}}$ n'est pas exempte de défaut puisqu'elle n'est pas définie lorsque $\mathbf{l}_i$ et $\mathbf{m}_i$ ne correspondent pas à un appariement correct. À l'inverse, notre approximation d_{ligne} devrait donner une valeur élevée dans ce

cas-là. De plus, l'attribution des points de fuite n'est pas parfaite car deux lignes quasiment parallèles dans la scène seront associées au même point de fuite alors que leur direction, même en étant similaire, est strictement différente. Néanmoins, nous verrons que cette distance permet en pratique de faire fonctionner une méthode *a contrario* plus efficacement qu'un RANSAC avec seuil.

Modèle de fond :

On utilise ici deux modèles de fond différent : $\mathcal{H}_0^{\text{pt}}$ pour les points et $\mathcal{H}_0^{\text{seg}}$ pour les lignes. Le modèle $\mathcal{H}_0^{\text{pt}}$ (resp. $\mathcal{H}_0^{\text{seg}}$) correspond à un ensemble contenant autant de points (resp. de segments) que détectés dans la scène. Cependant, ceux-ci sont indépendamment et uniformément distribués dans chaque image. C'est d'ailleurs cette hypothèse d'indépendance nécessaire au calcul du NFA qui nous a contraint a définir la distance d_{ligne}.

Nombre de Fausses Alarmes :

Le NFA peut alors être défini pour évaluer l'adéquation d'un groupe de *features* $\mathcal{F}$ à la pose relative (R,t) qui lui est associé. Les modèles *a contrario* ont déjà été utilisés pour une estimation robuste de la matrice fondamentale [53, 56] puis généralisés à n'importe quel modèle géométrique [54] (*e.g.* matrice essentielle, homographie). La formule générique est donnée par :

$$\text{NFA}(n,k,\varepsilon) = N_{\text{modele}}(n - N_{\text{echantillon}}) \binom{n}{k} \binom{k}{N_{\text{echantillon}}} \text{p}(\varepsilon)^{k-N_{\text{echantillon}}} \tag{5.30}$$

où $N_{\text{echantillon}}$ est le nombre d'échantillons nécessaires à l'estimation du modèle et N_{modele} correspond au nombre maximal de modèles estimés à partir d'un échantillonnage donné. k représente le nombre hypothétique d'*inliers* et $\text{p}(\varepsilon)$ est la probabilité qu'un *feature* généré par le modèle de fond $\mathcal{H}_0$ soit à une distance ε du modèle estimé.

Dans notre cas, le nombre maximal de modèles estimés N_{modele} est de 4 pour les lignes et 10 pour les points. Le nombre d'échantillon $N_{\text{echantillon}}$ est de 5 dans le cas des points tandis que dans pour les lignes, nous utilisons 2 paires de lignes parallèles pour estimer la rotation soit 4 lignes. On obtient alors deux formules du NFA pour les lignes et pour les points :

$$\text{NFA}_{\text{lignes}}(n_{\text{lignes}},k,\varepsilon) = 4(n_{\text{lignes}} - 4) \binom{n_{\text{lignes}}}{k} \binom{k}{4} \text{p}_{\text{lignes}}(\varepsilon)^{k-4} \tag{5.31}$$

$$\text{NFA}_{\text{points}}(n_{\text{points}},k,\varepsilon) = 10(n_{\text{points}} - 5) \binom{n_{\text{points}}}{k} \binom{k}{5} \text{p}_{\text{points}}(\varepsilon)^{k-5} \tag{5.32}$$

Le terme $\text{p}(\varepsilon)$ dépend de la distance du *feature* au modèle estimé. Cependant, la distance utilisée pour les lignes et pour les points correspond à une comparaison angulaire de deux directions 3D. Dans le cas des points, il s'agit de calculer la probabilité pour deux directions 3D aléatoires (*i.e.* les normales des deux plans épipolaires, voir Fig. 5.5) d'être alignées à ε près tandis que dans le cas des lignes, c'est la probabilité pour une direction 3D aléatoire (*i.e.* le point de fuite) d'être aligné à ε près avec une direction 3D fixe (*i.e.* la direction 3D de la ligne). Comme les directions 3D suivent le modèle de fond $\mathcal{H}_0$, elles sont indépendamment et uniformément distribuées dans l'espace et le cas des lignes est donc équivalent à celui des points.

Pour une direction u suivant le modèle de fond $\mathcal{H}_0$ et une direction d fixe, la probabilité qu'elles soient alignées à ε près peut être exprimé comme le rapport de l'aire de deux calottes sphériques de la sphère unité :

$$\text{p}(\varepsilon) = \frac{2\mathcal{A}_{\varepsilon-CAP}}{\mathcal{A}_{SPHERE}} = \frac{4\pi(1-\cos\varepsilon)}{4\pi} = 1 - \cos\varepsilon. \tag{5.33}$$

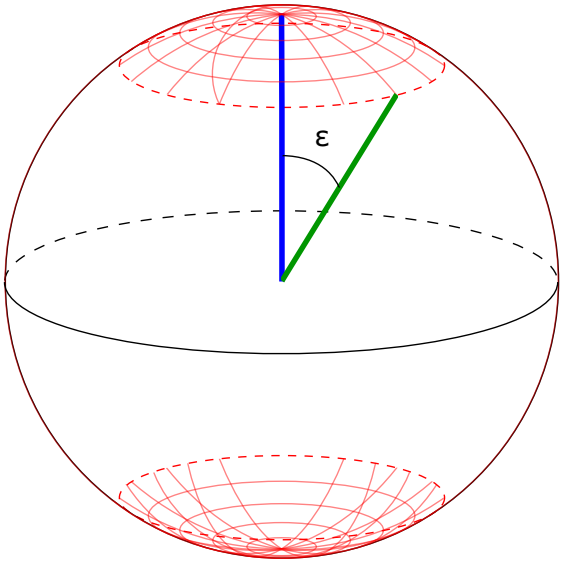


FIGURE 5.7 – En bleu la direction d fixe et en vert un exemple de u tiré aligné à moins de ε. Lorsque la direction u est tirée dans les calottes sphériques rouges, les directions sont les mêmes à ε près, sinon l'angle entre les deux directions est trop élevé. Notons également qu'on cherche une égalité de direction et donc les sens n'interviennent pas, c'est pourquoi la calotte sphérique du haut et du bas de la sphère sont en rouge.

où $\mathcal{A}_{SPHERE}$ représente l'aire de la sphère unité et $\mathcal{A}_{\varepsilon-CAP}$ l'aire de la calotte sphérique dont la taille varie avec ε (voir Fig. 5.7 pour plus de détails).

Nous obtenons ainsi deux formules de NFA pour l'estimation de rotation à partir de lignes ou de matrice essentielle pour les points. Or le NFA correspond à l'espérance du nombre de modèles ε-significatif pouvant être détectés à partir de données suivant le modèle de fond $\mathcal{H}_0$, on peut donc le décrire comme étant :

$$\text{NFA}_{\text{lignes}}(n_{\text{lignes}},k,\varepsilon) = \mathbb{E}[\mathcal{X}_{\text{lignes}}(n_{\text{lignes}},k,\varepsilon)] \tag{5.34}$$

où $\mathcal{X}_{\text{lignes}}(n_{\text{lignes}},k,\varepsilon)$ correspond à l'évènement "obtenir une rotation R à partir de 4 éléments parmi n_{lignes} lignes suivant le modèle de fond $\mathcal{H}_0$ telle que k lignes parmi les n_{lignes} lignes soient à une distance de moins de ε de R".

De manière équivalente on peut écrire que pour les points :

$$\text{NFA}_{\text{points}}(n_{\text{points}},k,\varepsilon) = \mathbb{E}[\mathcal{X}_{\text{points}}(n_{\text{points}},k,\varepsilon)] \tag{5.35}$$

Nous définissons alors le $\text{NFA}_{\text{lignes\&points}}$ comme étant l'espérance de l'intersection de ces deux évènements :

$$\begin{aligned}
&\text{NFA}_{\text{lignes\&points}}(n_{\text{lignes}},n_{\text{points}},k_{\text{lignes}},k_{\text{points}},\varepsilon_{\text{lignes}},\varepsilon_{\text{points}}) \\
&= \mathbb{E}[\mathcal{X}_{\text{lignes}}(n_{\text{lignes}},k_{\text{lignes}},\varepsilon_{\text{lignes}}) \cap \mathcal{X}_{\text{points}}(n_{\text{points}},k_{\text{points}},\varepsilon_{\text{points}})] \\
&= \mathbb{E}[\mathcal{X}_{\text{lignes}}(n_{\text{lignes}},k_{\text{lignes}},\varepsilon_{\text{lignes}})]\mathbb{E}[\mathcal{X}_{\text{points}}(n_{\text{points}},k_{\text{points}},\varepsilon_{\text{points}})] \\
&= \text{NFA}_{\text{lignes}}(n_{\text{lignes}},k_{\text{lignes}},\varepsilon_{\text{lignes}})\text{NFA}_{\text{points}}(n_{\text{points}},k_{\text{points}},\varepsilon_{\text{points}})
\end{aligned}$$

Le modèle sélectionné correspond alors à celui qui minimise le $\text{NFA}_{\text{lignes\&points}}$:

$$(R^\star,t^\star) = \arg\min_{R,t}(\text{NFA}_{\text{lignes}}(R)\text{NFA}_{\text{points}}(R,t)) \tag{5.36}$$

où les NFA des modèles sont définis par :

$$\mathrm{NFA}_{\mathsf{lignes}}(R) = \min_{\varepsilon>0} \mathrm{NFA}_{\mathsf{lignes}}(n_{\mathsf{lignes}}, k(R,\varepsilon), \varepsilon) \tag{5.37}$$

$$\mathrm{NFA}_{\mathsf{points}}(R,t) = \min_{\varepsilon>0} \mathrm{NFA}_{\mathsf{points}}(n_{\mathsf{points}}, k(R,t,\varepsilon), \varepsilon) \tag{5.38}$$

En pratique, ces formules ne pouvant pas être minimisées simplement, nous utilisons les approximations :

$$\mathrm{NFA}_{\mathsf{lignes}}(R) \approx \min_{k\in[5,n_{\mathsf{lignes}}]} \mathrm{NFA}_{\mathsf{lignes}}(n_{\mathsf{lignes}}, k, d_{\mathsf{ligne}}(\mathbf{l}_k, \mathbf{m}_k, R)) \tag{5.39}$$

$$\mathrm{NFA}_{\mathsf{points}}(R,t) \approx \min_{k\in[6,n_{\mathsf{lignes}}]} \mathrm{NFA}_{\mathsf{points}}(n_{\mathsf{points}}, k, d_{\mathsf{points}}(\mathbf{p}_k, \mathbf{q}_k, R, t)) \tag{5.40}$$

Notons ici que le *feature* d'indice k (*e.g.* appariement $(\mathbf{l}_k, \mathbf{m}_k)$ ou $(\mathbf{p}_k, \mathbf{q}_k)$ correspond au *feature* dont la distance au modèle testé est la k-ième plus petite.

5.4.3 Raffinement non-linéaire mixte point-ligne

Une fois la pose estimée à l'aide du RANSAC, *a contrario* ou non, sa valeur peut être raffinée par une méthode non linéaire. Pour éviter de reproduire les défauts présentés dans la section 5.2.4, nous avons utilisé les distances angulaires présentées précédemment (Eq. 5.22 pour les lignes et Eq. 5.25 pour les points).

Nous définissons une fonction de coût pour les paires de lignes parallèles :

$$\mathcal{C}_{\mathsf{lignes}}(R) = \sum_{(\mathbf{l}_i,\mathbf{l}_j,\mathbf{m}_i,\mathbf{m}_j)\ inlier} \left\| \frac{R(\mathbf{l}_i \times \mathbf{l}_j)}{\|\mathbf{l}_i \times \mathbf{l}_j\|_2} \times \frac{\mathbf{m}_i \times \mathbf{m}_j}{\|\mathbf{m}_i \times \mathbf{m}_j\|_2} \right\|_2^2 \tag{5.41}$$

$$= \sum_{(\mathbf{l}_i,\mathbf{l}_j,\mathbf{m}_i,\mathbf{m}_j)\ inlier} d_{\mathsf{lignes}}(\mathbf{l}_i, \mathbf{l}_j, \mathbf{m}_i, \mathbf{m}_j, R)^2 \tag{5.42}$$

et une fonction de coût pour les points :

$$\mathcal{C}_{\mathsf{points}}(R,t) = \sum_{(\mathbf{p}_i,\mathbf{q}_i)\ inlier} \left\| \frac{(R\mathbf{p}_i \times t)^\top}{\|R\mathbf{p}_i \times t\|_2} \times \frac{\mathbf{q}_i \times t}{\|\mathbf{q}_i \times t\|_2} \right\|_2^2 \tag{5.43}$$

$$= \sum_{(\mathbf{p}_i,\mathbf{q}_i)\ inlier} d_{\mathsf{points}}(\mathbf{p}_i, \mathbf{q}_i, R, t)^2 \tag{5.44}$$

Ces deux fonctions ont l'avantage d'être comparables et homogènes. Nous pouvons alors définir une fonction de coût global permettant d'obtenir les versions raffinées $(R^\star, t^\star)$ de la pose relative :

$$(R^\star, t^\star) = \arg\min_{R,t} \mathcal{C}_{\mathsf{lignes}}(R) + \mathcal{C}_{\mathsf{points}}(R,t) \tag{5.45}$$

La solution optimale est obtenue à l'aide de l'algorithme de Levenberg-Marquardt après l'avoir initialisé avec la pose obtenue après RANSAC *a contrario* ou non.

5.5 Expériences

Dans le but de valider l'efficacité de nos méthodes, nous avons réalisé de nombreuses expériences : des expériences synthétiques d'une part pour avoir un aperçu théorique du comportement relatif de chacune des méthodes, puis des expériences sur des données réelles pour confirmer l'efficacité des méthodes à base de lignes dans les scènes d'intérieur.

Dans les expériences, nous mesurons deux types d'erreur :

- L'erreur en rotation e_R qui combine les différences d'angle et d'axe entre la rotation R calculée et la vérité terrain R_{gt} :

$$e_R = \angle(R^\top R_{\text{gt}}) = \cos^{-1}\left(\frac{|\operatorname{Tr}(R^\top R_{\text{gt}}) - 1|}{2}\right) \tag{5.46}$$

- L'erreur en translation e_t qui représente la différence angulaire entre la direction de la translation t calculée et la verité terrain t_{gt} :

$$e_t = \angle(t, t_{\text{gt}}) = \cos^{-1}\left(\frac{|t^\top t_{\text{gt}}|}{\|t\|_2 \, \|t_{\text{gt}}\|_2}\right) \tag{5.47}$$

5.5.1 Méthodes comparées

Nous avons comparés diverses méthodes dans le but d'être le plus complet possible. Certaines utilisent uniquement des points ou uniquement des segments, d'autres utilisent à la fois les segments et les points ; enfin nous avons utilisé une méta-méthode à base de points de fuite :

- **PdF** : cette méta-méthode à base de point de fuite ne calcule que la rotation relative. Son but est de fournir une référence sur la précision des points de fuite. Nous considérons deux appariement de points de fuite et calculons la rotation associée (voir Section 5.3). Les appariements de points de fuite sont obtenus à l'aide des appariements de segments (voir section 3.7.3). Pour obtenir une borne supérieure à toute méthode utilisant les points de fuite, nous avons considéré toutes les rotations obtenues à partir de deux appariements de points de fuite et conservé la meilleure rotation selon la vérité terrain (fournie lors du calcul de cette méta-méthode),
- **5-point** : une version *a contrario* de la méthode d'estimation de matrice essentielle à 5 points [61]. Moulon *et al.* [58] montrent qu'elle donne de meilleurs résultats qu'avec un RANSAC usuel,
- **4-point** : une version *a contrario* de la méthode d'estimation d'homographie suivie d'une décomposition en rotation et translation [55]. Cette méthode est censée mieux se comporter sur des scènes planaires,
- **3-line** : la méthode d'Elqursh *et al.* [14] permettant d'estimer la rotation à partir d'un triplet de ligne localement *Manhattan*,
- **2x2-line** : notre méthode d'estimation de rotation à partir de paires de lignes parallèles (voir Section 5.3),
- **mixte** : notre méthode à base de lignes et de points avec seuil à fixer, utilisée comme référence pour la comparaison avec la version *a contrario* (voir Section 5.4.1),
- **AC-mixte** : notre version *a contrario* de la méthode mixte point-ligne (voir Section 5.4.2).

5.5.2 Paramètres

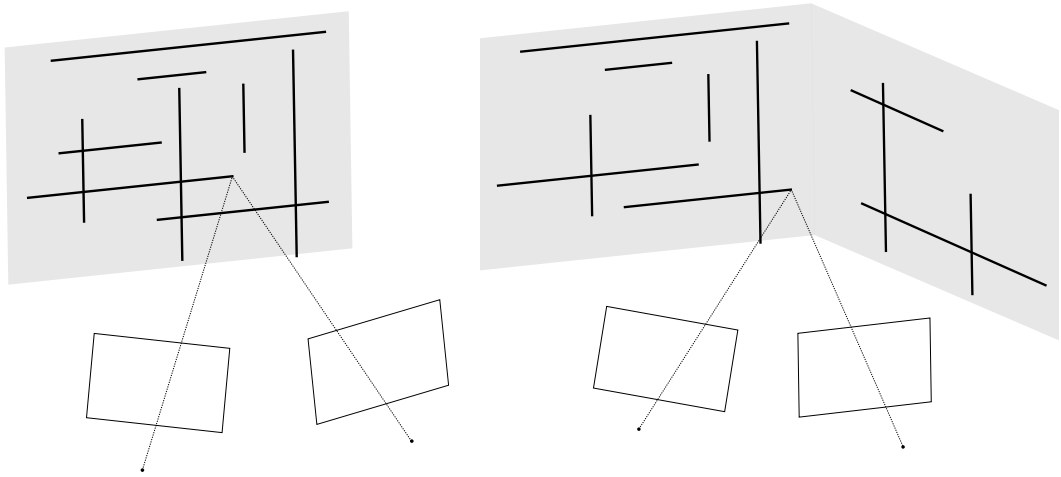
Dans toutes nos expériences, les paramètres sont fixés comme suit :

- Après détection, les points de fuite ainsi que les segments associés sont fusionnés lorsque la direction 3D correspondante est la même à 5° près. Cela permet d'éviter les cas dégénérés où deux paires de lignes parallèles appartiennent à des points de fuite différents mais sont en fait quasi-parallèles.
- Dans le cas des méthodes de RANSAC mixte, nous utilisons un seuil angulaire de 2°. Ce seuil est d'ailleurs le même que celui utilisé pour la méthode 3-line [14].
- Le nombre d'itération de RANSAC est calculé avec un niveau de confiance de $p = 95\%$ et en supposant un ratio d'au moins $w = 30\%$ d'*inliers* parmi les variables. Comme nous utilisons au plus 6 variables pour estimer un modèle (selon les méthodes utilisées), nous devons donc réaliser au moins $N_{\min} = \frac{\log(1-p)}{\log(1-w^n)} \approx 4108$ itérations. Nous arrondissons alors ce nombre d'itération à 5000 pour chaques méthodes.

5.5.3 Expériences sur données synthétiques

Pour les expériences synthétiques, sauf précision contraire, les détections et les appariements de points et de segments sont fournis directement à partir de la vérité terrain.

Données expérimentales



Cas Planaire Cas Manhattan ou Quasi-Manhattan

FIGURE 5.8 – Illustration des données synthétiques utilisées pour les expériences. Les cas les plus fréquents de la reconstruction d'intérieur sont proposés pour étudier le comportement des méthodes face au bruit et comparer les différentes précisions obtenues.

Nous avons considéré trois cas pour étudier les comportements de chacune des méthodes (voir Fig. 5.8). Celles-ci définissent différentes contraintes sur les directions et placements des *features* dans la scène synthétique :

- **Cas Planaire (P)** : la scène est constituée d'un seul plan et de deux directions orthogonales
- **Cas Manhattan (M)** : la scène est constituée de deux plans orthogonaux et de trois directions orthogonales deux à deux.
- **Cas Quasi-Manhattan (QM)** : la scène est constituée de deux plans avec un angle de 120°. Les directions d_1, d_2, d_3 sont telles que la direction verticale soit orthogonale aux deux autres (*i.e.* $d_1 \perp d_2, d_3$) et les deux autres appartiennent chacune à un des plans (*i.e.* $\widehat{d_2, d_3} = 120°$).

Nous avons utilisé des paramètres proches de ceux du dataset Office (voir section 5.5.4) pour générer les scènes (*i.e.* image de taille 15 Mpixel, scène de 2m sur 2m, même matrice de calibration interne K, angle de 45° entre les 2 caméras, distance scène-caméra de 2m, ...).

Le but de ces expériences est principalement de comparer les méthodes 2x2-line et 3-line pour l'estimation de rotation uniquement. En effet, comparer des méthodes de points et de lignes sur des données synthétiques nous a semblé peu instructif car les modèles de bruit appliqués aux segments et aux points ne sont *a priori* pas comparables.

Influence du bruit

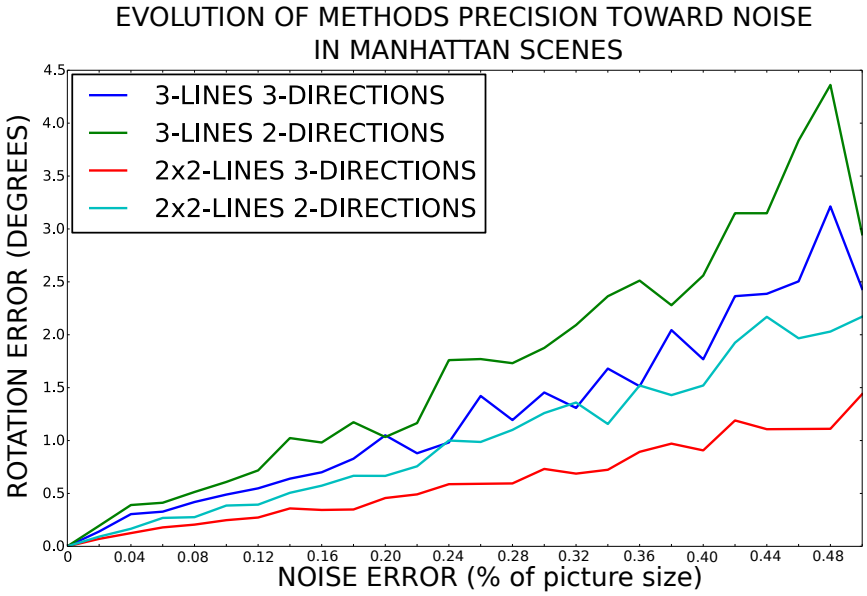


FIGURE 5.9 – Influence du bruit pour les méthodes 2x2-line et 3-line dans les cas planaire et Manhattan. La variance du bruit est indiquée en pourcentage de la taille de l'image et l'erreur en rotation est indiquée en degré.

Dans ces expériences, nous n'avons considéré que les cas planaire et Manhattan pour ne pas introduire de biais entre les deux méthodes. Nous avons générés 100 segments 3D aléatoires suivant le modèle de la scène avec une répartition uniforme dans les plans et selon les 2 ou 3 directions principales. Nous avons ensuite projeté les segments sur chacune des caméras puis ajouté un bruit Gaussien d'écart type σ et de moyenne nulle sur les extrémités. Notons que le bruit est différent pour les deux projections. Nous avons enfin généré 100 scènes suivant ce modèle et calculé la moyenne de l'erreur obtenue.

Nous avons étudié le comportement de l'erreur en fonction des différentes méthodes en faisant varier σ entre 0% et 0.5% de la taille de l'image (soit ici entre 0 et 20 pixels).

Les résultats sont affichés sur la Figure 5.9. Nous nous apercevons alors que la variété des directions principales permet aux deux méthodes de fournir de meilleurs résultats. De plus, malgré le fait que l'orthogonalité soit un critère que seule la méthode 3-line exploite, notre méthode 2x2-line donne toujours de meilleurs résultats.

Nous avons également mesuré l'impact du bruit sur notre méthode finale et plus particulièrement selon le ratio points/segments. Pour cela nous avons appliqué le même bruit de détection sur les extrémités des segments et sur les points.

Les résultats sont affichés sur la Figure 5.10 où on peut observer que les points sont plus précis que les segments. Notons cependant qu'appliquer la même erreur sur les points et les

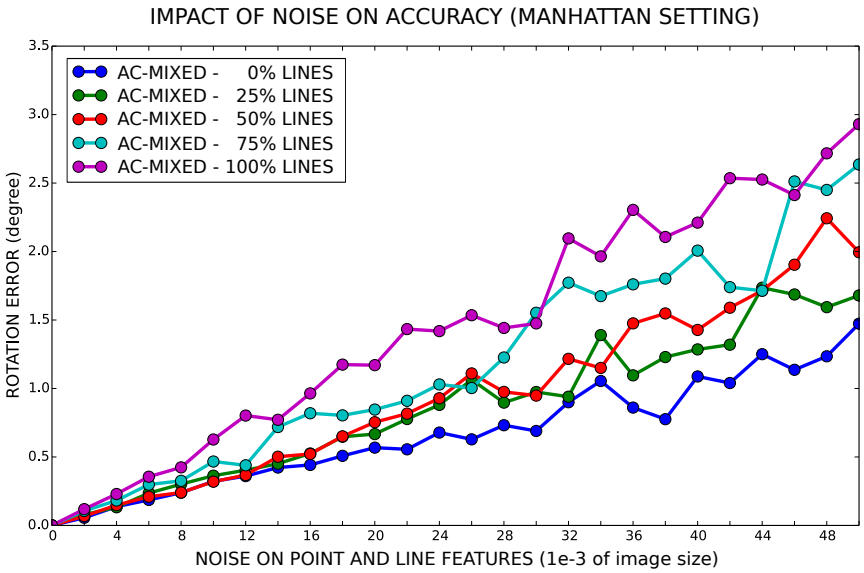


FIGURE 5.10 – Influence du bruit pour la méthodes AC-mixte. La variance du bruit est indiquée en pourcentage de la taille de l'image et l'erreur en rotation est indiquée en degré.

extrémités des segments n'a pas forcément de sens physique et pourrait expliquer l'écart de précision entre les deux types de *features*.

Influence du cadre non-Manhattan

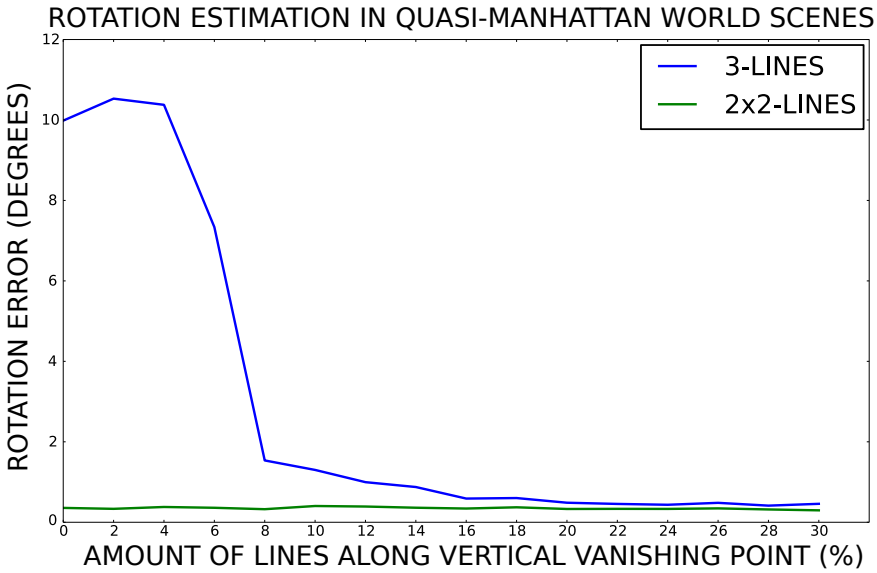


FIGURE 5.11 – Influence du cadre non strictement Manhattan pour les méthodes 2x2-line et 3-line. La méthode 3-line est fortement affectée en présence de peu de triplets orthogonaux mais obtient des résultats corrects dès que leur proportion dépasse un certain seuil. Comme prévu, notre méthode n'est pas affectée par la présence ou non d'un cadre Manhattan.

Nous avons également voulu mesurer l'impact des hypothèses contraignantes de la méthode

3-line. Pour cela, nous avons considéré le cas Quasi-Manhattan. Nous avons utilisé les mêmes paramètres que précédemment mais nous avons fixé cette fois la variance du bruit de reprojection à 0.2% et nous avons fait varier le ratio d'apparition de lignes le long de la direction principale commune (verticale). Ainsi, à 0% d'apparition, il n'y a aucun triplet vérifiant les contraintes de la méthode 3-line, et elle ne peut donc pas trouver la bonne rotation.

Notons également que cette fois, les triplets ont été générés à l'aide de la méthode originale (voir section 5.2.2) et l'information des directions n'a pas été fournie. De plus la méthode 2x2-line n'est là que pour servir de référence puisqu'elle n'est pas censée être affectée par cet aspect là ce qu'on peut vérifier expérimentalement.

Les résultats sont affichés sur la Figure 5.11. On peut alors voir qu'un trop faible ratio de lignes orthogonales affecte énormément la méthode 3-line tandis que 2x2-line reste inchangée. Le graphique montre de bons résultats dès que le ratio de lignes verticales dépasse 10%. Cependant, même si ce seuil peut sembler faible, il apparaît alors que la méthode 3-line est beaucoup moins robuste à de mauvaises détections et/ou mauvais appariements dans un cadre quasi-Manhattan ou non-Manhattan (voir section 5.5.4).

5.5.4 Expériences sur données réelles

Dans chacune des expériences sur des données réelles présentées ci-dessous, les points ont été détectés et mis en correspondance avec SIFT [46]. Les segments ont été détectés avec MLSD [68] et mis en correspondance avec LBD [84]. Les points de fuite ont été détectés à l'aide du détecteur d'Almansa *et al.* [3]. Enfin, la partie raffinement non linéaire a été réalisée à l'aide de la librairie CERES [1].

Présentation des données

Nous avons utilisés différents types de données réelles dans le but d'avoir des résultats les plus représentatifs possible. Pour cela nous avons utilisé :

- **Office** : une scène d'intérieur avec peu de texture, peu de recouvrement et respectant à peu près les contraintes de type *Manhattan world*.
- **Strecha** : plusieurs scènes extérieures très texturées avec un grand recouvrement et respectant généralement les contraintes de type *Manhattan world*. Cet ensemble de données est celui de Strecha *et al.* [73].
- **Building** : un bâtiment en forme de V assez texturé avec peu de recouvrement, exemple typique d'une scène urbaine ne vérifiant pas les contraintes de type *Manhattan-world*.
- **Car** : vue rapprochée d'une voiture dans la rue. Une scène relativement texturée mais avec peu de lignes parallèles et/ou orthogonales et un recouvrement limité.

Le tableau 5.2 reprend les différents jeux d'images en précisant les critères associés.

Précision des intersections

Elqursh *et al.* [14] présentent une méthode n'utilisant que des lignes, l'intérêt étant de définir une méthode se passant totalement de points pour réussir à calibrer là où les détections sont rares voire absentes. En pratique, pour les scènes utilisées, nous obtenons toujours au moins quelques appariements de points. C'est pourquoi nous avons voulu comparer la précision des intersections face à celle des points dans des scènes avec peu de détections et d'appariements (en pratique nous avons utilisé le *dataset* Office).

Nous avons donc comparé la méthode 3-line en utilisant d'une part la méthode originale (*i.e.* avec les intersections de lignes, voir section 5.2.3) et d'autre part une version modifiée de 3-line utilisant les détections et appariements de SIFT [46] au lieu des intersections de lignes.

FIGURE 5.12 – Quelques exemples des images des différents jeux de données. De haut en bas, *Office*, *Strecha* [73], *Building* et *Car*.

Dans les deux cas, la rotation est estimée de la même façon (*i.e.* à l'aide des triplets de lignes), seule l'estimation de translation est changée.

Les résultats sont affichés sur la Figure 5.13. Ils montrent que le choix des intersections pour le calcul de la translation n'est pas optimal et que même dans des scènes où les appariements SIFT ne sont pas suffisants pour calibrer seuls, ils permettent une estimation de translation bien plus précise une fois la rotation calculée.

Influence du cadre non Manhattan

Nous reprenons avec des données réelles certains des expériences synthétiques faites à la section 5.5.3. Pour cela, nous avons considéré les deux *datasets* qui ne vérifient pas les conditions *Manhattan-world* ; *Building* et *Car*. *Building* correspond à un cadre *quasi-Manhattan* puisqu'il contient des directions parallèles et orthogonales permettant la création de triplets pour la

Critère \ *Dataset*	*Office*	*Strecha*	*Building*	*Car*
Scène planaire	✓	✗	✓	✗
Peu de texture	✓	✗	✓	✗
Recouvrement limité	✓	✗	✗	✗
Manhattan-World	✓	✓	✗	✗

TABLE 5.2 – Tableau récapitulatif des différentes caractéristiques de chacun des *datasets* en lien avec le tableau 5.1. Notons que même si *Building* est peu texturé et relativement planaire, son recouvrement est suffisamment important pour générer de nombreux appariements de points répartis sur toute la scène. De plus, l'environnement aide également les méthodes de calibration à base de points.

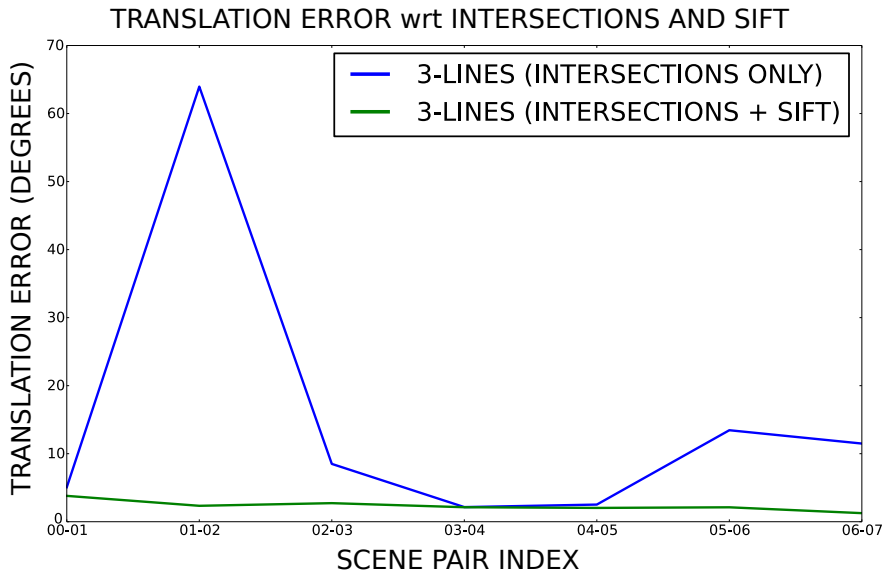


FIGURE 5.13 – Estimation de translation en utilisant les intersections de lignes et/ou les quelques appariements SIFT [46] disponibles quand il y en a. Même si les appariements SIFT ne sont pas suffisants pour calibrer les scènes d'Office, ils permettent une estimation plus précise de la translation une fois la rotation calculée.

méthode 3-line. Cependant, il contient également de faux triplets qui pourraient être détectés comme étant localement *Manhattan* alors qu'ils ne vérifient pas les conditions en pratique. *Car* ne contient presque pas de segments parallèles et encore moins de segments orthogonaux.

Les résultats sont rassemblés sur la Figure 5.14. On peut constater que la méthode 3-line n'est effectivement pas robuste à des scènes ne vérifiant pas pleinement les conditions *Manhattan-world*. Pourtant, ce type de scène est relativement fréquent dans les zones urbaines, qui justement privilégient la reconstruction à base de lignes. De plus, les résultats de *Car* montrent que la méthode 2x2-line, même si elle perd un peu en précision par rapport à *Building*, conserve un résultat correct. Ainsi, même la présence de peu de lignes parallèles peut suffire à 2x2-line pour obtenir une rotation relative fiable de la scène. Notons cependant que pour ce type de scène, il est plus approprié d'utiliser une méthode à base de point ou mixte, mais nous voulions tester les limites de 2x2-line.

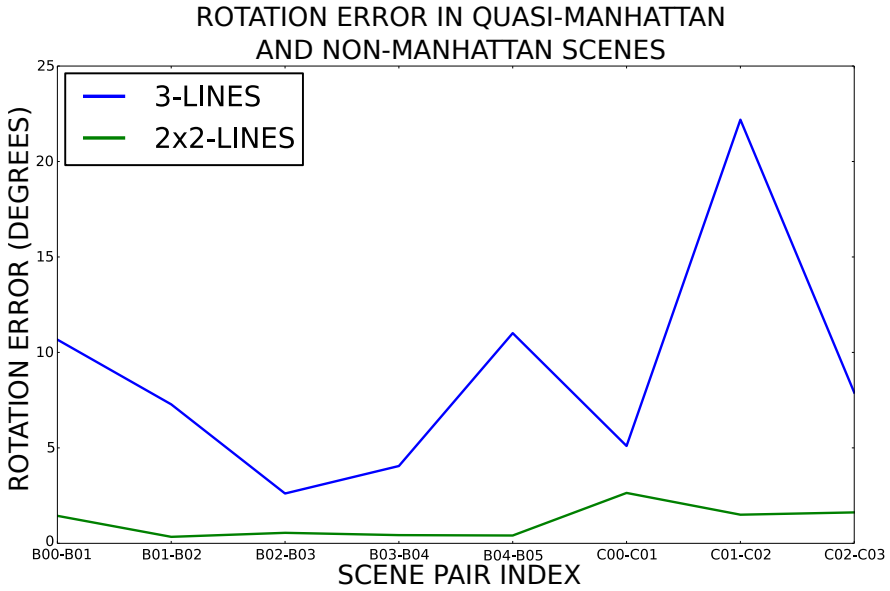


FIGURE 5.14 – *Building* correspond à une scène urbaine typiquement quasi-Manhattan tandis que *Car* correspond à une scène urbaine peu fournie en ligne. Nous pouvons ainsi tester les limites des deux méthodes 2x2-line et 3-line pour l'estimation de rotation dans des cadres difficiles.

Raffinement non linéaire

Nous avons comparé ici les différentes méthodes de raffinement non linéaire : celle présentée par Elqrush *et al.* [14] (voir section 5.2.4), dont nous avons présenté les défauts précédemment, et notre méthode de raffinement mixte point-ligne (voir section 5.4.3). L'idée principale est que le raffinement de 3-line considère surtout les points tandis que le raffinement de 2x2-line trouve un juste milieu. Pour cela nous avons comparé les méthodes 3-line et 2x2-line avec et sans raffinement. De plus, nous avons affiché les résultats de 5-point comme témoin de la précision des points pour chaque scène. Enfin, nous avons considéré les *dataset Office* et *HerzJesuP8* (de *Strecha* [73]) pour comparer les comportements des raffinements dans une scène privilégiant les lignes puis une scène privilégiant les points.

Les résultats sont rassemblés sur la Figure 5.15. Comme on peut le voir dans les graphiques, le raffinement de la méthode 3-line tend à améliorer la précision lorsque la méthode 5-point fonctionne (en particulier dans la scène *HerzJesuP8* où l'amélioration est flagrante). A l'inverse, pour les paires d'images où 5-point n'arrive pas à calibrer, le raffinement détériore la solution (voir dans *Office* et notamment les paires 3-4 et 5-6). Le raffinement d'Elqrush *et al.* [14] n'utilisent donc quasiment que l'information fournie par les points, ce qui la rend moins robuste, tandis que notre méthode utilise les deux types de *features* pour améliorer la précision du modèle trouvé.

Méthode mixte avec RANSAC ou AC-RANSAC

Nous avons aussi comparé les deux méthodes mixtes en détail. En effet, la méthode mixte RANSAC peut donner de bons résultats si le seuil est bien choisi. Les questions qui viennent alors à l'esprit sont de savoir, d'une part, si ce seuil est souvent le même d'une scène à l'autre et, d'autre part, comment sont situés les résultats optimaux du RANSAC par rapport à ceux fournis par AC-RANSAC. Pour cela nous avons comparés les résultats des deux méthodes (avant

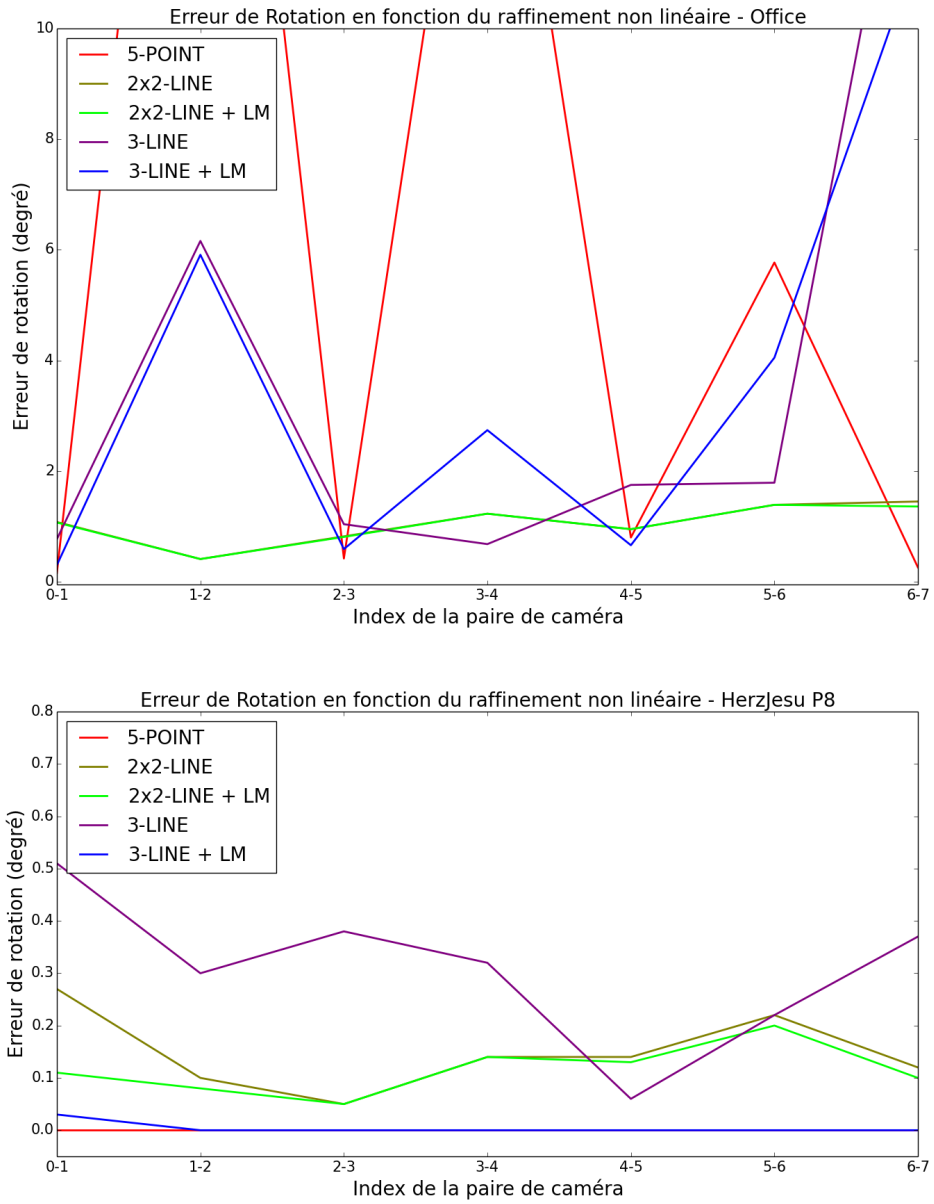


FIGURE 5.15 – Comparaison du raffinement non-linéaire (LM) utilisé par Elqrush *et al.* [14] et notre version. La méthode 5-point est utilisée comme témoin. On voit alors que le raffinement de 3-line tend à améliorer le résultat quand 5-point fonctionne correctement et à le détériorer sinon.

raffinement) sur les *datasets Office* et *CastleP18*. Le seuil τ du RANSAC a été testé sur plusieurs valeurs dans le but d'avoir une estimation du meilleur résultat obtenable par RANSAC. Les résultats sont affichés dans le Tableau 5.3.

On peut ainsi remarquer que les résultats du RANSAC dépendent fortement du seuil τ choisi. Le seuil utilisé entre les deux images est assez similaire (proche de $.02 \sim .05°$) cependant un choix de l'un ou l'autre pour les deux scène ne permet pas d'obtenir un résultat en moyenne aussi bon que celui d'AC-RANSAC. En effet, même si AC-RANSAC n'obtient pas toujours le meilleur résultat possible en terme de précision, il reste toujours proche de celui-ci. En moyenne, sur les deux scènes, il reste plus performant et a l'avantage de ne pas nécessiter un choix d'un

Dataset \ Seuil		AC	$\tau = 2°$	$\tau = 1°$	$\tau = .5°$	$\tau = .2°$	$\tau = .1°$	$\tau = .05°$	$\tau = .02°$	$\tau = .01°$
Office	e_R	**0.50**	1.79	1.41	0.97	0.85	0.67	0.55	1.23	0.91
	e_t	**0.85**	7.66	5.11	2.32	1.42	1.17	1.11	1.71	2.41
CastleP19	e_R	0.13	1.96	0.95	0.56	0.32	0.26	0.16	**0.09**	0.12
	e_t	0.53	11.88	4.22	2.77	1.36	0.91	0.60	**0.30**	0.37

TABLE 5.3 – Comparaison des méthodes RANSAC classique et *a contrario* (AC). Nous avons représenté en **gras** le meilleur résultat toutes méthodes confondues et en vert le meilleur résultat obtenu par le RANSAC classique.

seuil τ arbitraire.

Robustesse et précision des méthodes d'estimation de rotation

Pour comparer de manière équitable l'intérêt d'un *feature* pour un certain type de scène (*e.g.* peu texturé ou non, *Manhattan-world*, ...) nous ne pouvons utiliser que les méthodes d'estimation de rotation. En effet, il n'est pas possible d'estimer une translation autrement qu'avec des points et dans le cas des lignes ou des points de fuite, son calcul est dépendant du résultat de rotation. Nous avons donc mesuré les erreurs de rotation estimées à partir des différents *features* sur chacune des scènes. Les résultats sont affichés dans le Tableau 5.4.

Dataset \ Méthode	Points		Lignes		Points de fuite
	5-point	4-point	3-line	2×2-line	PdF
Strecha	**0.05**	1.86	0.35	0.28	1.29
Office	8.63	25.37	3.93	1.05	**0.65**
Building	**0.35**	1.00	8.04	0.56	0.54
Car	**0.23**	3.19	24.27	2.41	14.93

TABLE 5.4 – Comparaison des rotations obtenues par différentes méthodes en fonction des *features* utilisés (points, lignes ou points de fuite). Nous avons représenté en **gras** le meilleur résultat toutes méthodes confondues et en rouge les résultats incorrects dépassant le seuil de 5°.

Comme on peut le voir dans le tableau, les points sont les plus précis des *features* étudiés, souvent avec une forte marge. Cependant, dans certaines scènes difficiles, les points ne sont pas capables d'estimer correctement la matrice essentielle. A l'inverse, les méthodes de lignes sont plus robustes dans les scènes peu texturées. De plus, notre méthode 2×2-line, n'utilisant pas de contraintes de type *Manhattan-world* fonctionne correctement sur toutes les scènes étudiées. Notons également que notre méthode est également plus précise que 3-line dans toutes les scènes. Enfin, les méthodes de points de fuite obtiennent des résultats très variables ; parfois très précis, parfois totalement faux. Cette méthode a été utilisée comme référence face aux méthodes de lignes. Elle montre ainsi que l'utilisation de points de fuite est plus instable que l'utilisation de lignes. En effet, cette méthode réduit une grande quantité d'information fournie par les lignes en une plus faible moyenne fournie par les points de fuite et estime ensuite la rotation. A l'inverse, notre méthode cherche à utiliser toute l'information fournie par les lignes pour estimer la rotation.

Robustesse et précision des méthodes d'estimation de pose relative

Nous avons comparé les principales méthodes étudiées pour l'estimation de pose complète. La comparaison se fait ici sur la base des erreurs en rotation et en translation (toutes deux en degré, voir le début de la section 5.5). Les résultats sont affichés dans le Tableau 5.5.

Dataset \ Méthode		5-point	3-line	2×2-line	mixte	AC-mixte
Strecha	e_R	**0.02**	0.05	0.25	0.19	**0.02**
	e_t	**0.18**	0.36	1.03	0.80	0.21
Office	e_R	6.88	3.67	1.03	1.01	**0.39**
	e_t	27.19	16.26	3.26	3.13	**0.77**
Building	e_R	0.23	3.73	0.49	0.24	**0.21**
	e_t	**0.31**	18.72	1.57	0.83	0.45
Car	e_R	**0.19**	13.81	2.37	0.75	0.24
	e_t	**0.20**	69.47	18.03	0.89	0.28

TABLE 5.5 – Comparaison des poses obtenues par les différentes méthodes. Nous avons représenté en **gras** le meilleur résultat toutes méthodes confondues et en rouge les résultats aberrants dépassant le seuil de 5°.

Les résultats obtenus confirment les éléments recueillis précédemment :

- La méthode 5-point permet d'obtenir des résultats très précis. Cependant, elle n'est pas robuste aux scènes d'intérieurs cumulant un manque de texture, un faible recouvrement des images et des cas planaires critiques.
- La méthode 3-line est robuste en intérieur mais obtient de moins bons résultats que 2×2-line en particulier lorsque les hypothèses *Manhattan-world* ne sont pas vérifiées.
- Les méthodes mixtes arrivent à combiner les avantages des points (précision) à ceux des lignes (robustesse) pour obtenir de bons résultats partout. La version *a contrario* obtient même des résultats comparables à 5-point tout en réussissant à calibrer dans les scènes d'intérieur.

Utilité des lignes et points dans les méthodes mixtes

Nous avons voulu jauger l'utilité des lignes et des points dans les méthodes mixtes. Les deux *features* sont utilisés de manière combinée dans le calcul d'*inliers* mais leur utilisation est bien séparée dans le cadre de la génération de modèle (*i.e.* couple (R,t) pour les lignes, et E pour les points). Nous avons donc calculé le ratio d'hypothèses conservées pour les lignes et pour les points en fonction des scènes.

Les résultats sont affichés dans le Tableau 5.6. Mis à part dans *Car* où les lignes sont fortement désavantagées, les deux types de *feature* sont utilisés pour générer des modèles. On remarque également que la méthode AC-mixte a plus tendance à sélectionner des hypothèses de points ce qui rejoint les observations faites précédemment. En effet, les points sont généralement plus précis que les lignes et l'utilité des lignes est surtout la robustesse. Or cet aspect se reflète plus dans le score du NFA ou le nombre d'*inliers* que dans la génération du modèle. Cependant, malgré leur moins bonnes précisions, les lignes restent utiles pour générer des modèles.

Méthode / *Dataset*	mixte		AC-mixte	
	lignes	points	lignes	points
Strecha	**54%**	46%	10%	**90%**
Office	**63%**	36%	34%	**66%**
Building	**70%**	30%	20%	**80%**
Car	0%	**100%**	0%	**100%**

TABLE 5.6 – Ratio d'hypothèses conservées dans les méthodes mixtes. Nous avons représenté en **gras** le *feature* générant le plus souvent les meilleures hypothèses (*i.e.* . retenu par RANSAC ou AC-RANSAC pour fournir le modèle final).

5.6 Limites & Conclusion

Dans ce chapitre, nous avons introduit une nouvelle méthode de calibration bifocale basée sur les lignes. Contrairement à la méthode 3-line [14], elle ne nécessite pas d'hypothèses de type *Manhattan-world*. De plus, contrairement aux méthodes trifocales plus fréquemment utilisées, elle ne requiert pas un recouvrement de trois vues et ne nécessite que 2 appariements de couples de lignes parallèles et 2 de points, contrairement aux 13 requis par les méthodes trifocales [24].

Nous avons également défini une méthode permettant de combiner les points et les lignes en conservant l'avantage de chacun (voir Tab. 5.7). En particulier, nous avons développé une méthode de raffinement utilisant ces deux types de *features* sur un même plan d'égalité. De plus, nous avons ajouté un cadre *a contrario* aux méthodes mixtes permettant ainsi d'obtenir une méthode précise qui ne nécessite pas de paramètres. Nous avons finalement comparé les différentes méthodes existantes, sur de nombreuses données de type différent pour appuyer nos théories.

	Méthode / Critère	Points			Lignes		Mixte
		Essentielle [61]	Homographie [24]	Trifocal [24]	Bifocal [14]	Trifocal [85]	Point-Ligne [69]
	Scène planaire	✗	✓	✓	✓	✓	✓
	Peu de texture	✗	✗	✗	✓	✓	✓
	Recouvr. limité	✗	✗	✗	✓	✗	✓
M-W	✓	✓	✓	✓	✓	✓	✓
M-W	✗	✓	✓	✓	✗	✓	✓

TABLE 5.7 – Les méthodes mixtes, en combinant les avantages de points et des lignes, sont capables d'obtenir des calibrations très précises à la fois dans des scènes d'intérieur complexes pour les points mais aussi dans des scènes ne présentant aucune lignes.

Cependant, la méthode proposée a encore quelques limitations. L'hypothèse principale est la présence de lignes parallèles. Si cette hypothèse est vérifiée dans la plupart des scènes urbaines, elle rend néanmoins la méthode sensible aux détections incorrectes de points de fuite. De plus, cette méthode nécessite dans tous les cas la présence de points pour le calcul de la translation. Les points étant difficilement détectés et mis en correspondance dans les scènes d'intérieur avec peu de recouvrement, les translations obtenues peuvent être de moins bonne qualité que dans les scènes bien texturées.

Chapitre 6

Calibration multi-vue sans contraintes trifocales

Passer d'une calibration bifocale ou trifocale à une calibration à n vues a fait l'objet de nombreuses recherches. Il existe déjà de multiples méthodes pour traiter le sujet, qui ont toutes un point commun : l'utilisation de triplet de points et/ou de segments c'est à dire de points ou de segments vus simultanément dans au moins 3 images. Cette utilisation nécessite alors un recouvrement suffisamment important entre les images successives pour permettre d'observer ces triplets de *features*. Et si cette hypothèse est vérifiée dans de nombreux cas (*e.g.* prises de vues rapprochées, présence de texture qui augmente la densité des points, ...), elle l'est plus difficilement en intérieur, et en particulier avec des photographies prises par des personnes qui ne sont pas expertes du sujet. En effet, les scènes d'intérieur sont pauvres en texture et les changements de points de vue suffisamment brusques pour limiter le recouvrement entre images successives. Il est donc tout à fait envisageable d'obtenir un ou plusieurs triplets de caméras sans *features* en commun.

Nous présentons, dans ce chapitre, une méthode de calibration multi-vue qui ne nécessite pas la présence de triplets pour fonctionner. Nous montrons qu'en utilisant des contraintes de coplanarité observées partiellement par les différentes caméras, il est possible d'obtenir les informations d'échelles nécessaires à la calibration multi-vue. Nous proposons ensuite comment combiner ces informations à celle de potentiels triplets pour obtenir des résultats les plus précis possible. Enfin nous nous comparons à diverses méthodes de l'état de l'art sur des scènes d'intérieur pour montrer la robustesse de notre méthode ainsi que des scènes plus classiques pour comparer leurs précisions.

Contents

6.1 Méthodes existantes

Il existe de nombreuses méthodes permettant de transformer les poses relatives obtenues avec 2 ou 3 vues en un ensemble de poses globales dans un même repère et permettant la reconstruction 3D d'une scène. Cela recouvre généralement parmi les méthodes *incrémentales*, *hiérarchiques* ou *globales*.

Les méthodes incrémentales [72, 58, 81] initialisent la scène à l'aide de 2 ou 3 vues puis ajoutent les caméras suivantes en faisant des comparaisons 2D-3D. Ces méthodes utilisent les algorithmes de *Perspective-n-Points* (PnP) [40, 18] pour calculer la pose d'une caméra à partir des correspondances entre points 3D de la scène déjà calculée et observations 2D de la nouvelle caméra. De telles méthodes nécessitent au moins 3 correspondances 2D-3D et certaines en nécessitent même 6 [91]. La présence d'autant de correspondances implique donc l'existence de suffisamment de points observés par 3 caméras ou plus.

Les méthodes hiérarchiques, qui fusionnent des modèles partiels, nécessitent également de telles contraintes. Dans [26], les deux modèles se recoupent dans le sens où ils partagent des images en commun. Des paires de points 3D se reprojetant sur les mêmes observations sont alors utilisés pour relier les deux modèles, ce qui implique la présence de points observés dans au moins 3 caméras. Dans [76], les modèles partiels sont reconstruits à partir d'ensemble disjoints d'images. Ils sont ensuite fusionnés à l'aide de 4 points 3D en commun ce qui implique la présence de points vus par au moins 4 caméras. Enfin, certaines méthodes [17] utilisent des contraintes plus faibles, ne nécessitant pas que les points soient reconstruits dans les deux scènes. Cependant, la visibilité par au moins 3 caméras est toujours requise.

Les méthodes globales utilisent les informations bifocales pour obtenir un graphe épipolaire satisfaisant au mieux ces contraintes. Dans ce cas, les translations ne sont connues qu'à une échelle près et celles-ci sont alors calculées à l'aide des supposées nombreuses connexions dans le graphe. Ainsi, certaines méthodes reposent sur la redondance d'information [21, 11], d'autres sur une information fournie par des tenseurs trifocaux [71, 59] et nécessitent donc la présence d'au moins 4 points observés par 3 caméras. Un certain nombre d'autres méthodes [34, 67, 4] calculent les translations globales en résolvant des équations utilisant des points observés par deux caméras. Cependant, ils supposent implicitement qu'il existe suffisamment de points observés par 3 caméras pour obtenir les facteurs d'échelles communs entre toutes les caméras.

Dans le cadre des scènes d'intérieur, la présence de telles correspondances est assez limité. Ainsi, si le seuil de 3 correspondances au minimum est souvent vérifié, leur nombre ne permet pas toujours une estimation précise. De plus, comme nous le verrons par la suite, il est également possible d'obtenir des triplets d'images consécutives sans aucun *feature* en commun.

Pour pallier cette absence de triplets de points, certaines méthodes vont chercher du côté des lignes pour obtenir une reconstruction correcte.

Certaines méthodes de reconstruction [28, 42] utilisent les lignes plutôt que les points pour reconstruire la scène. Les résultats obtenus sont généralement plus intéressants dans le cadre des scènes urbaines. Cependant ces méthodes utilisent une calibration multi-vue faite à l'aide de points pour calculer les appariements de lignes puis les reconstruction 3D. Elles ne sont donc pas applicables en pratique dans les scènes d'intérieur où les points ne permettent pas une calibration correcte.

Certaines méthodes incrémentales utilisent les lignes plutôt que les points pour la calibration. Les algorithmes de PnP deviennent alors des algorithme de *Perspective-n-Lines* (PnL) [51, 86] mais requièrent toujours la présence d'au moins 3 correspondances 2D-3D. Ce minimum de 3 passe parfois à 6 pour limiter la sensibilité au bruit [51] voire à 9 requis et 25 souhaités pour plus de précision [65]. Enfin, même en utilisant à la fois les lignes et les points dans des algorithmes *Perspective-n-Features* (PnF), un minimum de 3 correspondances est toujours requis [66, 82, 47]. Les problèmes causés par l'absence de triplets de points ne sont pas toujours compensés

par l'utilisation de lignes car celles-ci étant plus difficiles à apparier, le nombre de triplets est toujours aussi réduit.

Plus récemment, Kim *et al.* [35] et Ikehata *et al.* [31] ont proposé une méthode de calibration multi-vue à partir de contraintes de coplanarités entre segments pour permettre des reconstructions en intérieur. Cependant, la seconde méthode nécessite un procédé très particulier pour prendre les photos : l'utilisateur doit se placer au centre de la pièce et prendre plusieurs centaines de photos en tournant sur lui-même. De plus, les deux méthodes utilisent pleinement des hypothèses de type *Manhattan-world* et ne peut donc se généraliser facilement à n'importe quelle scène d'intérieur.

6.2 Des poses relatives aux poses globales

Comme nous l'avons vu précédemment, la calibration multi-vues est réalisée en plusieurs étapes. Nous nous plaçons dans le cadre d'une chaîne globale et la première étape est donc l'obtention du graphe épipolaire contenant les transformations relatives entre caméras. Nous supposons ici que nous avons déjà accès au graphe épipolaire, celui-ci pouvant être obtenu à partir de n'importe quelle méthode de calibration bifocale (dans notre cas, nous utiliserons la méthode du chapitre précédent [69]).

Nous supposons que ce graphe est connexe mais, contrairement aux méthodes usuelles, nous ne supposons pas qu'il existe des contraintes trifocales pour chaque caméras. En effet, dans le cadre des scènes d'intérieurs, les images sont généralement pauvres en texture. De plus, la proximité de la scène avec la caméra entraîne souvent une prise de photo espacée et donc de forts changements de points de vue entre photos. Ces deux aspects combinés font qu'il y a généralement peu de *features* en commun d'une scène à l'autre, et *a fortiori* encore moins pour des triplets d'images.

À l'aide du graphe épipolaire, nous avons accès, pour une paire de caméras voisines (i,j), à la transformation relative $(R_{i,j}, t_{i,j})$. Notons cependant que $t_{i,j}$ n'est connu qu'à un facteur d'échelle près et nous supposerons donc par la suite que $\|t_{i,j}\|_2 = 1$. Le problème de calibration multi-vue se résume donc, dans notre cas, à la recherche de ces différents facteurs d'échelle que nous noterons $\lambda_{i,j}$.

En notant (R_i, C_i) les poses globales des caméras dans un repère commun, nous avons les relations suivantes :

$$R_j = R_{ij}R_i, \quad T_j = R_{ij}T_i + T_{ij}, \quad T_i = -R^\top C_i, \quad T_{i,j} = \lambda_{i,j}t_{i,j} \tag{6.1}$$

Les poses globales peuvent alors être obtenues à partir des relations précédentes :

$$R_1 = I \quad \text{et} \quad R_{i+1} = R_{i,i+1}R_i \tag{6.2}$$

$$C_1 = 0 \quad \text{et} \quad C_{i+1} = C_i - \lambda_{i,i+1}R_{i+1}^\top t_{i,i+1} \tag{6.3}$$

Comme la pose globale est définie à une échelle près, nous imposons $\lambda_{12} = 1$. Dans le cas où le graphe épipolaire contiendrait de nombreux cycles, il serait intéressant d'utiliser ces contraintes pour obtenir des rotations plus précises.

Dans la suite, nous supposerons la matrice de calibration interne connue et utiliserons donc les coordonnées normalisées (voir section 3.5.1). De plus, par souci de lisibilité, nous noterons un triplet de caméra $1,2,3$ au lieu de $i, i+1, i+2$.

6.2.1 Contraintes de coplanarité

Nous supposons ici que nous avons accès à 2 segments 3D L_a et L_b coplanaires mais non parallèles et nous noterons $\mathcal{P}$ leur plan commun. Nous supposons de plus que L_a n'apparaît que

dans les caméras 1 et 2 tandis que L_b apparaît dans les caméras 2 et 3 (*i.e.* dans ce cas, nous n'avons pas accès à des contraintes trifocales classiques). Nous noterons l_i^j le projeté de L_i sur la caméra j. Ces hypothèses sont illustrées dans la Figure 6.1.

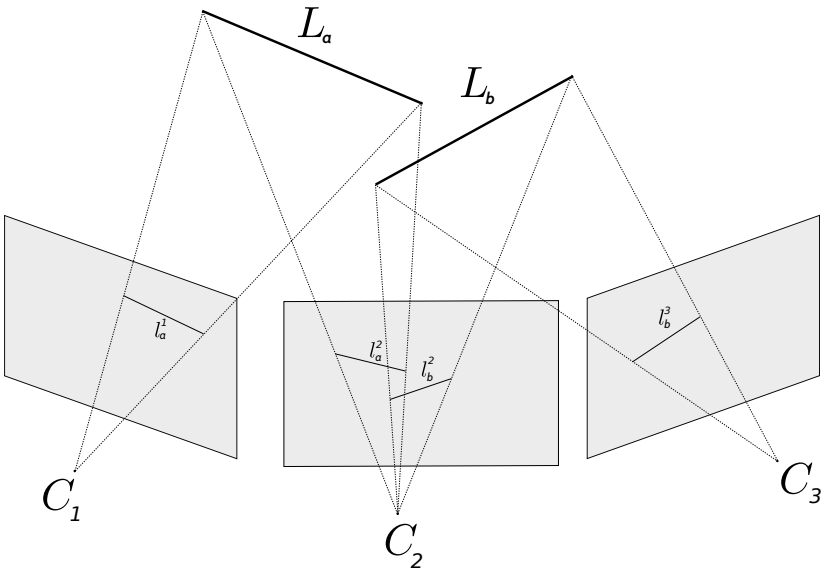


FIGURE 6.1 – Si L_a et L_b sont coplanaires, même en l'absence de contraintes trifocales, nous pouvons situer C_1, C_2, C_3 dans le même repère à partir de paires de projections sur les caméras 1-2 et 2-3.

Nous allons présenter dans la suite comment utiliser des contraintes coplanaires pour estimer le facteur d'échelle. Cette formule repose sur l'utilisation de plusieurs contraintes simples.

Pour un point P quelconque de la scène sur une ligne L_i dans le repère global, ses coordonnées peuvent s'écrire en fonction de sa projection $\mathbf{p}^j$ sur la caméra j et sa profondeur z_p^j :

$$P = R_j^\top (z_p^j \mathbf{p}^j - T_j) \tag{6.4}$$

Comme $P \in L_i$, on obtient également la relation pour n'importe quelle caméra k :

$$\mathbf{l}_i^{k\top} \mathbf{p}^k = \mathbf{l}_i^{k\top} (R_k P + T_k) = 0, \tag{6.5}$$

En utilisant les équations 6.4 et 6.5, il est alors possible d'exprimer la profondeur z_p^j en fonction des projections $\mathbf{l}_i^k, \mathbf{p}^j$ et des informations de poses relatives :

$$\begin{aligned} &\mathbf{l}_i^{k\top} (R_k R_j^\top (z_p^j \mathbf{p}^j - T_j) + T_k) = \mathbf{l}_i^{k\top} (R_{jk} z_p^j \mathbf{p}^j + T_{jk}) = 0 \\ \Leftrightarrow \quad & z_p^j = -\frac{\mathbf{l}_i^{k\top} T_{jk}}{\mathbf{l}_i^{k\top} (R_{jk} \mathbf{p}^j)}. \end{aligned} \tag{6.6}$$

Cette équation est très souvent utilisée en reconstruction 3D puisqu'elle permet de calculer les extrémités des segments que l'on reconstruit [28]. Notons également que cette formule ne sert qu'à reconstruire les segments vus dans deux images uniquement. Dans le cas où le segment est observé par 3 caméras ou plus, il faut utiliser des méthodes de triangulation [6].

Pour $i = a$ ou b, on sait alors que $P \in \mathcal{P}$. Pour n'importe quel point $M \in \mathcal{P}$ et en notant $n_\mathcal{P}$ la normale du plan, on a alors :

$$P \in \mathcal{P} \Leftrightarrow n_\mathcal{P}^\top (P - M) = 0 \tag{6.7}$$

En combinant les équations 6.6 et 6.7, on obtient alors :

$$\begin{aligned} n_{\mathcal{P}}^{\top}(R_j^{\top}(z_p^j p^j - T_j) - M) = 0 \quad &\Leftrightarrow \\ n_{\mathcal{P}}^{\top}(-\frac{l_i^{k\top} T_{jk}}{l_i^{k\top}(R_{jk}p^j)} R_j^{\top} p^j + C_j - M) = 0 \quad &\Leftrightarrow \\ n_{\mathcal{P}}^{\top}(C_j - M) = \frac{(l_i^{k\top} T_{jk})(n_{\mathcal{P}}^{\top}(R_j^{\top} p^j))}{l_i^{k\top}(R_{jk}p^j)} \end{aligned} \tag{6.8}$$

Cette équation fait intervenir deux inconnues $n_{\mathcal{P}}$ et M. Cependant, la connaissance des rotations globales nous donne accès aux directions des segments et ainsi à la normale du plan. En effet, pour un segment L_i vu dans les caméras j et k, sa direction est donnée par :

$$R_j^{\top} l_i^j \times R_k^{\top} l_i^k \tag{6.9}$$

Comme le plan $\mathcal{P}$ contient les segments L_a et L_b, sa normale $n_{\mathcal{P}}$ est orthogonale aux deux directions des segments, soit :

$$n_{\mathcal{P}} = d_{L_a} \times d_{L_b} = (R_1^{\top} l_a^1 \times R_2^{\top} l_a^2) \times (R_2^{\top} l_b^2 \times R_3^{\top} l_b^3) \tag{6.10}$$

Concernant la deuxième inconnue de l'équation 6.8, *i.e.* le point M, il est possible de la faire disparaître en utilisant cette équation pour un point $P_a \in L_a$ avec $j = 2$ et $k = 1$ et pour un point $P_b \in L_b$ avec $j = 2$ et $k = 3$. On obtient alors :

$$\begin{aligned} \frac{(l_b^{3\top} T_{23})(n_{\mathcal{P}}^{\top}(R_2^{\top} p_b^2))}{l_b^{3\top}(R_{23}p_b^2)} = \frac{(l_a^{1\top} T_{21})(n_{\mathcal{P}}^{\top}(R_2^{\top} p_a^2))}{l_a^{1\top}(R_{21}p_a^2)} \quad &\Leftrightarrow \\ \frac{l_b^{3\top} T_{23}}{l_a^{1\top} T_{21}} = \frac{(l_b{}^{3\top}(R_{23}p_b^2))(n_{\mathcal{P}}^{\top}(R_2^{\top} p_a^2))}{(l_a^{1\top}(R_{21}p_a^2))(n_{\mathcal{P}}^{\top}(R_2^{\top} p_b^2))} \quad &\Leftrightarrow \\ \frac{\lambda_{23}}{\lambda_{21}} = \frac{(l_b^{3\top}(R_{23}p_b^2))(n_{\mathcal{P}}^{\top}(R_2^{\top} p_a^2))(l_a^{1\top} t_{21})}{(l_a^{1\top}(R_{21}p_a^2))(n_{\mathcal{P}}^{\top}(R_2^{\top} p_b^2))(l_b^{3\top} t_{23})} \end{aligned} \tag{6.11}$$

L'équation 6.11 exprime ainsi le ratio signé de deux facteurs d'échelles successifs. Ce ratio permet donc d'obtenir l'échelle commune de reconstruction sur un triplet d'images et permet ainsi une reconstruction globale d'un système à n caméras.

Notons également que cette équation ne fait pas intervenir de triplet de *features*, uniquement des informations bifocales ainsi qu'une contrainte de coplanarité sur l'image centrale. De plus, si l'équation 6.11 fait intervenir des points particuliers des segments p_a^2 et p_b^2, seul le fait que le point appartienne au segment est important. En effet, en prenant $Q = P_a + \mu d_a$, $\mu \in \mathbb{R}$ un point quelconque de L_a, on peut exprimer son projeté sur la caméra 2 en fonction de p_a^2 :

$$\tilde{q}^2 = R_2(Q - C_2) = p_a^2 + \mu R_2 d_a \tag{6.12}$$

Les facteurs de 6.11 faisant intervenir p_a^2 deviennent, en le remplaçant par $\tilde{q}^2$:

- $n_{\mathcal{P}} \cdot (R_2^{\top} \tilde{q}^2) = n_{\mathcal{P}} \cdot (R_2^{\top} p_a^2) + n_{\mathcal{P}} \cdot d_a = n_{\mathcal{P}} \cdot (R_2^{\top} p_a^2)$ car $d_a \perp n_{\mathcal{P}}$ par construction du plan $\mathcal{P}$.
- $l_a^1 \cdot (R_{21}\tilde{q}^2) = l_a^1 \cdot (R_{21} p_a^2) + R_1^{\top} l_a^1 \cdot d_a = l_a^1 \cdot (R_{21} p_a^2)$ car $R_1^{\top} l_a^1$ correspond à la normale du plan (C_1, L_a).

Ainsi la formule 6.11 est bien invariante au choix du point du segment sur L_a et par symétrie sur L_b.

Remarquons enfin que cette formule n'est pas définie dans tous les cas possibles. En effet, lorsque certaines configurations géométriques sont atteintes, l'un des produits scalaires de la formule est nul rendant cette contrainte inutilisable. Il existe trois configurations géométriques qui rendent cette formule dégénérée et celles-ci s'appliquent de façon symmétrique à la paire de caméras 1-2 et à la paire 2-3. Pour simplifier, nous ne décrirons donc que le cas des caméras 1-2 :

- Le cas $l_a^1 \cdot (R_{21} p_a^2) = 0$ correspond au cas où le plan $(C_2 l_a^2)$ est parallèle au plan $(C_1 l_a^1)$. Cela signifie en particulier que la ligne 3D L_a définie comme l'intersection de ces deux plans ne peut être définie de manière unique. La contrainte coplanaire faisant intervenir cette ligne n'a donc aucun sens.
- Le cas $n_{\mathcal{P}} \cdot (R_2^\top p_a^2) = 0$ correspond au cas où L_a est toujours à une distance constante du plan $\mathcal{P}$ et ce, quel que soit la valeur de λ. La contrainte coplanaire n'a ici aucun sens, puisque le choix de la valeur de λ ne permet pas de rendre L_a et L_b coplanaires.
- Le cas $l_a^1 \cdot t_{21} = 0$ correspond au cas où la ligne 3D L_a est constante quelle que soit la valeur de λ, seules les extrémités des segments 3D varient.

Ces configurations particulières empêchent une estimation correcte du facteur d'échelle λ. Cependant, elles ne se produisent que très rarement et il est possible de les éviter en écartant les paires de lignes coplanaires qui annulent l'un des 6 produits scalaires. De plus, nous n'avons pas besoin de connaître la valeur de λ pour détecter ces cas particuliers, qui peuvent donc être écartés avant l'estimation du facteur d'échelle.

6.2.2 Contraintes trifocales

Bien que nous pouvons nous passer d'informations trifocales, il serait dommage de ne pas en utiliser quand il y en a de disponible. Nous supposons ici que nous avons accès à un triplet de points ou de segments dans le triplet d'images successives $1,2,3$. Nous considérons alors une contrainte trifocale simplifiée ne faisant apparaître que les facteurs d'échelles λ_{12} et λ_{23}. Quel que soit le type de *feature* considéré, le procédé est similaire :

1 Nous imposons de manière arbitraire la position des deux premières caméras en fixant $\lambda_{12} = 1$, $C_1 = 0$ et $C_2 = -R_2^\top t_{12}$. Nous pouvons alors calculer une reconstruction 3D approximative du *feature* à partir des deux premières caméras,

2 Nous reprojetons le *feature* 3D calculé précédemment sur la troisième caméra,

3 Nous calculons enfin le facteur d'échelle λ_{23} tel que la reprojection calculée précédemment corresponde au mieux à la détection sur la troisième caméra.

A l'étape 1 nous fixons arbitrairement $\lambda_{12} = 1$ et le facteur d'échelle calculé λ_{23} correspond en fait au ratio d'échelle $\tau_{123} = \lambda_{23}/\lambda_{12}$. Pour obtenir un résultat plus robuste et plus précis nous calculons également le ratio τ_{321} correspondant au cas où les caméras 1 et 3 ont été échangées. Lorsque τ_{123} et $1/\tau_{321}$ diffèrent trop, le triplet est considéré comme mauvais et le ratio d'échelle n'est pas calculé. Sinon, nous conservons le ratio $\tau = (\tau_{123} + 1/\tau_{321})/2$.

Notons que nous n'avons utilisé ici que les directions des translations t_{12} et t_{23}. En effet, l'existence de triplet de *features* n'implique pas que la calibration bifocale entre les caméras 1 et 3 a fonctionné et a donné un résultat précis (particulièrement dans les scènes d'intérieur). Cependant, dans le cas où t_{13} est disponible et considérée comme fiable, il est possible d'utiliser cette méthode avec 4 permutations supplémentaires ou d'utiliser des méthodes plus classiques.

A l'étape 3, comme nous le verrons par la suite, le facteur d'échelle λ_{23} est exprimé comme solution d'une équation de la forme suivante :

$$\lambda_{23} = \underset{\lambda \in \mathbb{R}}{\arg\min} \frac{\|u \times (v + \lambda w)\|_2}{\|u\|_2 \, \|v + \lambda w\|_2}, \tag{6.13}$$

où u, v, w sont des vecteurs connus de $\mathbb{R}^3$.

Proposition : L'équation 6.13 admet une unique solution dans $\mathbb{R}$ ssi l'hypothèse suivante est vérifiée :

$$(u, w, v \times w) \text{ forment une famille libre.} \tag{6.14}$$

Démonstration. Cette propriété se visualise très bien graphiquement (voir Figure 6.2). Nous avons représenté les deux vecteurs v et w normalisés ainsi que les positions de $\|(v+\lambda w)\|_2/\|v+\lambda w\|_2$ lorsque λ varie. On notera dans la suite $v_\lambda = v+\lambda w$.

Notons que nous avons supposé ici que (v,w) formait une famille libre. Dans le cas contraire, l'équation 6.13 n'admet trivialement pas de solution unique (fonction constante de λ).

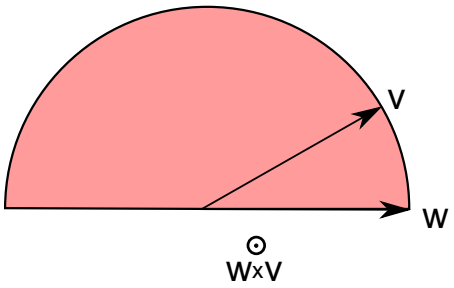


FIGURE 6.2 – On a représenté ici les deux vecteurs v et w normalisés. Le demi-cercle rouge représente les positions de $(v+\lambda w)\|v+\lambda w\|_2$ lorsque λ varie. Il passe de $-w$ lorsque $\lambda \to -\infty$ à w lorsque $\lambda \to +\infty$ en passant par v lorsque $\lambda = 0$.

Chercher la solution de l'équation 6.13, c'est chercher le λ qui minimise l'angle $\angle(u, v_\lambda)$ en valeur absolue. Pour plus de clarté, on supposera dans la suite que $\|u\|_2 = 1$ ($u = 0$ contredit l'hypothèse (6.14)).

Graphiquement, lorsque u est dans le plan (v,w), l'équation admet un minimum lorsque $v_\lambda = \pm u$. Cette solution est atteinte ssi $u \times w \neq 0$. De plus lorsque $u = v \times w$, l'équation n'admet pas de solution unique car la fonction est constante égale à 1. Les autres cas ne sont que des combinaisons linéaires des précédents cas.

Pour démontrer plus rigoureusement le résultat, nous pouvons décomposer u dans la base $(v, w, v \times w)$. Cette famille est bien libre puisque nous avons écarté le cas où v et w sont liés. La décomposition est alors donnée par :

$$u = c_1 v + c_2 w + c_3 v \times w \tag{6.15}$$

on cherche alors le minimum de la fonction F définie par :

$$F(\lambda) = \frac{\|u \times v_\lambda\|_2}{\|v_\lambda\|_2} = \frac{\|(c_1 v + c_2 w + c_3 v \times w) \times (v+\lambda w)\|_2}{\|v_\lambda\|_2} \tag{6.16}$$

$$= \frac{\|(c_1\lambda - c_2)(v \times w) + c_3 (v \times w) \times (v+\lambda w)\|_2}{\|v_\lambda\|_2} \tag{6.17}$$

Le minimum de F est le même que celui de F^2 et à l'aide du théorème de Pythagore, on obtient :

$$F^2(\lambda) = \frac{\|(c_1\lambda - c_2)(v \times w)\|_2^2}{\|v_\lambda\|_2^2} + \frac{\|c_3 (v \times w) \times (v+\lambda w)\|_2^2}{\|v_\lambda\|_2^2} \tag{6.18}$$

$$= \frac{\|(c_1\lambda - c_2)(v \times w)\|_2^2}{\|v_\lambda\|_2^2} + c_3^2 \|v \times w\|_2^2 \tag{6.19}$$

Le second terme étant constant, on a alors deux cas :

- $c_1 = 0$, F est alors constante et n'admet pas de minimum unique,
- $c_1 \neq 0$, F admet un minimum unique en $\lambda = c_2/c_1$.

Dans le premier cas on a $u = c_2 w + c_3 v \times w$ ce qui contredit l'hypothèse (6.14).

Dans le second cas, nous devons démontrer que l'hypothèse (6.14) est bien vérifiée. Supposons donc par l'absurde qu'elle ne l'est pas :

$$\exists(\lambda_1,\lambda_2,\lambda_3) \neq (0,0,0) \text{ t.q. } \lambda_1 u + \lambda_2 w + \lambda_3 v \times w = 0 \tag{6.20}$$

Comme nous avons écarté le cas (v,w) liés, alors $\lambda_1 \neq 0$. Ainsi nous avons deux décompositions différentes (car $c_1 \neq 0$) de u dans la base $(v,w,v\times w)$:

$$u = c_1 v + c_2 w + c_3 v \times w = -\frac{\lambda_2}{\lambda_1} w - \frac{\lambda_3}{\lambda_1} v \times w \tag{6.21}$$

Ce cas de figure étant impossible, l'hypothèse (6.14) est donc bien vérifiée ?

Finalement, en récapitulant chacun des cas :

- Si v et w sont liés alors (6.14) n'est pas vérifiée et il n'y a pas de minimum unique
- Si (v,w) libre ; soit $c_1 = 0$ qui implique que (6.14) ne soit pas vérifiée mais il n'y a pas non plus de minimum unique, soit $c_1 = 0$ qui implique que (6.14) soit vérifiée et il y a un minimum unique

Finalement, nous avons exploré tous les cas possibles et montré que la proposition était correcte. □

Notons de plus que la solution à cette équation (lorsqu'elle est unique) est très simple à avoir puisque la dérivée de $F^2(\lambda)$ est une fraction rationnelle dont le dénominateur est de degré 2 (le terme de degré 3 se simplifie).

Contrainte trifocale de points

On considère ici le triplet d'appariement de points $\hat{p} = (p_1, p_2, p_3)$ pour les caméras $1,2,3$. Il est possible de calculer un point 3D $\tilde{P}$ à partir des projections p_1 et p_2 et en supposant que $\lambda_{12} = 1$ (Étape 1). Sa projection sur la caméra 3 s'exprime en fonction de λ_{23} (Étape 2) :

$$\tilde{p_3} = R_3(\tilde{P} - C_3) = R_3(\tilde{P} - C_2) - \lambda_{23} t_{23}$$

Nous cherchons alors le facteur d'échelle $\lambda^\star_{23}$ qui rend la projection $\tilde{p_3}$ la plus proche possible de l'observation p_3. Plutôt que de minimiser la distance pixellique entre les deux points, nous préférons minimiser l'angle $\widehat{p_3 C_3 \tilde{p_3}}$ (voir Fig. 6.3). En effet, nous préférons utiliser une mesure homogène avec celle utilisée pour les segments dont la distance pixellique est mal définie. Ainsi, nous obtenons l'équation suivante :

$$\begin{aligned} \lambda^*_{23} &= \arg\min_{\lambda_{23}\in\mathbb{R}} \frac{\|p_3 \times \tilde{p_3}\|_2}{\|p_3\|_2 \, \|\tilde{p_3}\|_2} \\ &= \arg\min_{\lambda_{23}\in\mathbb{R}} \frac{\|p_3 \times (R_3(\tilde{P} - C_2) - \lambda_{23} t_{23})\|_2}{\|p_3\|_2 \, \|R_3(\tilde{P} - C_2) - \lambda_{23} t_{23}\|_2} \end{aligned} \tag{6.22}$$

Cette équation est bien de la forme de Eq. (6.13), nous pouvons donc la résoudre à l'aide de la méthode présentée précédemment (Étape 3).

Contrainte trifocale de lignes

On considère ici le triplet d'appariement de lignes $\hat{l} = (l_1, l_2, l_3)$ pour les caméras $1,2,3$. Il est possible de calculer une ligne 3D $\tilde{L}$ à partir des projections l_1 et l_2 et en supposant que $\lambda_{12} = 1$ (Étape 1). Sa projection sur la caméra 3 s'exprime en fonction de λ_{23} (Étape 2) :

$$\tilde{l_3} \propto R_3[d_{\tilde{L}} \times (\tilde{P} - C_3)]$$

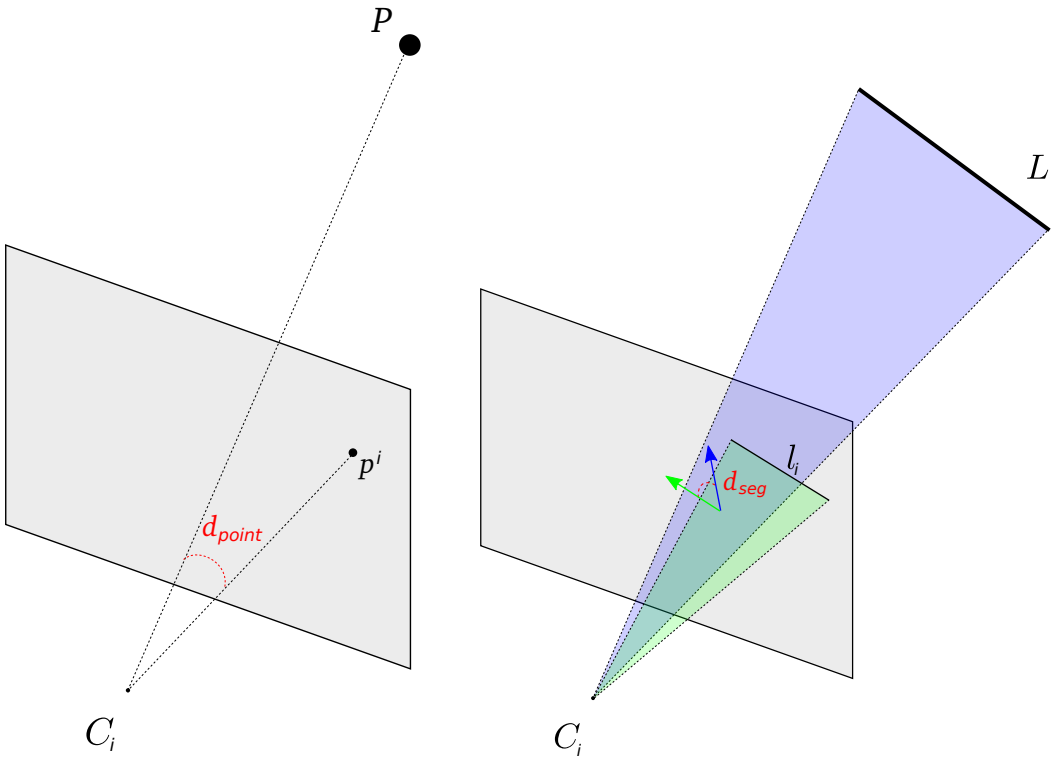


FIGURE 6.3 – Illustration des distances angulaires pour les points et les segments. La distance des points correspond à l'écart angulaire entre la direction caméra-observation et la direction caméra-point 3D. Pour les lignes, la distance correspond à l'écart angulaire entre les normales des plans caméra-observation et caméra-ligne 3D. Sur le dessin, la flèche verte représente la normale du plan vert (*i.e.* plan caméra-observation) et la flèche bleue représente la normale du plan bleu (*i.e.* plan caméra-ligne 3D).

où $\tilde{P}$ est un point quelconque de $\tilde{L}$ et $d_{\tilde{L}}$ correspond à la direction 3D de $\tilde{L}$. Nous cherchons alors le facteur d'échelle $\lambda_{23}^{\star}$ qui rend la projection $\tilde{l}_3$ la plus proche possible de l'observation l_3. Nous cherchons pour cela à minimiser l'angle entre $\tilde{l}_3$ et l_3 correspondant à l'écart angulaire entre les plans formés par $\mathcal{C}_3$ et $\tilde{l}_3$ ou l_3 (voir Fig. 6.3). Ainsi, nous obtenons l'équation suivante :

$$\begin{aligned}
\lambda_{23}^{*} &= \underset{\lambda_{23}\in\mathbb{R}}{\arg\min} \frac{\|l_3 \times \tilde{l}_3\|_2}{\|l_3\|_2 \, \|\tilde{l}_3\|_2} \\
&= \underset{\lambda_{23}\in\mathbb{R}}{\arg\min} \frac{\|l_3 \times (R_3[d_{\tilde{L}} \times (\tilde{P} - \mathcal{C}_3)])\|_2}{\|l_3\|_2 \, \|R_3[d_{\tilde{L}} \times (\tilde{P} - \mathcal{C}_3)]\|_2} \\
&= \underset{\lambda_{23}\in\mathbb{R}}{\arg\min} \frac{\|l_3 \times (R_3[d_{\tilde{L}} \times (\tilde{P} - \mathcal{C}_2)] - \lambda_{23} R_3[d_{\tilde{L}} \times t_{23}])\|_2}{\|l_3\|_2 \, \|R_3[d_{\tilde{L}} \times (\tilde{P} - \mathcal{C}_2)] - \lambda_{23} R_3[d_{\tilde{L}} \times t_{23}]\|_2}
\end{aligned} \tag{6.23}$$

Cette équation est bien de la forme de Eq. (6.13), nous pouvons donc la résoudre à l'aide de la méthode présentée précédemment (Étape 3). Notons également que cette équation fait apparaître un point $\tilde{P}$ de $\tilde{L}$. Cependant, le produit vectoriel avec $d_{\tilde{L}}$ rend cette formule invariante au choix du point sur la ligne $\tilde{L}$. Le facteur d'échelle ne dépend donc que des lignes et pas des segments. Il n'est donc pas affecté par une mauvaise détection des extrémités des segments.

6.3 Estimation robuste

Pour estimer de manière robuste et précise les facteurs d'échelle dans les différents triplets nous utilisons une approche de type RANSAC.

Sélection des *features*

Aucun oracle n'est utilisé pour sélectionner les paires de lignes coplanaires. Nous utilisons l'ensemble des paires de lignes pouvant générer une contrainte coplanaire (*e.g.* les paires d'appariements de segments tel qu'un des appariements concerne les caméras 1 et 2 tandis que le second concerne les caméras 2 et 3). C'est l'utilisation du RANSAC qui permet de filtrer les bonnes paires coplanaires des mauvaises paires.

Pour les points, nous construisons les triplets à l'aide des paires *inliers* de la calibration bifocale. Le RANSAC permet alors de sélectionner les triplets les plus précis.

Pour les triplets de segments nous procédons à un nouvel appariement à calibration bifocale connue. En effet, un appariement peut être considéré comme correct pour la calibration bifocale mais mauvais pour l'estimation du facteur d'échelle (*e.g.* un segment mis en correspondance avec un segment parallèle à l'appariement correct). Pour cela, nous nous nous sommes inspirés de la méthode [29] (voir Fig. 6.4). Pour chaque segment d'une image, nous considérons la bande définie dans l'autre image par les droites épipolaires de chacune de ses extrémités. Parmi les candidats à l'appariement, nous rejetons alors tous les segments dont l'intersection avec la bande est trop faible. L'appariement final est le même que celui de la méthode LBD [84] mais utilise le critère d'intersection avec la bande épipolaire au lieu des critères géométriques proposés par les auteurs.

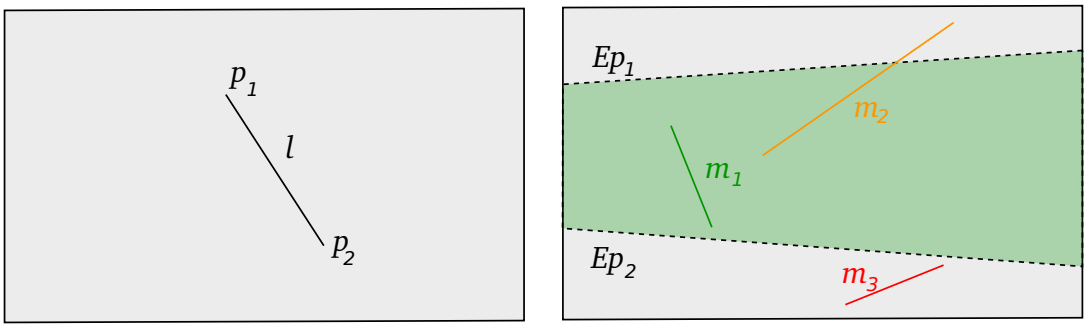


FIGURE 6.4 – Appariement des lignes après calibration bifocale. Les lignes Ep_1 et Ep_2 représentent les droites épipolaires des points p_1 et p_2. Le segment m_1 est un bon candidat pour l tandis que m_3 est un mauvais candidat. Le segment m_2 est entre les deux cas et l'utilisation d'un seuil de recouvrement permet de faire un choix.

Fonction d'erreur des lignes coplanaires

On considère ici un quadruplet donné de segments $\check{l} = (l_a^1, l_a^2, l_b^2, l_b^3)$ tel que l_a^1, l_a^2 corresponde à un appariement sur les caméras 1-2 et l_b^2, l_b^3 à un appariement sur les caméras 2-3 (voir Fig. 6.1, p.100). Nous pouvons alors calculer les lignes 3D L_a et L_b correspondant aux deux appariements. Nous définissons alors le point 3D P_{ab} comme étant le point de L_a le plus proche de L_b :

$$P_{ab} = \underset{P \in L_a}{\arg\min}\, d(P, L_b) \tag{6.24}$$

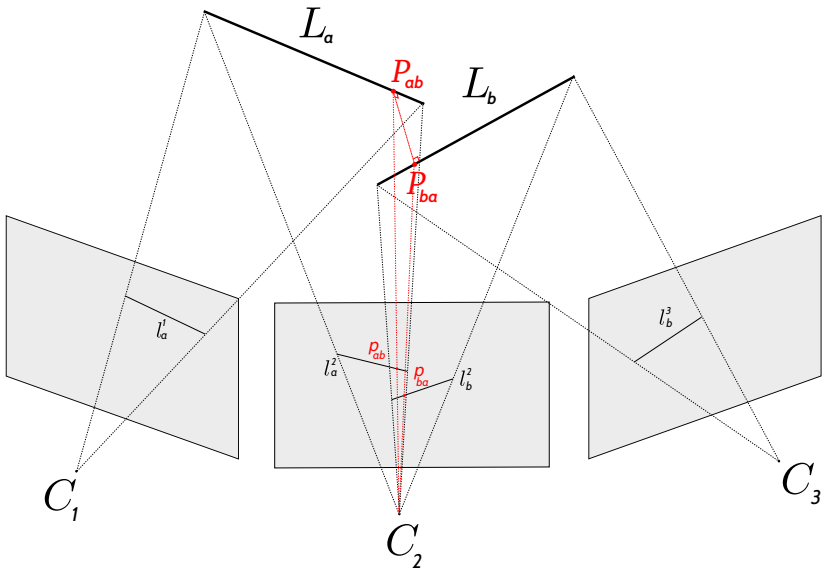


FIGURE 6.5 – Illustration de la distance coplanaire. La distance calculée correspond à la reprojection sur l'image centrale de la distance entre les deux droites supposées coplanaires.

De la même façon, nous définissons le point 3D $P_{ba} = \arg\min_{P \in L_b} d(P, L_a)$. Dans le cas où L_a et L_b sont parfaitement coplanaires et où le facteur d'échelle a parfaitement été estimé, on devrait avoir $P_{ab} = P_{ba}$. Nous considérons alors les projections p^2_{ab} et p^2_{ba} de P_{ab} et P_{ba} sur la caméra centrale et définissons l'erreur comme la distance pixellique suivante :

$$d_{\mathsf{cop}}(\check{l}) = d_{\mathsf{cop}}(L_a, L_b) = d(p^2_{ab}, p^2_{ba}) \tag{6.25}$$

La distance est représentée graphiquement sur la figure 6.5.

Pour éviter d'obtenir un trop grand nombre de paires potentielles de lignes coplanaires, nous générons pour chaque segment dans l'image centrale une paire avec les N segments les plus proches (en utilisant la distance définie par le minimum des distance des extrémités et avec $N = 10$ dans notre implémentation). De plus, pour éviter les cas dégénérés, nous ne considérons pas les paires de lignes dont les directions sont trop proches (nous avons utilisé un seuil minimum de 15° dans notre implémentation). Rappelons que nous avons accès aux directions des segments 3D à partir des rotations globales et donc que cette génération de paires coplanaires se fait avant toute estimation de facteur d'échelle.

Fonction d'erreur des triplets de points

Pour un triplet de points $\hat{p} = (p^1, p^2, p^3)$ mis en correspondance dans les caméras 1-2-3, nous calculons une estimation du point correspondant P à partir des projections p^1 et p^2. Nous calculons alors la reprojection p^3_{12} sur la caméra 3. Nous définissons enfin l'erreur comme étant la distance pixellique entre la reprojection p^3_{12} et l'observation p^3 (voir Fig. 6.6). Pour plus de robustesse, nous symétrisons cette mesure en interchangeant les caméras 1 et 3 et en prenant la moyenne des deux distances : $d_{\mathsf{point}}(\hat{p}) = (d(p^1_{23}, p_1) + d(p^3_{12}, p_3))/2$.

Il est possible de précalculer en partie cette distance pour limiter le temps de calcul. En effet, p^3_{12} peut s'écrire sous la forme $\alpha + \lambda\beta$ où les termes α et β sont indépendants de λ. Nous précalculons donc α et β et le calcul de la distance ne correspond alors plus qu'à calculer $\alpha + \lambda\beta$ pour un λ donné.

Notons que cette distance pourrait être symétrisée en utilisant les caméras 1-3 pour calculer le point 3D et la caméra 2 pour calculer la distance reprojection-observation. Cependant,

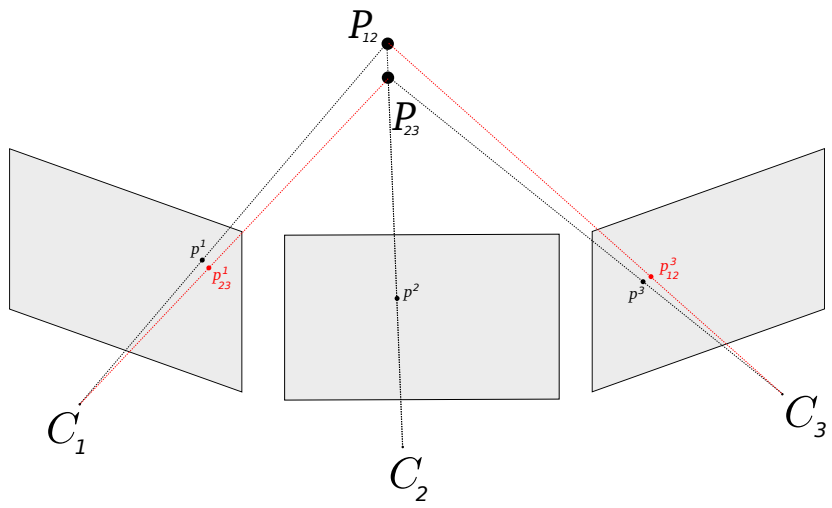


FIGURE 6.6 – Illustration de la distance d'un triplet de point pour un facteur d'échelle donné.

lorsque la direction t_{13} n'est pas connue, cette distance ne peut être précalculée. Pour des raisons d'efficacité de calcul, nous considérons donc la distance $d_{\text{point}}(\hat{p}) = (d(p^1_{23}, p_1) + d(p^3_{12}, p_3))/2$ lorsque t_{13} n'est pas connue et $d_{\text{point}}(\hat{p}) = (d(p^1_{23}, p_1) + d(p^2_{13}, p_2) + d(p^3_{12}, p_3))/3$ dans le cas contraire. Rappelons que l'existence de triplets dans les caméras 1-2-3 est juste une condition nécessaire pour obtenir la pose relative des caméras 1-3 mais pas une condition suffisante. De même la distance aurait pu être calculée en considérant le point P construit à partir des points p_1, p_2 et p_3 mais ce calcul perdrait en efficacité (*i.e.* le précalcul n'est pas possible).

Fonction d'erreur des triplets de lignes

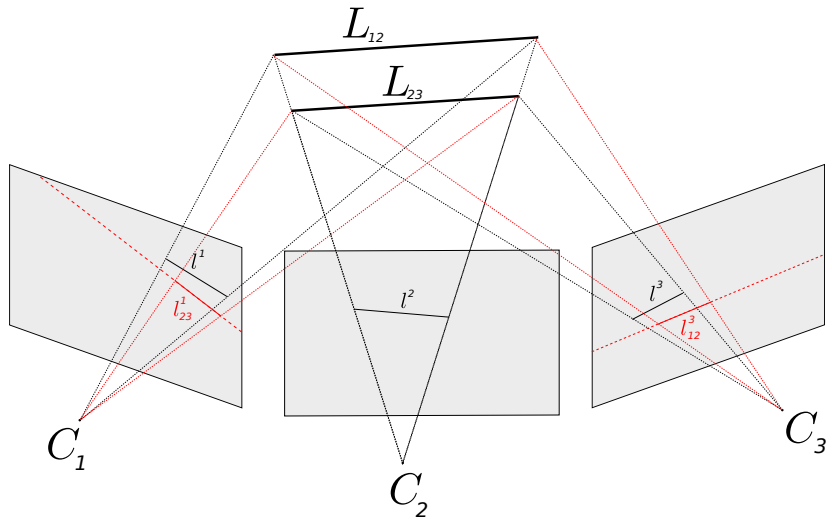


FIGURE 6.7 – Illustration de la distance d'un triplet de point pour un facteur d'échelle donné.

Pour un triplet de lignes $\hat{l} = (l_1, l_2, l_3)$ mis en correspondance dans les caméras 1-2-3, nous calculons une estimation du segment correspondant L à partir des projections l_1 et l_2. Nous calculons alors la reprojection l_{12}^3 sur la caméra 3. Nous définissons enfin l'erreur comme étant la distance pixellique entre la ligne l_{12}^3 et les extrémités du segment observé l_3. Pour plus de robustesse, nous symétrisons cette mesure comme nous l'avons fait avec les points en interchangeant les caméras 1 et 3 pour définir la distance moyenne $d_{\mathsf{seg}}(\hat{l})$.

Notons que les remarques faites pour les triplets de points concernant le précalcul et la symétrisation de la fonction de distance s'appliquent également aux triplets de lignes.

RANSAC

Dans les cas classiques d'utilisation de RANSAC, le nombre de modèle à tester est généralement très élevé (*e.g.* $\binom{n}{7}$ pour la matrice fondamentale ou $\binom{n}{5}$ pour la matrice essentielle avec n le nombre de *feature*, dépassant 1000 dans la plupart des scènes). Dans notre cas, le nombre de modèles est égal au nombre n de *features* considérés. De plus n est généralement inférieur à 10000, nous n'utilisons donc pas la partie *choix aléatoire des features* dans notre RANSAC. Nous réalisons exactement n itérations pour tester tous les modèles possibles.

Dans le cas où ce nombre de *feature* serait trop important, il est cependant toujours possible de revenir à une utilisation plus classique en sélectionnant aléatoirement un sous-ensemble de *feature*. Cette sélection ne devrait pas trop altérer la qualité des résultats à condition de conserver un sous-ensemble de taille suffisamment importante.

6.4 Méthode *a contrario*

Données d'entrée :

Les données qui suivent un modèle *a contrario* correspondent à des appariements de points et de segments. De même que dans le chapitre précédent, pour des questions d'indépendance, nous ne considérons pas que les paires de segments coplanaires suivent le modèle *a contrario* mais uniquement les segments, pris séparément, de ces mêmes paires.

Modèle estimé :

Le modèle estimé correspond au facteur d'échelle commun aux trois caméras, ou de manière équivalente à la position relative des trois caméras. Quel que soit le *feature* utilisé, le nombre de modèles estimés est de 1 car il n'y a jamais d'ambiguïté sur le ratio d'échelle obtenu à partir d'un *feature*.

Distance données-modèle :

Pour les triplets de points et de lignes (*e.g.* contraintes trifocales), la distance utilisée correspond aux distances présentées précédemment, soit d_{point} pour les points (voir section 6.3) et d_{seg} pour les lignes (voir section 6.3). Dans le cas des lignes coplanaires, comme les données concernent les lignes et non les paires de lignes, nous avons besoin d'une distance portant sur les lignes seules. Pour résoudre ce problème, nous utilisons la même méthode que pour la calibration bifocale en définissant la distance d'une ligne coplanaire comme étant :

$$d_{\mathsf{cop}}(L_i) = \min_{L_j \text{ coplanaire avec } L_i} d_{\mathsf{cop}}(L_i, L_j) \tag{6.26}$$

Cette distance n'a pas de sens physique particulier mais elle fournit un ordre de grandeur d'une mesure de l'adéquation de la ligne, par rapport à d'autres lignes supposées coplanaires. Elle permet alors de faire fonctionner correctement le modèle *a contrario*.

Modèle de fond :

Nous utilisons ici 3 modèles de fond indépendants pour chaque *features*. Le modèle de fond $\mathcal{H}_0^l$ considère des triplets de lignes indépendamment et uniformément distribués dans la scène. De même, nous utilisons le modèle de fond $\mathcal{H}_0^p$ pour les triplets de points. Dans le cas des lignes coplanaires, le modèle de fond $\mathcal{H}_0^c$ considère un ensemble de lignes indépendamment et uniformément distribués dans la scène. Les modèles $\mathcal{H}_0^l$ et $\mathcal{H}_0^c$ sont similaires mais le nombre de *features* considérés sont différents ; les nombres de triplets de lignes pour le premier et le nombre de lignes associés à une paire coplanaire pour le second.

Nombre de Fausses Alarmes :

Nous reprenons la formules du NFA générique [54] pour l'appliquer aux différents *features*. Dans le cas des triplets de points, un triplet suffit à estimer le facteur d'échelle :

$$\mathrm{NFA}_{\mathsf{point}}(n,k,\varepsilon) = (n_{\mathsf{point}}-1)\binom{n_{\mathsf{point}}}{k}\binom{k}{1}\mathrm{p}_{\mathsf{point}}(\varepsilon)^{k-1} \tag{6.27}$$

La distance utilisée correspond à une distance entre deux points. La probabilité pour deux points dans une image de taille $\mathcal{A} = w \times h$ d'être à une distance de moins de ε correspond à :

$$\mathrm{p}_{\mathsf{point}}(\varepsilon) = \frac{\pi\varepsilon^2}{\mathcal{A}} \tag{6.28}$$

On obtient alors la formule finale pour le NFA des triplets de points :

$$\mathrm{NFA}_{\mathsf{point}}(n_{\mathsf{point}},k,\varepsilon) = (n_{\mathsf{point}}-1)\binom{n_{\mathsf{point}}}{k}k\left(\frac{\pi\varepsilon^2}{\mathcal{A}}\right)^{k-1} \tag{6.29}$$

où n_{point} correspond au nombre de triplets de points.

Dans le cas des triplets de segments, un seul triplet est suffisant pour estimer le facteur d'échelle et la distance utilisée d_{seg}, correspond à une distance entre un point et une ligne. La probabilité pour un point d'être à une distance de moins de ε d'une ligne dans une image de taille $\mathcal{A}$ et de diamètre $D = \frac{w+h}{2}$ correspond à :

$$\mathrm{p}_{\mathsf{seg}}(\varepsilon) = \frac{D\varepsilon}{\mathcal{A}} \tag{6.30}$$

On obtient alors la formule finale pour le NFA des triplets de segments :

$$\mathrm{NFA}_{\mathsf{seg}}(n_{\mathsf{seg}},k,\varepsilon) = (n_{\mathsf{seg}}-1)\binom{n_{\mathsf{seg}}}{k}k\left(\frac{D\varepsilon}{\mathcal{A}}\right)^{k-1} \tag{6.31}$$

où n_{seg} correspond au nombre de triplets de segments.

Dans le cas des contraintes coplanaires, le comptage des modèles possibles est différent. Un comptage usuel donnerait $\binom{n}{k}\binom{k}{2}$ mais il inclurait alors des contraintes coplanaires immédiatement rejetées car correspondant à des cas dégénérés. Notre formule tient donc compte de ces cas-là en imposant d'abord de prendre deux segments générant une contrainte coplanaire (celle générant le modèle) puis tous les segments restants :

$$\mathrm{NFA}_{\mathsf{cop}}(n_{\mathsf{cop}},k,\varepsilon) = (n_{\mathsf{cop}}-2)\binom{n_{\mathsf{cop}}}{k-2}\binom{n_{\mathsf{cop}}}{1}\binom{N}{1}\mathrm{p}_{\mathsf{cop}}(\varepsilon)^{k-2} \tag{6.32}$$

où N correspond aux nombre de paires coplanaires considérées pour un segment donné et n_{cop} correspond au nombre de segments dans l'image centrale qui forment au moins une contrainte coplanaire avec des segments des autres images.

Par ailleurs, la distance utilisée dans $\mathrm{p}_{\mathsf{cop}}(\varepsilon)$ correspondant à une distance entre deux points, la formule est la même que pour les points (6.28). La formule finale pour le NFA des contraintes coplanaires est alors :

$$\mathrm{NFA}_{\mathsf{cop}}(n_{\mathsf{cop}},k,\varepsilon) = (n_{\mathsf{cop}}-2)\binom{n_{\mathsf{cop}}}{k-2} N n_{\mathsf{cop}} \left(\frac{\pi\varepsilon^2}{\mathcal{A}}\right)^{k-2} \tag{6.33}$$

En pratique, on cherche à évaluer le NFA associé à un facteur d'échelle λ qui est donné par les formules :

$$\mathrm{NFA}_{\mathsf{point}}(\lambda) = (n_{\mathsf{point}}-1)\min_{k\in[2,n_{\mathsf{point}}]}\binom{n_{\mathsf{point}}}{k} k \left[\frac{\pi d_{\mathsf{point}}(\hat{p}_k,\lambda)^2}{\mathcal{A}}\right]^{k-1} \tag{6.34}$$

$$\mathrm{NFA}_{\mathsf{seg}}(\lambda) = (n_{\mathsf{seg}}-1)\min_{k\in[2,n_{\mathsf{seg}}]}\binom{n_{\mathsf{seg}}}{k} k \left[\frac{2\mathcal{D}\, d_{\mathsf{seg}}(\hat{l}_k,\lambda)}{\mathcal{A}}\right]^{k-1} \tag{6.35}$$

$$\mathrm{NFA}_{\mathsf{cop}}(\lambda) = (n_{\mathsf{cop}}-2)\min_{k\in[3,n_{\mathsf{cop}}]} n_{\mathsf{cop}} N \binom{n_{\mathsf{cop}}}{k-2} \left[\frac{\pi d_{\mathsf{cop}}(L_k,\lambda)^2}{\mathcal{A}}\right]^{k-2} \tag{6.36}$$

où $\hat{p}_k$, $\hat{l}_k$ et L_k correspondent au k-ième *feature* associé lorsqu'on les a ordonnés par distance croissante.

La formule du NFA global est alors le produit des NFA pour chacun des *features* comme lors de la calibration bifocale (les NFA peuvent être vus comme des espérances de variables aléatoires indépendantes, voir section 5.4.2, page 81) :

$$\mathrm{NFA}(\lambda) = \mathrm{NFA}_{\mathsf{cop}}(\lambda).\mathrm{NFA}_{\mathsf{point}}(\lambda).\mathrm{NFA}_{\mathsf{seg}}(\lambda) \tag{6.37}$$

Enfin, le facteur d'échelle finalement sélectionné correspond au λ qui minimise le NFA global.

6.5 Ajustement de faisceaux

Une fois le facteur d'échelle estimé par RANSAC *a contrario* pour chaque triplet d'images, nous avons accès aux positions globales des caméras. Les positions et orientations des caméras sont alors raffinées lors d'une étape d'ajustement de faisceaux permettant un gain de précision.

Lors de cette étape, nous utilisons toutes les contraintes *inliers* sélectionnées lors de l'étape de RANSAC. De plus nous combinons ces contraintes pour former des *tracks* afin d'obtenir un raffinement de meilleure qualité. Ainsi les triplets de points ou de segments ayant une ou plusieurs observations en commun (*e.g.* les points ou segments vu par 4 caméras ou plus) sont fusionnées pour obtenir des n-uplets de points ou segments. Ces n-uplets génèrent alors des contraintes de reprojections sur les caméras concernées. Nous avons utilisé les formulations usuelles des contraintes de reprojection pour cette étape. Ainsi, la distance de reprojection d'un point P est donnée par la distance pixellique entre la projection $\mathrm{P}_i P$ du point 3D et l'observation p_i concernée (voir Figure 6.6) :

$$d_{\mathsf{point}}(P,i) = \|\mathrm{P}_i P - p_i\|_2 \tag{6.38}$$

La distance de reprojection des segments est la même que celle de [85] et correspond à la moyenne des distances entre les extrémités des observations et la projection du segment 3D :

$$d_{\mathsf{seg}}(L,i) = \frac{1}{2}\sum_{k=1}^{2} \frac{|\tilde{\mathrm{P}}_i L^\top \mathbf{l}_{i,k}|}{\mathbf{l}_{i,k,z}\sqrt{(\tilde{\mathrm{P}}_i L)_x^2 + (\tilde{\mathrm{P}}_i L)_y^2}} \tag{6.39}$$

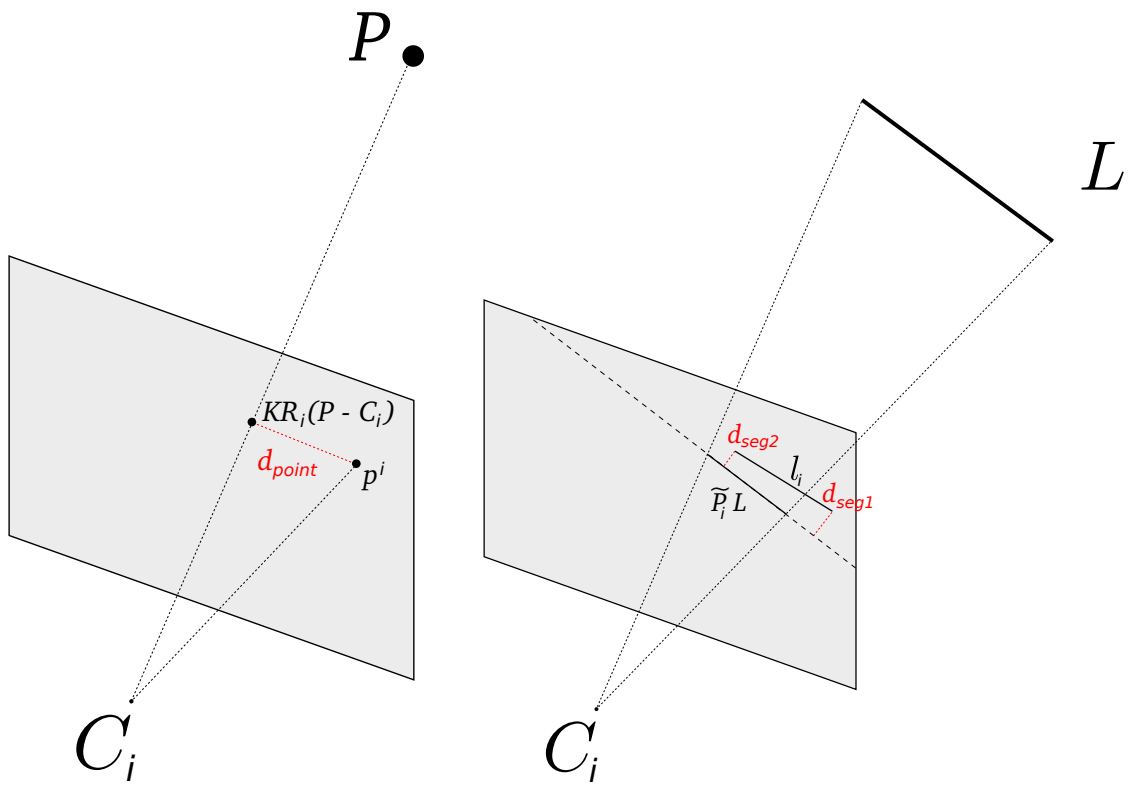


FIGURE 6.8 – Illustration des distances de reprojections pour les points et les segments. La distance de reprojection des points correspond à la distance du projeté à l'observation. Pour les lignes, c'est la moyenne arithmétique entre la distance des extrémités de l'observation et la ligne projetée.

où les indices x, y, z désignent les composantes des vecteurs associés. Rappelons que $\tilde{\mathrm{P}}L$ est une notation indiquant le projeté de la ligne 3D mais ne correspond pas à une formule mathématique réelle.

Les contraintes coplanaires peuvent être utilisées de différentes façons lors de l'étape d'ajustement de faisceaux. Nous avons considéré deux approches :

- l'utilisation des contraintes *inliers* considérés séparément et traitées avec les contraintes de reprojections des segments considérés,
- l'agrégation de segments coplanaires en plans, combinée avec l'utilisation d'une contrainte forte d'appartenance aux plans.

La première approche a l'avantage d'être directe et fonctionne correctement, seule ou combinée avec les contraintes de reprojections des n-uplets. L'idée de la seconde approche est de combiner les différentes contraintes coplanaires pour obtenir une structure plus rigide. Cependant elle produit de moins bons résultats. L'hypothèse principale pour expliquer cet échec est que l'agrégation en plans multisegments néglige de nombreuses erreurs de coplanarité. De même que les points de fuite sont moins précis que les paires de lignes parallèles lors de la calibration bifocale, les plans sont moins précis que les contraintes coplanaires prises isolément.

S'inspirant de travaux récents [59], nous raffinons en premier les *features* 3D ainsi que les positions des caméras en fixant leurs orientations, puis nous raffinons la structure complète en permettant à la rotation de varier également. Enfin, il est possible de faire un dernier raffinement pour les matrices de calibration internes K lorsque leur estimation initiale n'est pas jugée suffisante.

6.6 Expériences

Nous avons testé nos méthodes sur différents types de données, synthétiques et réelles.

6.6.1 Calcul de l'erreur

Dans le cas des données synthétiques, nous ne nous intéressons qu'au facteur d'échelle et comparons donc la valeur trouvée λ à la vérité terrain λ^{GT} en utilisant le ratio suivant :

$$\tau = \frac{|\lambda - \lambda^{GT}|}{|\lambda^{GT}|} \tag{6.40}$$

Lorsque le jeu de données réelles contient une vérité terrain, il est possible d'obtenir une erreur moyenne du positionnement des caméras. Pour un ensemble de caméras $(\mathcal{C}_i)$ à comparer à l'ensemble de caméras vérité terrain $(\mathcal{C}_i^{GT})$, on cherche la transformation $[s,R,t]$ permettant de juxtaposer au mieux chaque caméras, soit :

$$[s^\star, R^\star, t^\star] = \underset{s\in\mathbb{R}, R\in\mathcal{SO}_3(\mathbb{R}), t\in\mathbb{R}^3}{\arg\min} \sum_i \|s(R\mathcal{C}_i + t) - \mathcal{C}_i^{GT}\|_2 \tag{6.41}$$

Une fois la transformation estimée, il est alors possible d'obtenir l'écart moyen entre chacune des positions estimées des caméras et les positions vérité terrain.

Enfin, dans le cas où il n'existe pas de vérité terrain, une évaluation quantitative peut être réalisée à l'aide du résidu pixellique de reprojection. Nous l'avons cependant accompagné d'une reconstruction 3D des différents *features* qui permet alors de contrôler aussi visuellement la qualité du résultat.

6.6.2 Expériences sur données synthétiques

Nous avons souhaité étudier le comportement des contraintes coplanaires seules dans un premier temps, puis comparées aux autres contraintes dans un second temps. Pour cela, nous avons utilisé des données synthétiques pour lesquelles nous avons fait varier divers paramètres :

- le nombre de plans dans la scène,
- le nombre de lignes détectées,
- le bruit de détection correspondant aux détections parfois peu précises des détecteurs de segments,
- le degré de planarité des plans correspondant aux paires de lignes *presque* coplanaires.

Comme nous le verrons par la suite, les contraintes coplanaires obtiennent de moins bons résultats lorsque les paramètres utilisés rendent le problème plus difficile. Cependant, cette décroissance n'est pas brutale.

Paramètres des scènes synthétiques :

Bien que le *dataset* soit synthétique, nous avons utilisé des paramètres réalistes pour la création de la scène et les caméras. Ces paramètres sont proches de ceux de la scène *Office-P19*. Les paramètres utilisés sont les suivants :

- 3 caméras observant une scène planaire par morceaux avec un écart de $15^\circ + \text{random}[0, 15^\circ]$ entre les orientations successives des caméras,
- la scène fait 2 m et est à une distance de 3 m des caméras,
- la matrice de calibration interne K est similaire à celle de la caméra utilisée pour *Office-P19* soit une focale de 20 mm et des images de résolution 4000×4000 pixels, avec un capteur de taille similaire.

Pour chaque expériences, nous générons $n_{\mathcal{P}}$ plans aléatoires et n_{seg} segments aléatoires dans chaque plans. La moitié des segments n'est observée que par les caméras 1 et 2 tandis que l'autre moitié n'est observée que par les caméras 2 et 3 (comme dans la configuration de la Fig. 6.1). Nous avons étudié trois configurations différentes, proches des observations faites dans nos scènes d'intérieurs :

- 2 plans contenant 10 segments chacun, générant en moyenne 90 contraintes coplanaires,
- 2 plans contenant 20 segments chacun, générant en moyenne 200 contraintes coplanaires,
- 3 plans contenant 10 segments chacun, générant en moyenne 150 contraintes coplanaires,
- 3 plans contenant 20 segments chacun, générant en moyenne 300 contraintes coplanaires.

Notons que le nombre de contraintes coplanaires varie d'une scène à l'autre puisque le nombre de contraintes rejetées pour cause de parallélisme ou d'une autre hypothèse non vérifiée varie également.

Nous ajoutons également deux types de bruit aux segments 3D et aux projections :

- un bruit de projection modélisé par un bruit Gaussien centré d'écart type σ_R (en pixels) affectant les extrémités des segments projetés. Notons que ce bruit est ajouté indépendamment à chaque projection sur chaque caméra.
- un écart de planarité modélisé par un bruit Gaussien centré d'écart type $\sigma_{\mathcal{P}}$ (en mm) affectant les extrémités des segments 3D et les écartant du plan parfait initial.

Enfin, nous générons 500 scènes aléatoires pour chaque configuration et utilisons l'erreur moyenne définie précédemment (6.40) comme mesure de qualité.

Contraintes coplanaires seules

Nous étudions ici la sensibilité des contraintes coplanaires en présence de bruit de projection (Fig. 6.9) et/ou de bruit de planarité (Fig. 6.10).

Manque de précision des détections.

Comme nous pouvons le voir dans la Fig. 6.9, en diminuant la précision de détection des segments de $\sigma_R = 0$ à 5 pixels, que l'erreur de planarité soit absente ($\sigma_{\mathcal{P}} = 0$ mm) ou moyenne ($\sigma_{\mathcal{P}} = 20$ mm), l'erreur d'estimation du facteur d'échelle croît doucement.

Erreur de planarité.

Comme nous pouvons le voir dans la Fig. 6.10, en diminuant la planarité des segments de $\sigma_{\mathcal{P}} = 0$ à 50 mm, que l'erreur de détection soit absente ($\sigma_R = 0$ pixels) ou moyenne ($\sigma_R = 2$ pixels), l'erreur d'estimation du facteur d'échelle croît doucement.

Nombre de lignes.

Comme nous pouvons le voir dans les Fig. 6.9 et 6.10, plus un plan contient de lignes, moins l'erreur est élevée. En effet, un plus grand nombre de lignes permet de mieux lisser et moyenner l'erreur.

Nombre de plans.

A l'inverse, un plus grand nombre de plans, avec un même nombre de lignes dans chacun, augmente le nombre de contraintes mais accroît également l'erreur. L'augmentation du nombre de plans augmente en fait surtout le nombre d'*outliers* et réduit la qualité du résultat. Dans le cas où la planarité est parfaite (*e.g.* pas de bruit de planarité), ces ajouts correspondent pour la plupart à des *inliers* précis ce qui explique que l'erreur soit plus faible.

Comparaison des différentes contraintes

Nous comparons ici la précision des différents types de contraintes utilisées pour l'estimation du facteur d'échelle (*i.e.* triplets de points, triplets de segments, paires de lignes coplanaires).

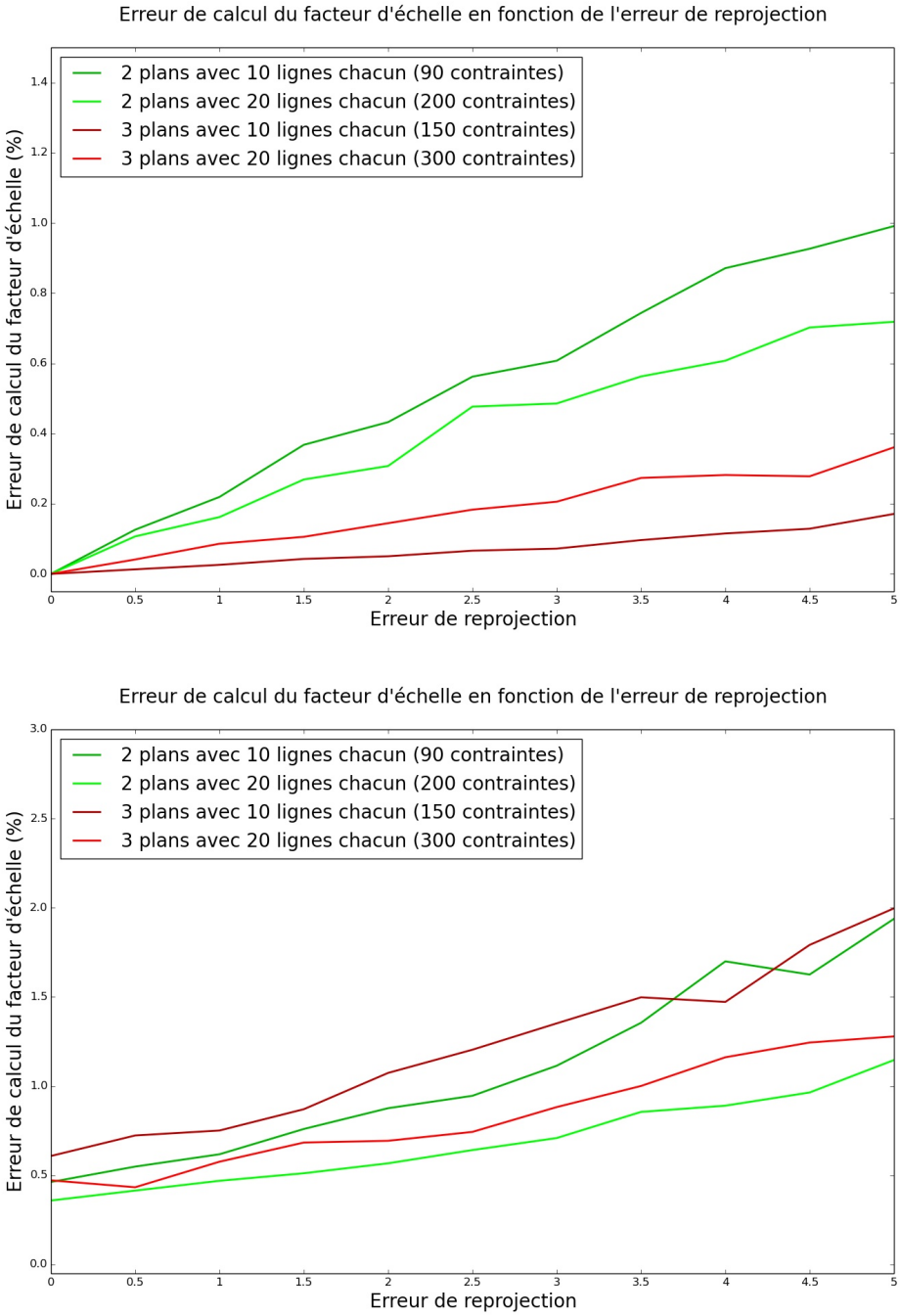


FIGURE 6.9 – Sensibilité de l'estimation du facteur d'échelle quand l'erreur de reprojection des segments augmente. En haut, en l'absence de bruit planaire et en bas avec une déviation de la planarité de $\sigma_{\mathcal{P}} = 20$mm.

Nous utilisons les mêmes paramètres pour générer les scènes. Les configurations font également intervenir les triplets de points et de segments de la façon suivante :

- 2 plans contenant chacun 10 points et 10 segments générant $n_{\text{point}} = 20$ triplets de points, $n_{\text{seg}} = 20$ triplets de lignes et une moyenne de $n_{\text{cop}} = 200$ segments coplanaires.
- 3 plans contenant chacun 10 points et 10 segments générant $n_{\text{point}} = 30$ triplets de points,

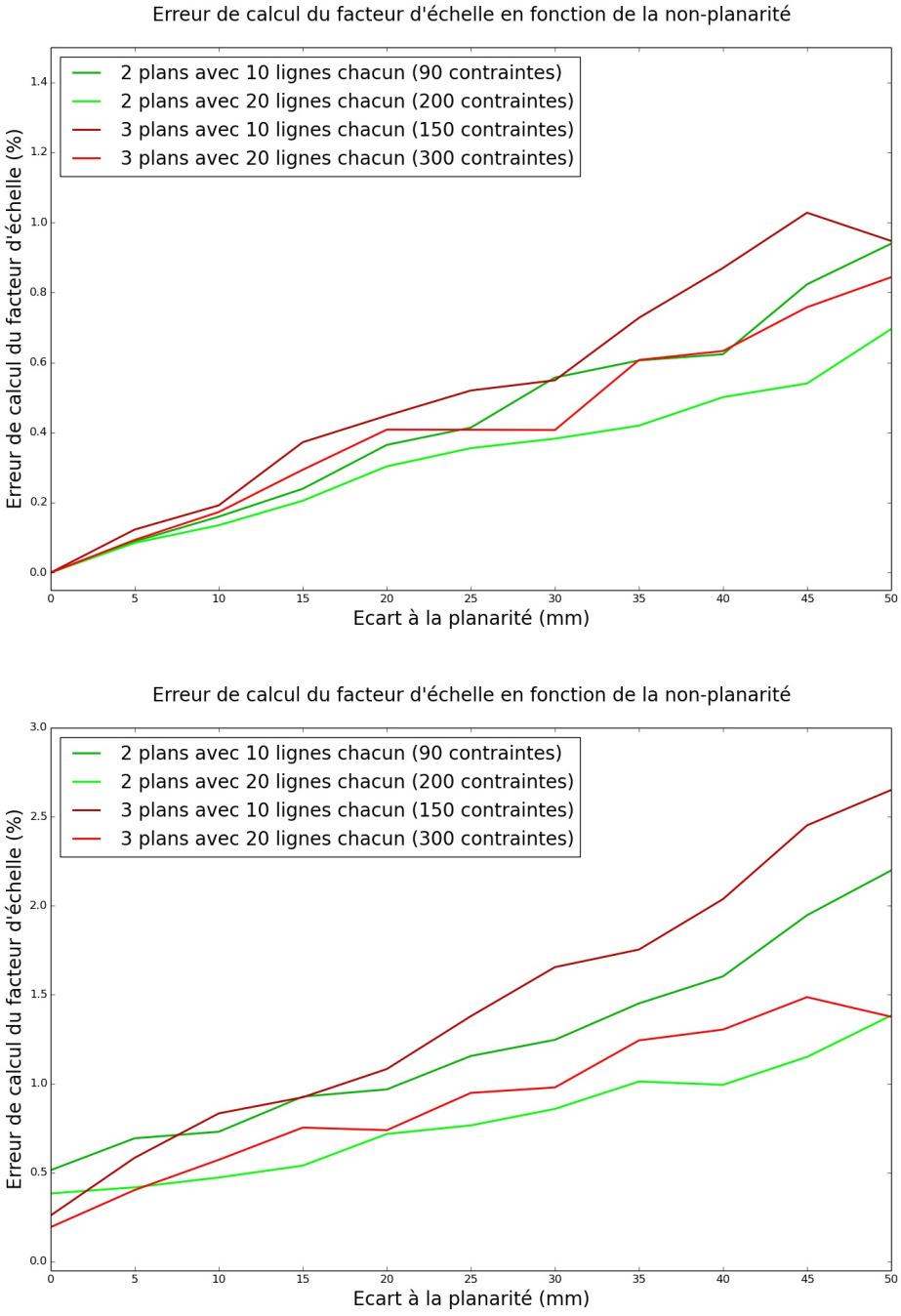


FIGURE 6.10 – Sensibilité de l'estimation du facteur d'échelle quand l'erreur de planarité des segments augmente. En haut, en l'absence de bruit de projection et en bas avec un bruit de reprojection de $\sigma_R = 2$ pixels.

$n_{\text{seg}} = 30$ triplets de lignes et une moyenne de $n_{\text{cop}} = 300$ segments coplanaires.

Les appariements des points et des segments sont parfaits et les contraintes de triplets ne contiennent donc pas d'*outliers*. Les contraintes coplanaires sont générées à partir des triplets de segments sans aucune connaissance de la géométrie de la scène (*i.e.* comme dans le cas de données réelles). Cependant la génération de contraintes coplanaires se fait sans connaissance préalable

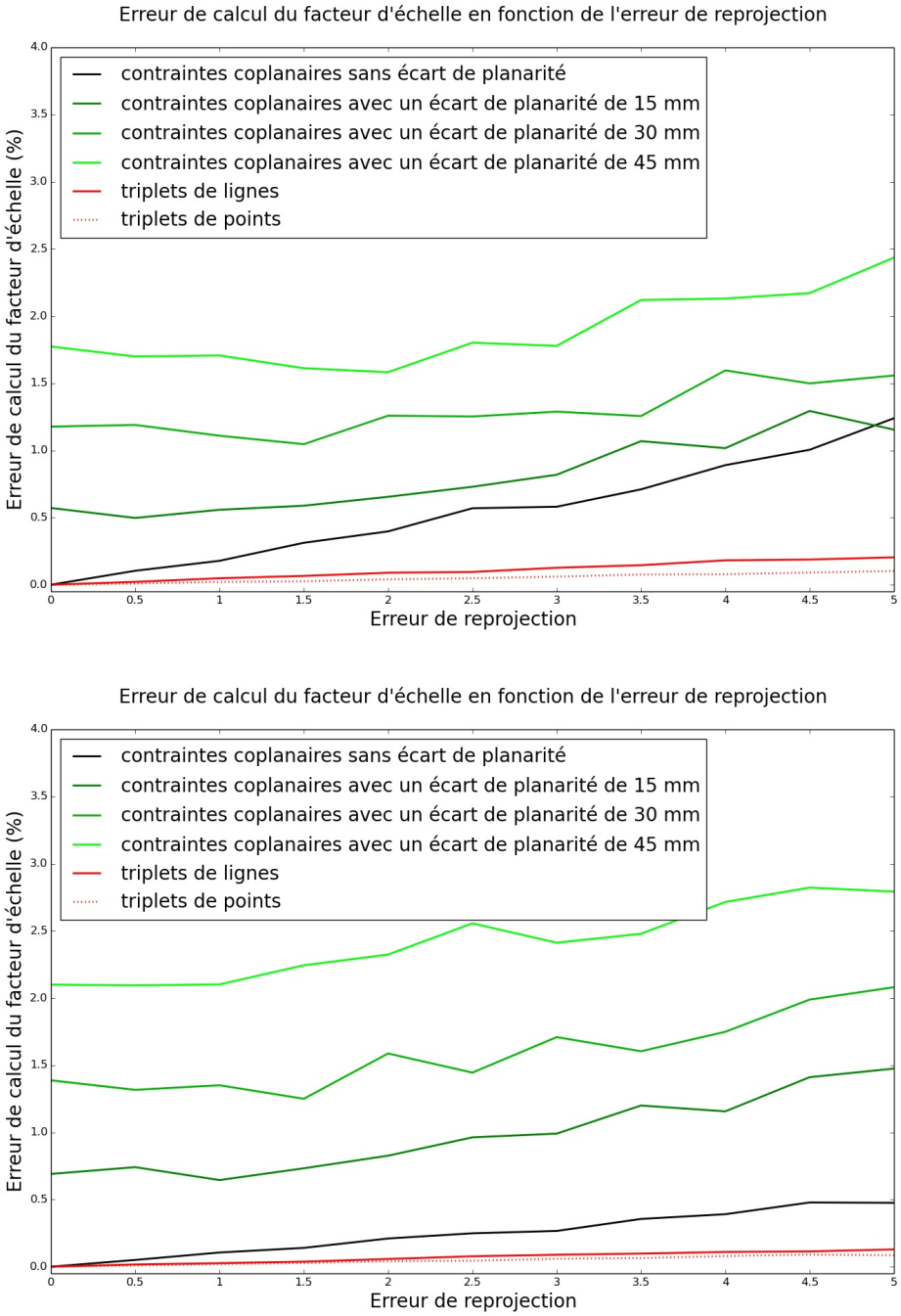


FIGURE 6.11 – Comparaison des différents *features* en fonction de l'erreur de projection. En haut, la configuration à 2 plans et en bas la configuration à 3 plans. Les contraintes coplanaires génèrent plusieurs cas car elles sont également sensibles au degré de planarité des plans.

de la vérité terrain et ils contiennent donc une large proportion d'*outliers* ($\frac{1}{2}$ pour la configuration à 2 plans et $\frac{2}{3}$ pour la configuration à 3 plans).

Le bruit de projection est le même que précédemment pour les segments. Il affecte donc exactement de la même manière les contraintes trifocales de segments et les contraintes coplanaires. Pour les points, nous appliquons un bruit Gaussien de même écart-type σ_R sur les projections.

Des imperfections sont également ajoutées à la planarité des plans à travers le bruit Gaussien centré d'écart type $\sigma_{\mathcal{P}}$. Nous avons étudié les cas de plans d'épaisseur 0 mm, 15 mm, 30 mm, 45 mm. Les triplets de points et de segments n'étant pas affectés par le bruit de planarité, nous n'avons tracé qu'une seule courbe pour chacun, indépendamment de l'écart de coplanarité.

Les résultats peuvent être observés dans la Fig. 6.11. Nous avons affiché les courbes des différentes contraintes sur le même graphique pour plus de clarté. Cependant, il ne paraît pas évident que la comparaison entre triplets de segments et triplets de points soit pertinente. En effet, les modèles de bruit utilisés pour les points et les lignes ne sont *a priori* pas comparables. De plus, le ratio d'*outliers* est relativement élevé dans le cas des contraintes coplanaires alors qu'il est absent dans le cas des triplets de points ou de segments. Ainsi, le comportement général des différentes *features* a du sens mais les comparaisons relatives sont biaisées. On retrouve cependant l'ordre observé dans le cadre des données réelles ; les points sont les plus précis et les contraintes coplanaires les moins précises.

6.6.3 Expériences sur données réelles

Nous avons étudié notre méthode de calibration globale sur deux types de jeux de données :

- Différentes scènes d'intérieur avec peu de texture et peu de recouvrement entre images successives. *Office-P19* et *Meeting-P31*, un bureau et une salle de réunion vérifiant des contraintes de type *Manhattan-world* et *Trapezoid-P17*, un bureau ne vérifiant pas les contraintes de type *Manhattan-world*. Ces *datasets* sont constitués de n images (où n correspond à *dataset-Pn*) avec une résolution de 5184×3456 pixels. Ce *dataset* ne comporte pas de vérité terrain et sera donc évalué quantitativement à l'aide des résidus et qualitativement à l'aide de visuels,
- Différentes scènes de la cour intérieure d'un château provenant de l'ensemble de données de Strecha *et al.* [73]. Les images de ce *dataset* sont plus rapprochées les unes des autres (*i.e.* recouvrement important entre images successives) et plus texturées. Une vérité terrain précise permet de mesurer quantitativement la qualité des résultats. Les images ont une résolution de 3072×2048 pixels.

Nous avons également corrigé la distorsion sur chacune des images, ce phénomène détériorant fortement les résultats de calibration à base de lignes.

Comparaison entre RANSAC et RANSAC *a contrario*

Nous avons comparé notre méthode d'estimation du facteur d'échelle selon que nous employons une méthode RANSAC classique ou une méthode RANSAC *a contrario*. Pour cela nous avons étudié les résultats obtenus à l'aide de différents seuils fixes (en pixels) tout en imposant qu'il soit le même pour chaque *feature*. En effet, il est probable que les meilleurs résultats obtenus à l'aide d'un RANSAC le soient en trouvant le seuil optimal pour chaque *feature* pris séparément, cependant une telle recherche serait trop fastidieuse pour rendre la méthode applicable en pratique.

Nous avons résumé les résultats dans le Tableau 6.1. Bien que AC-RANSAC ne donne pas toujours les meilleurs résultats pour chaque scène, il reste toujours proche du meilleur résultat et il est le meilleur en moyenne. Il permet en fait d'adapter les seuils de chaque *feature* indépendamment et ce, automatiquement.

Contribution de chacune des contraintes

Nous avons comparé les résultats de calibration obtenus à l'aide des différentes contraintes prises séparément ou toutes ensemble. Nous pouvons ainsi comparer l'apport respectif de chaque

Méthode / Scène	seuil RANSAC (pixels)					AC-RANSAC
	0.5	1	3	6	9	
Castle P19	146	90.5	107	**90**	196	101
Castle P30	91	83	73	76	144	**72**
Entry P10	**7.6**	7.9	8.4	9.3	10.6	7.8
Fountain P11	2.2	**2.1**	2.4	2.5	3.8	2.3
Herz-Jesu P8	4.0	4.1	**3.9**	7.1	6.4	4.2
Herz-Jesu P25	8.5	9.0	8.6	13.3	16.9	**7.9**
Moyenne	43.2	32.8	33.9	33.0	63.0	**32.5**

TABLE 6.1 – Comparaison des résultats (erreur en mm) obtenus à l'aide d'un RANSAC à seuil fixe ou d'un RANSAC *a contrario* (AC) sur les scènes de Strecha [73]. Les meilleurs résultats pour une scène sont indiqués en **gras**.

Contrainte / Scène	Triplets de points	Triplets de lignes	Triplets de points+lignes	Paires coplanaires	Toutes
Castle P19	97.9	**80**	98	129	101
Castle P30	80.3	111.2	78	134	**72**
Entry P10	7.9	12.1	8.0	8.9	**7.8**
Fountain P11	**2.3**	2.5	**2.3**	52	**2.3**
Herz-Jesu P8	**4.1**	4.3	**4.1**	5.4	4.2
Herz-Jesu P25	8.4	15.4	8.4	36	**7.9**
Moyenne	33.5	37.6	33.1	60.9	**32.5**

TABLE 6.2 – Comparaison de l'apport des différentes contraintes (erreur en mm) sur les scènes de Strecha [73]. Hormis sur la scène de *Castle-P19*, l'ensemble des contraintes permet d'obtenir le meilleur résultat ou un équivalent proche. Les meilleurs résultats pour une scène sont indiqués en **gras**.

contrainte et vérifier ainsi qu'aucune d'elles n'a de contribution négative au niveau de la précision des résultats.

Les résultats sont rassemblés sur le Tableau 6.2. On peut observer que les triplets de points sont généralement plus précis que les triplets de lignes et eux-même plus précis que les paires coplanaires. Cependant, malgré ces différences d'erreurs, l'ensemble des contraintes permet d'obtenir les meilleurs résultats (hormis sur *Castle-P19*) et l'utilisation de triplets de lignes et de points uniquement ne permet pas d'obtenir un résultat aussi précis. Les contraintes coplanaires ont donc un double avantage : un gain de précision et, comme nous le verrons ci-dessous, de robustesse.

Comparaison avec des méthodes état-de-l'art

Nous avons comparé nos résultats avec ceux obtenus à l'aide d'autre méthodes état-de-l'art. Pour cela nous nous sommes comparés à des méthodes globales [59, 62, 11, 4, 33] et incrémentales [72, 81]. Ces méthodes n'utilisent que les points pour calculer la calibration. En effet, à notre connaissance, il n'existe qu'une chaîne de calibration utilisant les lignes [85] mais ses auteurs ne fournissent ni code ni résultats sur le *dataset* de Strecha *et al.* [73].

Les résultats sont rassemblés sur le Tableau 6.3. On peut remarquer que notre méthode obtient des résultats équivalents à ceux des autres méthodes globales et bien meilleurs que les

Méthode / Scène	Notre méthode	OpenMVG [59]	Olsson [62]	Cui [11]	Arie [4]	Jiang [33]	Bundler [72]	VSfM [81]
Castle P19	**22.2**	25.6	76.2	—	—	—	344	258
Castle P30	23.8	21.9	66.8	**21.2**	—	235	300	522
Entry P10	**5.6**	5.9	6.9	—	—	—	55.1	63.0
Fountain P11	2.4	2.5	**2.2**	2.5	4.8	14	7.0	7.6
Herz-Jesu P8	3.8	**3.5**	3.9	—	—	—	16.4	19.3
Herz-Jesu P25	5.4	5.3	5.7	**5.0**	7.8	64	21.5	22.4

TABLE 6.3 – Erreur moyenne (en mm) de position des caméras après ajustement de faisceaux : comparaison avec des méthodes globales[59, 62, 11, 4, 33] et incrémentales [72, 81]. Les meilleurs résultats sont indiqués en **gras**.

méthodes incrémentales. Ainsi, l'utilisation de contraintes coplanaires et de triplets de lignes permet de faire fonctionner notre méthode dans des scènes plus difficiles tout en conservant une précision équivalente aux meilleures méthodes globales.

Comparaison en cas de recouvrement faible

Pour modéliser l'effet de faible recouvrement très fréquent dans les scènes d'intérieur, nous avons utilisé deux scènes de *Strecha* en supprimant une partie des images. Les méthodes comparées dans la section précédente nécessitent la présence d'un certain nombre de triplets de points sur chaque triplet d'images consécutives. Dans le cas contraire, la caméra centrale ne peut pas être calibrée avec le reste.

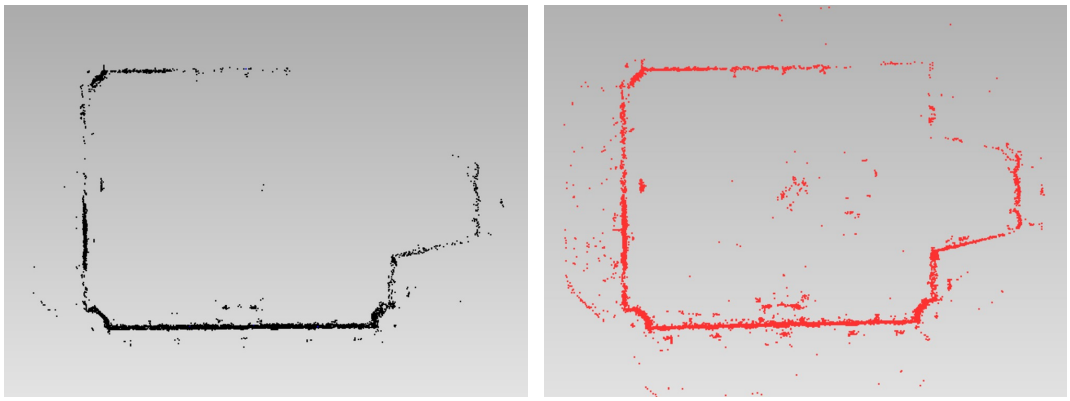


FIGURE 6.12 – Vu de haut du nuage de point calculé à partir de la calibration de *Castle-P(19−3)* par OpenMVG (à gauche) et par notre méthode (à droite). Le coin en haut à droite correspond à l'observation d'une caméra isolée. Elle n'est pas positionnée par d'OpenMVG mais notre méthode peut calculer sa position.

Ainsi, en enlevant simplement 3 images de *Castle-P19*, il est possible d'empêcher la calibration d'une des caméras avec n'importe quelle méthode état-de-l'art. Cependant notre méthode, sans l'aide des triplets de points et de lignes, est capable de trouver la position de cette caméra. L'erreur moyenne de position des caméras passe alors de 2.3 cm à 13.2 cm car la caméra isolée est située avec moins de précision. Le visuel (voir Fig. 6.12) nous confirme cependant qu'elle est située correctement.

En poursuivant l'expérience, il est possible d'enlever jusqu'à 11 des 19 images de *Castle-*

Scene \ Method	Notre méthode	OpenMVG [59]
Castle P19	0.30	0.21
Castle P30	0.29	0.21
Entry P10	0.006	0.90
Fountain P11	0.26	0.19
Herz-Jesu P8	0.28	0.20
Herz-Jesu P25	0.29	0.20
Office P19	1.07	6/20
Meeting P31	1.79	9/31
Trapezoid P17	3.60	3/17

TABLE 6.4 – Erreur moyenne de reprojection des points 3D sur toutes les caméras (en pixels). En rouge, en cas d'échec, la fraction des caméras calibrées. Notons que si notre erreur résiduelle dans les scènes d'intérieur est entre 4 et 10 fois plus élevée que celles du *dataset* de Strecha *et al.* [73], les images ont également une résolution 3 fois plus grande.

P19 en conservant une erreur moyenne de position des caméras restantes de 18.4 cm, ce qui reste encore meilleur que les méthodes incrémentales avec toutes les images.

FIGURE 6.13 – Seule notre méthode est capable de calibrer ce triplet avec peu de recouvrement et de forts changements de point de vue. Ce triplet correspond aux 3 images les plus éloignées de Herz-Jesu-P8 (première, centrale, dernière).

De la même façon, en ne conservant que les 3 images les plus éloignées de *Herz-Jesu-P8* (voir Fig. 6.13) il est possible d'obtenir une calibration encore précise à 6 mm tandis que les autres méthodes échouent car aucun triplet de points ni de lignes n'est trouvé entre ces 3 images.

Comparaison sur des scènes d'intérieur

Nous avons utilisé notre méthode sur l'ensemble des scènes d'intérieur. Cependant, en l'absence de vérité terrain, nous avons seulement calculé les résidus de reprojection des points. Nous avons aussi calculé les résidus sur les images de Strecha pour avoir un comparatif. Les résidus moyens en pixels sont indiqués dans le tableau 6.4.

Comme on peut le voir au niveau des résidus, notre méthode arrive à calibrer une scène que des méthodes plus classiques ne peuvent calibrer entièrement (manque de *features*, absence de triplets, ...). Notons également que l'écart de résidus entre les scènes d'intérieur et les scènes du *dataset* de Strecha *et al.* [73] s'expliquent en partie par l'augmentation de la résolution des images d'un rapport 3. De plus les visuels (voir Fig. 6.14) nous confirment que la calibration est qualitativement représentative de la réalité.

6.7 Limites & Conclusion

Nous avons présenté une méthode de calibration multi-vues qui utilise différents types de contraintes pour être plus robuste dans les scènes d'intérieur. L'utilisation des points, classique, permet d'obtenir une bonne précision lorsqu'ils sont disponibles, tandis que l'utilisation des lignes rend ce procédé plus robuste dans les scènes planaires par morceaux et peu texturées comme les scènes d'intérieur. De plus, les contraintes coplanaires permettent de se libérer de la nécessité d'utiliser des triplets d'images pour calibrer une scène entière. Enfin nous avons montré comment combiner ces 3 types de contraintes pour obtenir un résultat utilisant au mieux les avantages de chacune.

Cependant, notre calibration multi-vue n'est pas encore suivie d'une reconstruction efficace de scènes d'intérieur. Le problème principal pour une reconstruction dense reste l'absence de points d'accroches notamment sur des murs de couleur uniforme. Néanmoins, nous pensons que les contraintes coplanaires utilisées lors de la calibration pourraient permettre de faciliter la reconstruction à l'aide d'aprioris planaires sur les zones uniformes.

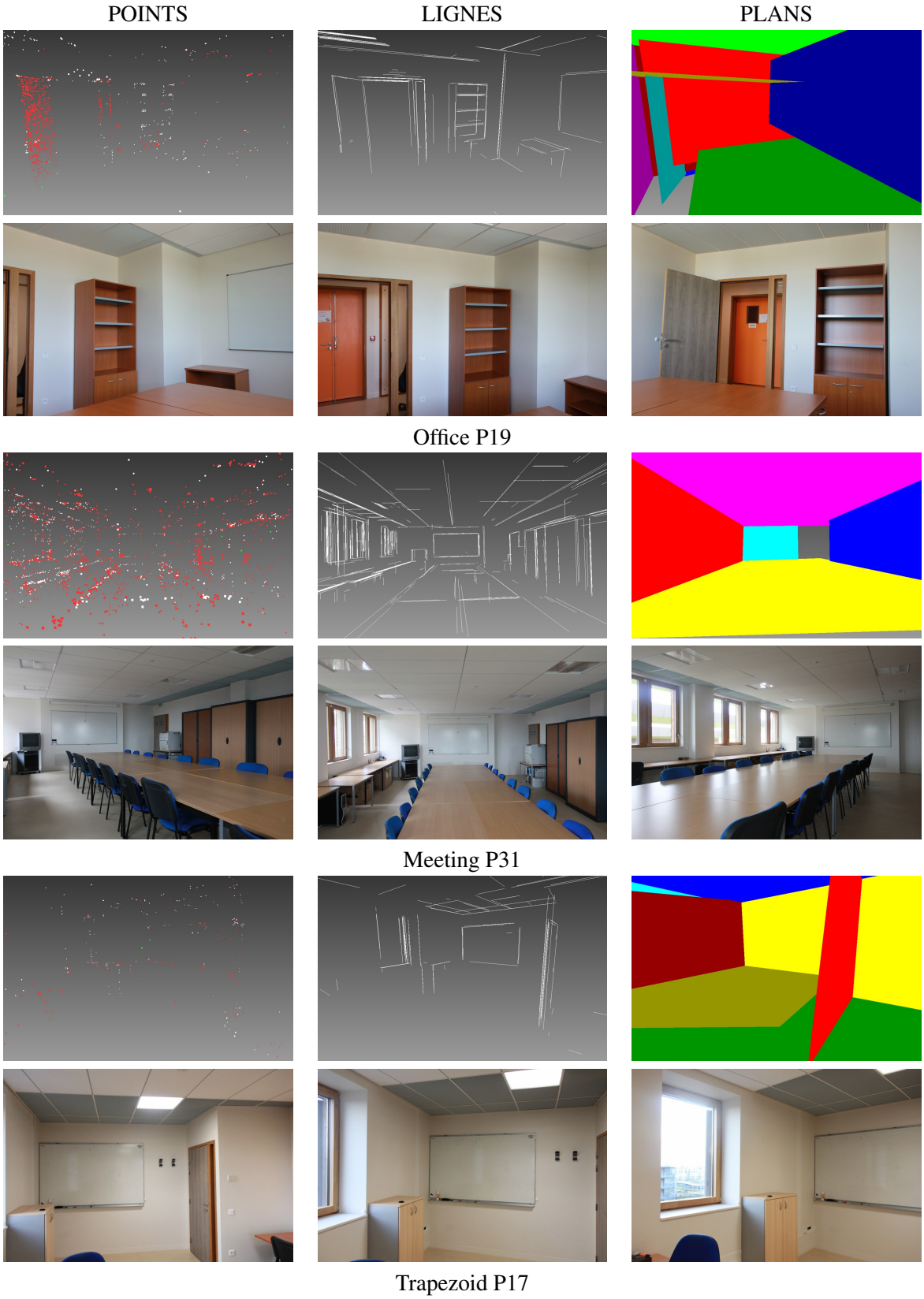


FIGURE 6.14 – Reconstruction 3D des scènes d'intérieur en séparant les points, les lignes et les contraintes coplanaires. Les plans affichés correspondent à la fusion de plusieurs contraints coplanaires appartenant à des plans similaires. Nous avons ajouté quelques images des *datasets* pour donner une idée générale des scènes.

Chapitre 7

Conclusion & Perspectives

Dans cette thèse, nous avons renforcé, sans pour autant compromettre la précision, la robustesse des méthodes de calibration multi-vue dans le cadre des scènes d'intérieur. Pour cela nous avons choisi d'utiliser les segments de droites, plus nombreux, mieux répartis dans ce type de scène et plus robustes à de forts changements de points de vue. Des améliorations sont néanmoins encore possible.

7.1 Détection et appariement

Confronté à des problèmes de détection dans les scènes d'intérieur et plus particulièrement avec des images de grande résolution, nous avons défini une méthode multi-échelle permettant d'obtenir de bonnes détections indépendamment de la taille de l'image et de la difficulté de la scène (*e.g.* présence de bruit, faible contraste...).

Même si la méthode développée permet d'obtenir de bons résultats, elle possède encore quelques inconvénients qui la rendent encore améliorable.

De plus, les méthodes d'appariement de segments sont également loin d'être parfaites et de nombreux segments ne sont pas bien mis en correspondance. En effet, comparés aux méthodes de détection et d'appariement de points [46, 7], les résultats semblent nettement moins bons. Cette observation peut s'expliquer par une difficulté intrinsèque plus grande dans le cas des segments que dans le cas des points. En effet, les extrémités des segments sont rarement fiables, les différences de points de vue engendrent des déformations des surfaces et des positions relatives entre lignes bien plus importantes qu'avec les points et enfin, l'appariement lui-même est mal défini : voulons nous mettre des segments en correspondance bijective alors que les extrémités sont mal définies ou préférons nous mettre en correspondance n segments d'une image avec m de l'autre à condition qu'ils appartiennent tous à la même droite 3D ?

La qualité de l'appariement dépend de la qualité de la méthode utilisée mais également de la qualité de la détection. Or ces deux aspects sont cruciaux pour la méthode de calibration. Nous nous sommes aperçus que certaines scènes étaient mal calibrées à cause d'un appariement de très mauvaise qualité et il nous semble donc que cette étape est actuellement limitante pour le bon fonctionnement d'une méthode de calibration à partir de segments.

7.2 Calibration bifocale

Dans le but de pouvoir calibrer des scènes planaires et peu texturées, nous nous sommes tournés vers l'utilisation de segments. Cependant, les segments seuls n'imposent pas suffisamment de contraintes sur la 3D d'une scène observée par deux images. En utilisant le parallélisme

des segments, nous avons développé une méthode précise et robuste de calibration à deux vues qui ne suppose pas une scène *Manhattan-world*.

Néanmoins si des contraintes de parallélisme sont toujours observables dans la pratique, elles nécessitent au préalable la détection des points de fuite de la scène. Or, tous les segments détectés n'ont pas nécessairement de point de fuite associé et certains points de fuite sont dédoublés et d'autres fusionnés. Ainsi, cette étape a tendance à réduire le nombre de contraintes disponibles (*i.e.* pour pouvoir être utilisé, un segment doit avoir un appariement **et** un point de fuite) tout en les diluant avec de mauvaises contraintes (*i.e.* lorsque le point de fuite associé n'est pas le bon).

De plus, si cette méthode exploite les segments, elle a toujours besoin d'appariements de points pour estimer la direction de la translation. Cependant, les points ne sont pas toujours bien détectés ou mis en correspondance dans les scènes d'intérieur et nos expériences ont montré que l'utilisation d'intersections de droites ne fonctionnait pas aussi bien. Une détection de points et/ou d'intersections de droite coplanaires de meilleure qualité permettrait d'améliorer la précision et la robustesse de cette méthode.

7.3 Calibration multi-vue et reconstruction dense

Enfin, motivés par des scènes pauvres en *features* nous avons cherché à nous passer des contraintes trifocales habituellement nécessaires à une calibration multi-vue. Pour cela, nous avons exploité les contraintes coplanaires de certains segments. De plus nous avons reformulé les contraintes trifocales usuelles pour pouvoir les combiner à nos contraintes coplanaires. Nous avons montré que l'utilisation conjointe des trois types de contraintes permettait une calibration plus robuste et aussi précise que les méthodes état-de-l'art.

Cependant nous avons observé que l'utilisation des points dans l'ajustement de faisceaux permettait un gain de précision nettement plus important que celui des triplets de segments ou des segments coplanaires alors que la précision des différents *features* est similaire avant ajustement de faisceaux. De plus, contrairement au cas des points, l'erreur de reprojection des lignes n'a pas de formule *parfaite* (*i.e.* comment définir une distance entre deux segments ? faut il prendre en compte les extrémités ? les différences angulaires de direction ? ...). Dans le cas des contraintes coplanaires, la distance de reprojection est encore plus complexe à définir et nous pensons donc qu'une formulation plus adaptée de ces deux distances permettrait un gain de précision non négligeable.

De plus, certaines contraintes propres aux lignes pourraient être ajoutées pour permettre une reconstruction de meilleure qualité. Ainsi, des contraintes à base de parallélisme, d'orthogonalité ou exploitant la présence hypothétique de plans pourraient être utilisées. En particulier, les contraintes coplanaires ne se limitent qu'à des paires de segments. En regroupant les segments coplanaires appartenant à un même plan il devrait être possible d'obtenir un résultat plus précis. De plus, l'utilisation de plans devrait permettre une reconstruction dense plus robuste, notamment dans les zones peu texturées, fréquentes dans les scènes d'intérieur.

Bibliographie

[1] Sameer AGARWAL, Keir MIERLE et et AL. *Ceres Solver*. http://ceres-solver.org.

[2] Cuneyt AKINLAR et Cihan TOPAL. « EDLines : A Real-time Line Segment Detector with a False Detection Control ». In : *Pattern Recogn. Lett.* 32.13 (oct. 2011), p. 1633–1642. ISSN : 0167-8655. DOI : 10.1016/j.patrec.2011.06.001. URL : http://dx.doi.org/10.1016/j.patrec.2011.06.001.

[3] A. ALMANSA, A. DESOLNEUX et S. VAMECH. « Vanishing point detection without any a priori information ». In : *Transactions on Pattern Analysis and Machine Intelligence* 25.4 (2003), p. 502–507. URL : http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1190575.

[4] Mica ARIE-NACHIMSON, Shahar Z. KOVALSKY, Ira KEMELMACHER-SHLIZERMAN, Amit SINGER et Ronen BASRI. « Global Motion Estimation from Point Matches ». In : *3DIMPVT*. 2012.

[5] D. H. BALLARD. « Readings in Computer Vision : Issues, Problems, Principles, and Paradigms ». In : sous la dir. de Martin A. FISCHLER et Oscar FIRSCHEIN. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1987. Chap. Generalizing the Hough Transform to Detect Arbitrary Shapes, p. 714–725. ISBN : 0-934613-33-8. URL : http://dl.acm.org/citation.cfm?id=33517.33574.

[6] Adrien BARTOLI et Peter F. STURM. « Structure-from-motion using lines : Representation, triangulation, and bundle adjustment ». In : *Computer Vision and Image Understanding* 100.3 (2005), p. 416–441. DOI : 10.1016/j.cviu.2005.06.001. URL : http://dx.doi.org/10.1016/j.cviu.2005.06.001.

[7] Herbert BAY, Andreas ESS, Tinne TUYTELAARS et Luc VAN GOOL. « Speeded-Up Robust Features (SURF) ». In : *Comput. Vis. Image Underst.* 110.3 (juin 2008), p. 346–359. ISSN : 1077-3142. DOI : 10.1016/j.cviu.2007.09.014. URL : http://dx.doi.org/10.1016/j.cviu.2007.09.014.

[8] Herbert BAY, Vittorio FERRARI et Luc VAN GOOL. « Wide-Baseline Stereo Matching with Line Segments ». In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2005)*. Washington, DC, USA : IEEE Computer Society, 2005, p. 329–336. ISBN : 0-7695-2372-2. DOI : 10.1109/CVPR.2005.375.

[9] J. CANNY. « A Computational Approach to Edge Detection ». In : *IEEE Trans. Pattern Anal. Mach. Intell.* 8.6 (juin 1986), p. 679–698. ISSN : 0162-8828. DOI : 10.1109/TPAMI.1986.4767851. URL : http://dx.doi.org/10.1109/TPAMI.1986.4767851.

[10] M. CHANDRAKER, Jongwoo LIM et D. KRIEGMAN. « Moving in stereo : Efficient structure and motion using lines ». In : *IEEE 12th International Conference on Computer Vision (ICCV 2009)*. 2009, p. 1741–1748. DOI : 10.1109/ICCV.2009.5459390.

[11] Zhaopeng CUI, Nianjuan JIANG, Chengzhou TANG et Ping TAN. « Linear Global Translation Estimation with Feature Tracks ». In : *British Machine Vision Conference (BMVC 2015)*. Swansea, UK, September 7-10, 2015, 2015. DOI : 10.5244/C.29.46.

[12] Agnès DESOLNEUX, Lionel MOISAN et Jean-Michel MOREL. « Meaningful Alignments ». In : *International Journal of Computer Vision* 40.1 (2000), p. 7–23. ISSN : 1573-1405. DOI : 10.1023/A:1026593302236. URL : http://dx.doi.org/10.1023/A:1026593302236.

[13] Jamil DRARÉNI, Renaud KERIVEN et Renaud MARLET. « Indoor Calibration using Segments Chains ». In : *33rd Annual Symposium of the German Association for Pattern Recognition (GCPR/DAGM 2011)*. Août 2011.

[14] A. ELQURSH et A. ELGAMMAL. « Line-based relative pose estimation ». In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011, p. 3049–3056. DOI : 10.1109/CVPR.2011.5995512.

[15] Olivier D. FAUGERAS, Rachid DERICHE, Herve MATHIEU, Nicholas AYACHE et Gregory RANDALL. « The Depth and Motion Analysis Machine ». In : *International Journal of Pattern Recognition and Artificial Intelligence* 06.02n03 (1992), p. 353–385. DOI : 10.1142/S0218001492000229.

[16] Johan FREDRIKSSON, Olof ENQVIST et Fredrik KAHL. « Fast and Reliable Two-View Translation Estimation ». In : *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[17] A. FUSIELLO, F. CROSILLA et F. MALAPELLE. « Procrustean Point-Line Registration and the NPnP Problem ». In : *International Conference on 3D Vision (3DV 2015)*. Oct. 2015, p. 250–255. DOI : 10.1109/3DV.2015.35.

[18] V. GARRO, F. CROSILLA et A. FUSIELLO. « Solving the PnP Problem with Anisotropic Orthogonal Procrustes Analysis ». In : *International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission (3DIMPVT 2012)*. Oct. 2012, p. 262–269. DOI : 10.1109/3DIMPVT.2012.40.

[19] Rafael Grompone von GIOI, Jérémie JAKUBOWICZ, Jean-Michel MOREL et Gregory RANDALL. « LSD : a Line Segment Detector ». In : *Image Process. On Line (IPOL 2012)* 2 (2012). http://dx.doi.org/10.5201/ipol.2012.gjmr-lsd, p. 35–55.

[20] Rafael Grompone von GIOI, Jérémie JAKUBOWICZ, Jean-Michel MOREL et Gregory RANDALL. « On Straight Line Segment Detection ». In : *Journal of Mathematical Imaging and Vision* 32.3 (nov. 2008), p. 313–347. ISSN : 0924-9907 (Print) 1573-7683 (Online). URL : http://www.springerlink.com/content/x660u50112527k23/.

[21] Venu Madhav GOVINDU. « Combining Two-view Constraints For Motion Estimation ». In : *Conference on Computer Vision and Pattern Recognition (CVPR 2001)*. 2001.

[22] Allen R. HANSON, Edward M. RISEMAN et J. Brian BURNS. « Extracting Straight Lines ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8 (1986), p. 425–455. ISSN : 0162-8828. DOI : doi.ieeecomputersociety.org/10.1109/TPAMI.1986.4767808.

[23] Chris HARRIS et Mike STEPHENS. « A combined corner and edge detector ». In : *In Proc. of Fourth Alvey Vision Conference*. 1988, p. 147–151.

[24] R. I. HARTLEY et A. ZISSERMAN. *Multiple View Geometry in Computer Vision*. Second. Cambridge University Press, ISBN : 0521540518, 2004.

[25] Richard I. HARTLEY. « In Defense of the Eight-Point Algorithm ». In : *IEEE Trans. Pattern Anal. Mach. Intell.* 19.6 (juin 1997), p. 580–593. ISSN : 0162-8828. DOI : 10.1109/34.601246. URL : http://dx.doi.org/10.1109/34.601246.

[26] Michal HAVLENA, Akihiko TORII et Tomáš PAJDLA. « Efficient Structure from Motion by Graph Optimization ». In : *11th European Conference on Computer Vision (ECCV 2010)*. Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II, 2010, p. 100–113. ISBN : 978-3-642-15552-9. DOI : 10.1007/978-3-642-15552-9_8.

[27] J. A. HESCH et S. I. ROUMELIOTIS. « A Direct Least-Squares (DLS) method for PnP ». In : *2011 International Conference on Computer Vision*. 2011, p. 383–390. DOI : 10.1109/ICCV.2011.6126266.

[28] Manuel HOFER, Michael MAURER et Horst BISCHOF. « Line3D : Efficient 3D Scene Abstraction for the Built Environment. » In : *GCPR*. Sous la dir. de Juergen GALL, Peter V. GEHLER et Bastian LEIBE. T. 9358. Lecture Notes in Computer Science. Springer, 2015, p. 237–248. ISBN : 978-3-319-24946-9. URL : http://dblp.uni-trier.de/db/conf/dagm/gcpr2015.html#HoferMB15.

[29] Manuel HOFER, Michael DONOSER et Horst BISCHOF. « Semi-Global 3D Line Modeling for Incremental Structure-from-Motion ». In : *British Machine Vision Conference (BMVC 2014)*. Nottingham, UK, September 1-5, 2014, 2014.

[30] Michael HORN'ACEK. « Extracting Vanishing Points across Multiple Views ». Mém.de mast. Favoritenstrasse 9-11/186, A-1040 Vienna, Austria : Institute of Computer Graphics et Algorithms, Vienna University of Technology, sept. 2010. URL : https://www.cg.tuwien.ac.at/research/publications/2010/hornacek-2010-evp/.

[31] Satoshi IKEHATA, Ivaylo BOYADZHIEV, Qi SHAN et Yasutaka FURUKAWA. « Panoramic Structure from Motion via Geometric Relationship Detection ». In : *CoRR* abs/1612.01256 (2016). URL : http://arxiv.org/abs/1612.01256.

[32] ITSEEZ. *Open Source Computer Vision Library*. https://github.com/itseez/opencv. 2015.

[33] Nianjuan JIANG, Zhaopeng CUI et Ping TAN. « A Global Linear Method for Camera Pose Registration ». In : *IEEE International Conference on Computer Vision (ICCV 2013)*. Sydney, Australia, December 1-8, 2013, 2013, p. 481–488. DOI : 10.1109/ICCV.2013.66.

[34] Fredrik KAHL et Richard I. HARTLEY. « Multiple-View Geometry Under the L_∞-Norm ». In : *IEEE Trans. PAMI* 30.9 (2008), p. 1603–1617.

[35] Chelhwon KIM et Roberto MANDUCHI. « Planar Structures from Line Correspondences in a Manhattan World ». In : *Computer Vision – ACCV 2014 : 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part I*. Sous la dir. de Daniel CREMERS, Ian REID, Hideo SAITO et Ming-Hsuan YANG. Cham : Springer International Publishing, 2015, p. 509–524. ISBN : 978-3-319-16865-4. DOI : 10.1007/978-3-319-16865-4_33. URL : http://dx.doi.org/10.1007/978-3-319-16865-4_33.

[36] Jana KOSECKÁ et Zhang WEI. « Extraction, matching, and pose recovery based on dominant rectangular structures. » In : *Computer Vision and Image Understanding* 100.3 (2005), p. 274–293. URL : http://dblp.uni-trier.de/db/journals/cviu/cviu100.html#KoseckaZ05.

[37] Gim Hee LEE. « A Minimal Solution for Non-perspective Pose Estimation from Line Correspondences ». In : *Computer Vision – ECCV 2016 : 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V*. Sous la dir. de Bastian LEIBE, Jiri MATAS, Nicu SEBE et Max WELLING. Cham : Springer International Publishing, 2016, p. 170–185. ISBN : 978-3-319-46454-1. DOI : 10.1007/978-3-319-46454-1_11. URL : http://dx.doi.org/10.1007/978-3-319-46454-1_11.

[38] Jeong-Kyun LEE et Kuk-Jin YOON. « Real-Time Joint Estimation of Camera Orientation and Vanishing Points ». In : *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[39] Vincent LEPETIT, Francesc MORENO-NOGUER et Pascal FUA. « EPnP : An Accurate O(n) Solution to the PnP Problem ». In : *International Journal of Computer Vision* 81.2 (2008), p. 155. ISSN : 1573-1405. DOI : 10.1007/s11263-008-0152-6. URL : http://dx.doi.org/10.1007/s11263-008-0152-6.

[40] Vincent LEPETIT, Francesc MORENO-NOGUER et Pascal FUA. « EPnP : An Accurate O(n) Solution to the PnP Problem ». In : *International Journal of Computer Vision (IJCV 2008)* 81.2 (2008), p. 155–166. ISSN : 1573-1405. DOI : 10.1007/s11263-008-0152-6.

[41] Jose LEZAMA, Rafael Grompone von GIOI, Gregory RANDALL et Jean-Michel MOREL. « Finding Vanishing Points via Point Alignments in Image Primal and Dual Domains ». In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2014)*. 2014.

[42] Kai LI et Jian YAO. « Line segment matching and reconstruction via exploiting coplanar cues ». In : *{ISPRS} Journal of Photogrammetry and Remote Sensing* 125 (2017), p. 33 –49. ISSN : 0924-2716. DOI : http://dx.doi.org/10.1016/j.isprsjprs.2017.01.006. URL : http://www.sciencedirect.com/science/article/pii/S0924271616301708.

[43] Shiqi LI, Chi XU et Ming XIE. « A Robust O(n) Solution to the Perspective-n-Point Problem. » In : *IEEE Trans. Pattern Anal. Mach. Intell.* 34.7 (2012), p. 1444–1450. URL : http://dblp.uni-trier.de/db/journals/pami/pami34.html#LiXX12.

[44] Zhe LIU et Renaud MARLET. « Virtual Line Descriptor and Semi-Local Matching Method for Reliable Feature Correspondence ». In : *British Machine Vision Conference 2012*. United Kingdom, sept. 2012, p. 16.1–16.11. URL : https://hal.archives-ouvertes.fr/hal-00743323.

[45] Juan LÓPEZ, Roi SANTOS, Xose R. FERNÁNDEZ-VIDAL et Xose Manuel PARDO. « Two-view line matching algorithm based on context and appearance in low-textured images ». In : *Pattern Recognition* 48.7 (2015), p. 2164–2184. DOI : 10.1016/j.patcog.2014.11.018. URL : http://dx.doi.org/10.1016/j.patcog.2014.11.018.

[46] David G. LOWE. « Distinctive Image Features from Scale-Invariant Keypoints ». In : *Int. J. Comput. Vision* 60.2 (nov. 2004), p. 91–110. ISSN : 0920-5691. DOI : 10.1023/B:VISI.0000029664.99615.94. URL : http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94.

[47] Y. LU et D. SONG. « Robust RGB-D Odometry Using Point and Line Features ». In : *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, p. 3934–3942. DOI : 10.1109/ICCV.2015.448.

[48] F. Landis MARKLEY. « Attitude determination using vector observations and the singular value decomposition ». In : *The Journal of the Astronautical Sciences* 36.3 (1988), p. 245–258.

[49] Krystian MIKOLAJCZYK et Cordelia SCHMID. « A Performance Evaluation of Local Descriptors ». In : *IEEE Trans. Pattern Anal. Mach. Intell.* 27.10 (oct. 2005), p. 1615–1630. ISSN : 0162-8828. DOI : 10.1109/TPAMI.2005.188. URL : http://dx.doi.org/10.1109/TPAMI.2005.188.

[50] Faraz M. MIRZAEI et Stergios I. ROUMELIOTIS. « Globally optimal pose estimation from line correspondences. » In : *ICRA*. IEEE, 2011, p. 5581–5588. URL : http://dblp.uni-trier.de/db/conf/icra/icra2011.html#MirzaeiR11.

[51] Faraz M. MIRZAEI et Stergios I. ROUMELIOTIS. « Globally optimal pose estimation from line correspondences ». In : *IEEE International Conference on Robotics and Automation (ICRA 2011)*. IEEE, 2011, p. 5581–5588.

[52] Faraz M. MIRZAEI et Stergios I. ROUMELIOTIS. « Optimal estimation of vanishing points in a Manhattan world ». In : *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*. 2011, p. 2454–2461. DOI : 10.1109/ICCV.2011.6126530. URL : http://dx.doi.org/10.1109/ICCV.2011.6126530.

[53] Lionel MOISAN et Bérenger STIVAL. « A Probabilistic Criterion to Detect Rigid Point Matches Between Two Images and Estimate the Fundamental Matrix ». In : *Int. J. Comput. Vision* 57.3 (mai 2004), p. 201–218. ISSN : 0920-5691. DOI : 10.1023/B:VISI.0000013094.38752.54. URL : http://dx.doi.org/10.1023/B:VISI.0000013094.38752.54.

[54] Lionel MOISAN, Pierre MOULON et Pascal MONASSE. « Automatic Homographic Registration of a Pair of Images, with A Contrario Elimination of Outliers ». In : *Image Process. On Line (IPOL)* 2 (2012). http://dx.doi.org/10.5201/ipol.2012.mmm-oh, p. 56–73.

[55] Lionel MOISAN, Pierre MOULON et Pascal MONASSE. « Automatic Homographic Registration of a Pair of Images, with A Contrario Elimination of Outliers ». In : *Image Process. On Line (IPOL)* 2 (2012). http://dx.doi.org/10.5201/ipol.2012.mmm-oh, p. 56–73.

[56] Lionel MOISAN, Pierre MOULON et Pascal MONASSE. « Fundamental Matrix of a Stereo Pair, with A Contrario Elimination of Outliers ». In : *Image Process. On Line (IPOL)* 6 (2016). http://dx.doi.org/10.5201/ipol.2016.147, p. 89–113.

[57] J.M.M. MONTIEL, J.D. TARDÓS et L. MONTANO. « Structure and motion from straight line segments ». In : *Pattern Recognition (PR 2000)* 33.8 (2000), p. 1295–1307. ISSN : 0031-3203. DOI : http://dx.doi.org/10.1016/S0031-3203(99)00117-X.

[58] Pierre MOULON, Pascal MONASSE et Renaud MARLET. « Adaptive Structure from Motion with a Contrario Model Estimation ». In : *Proceedings of the 11th Asian Conference on Computer Vision - Volume Part IV*. ACCV'12. Daejeon, Korea : Springer-Verlag, 2013, p. 257–270. ISBN : 978-3-642-37446-3. DOI : 10.1007/978-3-642-37447-0_20. URL : http://dx.doi.org/10.1007/978-3-642-37447-0_20.

[59] Pierre MOULON, Pascal MONASSE et Renaud MARLET. « Global fusion of relative motions for robust, accurate and scalable structure from motion ». In : *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2013, p. 3248–3255.

[60] Marcos NIETO et Luis SALGADO. « Non-linear optimization for robust estimation of vanishing points ». In : *Proceedings of the International Conference on Image Processing, ICIP 2010, September 26-29, Hong Kong, China*. 2010, p. 1885–1888. DOI : 10.1109/ICIP.2010.5652381. URL : http://dx.doi.org/10.1109/ICIP.2010.5652381.

[61] David NISTÉR. « An Efficient Solution to the Five-Point Relative Pose Problem ». In : *IEEE Trans. Pattern Anal. Mach. Intell.* 26.6 (juin 2004), p. 756–777. ISSN : 0162-8828. DOI : 10.1109/TPAMI.2004.17. URL : http://dx.doi.org/10.1109/TPAMI.2004.17.

[62] Carl OLSSON et Olof ENQVIST. « Stable Structure from Motion for Unordered Image Collections ». In : *SCIA*. LNCS 6688. 2011, p. 524–535.

[63] J. PHILIP et Kungl. Tekniska högskolan. Department of MATHEMATICS. *Critical Point Configurations of the 5-, 6-, 7-, and 8-point Algorithms for Relative Orientation*. TRITA / MAT / MA : TRITA. Department of Mathematics, Royal Institute of Technology, 1998. URL : https://books.google.fr/books?id=ETzCHAAACAAJ.

[64] V. PRADEEP et J. LIM. « Egomotion using assorted features ». In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2010)*. Juin 2010, p. 1514–1521. DOI : 10.1109/CVPR.2010.5539792.

[65] Bronislav PŘIBYL, Pavel ZEMČÍK et Martin ČADIK. « Camera Pose Estimation from Lines using Plücker Coordinates ». In : *British Machine Vision Conference (BMVC 2015)*. Sous la dir. de Xianghua XIE, Mark W. JONES et Gary K. L. TAM. BMVA Press, sept. 2015, p. 45.1–45.12. ISBN : 1-901725-53-7. DOI : 10.5244/C.29.45.

[66] Srikumar RAMALINGAM, Sofien BOUAZIZ et Peter F. STURM. « Pose estimation using both points and lines for geo-localization ». In : *IEEE International Conference on Robotics and Automation (ICRA 2011)*. Shanghai, China, 9-13 May 2011, 2011, p. 4716–4723. DOI : 10.1109/ICRA.2011.5979781.

[67] Antonio L. RODRÍGUEZ, Pedro E. López-de TERUEL et Alberto RUIZ. « Reduced epipolar cost for accelerated incremental SfM ». In : *Conference on Computer Vision and Pattern Recognition (CVPR 2011)*. 2011.

[68] Yohann SALAÜN, Renaud MARLET et Pascal MONASSE. « Multiscale Line Segment Detector for robust and accurate SfM ». In : *23rd International Conference on Pattern Recognition (ICPR)*. 2016.

[69] Yohann SALAÜN, Renaud MARLET et Pascal MONASSE. « Robust and Accurate Line- and/or Point-Based Pose Estimation without Manhattan Assumptions ». In : *European Conference on Computer Vision (ECCV)*. 2016.

[70] Peter SCHÖNEMANN. « A generalized solution of the orthogonal procrustes problem ». In : *Psychometrika* 31.1 (1966), p. 1–10. URL : http://EconPapers.repec.org/RePEc:spr:psycho:v:31:y:1966:i:1:p:1-10.

[71] K. SIM et R. HARTLEY. « Recovering Camera Motion Using L_∞ Minimization ». In : *Conference on Computer Vision and Pattern Recognition (CVPR 2006)*. T. 1. 2006.

[72] Noah SNAVELY, Steven M. SEITZ et Richard SZELISKI. « Photo Tourism : Exploring Photo Collections in 3D ». In : *ACM Transations on Graphics (TOG 2006)* 25.3 (juil. 2006), p. 835–846. ISSN : 0730-0301. DOI : 10.1145/1141911.1141964.

[73] C. STRECHA, W. von HANSEN, L. VAN GOOL, P. FUA et U. THOENNESSEN. « On benchmarking camera calibration and multi-view stereo for high resolution imagery ». In : *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2008)*. 2008, p. 1–8. DOI : 10.1109/CVPR.2008.4587706.

[74] Changming SUN. « 2 Point Linear Algorithm for Camera Translation Vector Estimation with Known Rotation ». In : *Robotica* 18.5 (sept. 2000), p. 557–561. ISSN : 0263-5747. DOI : 10.1017/S0263574799002520. URL : https://doi.org/10.1017/S0263574799002520.

[75] Jean-Philippe TARDIF. « Non-iterative approach for fast and accurate vanishing point detection ». In : *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*. 2009, p. 1250–1257. DOI : 10.1109/ICCV.2009.5459328. URL : http://dx.doi.org/10.1109/ICCV.2009.5459328.

[76] Roberto TOLDO, Riccardo GHERARDI, Michela FARENZENA et Andrea FUSIELLO. « Hierarchical Structure-and-motion Recovery from Uncalibrated Images ». In : *Computer Vision and Image Understanding (CVIU 2015)* 140.C (nov. 2015), p. 127–143. ISSN : 1077-3142. DOI : 10.1016/j.cviu.2015.05.011.

[77] P.H.S. TORR et D.W. MURRAY. « The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix ». In : *International Journal of Computer Vision* 24.3 (1997), p. 271–300. ISSN : 1573-1405. DOI : 10.1023/A:1007927408552. URL : http://dx.doi.org/10.1023/A:1007927408552.

[78] Lu WANG, Ulrich NEUMANN et Suya YOU. « Wide-baseline image matching using Line Signatures. » In : *ICCV*. IEEE Computer Society, 2009, p. 1311–1318. ISBN : 978-1-4244-4419-9. URL : http://dblp.uni-trier.de/db/conf/iccv/iccv2009.html#WangNY09.

[79] Zhiheng WANG, Fuchao WU et Zhanyi HU. « MSLD : A robust descriptor for line matching ». In : *Pattern Recognition* 42.5 (2009), p. 941–953. ISSN : 0031-3203. DOI : http://dx.doi.org/10.1016/j.patcog.2008.08.035. URL : http://www.sciencedirect.com/science/article/pii/S0031320308003658.

[80] J. WENG, T. S. HUANG et N. AHUJA. « Motion and structure from line correspondences ; closed-form solution, uniqueness, and optimization. » In : *PAMI* 14.3 (1992), p. 318–336.

[81] Changchang WU. « Towards Linear-Time Incremental Structure from Motion ». In : *International Conference on 3D Vision (3DV 2013)*. 2013, p. 127–134. DOI : 10.1109/3DV.2013.25.

[82] Chi XU, Lilian ZHANG, Li CHENG et Reinhard KOCH. « Pose Estimation from Line Correspondences : A Complete Analysis and A Series of Solutions ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI 2016)* (juin 2016).

[83] Yiliang XU, Sangmin OH et A. HOOGS. « A Minimum Error Vanishing Point Detection Approach for Uncalibrated Monocular Images of Man-Made Environments ». In : *IEEE Computer Vision and Pattern Recognition (CVPR 2013)*. 2013, p. 1376–1383. DOI : 10.1109/CVPR.2013.181.

[84] Lilian ZHANG et Reinhard KOCH. « An Efficient and Robust Line Segment Matching Approach Based on LBD Descriptor and Pairwise Geometric Consistency ». In : *J. Vis. Comun. Image Represent.* 24.7 (oct. 2013), p. 794–805. ISSN : 1047-3203. DOI : 10.1016/j.jvcir.2013.05.006. URL : http://dx.doi.org/10.1016/j.jvcir.2013.05.006.

[85] Lilian ZHANG et Reinhard KOCH. « Structure and motion from line correspondences : Representation, projection, initialization and sparse bundle adjustment ». In : *Journal of Visual Communication and Image Representation* 25.5 (2014), p. 904–915. ISSN : 1047-3203. DOI : http://dx.doi.org/10.1016/j.jvcir.2014.02.013. URL : http://www.sciencedirect.com/science/article/pii/S1047320314000510.

[86] Lilian ZHANG, Chi XU, Kok-Meng LEE et Reinhard KOCH. « Robust and Efficient Pose Estimation from Line Correspondences ». In : *11th Asian Conference on Computer Vision (ACCV 2012)*. Sous la dir. de Kyoung Mu LEE, Yasuyuki MATSUSHITA, James M. REHG et Zhanyi HU. Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part III : Springer, 2012, p. 217–230. ISBN : 978-3-642-37431-9. DOI : 10.1007/978-3-642-37431-9_17.

[87] Lilian ZHANG, Chi XU, Kok-Meng LEE et Reinhard KOCH. « Robust and Efficient Pose Estimation from Line Correspondences ». In : *Computer Vision – ACCV 2012 : 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part III*. Sous la dir. de Kyoung Mu LEE, Yasuyuki MATSUSHITA, James M. REHG et Zhanyi HU. Berlin, Heidelberg : Springer Berlin Heidelberg, 2013, p. 217–230. ISBN : 978-3-642-37431-9. DOI : 10.1007/978-3-642-37431-9_17. URL : http://dx.doi.org/10.1007/978-3-642-37431-9_17.

[88] Lilian ZHANG, Huimin LU, Xiaoping HU et Reinhard KOCH. « Vanishing Point Estimation and Line Classification in a Manhattan World with a Unifying Camera Model ». In : *International Journal of Computer Vision* 117.2 (2016), p. 111–130. ISSN : 1573-1405. DOI : 10.1007/s11263-015-0854-5. URL : http://dx.doi.org/10.1007/s11263-015-0854-5.

[89] Zhengyou ZHANG. « Estimating motion and structure from correspondences of line segments between two perspective images ». In : *5th International Conference on Computer Vision (ICCV 1995)*. 1995, p. 257–262. DOI : 10.1109/ICCV.1995.466777.

[90] Zhengyou ZHANG. « Estimating motion and structure from correspondences of line segments between two perspective images ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI 1995)* 17.12 (déc. 1995), p. 1129–1139. ISSN : 0162-8828. DOI : 10.1109/34.476506.

[91] Y. ZHENG, Y. KUANG, S. SUGIMOTO, K. ÅSTRÖM et M. OKUTOMI. « Revisiting the PnP Problem : A Fast, General and Optimal Solution ». In : *IEEE International Conference on Computer Vision (ICCV 2013)*. Déc. 2013, p. 2344–2351. DOI : 10.1109/ICCV.2013.291.