Etalonnage de caméras à champs disjoints et reconstruction 3D : Application à un robot mobile

Pierre Lébraly

N° d'ordre : D.U : 2216
EDSPIC : 551

UNIVERSITÉ BLAISE PASCAL - CLERMONT II

Ecole Doctorale
Sciences Pour L'Ingénieur De Clermont-Ferrand

Thèse

présentée par :

PIERRE LÉBRALY

Ingénieur ENSEA

pour obtenir le grade de

DOCTEUR D'UNIVERSITÉ

Spécialité : Vision pour la robotique

Étalonnage de caméras à champs disjoints et reconstruction 3D - Application à un robot mobile

---

Soutenue publiquement le 18 janvier 2012 devant le jury :

M.	Omar AIT-AIDER	Examinateur
M.	François CHAUMETTE	Président du jury
M.	Michel DHOME	Directeur de thèse
M.	Richard HARTLEY	Rapporteur et examinateur
M.	Eric ROYER	Examinateur
M.	Peter STURM	Rapporteur et examinateur

Remerciements

Ce mémoire présente mes travaux menés dans le groupe GRAVIR du LASMEA au sein de l'équipe ComSee qui se consacre à la vision par ordinateur. Ils ont été cofinancés par l'Union européenne (l'Europe s'engage en Auvergne avec les Fonds européen de développement régional), la région Auvergne et Clermont Communauté dans le cadre du projet VIPA (Véhicule Individuel Public Autonome).

Je tiens à remercier François Chaumette pour avoir présidé le jury, mais aussi pour avoir lu et annoté en détails ce manuscrit comme l'aurait fait un rapporteur. Je remercie également Richard Hartley et Peter Sturm pour avoir accepté d'être rapporteur de ma thèse, mais aussi la qualité de leur rapport et annotations. Je remercie les membres de mon jury, pour l'intérêt porté à ce manuscrit, pour la pertinence de leurs remarques et pour toute la sympathie qu'ils m'ont adressée.

Un grand merci à l'équipe VIPA au grand complet, avec une mention toute particulière à Datta et Clément qui, à travers nos nombreuses réflexions, travaux, manips et démos réalisés ensemble, ont largement outrepassé leur tâche et réalisé un véritable travail de recherche et développement (SFM hierarchique, Boucleur, code-barre circulaire des cibles,...). Merci également à Eric et Omar (mes encadrants), Michel (mon directeur de thèse) et Thierry (mon chef d'équipe ComSee) pour leur soutien, conseils et aide précieuse tout au long de la thèse et notamment lors des relectures d'articles.

Merci également à mes deux collègues de bureau : Baptiste et notre ex-responsable de notre bureau shifu-Shuda, pour les discutions riches et animées, les relectures d'articles, et nombreuses remarques pertinentes ; Maxime et Jean-Thierry pour nos échanges mathématiques ; Christophe pour ses conseils informatiques ; Laurent (source intarissable de blagues...) et Ahmed qui ont fini de porter en C++ une version affine de la détection des cibles (à partir de mon code C/Matlab™) ; l'équipe technique Serge et François pour tous leurs travaux ayant permis les manips et démonstrations sur PAVIN ; Jonathan pour son aide lors d'une expérimentation ; Sameer qui a corrigé quelques fautes d'un article dans la langue de Shakespeare ; les 4 stagiaires et étudiants (de l'Isima et du master informatique et système) que j'ai co-encadrés pendant cette thèse pour leur écoute et travaux respectifs ; mes collègues de l'IUT (lors des enseignements à ma charge en parallèle de la thèse) pour leur accueil et sympathie ; Youcef pour l'organisation lors de ma participation à une démonstration sur un VipaLab en Corée du Sud.

De manière générale (et surtout, pour n'oublier personne), merci à l'ensemble des doctorants, ex-doctorants, post-doc, CDD, titulaires, équipe technique et administrative pour la convivialité, l'ambiance et l'enrichissante aventure humaine durant ces trois années.

Enfin, je tiens tout particulièrement à remercier Sandra, ma famille et belle-famille !

Table des matières

Table des matières **V**

Liste des figures **XI**

Liste des tableaux **XV**

Liste des algorithmes **XVII**

Notations **XIX**

1 Introduction **1**
1.1 Contexte 1
1.1.1 Le projet VIPA 1
1.1.2 La plateforme expérimentale 2
1.1.3 Les véhicules 3
1.2 Problématique 4
1.3 Plan du mémoire 4
1.4 Contributions 5

2 Etalonnage intrinsèque d'une caméra **7**
2.1 Modèles intrinsèques 7
2.1.1 Modèle sténopé 7
2.1.2 Modèle sténopé et distorsions radiales 9
2.1.2.1 Modèle direct 10
2.1.2.2 Modèle indirect 11
2.1.2.3 Discussions et remarques 12
2.1.3 Modèle unifié 13
2.2 Mire d'étalonnage 16
2.2.1 Cibles 16
2.2.1.1 Pourquoi utiliser des cercles concentriques ? 16
2.2.1.2 Détection automatique 17

2.2.1.3 Raffinement sous-pixellique automatique . . . 18
2.2.1.4 Labellisation automatique . . . 22
2.2.2 Mire de cibles et protocole d'étalonnage . . . 23
2.3 Etalonnage intrinsèque et cibles . . . 24
2.3.1 Principe . . . 25
2.3.2 Avantages/inconvénients par rapport à l'existant . . . 26
2.3.3 Résultats . . . 27
2.4 Correction automatique d'images distordues . . . 30
2.4.1 Etat de l'art . . . 30
2.4.2 Notre approche . . . 31
2.4.3 Résultats . . . 35
2.4.4 Perspectives . . . 35

3 Etat de l'art : modélisation et utilisation des systèmes multi-caméra à champs non-recouvrants 39
3.1 Modélisations multi-caméra . . . 39
3.1.1 Modèles existants . . . 40
3.1.2 Modèle retenu . . . 41
3.2 Utilisation des systèmes multi-cameras . . . 42
3.2.1 A champs recouvrants . . . 42
3.2.2 A champs non-recouvrants . . . 42
3.2.2.1 Statiques . . . 42
3.2.2.2 Embarquées . . . 43

4 Étalonnage extrinsèque multi-caméra et miroir 45
4.1 Introduction et état de l'art . . . 45
4.2 Formalisation mathématique . . . 47
4.2.1 Vue d'ensemble . . . 47
4.2.2 Paramètres intrinsèques d'une caméra virtuelle . . . 48
4.2.3 Calcul de la pose du miroir . . . 50
4.3 Étalonnage de caméras à champs disjoints . . . 50
4.3.1 Avec une mire connue . . . 50
4.3.2 Sans mire connue . . . 52
4.3.3 Appliqué à un robot mobile . . . 53
4.4 Influence de la réfraction . . . 54
4.4.1 Caméra perspective et réfraction induite par le miroir . . . 54
4.4.2 Influence de la réfraction sur le plan image . . . 55
4.4.3 Estimation de pose et réfraction . . . 57
4.5 Résultats . . . 58
4.5.1 Avec des données synthétiques . . . 58
4.5.2 Avec des données réelles . . . 59

4.5.3 Réfraction et étalonnage 62
4.6 Conclusions et perspectives 63
4.6.1 Conclusions 63
4.6.2 Perspectives 63

5 Etalonnage extrinsèque multi-caméra grâce aux mouvements 65
5.1 Introduction et état de l'art 65
5.2 Manœuvres et cibles 69
5.2.1 Formalisation du problème 69
5.2.2 Initialisation linéaire et cas singuliers 71
5.2.2.1 Mouvements 3D : cas général 71
5.2.2.2 Mouvements singuliers 74
5.2.2.3 Solutions pour traiter les cas singuliers 77
5.3 Structure from motion et auto-étalonnage extrinsèque 82
5.3.1 Formalisation du problème 82
5.3.2 Algorithme proposé 83
5.3.3 Structure from motion 84
5.3.4 Fermeture de boucles 86
5.4 Ajustement de faisceaux multi-caméra (MCBA) 89
5.4.1 MCBA et état de l'art 89
5.4.2 Vue d'ensemble 90
5.4.3 Algorithme LM et équations normales 91
5.4.4 Exploitation de la structure creuse 93
5.4.4.1 Des équations normales jusqu'au Système Caméras/Poses . . 93
5.4.4.2 Expression creuse des équations normales 94
5.4.5 Résolution du Système Caméras/Poses 98
5.4.5.1 Factorisation de Cholesky et complément de Schur 98
5.4.5.2 Gradient conjugué préconditionné 99
5.4.6 Complexité d'une itération du LM 101
5.4.7 Comparaison des implémentations 106
5.4.7.1 Manœuvres 106
5.4.7.2 Parcours 107
5.5 Résultats 108
5.5.1 Manœuvres et cibles 108
5.5.1.1 Résultats avec des données synthétiques 108
5.5.1.2 Résultats avec des données réelles 109
5.5.1.3 Paire stéréo 109
5.5.1.4 Système multi-caméra embarqué 111
5.5.2 Structure From Motion et auto-étalonnage extrinsèque 112
5.5.2.1 Résultats avec des données synthétiques 112
5.5.2.2 Résultats avec des données réelles 116

5.5.3 Résultats après étalonnage . . . 116
5.5.3.1 Reconstruction de quelques centaines de mètres . . . 116
5.5.3.2 Reconstructions de plusieurs kilomètres . . . 117
5.5.3.3 Reconstruction multi-caméra et dérive du facteur d'échelle . 117
5.6 Conclusions et perspectives . . . 118
5.6.1 Conclusions . . . 118
5.6.2 Perspectives . . . 119

6 Etalonnage du véhicule **129**
6.1 Etat de l'art . . . 129
6.2 Première approche : ligne droite et foyer d'expansion . . . 131
6.2.1 Principe . . . 131
6.2.2 Résultats . . . 133
6.2.3 Perspectives . . . 134
6.3 Modèle cinématique et étalonnage "essieu-caméra" . . . 136
6.3.1 Principe . . . 136
6.3.2 Résultats . . . 141
6.3.2.1 Résultats avec des données synthétiques . . . 141
6.3.2.2 Résultats avec des données réelles . . . 145
6.3.3 Conclusion et perspectives . . . 151

7 Conclusion et perspectives **153**
7.1 Conclusion . . . 153
7.2 Perspectives . . . 154

Annexe A **157**
A.1 Changement de base et matrice de passage . . . 157
A.2 Quelques transformations géométriques . . . 158
A.2.1 Transformation euclidienne . . . 158
A.2.2 Similitude directe . . . 158
A.2.3 Affinité . . . 158
A.2.4 Transformation projective . . . 159
A.2.5 De la transformation euclidienne à la transformation projective . . . 159
A.3 Valeurs et vecteurs propres . . . 160
A.3.1 Matrice de rotation . . . 160
A.3.2 Produit de Kronecker . . . 161

Annexe B **163**
B.1 Changement de repère monde/caméras . . . 163
B.2 Fonction de coût . . . 163
B.3 Jacobienne analytique . . . 164

B.3.1 Propositions préalables 164
B.3.2 Blocs analytiques du Jacobien 165
B.4 MCBA : Détail de l'expression des blocs 166
B.4.1 De l'équation normale 166
B.4.2 Du Système Caméra-Pose (CPS) 167

Publications dans le cadre de cette thèse **169**

Bibliographie **171**

Glossaire **187**

Liste des figures

1.1 Logos des membres du consortium VIPA, des partenaires et des financeurs. . . 2
1.2 Plateforme d'Auvergne pour les Vehicule Intelligent (PAVIN), cite des Cézeaux. 2
1.3 Véhicules développés dans le cadre du projet VIPA. 3

2.1 Modèle perspectif d'une caméra. 8
2.2 Caméra perspective et distorsions directes. 10
2.3 Caméra perspective et distorsions indirectes. 11
2.4 Modèle de caméra unifié. 14
2.5 Cercles concentriques auxquels une homographie a été appliquée. 18
2.6 Intensité lumineuse $\mathcal{L}(r)$ d'une cible, en fonction du rayon r dans le modèle. . 19
2.7 Homographie entre le modèle fronto-parallèle d'une cible et son acquisition dans une image. 20
2.8 Détection sous-pixellique d'ellipses concentriques, avant et après convergence. 21
2.9 Exemple de cible (disque noir, entouré d'une couronne noire, elle-même entourée par un code-barre circulaire). 22
2.10 Cible identifiée pour une vue directe ou indirecte (par son reflet sur un miroir). 23
2.11 16 prises de vue conseillées pour étalonner intrinsèquement une caméra. 24
2.12 Comparaison entre les méthodes "classiques" d'étalonnage intrinsèque basés mire et notre méthode. 26
2.13 Correction de la distorsion après l'étalonnage intrinsèque (algorithme 1). . . . 29
2.14 Correction automatique d'images distordues. 32
2.15 Principe de correction automatique des distorsions d'une image. 33
2.16 Paramétristion d'une droite portée par la primitive (u'_i, v'_i, θ'_i). 34
2.17 Principe du calcul d'une primitive (u'_i, v'_i, θ'_i) déformée par la fonction F_Υ, à partir de la primitive détectée (u_i, v_i, θ_i). 34
2.18 Fonction de coût pour l'image 2.14a et le modèle unifié en fonction du paramètre ξ. 36
2.19 Cartes de densité de probabilité des ρ_i, θ_i avec (a) les paramètres initiaux $\Upsilon_0 = \xi = 0$ et (b) après convergence avec $\Upsilon = \xi = 0.400$ (modèle unifié). 37

2.20 (a) Primitives issues de l'image 2.14a ayant subi la déformation avec le modèle indirect après convergence de l'algorithme. (b) Image corrigée à partir des paramètres du modèle indirect estimés par l'algorithme d'auto-correction des distorsions. 37

3.1 Systèmes multi-caméra à champs non-recouvrants ou partiellement recouvrants. 40

4.1 Quelques utilisations de miroirs plans en vision artificielle. 46
4.2 Schéma du système multi-caméra / miroir. 47
4.3 Image issue d'une caméra réelle (a) et image virtuelle (b) obtenue par symétrie par rapport à l'axe vertical passant par le point principal. 49
4.4 Réfraction d'un miroir plan observé par une caméra centrale 55
4.5 Erreur de reprojection dans le plan image si la réfraction est négligée. 56
4.6 Erreur err_{pix} dans le plan image si l'on négligeait la réfraction (en fonction de l'angle d'incidence i_{c_0} et de la distance CP' entre la caméra et le point). 57
4.7 Résultats avec des données synthétiques : erreur moyenne sur l'estimation de pose en fonction du niveau de bruit 59
4.8 Banc de métrologie et son schéma cinématique 60
4.9 Expérience avec des poses relatives - translations pures 60
4.10 Expérience avec une scène inconnue en vue directe (a), et indirecte (b). La résolution des images est de 1600x1200. 61
4.11 Impact de la réfraction sur la précision de l'Algorithme 3 (données synthétiques). 62

5.1 Système multi-caméra à champs non-recouvrants se déplaçant dans un environnement statique. 70
5.2 Vue d'ensemble de l'étalonnage extrinsèque. 71
5.3 Mouvements de vissage selon l'axe $\mathbf{a}$ (cas singulier n°1) appliqués au système multi-caméra (vue de dessus) 76
5.4 Mouvement plan (cas singulier n°4). 76
5.5 Système multi-caméra à champs non-recouvrants ayant subit une permutation entre les poses n°0 et n°k. Pour la pose n°0, C_1 observe S_1 et C_2 observe S_2, et inversement pour la pose n°k. 80
5.6 La permutation des scènes observées (via un demi-tour) permet de contraindre la hauteur relative des caméras. 81
5.7 Schéma prouvant intuitivement que la permutation des scènes (via un demi-tour) permet d'estimer une unique hauteur relative des caméras. 82
5.8 Caméras $C_1..C_j..C_N$ à champs disjoints se déplaçant le long d'une scène statique. Les flèches pleines indiquent les transformations optimisées par le MCBA (Section 5.4). Les points 3D $\overline{P}_1..\overline{P}_i..\overline{P}_M$ sont également optimisés. . 83
5.9 Algorithme de fermeture de boucles multi-caméra en fonctionnement sur une trajectoire synthétique. 87

5.10 Schéma prouvant que la permutation des scènes (via un demi-tour) ne permet pas forcément de contraindre l'écart latéral des caméras. 88
5.11 Structure primaire de la matrice jacobienne J pour le MCBA. 95
5.12 Forme creuse des équations normales (5.38) de l'ajustement de faisceaux multi-caméra (MCBA). 96
5.13 Illustration des différents index utilisés entre les poses du système multi-caméra, les points 3D et les caméras. 105
5.14 Matrice $\bar{\mathbf{S}}$ pour une séquence de 376 poses clés, dans le cas d'une reconstruction sans fermeture de boucle (a) ou bien dans le cas de fermeture de boucle (b). . . 107
5.15 Résultats avec des données synthétiques : erreur d'étalonnage extrinsèque en fonction du niveau de bruit. 109
5.16 Étalonnage d'une paire stéréo à champs recouvrants, sous l'hypothèse de champs de vue disjoints. 110
5.17 Étalonnage d'une paire stéréo avec des caméras embarquées à champs non-recouvrants. 112
5.18 (a)Une des images synthétiques utilisée comme entrée du SFM. Carte 3D reconstruite avec (b) le SFM aux paramètres extrinsèques approximatifs constants, puis optimisée par le MCBA (c), ou bien bouclée (avec l'algorithme 6) puis optimisée par le MCBA (d). Les points rouges et bleus sont respectivement observés par la caméra avant et arrière. 114
5.19 Nombre d'inliers 2D pour les différentes reconstructions. 115
5.20 (a) Image issue d'une séquence réelle (caméras Pixelink™). Vue de dessus de la carte reconstruite par le SFM avec : (b) uniquement la caméra avant, (c) uniquement la caméra arrière, (d) les deux caméras et des paramètres extrinsèques constants et approximatifs. En appliquant le MCBA à (d), on obtient la carte illustrée par (e) et (f). 122
5.21 Vue de côté de la carte reconstruite par le SFM avec des paramètres extrinsèques approximatifs constants (a), puis optimisée par le MCBA sans (b) ou avec (c) fermeture de boucle. Pour que la dérive soit bien visible, trois tours du rond-point ont été faits avec le véhicule. 123
5.22 Carte reconstruite par le SFM avec des paramètres extrinsèques approximatifs et constants (a), puis optimisée par le MCBA avec fermeture de boucle (b). . . 124
5.23 Reconstruction sur PAVIN d'une séquence de 175 m après avoir étalonné les caméras du VipaLab. 125
5.24 Interface logicielle du VIPA se localisant sur une trajectoire d'environ 1.3 km au challenge Bibendum de Berlin. 126
5.25 Localisation bi-caméra du véhicule dans la carte 3D apprise au préalable. . . . 126
5.26 Trajectoire d'environ 3.5 km sur le campus des Cézeaux. En revenant au point de départ, on constate une dérive d'environ 4 m, sans qu'aucun algorithme de fermeture de boucle ne soit appliqué. Image satellite du Géoportail de l'IGN. . 127

6.1 Caméra statique observant des points 3D se déplaçant rectilignement selon un même vecteur vitesse $\mathbf{v}$. 131
6.2 Principe de détection du foyer d'expansion. 133
6.3 Quelle précision de détection du foyer d'expansion est nécessaire pour une caméra perspective, avec une précision angulaire et un angle de lacet φ donnés ? . 134
6.4 Détermination partielle de l'orientation de la caméra par rapport au véhicule se déplaçant en ligne droite, à partir du foyer d'expansion (pour trois séquences d'images). 135
6.5 Schéma illustrant un véhicule doté d'une caméra se déplaçant entre les instants k et $k+1$. Les poses de l'essieu et de la caméra sont définies dans un repère monde. 136
6.6 Schéma illustrant le modèle cinématique du véhicule (modèle tricycle). 138
6.7 Vue de dessus de la trajectoire synthétique de l'essieu du véhicule et de la trajectoire suivis par la caméra. Le mouvement est plan, mais la pose de la caméra par rapport au véhicule est bien une transformation euclidienne de l'espace. . . 142
6.8 Expériences à l'aide du simulateur. 143
6.9 (a) Vue de dessus du véhicule, avec les deux dispositions pour la paire de caméras à champs non-recouvrant. Pour la disposition n°1, la caméra avant est légèrement orientée vers la gauche (avec angle de lacet d'environ $4°$). (b) VipaLab avec les caméras Marlin™ dans la disposition n°1 (rouge), les caméras sont ensuite déplacées dans la position n°2 (bleu). 146
6.10 (a) Vue 3D de la reconstruction de la trajectoire en zigzag servant à l'étalonnage essieu-caméra (pour la disposition n°1). Exemples d'images avant (b) et arrière (c) issues de la séquence. 147
6.11 Traces GPS de différents rejeux de la trajectoire de référence 1_{ref}. 149
6.12 Zoom sur les traces GPS de différents rejeux de la trajectoire de référence 1_{ref}. 150

A.1 Illustration de quelques transformations géométriques du plan. 162

Liste des tableaux

2.1 Résultats de trois étalonnages intrinsèques qui utilisent le modèle direct, indirect ou bien unifié. 28

4.1 Récapitulatif des notations utilisées dans ce chapitre. 47
4.2 Résultats expérimentaux de la procédure d'étalonnage appliquée à un robot mobile (*cf* Section 4.3.3) 61

5.1 Nombre de degrés de liberté observables des paramètres extrinsèques avec K mouvements du système multi-caméra. L'échelle est supposée connue. 75
5.2 Récaptitulatif des index utilisés par le MCBA et la valeur moyenne de leur cardinal. 102
5.3 Complexité de chaque étape d'une itération du LM du MCBA implémenté avec la structure creuse secondaire. 103
5.4 Gains en temps de calcul d'une implémentation creuse pour une itération du MCBA en fonction du nombre de points 3D et du nombre de zéros de la hessienne. 106
5.5 Comparaison de la précision des étalonnages extrinsèques en fonction de l'hypothèse de champs de vue recouvrants, du mouvement et de la permutation de scènes. 111
5.6 Comparaison de la précision et de la répétabilité des étalonnages extrinsèques en fonction de l'hypothèse de champs de vue recouvrants pour deux jeux de données différents. 111
5.7 Comparaison de la précision obtenue avec plusieurs reconstructions : SFM avec des paramètres extrinsèques parfaitement connus, SFM avec des paramètres extrinsèques approximatifs constants, qui est bouclée ou non (avec l'algorithme 6), puis optimisée par le MCBA. 113

6.1 Résultats de l'étalonnage "essieu-caméra" sur une trajectoire de synthèse. . . . 143
6.2 Résultats de l'étalonnage "essieu-caméra" sur une trajectoire de synthèse (en déterminant position et orientation de la caméra). 144
6.3 Résultats sur la précision de la rotation $\hat{R}_e^c$ estimée par l'étalonnage sur trois configurations. Les séquences d'images sont obtenues avec le simulateur. . . . 144

Liste des Algorithmes

1 Étalonnage intrinsèque d'une caméra 25
2 Correction automatique des distorsions à partir d'une image 33

3 Étalonnage extrinsèque avec une mire connue 51
4 Etalonnage extrinsèque sans mire connue 53

5 Structure From Motion Hiérarchique 85
6 Détection des boucles 87
7 Algorithme de Levenberg-Marquardt 91
8 Ajustement de faisceaux multi-caméra (MCBA) : une itération de l'algorithme de Levenberg-Marquardt sous la forme creuse (structure secondaire) 97
9 Algorithme de du gradient conjugué préconditionné (PCG), pour déterminer x qui minimise $\frac{1}{2}x^\top Ax - b^\top x$. 100

10 Étalonnage essieu-caméra 140

Notations

Notations communes à l'ensemble des chapitres :

$\mathbb{R}^n$	Espace vectoriel à n dimensions sur le corps des réels.
$\mathbb{P}^n$	Espace projectif à n dimensions sur le corps des réels.
$\mathbf{x}$	Un vecteur.
$\|\mathbf{x}\|$	Norme 2 d'un vecteur ($\|\mathbf{x}\| = \sqrt{\mathbf{x}^\top \mathbf{x}}$).
$A \in \mathbb{R}^{m \times n}$	Une matrice à $m \in \mathbb{N}^*$ lignes et $n \in \mathbb{N}^*$ colonnes.
$A \otimes B$	Produit de Kronecker de deux matrices (*cf* Annexe A.3.2).
N	Nombre de caméras du système multi-caméra.
K	Nombre de poses du système multi-caméra.
M	Nombre de points 3D.
P	Point de l'espace (parfois confondu avec le vecteur de ses coordonnées cartésiennes).
$\overline{P}$	Coordonnées homogènes du point 3D P.
p	Point 2D (parfois confondu avec le vecteur de ses coordonnées cartésiennes).
$\overline{p}$	Coordonnées homogènes du point 2D p.
$\sim$	Equivalence entre deux vecteurs de coordonnées d'un même point d'un espace projectif.
$a \overset{def}{=} b$	La nouvelle entité a est définie comme étant égale à b.
T_1^2	Matrice de passage homogène de la base $\mathcal{B}_1$ à la $\mathcal{B}_2$ (également appelée *transformation homogène*, *cf* Annexe A.1).
R	Matrice de rotation.
$\mathbf{t}$	Vecteur de translation.
K	Matrice des paramètres intrinsèques.
f	Distance focale.
(u_0, v_0)	Coordonnées du point principal.
S	Scène regroupant des points 3D.
P_i	$i^{ème}$ point 3D.
$[\![a..b]\!]$	Ensemble des entiers relatifs compris entre les entiers a et b.

Notations spécifiques au chapitre 4 :

C^{1r}, C^{2r}	Première et deuxième caméras réelles.
Π_j	Miroir observé par la $j^{ème}$ image de C^{1r}.
C^{1v_j}	$j^{ème}$ caméra virtuelle de C^{1r} par rapport à Π_j.
M_l	$l^{ème}$ marqueurs collés sur le miroir.
$p_i^{1v_j}$	Projection du point P_i dans le plan image de C^{1v_j}.
p_i^{2r}	Projection du point P_i dans le plan image de C^{2r}.
m_l^{1r}	Projection d'un marqueur M_l dans le plan image de C^{1r}.
T_{1r}^{2r}	Transformation homogène entre C^{1r} et C^{2r}.
$T_{1r}^{\Pi_j}$	Transformation homogène entre C^{1r} et Π_j.
T_{2r}^{S}	Transformation homogène entre C^{2r} et S.

Notations spécifiques au chapitre 5 :

ΔT_j	Transformation homogène entre les caméras C_1 et C_j. (également appelés *paramètres extrinsèques*).
ΔR_j	Matrice de rotation entre les caméras C_1 et C_j. (également appelé *rotation extrinsèque*).
$\Delta \mathbf{t}_j$	Vecteur de translation entre les caméras C_1 et C_j. (également appelé *translation extrinsèque*).
T_j^k	Transformation homogène entre le repère monde et C_j^k.
T_1^k	$k^{ème}$ pose du système multi-caméra.
S_j	$j^{ème}$ scène (observée par C_j).
C_j	$j^{ème}$ caméra.
C_j^k	$j^{ème}$ caméra lors de la $k^{ème}$ pose.

Chapitre 1

Introduction

Ce premier chapitre situe dans quel contexte scientifique mes travaux ont été réalisés. Ils sont rattachés au projet VIPA, dont les enjeux, les moyens mis en œuvre et produits développés seront présentés pour définir la problématique de cette thèse. Les méthodes répondant à cette problématique seront énoncées à travers le plan du mémoire et nos différentes contributions.

1.1 Contexte

1.1.1 Le projet VIPA

Durant les années 2008-2011, le LASMEA s'est investi dans le projet VIPA (Véhicules Individuels Publics Automatiques) en partenariat avec la PME innovante Apojee et la société Automobiles Ligier [lig] (*cf* Figure 1.1). Durant ce projet, nous avons mis au point des véhicules capables de naviguer automatiquement sans aucune infrastructure extérieure dédiée. Ce projet novateur a pour ambition de proposer de nouveaux services en terme de mobilité en couplant l'idée de *véhicule-partage* (comme pour les vélos dans certaines villes) et la notion de *conduite automatique*. Le VIPA est assimilable à un *ascenseur horizontal*, qui sera commercialisé en moyenne série et déployé sur des zones urbaines : parkings, zones piétonnes, desserte d'un hôpital ou d'un terminal d'aéroport à partir d'un parking éloigné, visite automatique de sites touristiques ou de parcs d'attractions. Plusieurs VIPA (illustrés par la Figure 1.3a) ont déjà prouvé leur utilité lors de leur déploiement au challenge Bibendum 2011.

Des systèmes classiquement utilisés, tels que le GPS centimétrique, ne sont pas exploitables dans un environnement urbain car ils dépendent fortement de la constellation des satellites, des problèmes comme le multi-trajet et le masquage de ces satellites. Pour ce projet, le LASMEA a apporté son savoir-faire (développé depuis les années 2000) en proposant un système de localisation et de navigation précis pour les applications souhaitées. Il s'agit d'un système basé sur un ensemble de caméras vidéo. Une étape d'apprentissage permet de connaître l'environnement dans lequel évolue le véhicule en le conduisant manuellement. L'apprentissage est réalisé à partir d'une séquence d'images enregistrées par les caméras en mouvement le long de la trajectoire

désirée. Il s'agit en quelque sorte de l'étape construisant le *rail virtuel* qu'empruntera ensuite le véhicule. Une fois cette étape effectuée, le système (qui n'est pas nécessairement celui ayant servi pour l'apprentissage) est capable de calculer sa pose (position et orientation) avec une précision centimétrique en comparant l'image courante prise par les caméras et la base d'apprentissage. La pose fournie est ensuite utilisée pour guider un véhicule automatique le long de ce rail virtuel. Le système de navigation fusionne également des données issues d'autres capteurs.

FIGURE 1.1 – Logos des membres du consortium VIPA, des partenaires et des financeurs.

1.1.2 La plateforme expérimentale

Pendant le développement du projet VIPA, de nombreux tests et expérimentations se sont déroulés sur la Plateforme d'Auvergne pour les Véhicule Intelligent (PAVIN, *cf* Figure 1.2). Cette plateforme, située aux Cézeaux face au LASMEA, offre un environnement urbain réaliste de $317m$ de voies goudronnées, entre façades d'immeubles, avec carrefour, rond point, zone de montée de passagers. Voici une liste des équipements du site : informatique et commande centralisée, caméras, couverture Wi-Fi, feux de signalisation, éclairage, GPS centimétrique, sonorisation, borne d'appel, bornes "Energie".

FIGURE 1.2 – Plateforme d'Auvergne pour les Vehicule Intelligent (PAVIN), cite des Cézeaux.

1.1.3 Les véhicules

Au cours du projet, deux types de véhicule électriques ont vu le jour : le VIPA (Figure 1.3a) destiné à être commercialisé, et le VipaLab (Figure 1.3b) qui est un véhicule expérimental destiné à la recherche. Afin de se mouvoir seuls, les véhicules sont équipés de différents capteurs proprioceptifs et extéroceptifs. Les véhicules VIPA sont équipés :

- de deux caméras (l'une à l'avant, l'autre à l'arrière),
- d'un odomètre sur chaque roue,
- d'un capteur mesurant l'angle de braquage des roues,
- d'un GPS bas coût,
- d'un actionneur avant de direction,
- d'un LIDAR.

Les véhicules expérimentaux VipaLab sont équipés :

- de deux caméras (l'une à l'avant, l'autre à l'arrière),
- d'un odomètre moteur (qui ne renvoie que la valeur absolue de la vitesse angulaire du moteur),
- d'un capteur mesurant l'angle de braquage des roues,
- d'un DGPS,
- d'un actionneur avant de direction,
- d'un LIDAR.

Les caméras sont synchronisées par un trigger externe et sont dirigées dans des directions opposées.

(a) Véhicules VIPA au challenge Bibendum 2011 à Berlin. (b) Véhicule VipaLab.

FIGURE 1.3 – Véhicules développés dans le cadre du projet VIPA.

De nombreuses méthodes de navigation monoculaire utilisant des amers naturels ont été proposées récemment, comme les travaux [RLDL07] effectués au LASMEA. Cependant, l'utilisation d'une seule caméra n'est pas suffisamment robuste aux conditions d'illumination extérieures ; notamment aux problèmes de surexposition (que l'on rencontre parfois au lever ou au coucher du soleil ou bien lors de la sortie d'un bâtiment vers un espace ensoleillé). C'est pour résoudre ce problème que les véhicules VIPA et VipaLab sont dotés de deux caméras, l'une à l'avant (dirigée vers l'avant), et l'autre à l'arrière (dirigée vers l'arrière). De cette manière, si l'une des caméras est éblouie par le soleil, une autre caméra pourra fournir des données pertinentes. Outre la symétrie du véhicule, un autre avantage de ce système multi-caméra est qu'il fournit des informations redondantes, et rend donc la localisation plus robuste.

1.2 Problématique

Si la localisation s'effectue en ligne, la phase de reconstruction de la séquence d'apprentissage et les étalonnages nécessaires sont effectués hors ligne. La phase d'apprentissage consiste à construire une carte d'amers visuels à partir de primitives (de types points d'intérêt) extraits des images enregistrées par les différentes caméras (lors d'une navigation supervisée). On remonte ensuite à la géométrie euclidienne des points 3D, et aux poses du système multi-caméra, notamment à l'aide d'algorithmes de type Structure From Motion et ajustement de faisceaux.

Mais afin de pouvoir reconstruire cette carte 3D, puis utiliser les algorithmes de localisation, il faut préalablement étalonner le système multi-caméra. Pour des caméras à champs de vue recouvrants, cette opération est généralement mise en œuvre grâce à des appariements de primitives entre différentes vues d'une même scène. Cependant, si les champs de vision des différents capteurs sont disjoints, un tel appariement est impossible à réaliser.

Les travaux de cette thèse ont pour but de développer et de mettre en œuvre des méthodes souples permettant d'étalonner cet ensemble de caméras dont les champs de vue sont totalement disjoints. Nous verrons comment obtenir un étalonnage précis malgré cette contrainte.

1.3 Plan du mémoire

Après une étape préalable d'étalonnage intrinsèque (Chapitre 2), et un état de l'art sur les systèmes multi-caméra (Chapitre 3), nous développons et mettons en œuvre différentes méthodes d'étalonnage extrinsèque (déterminant les poses relatives des caméras à champs de vue disjoints).

Dans le chapitre 4, nous utilisons un miroir plan pour créer un champ de vision commun aux différentes caméras.

Le chapitre 5 présente deux approches pour étalonner extrinsèquement le système multi-caméra grâce à son mouvement. La première consiste (*cf* Section 5.2) à manœuvrer le véhicule

pendant que chaque caméra observe une scène statique composée de cibles (dont la détection est sous-pixellique). Dans la deuxième approche (*cf* Section 5.3), nous montrons que l'étalonnage extrinsèque peut être obtenu simultanément à la reconstruction 3D (par exemple lors de la phase d'apprentissage), en utilisant des amers visuels (comme des points d'intérêt). Ces deux approches utilisent un outil commun : l'ajustement de faisceaux multi-caméra, proposé en Section 5.4.

Enfin, le chapitre 6 termine par un étalonnage déterminant la rotation du système multi-caméra par rapport au véhicule.

1.4 Contributions

Les travaux présentés dans ce mémoire ont donné lieu aux publications suivantes, répertoriées à la page 169 :

- [5, 6, 1] traitent des travaux portant sur l'étalonnage à l'aide d'un miroir plan. La principale contribution consiste à utiliser une scène de géométrie inconnue et de créer un champ de vision commun, entre les différents capteurs statiques ou embarqués. De plus, l'influence de la réfraction induite par la lame de verre du miroir est étudiée, et une méthode d'estimation de pose d'un objet réfracté est proposée.
- [3, 7, 2] traitent des travaux d'étalonnage qui exploitent les déplacements du véhicule dans un environnement statique. Les principales contributions sont :

  i/ l'étude des mouvements singuliers,

  ii/ un ajustement de faisceaux multi-caméra qui optimise les scènes, les poses du système multi-caméra, et étalonne extrinsèquement les caméras,

  iii/ et deux méthodes d'étalonnage basées soit sur les cibles, soit sur des points d'intérêt. Une demande de brevet [8] a été déposée sur les dernières avancées de ces travaux.

Ce manuscrit présente en outre différentes contributions telles que les cibles (*cf* Section 2.2.1, publiées dans [1, §VII]), un étalonnage intrinsèque (*cf* Section 2.3), la correction automatique d'images distordues (*cf* Section 2.4, et améliorations publiées dans [4]). De plus, nous étudions la faisabilité pour déterminer l'orientation d'une caméra se translatant à partir du foyer d'expansion (*cf* Section 6.2) en imposant un déplacement rectiligne. Au contraire, une seconde approche proposée en Section 6.3 s'affranchit de cette contrainte en s'appuyant sur le modèle cinématique du véhicule.

Chapitre 2

Etalonnage intrinsèque d'une caméra

Ce deuxième chapitre dresse un bref état de l'art sur la modélisation et l'étalonnage intrinsèque d'une caméra. Nous nous appuierons sur les méthodes classiquement utilisées, pour mettre en exergue celles que nous proposons (basées sur une mire d'étalonnage constituée de plusieurs cibles, ou bien sur une méthode automatique de correction d'image distordue).

2.1 Modèles intrinsèques

Il existe différentes conceptions de caméras. En vision par ordinateur, la plupart disposent au moins d'un système optique (objectif) et d'un capteur photographique (CCD ou CMOS). Les modèles intrinsèques des caméras approximent le processus physique de formation de l'image. Dans cette section, nous présentons uniquement les modèles de paramétrisation intrinsèque des caméras que nous avons utilisés. Cependant, il en existe de nombreux, qu'ils soient paramétriques ou non (fonctions analytiques, surfaces paramétrées, interpolation de tables de correspondance...). Une présentation plus exhaustive des caméras et des modèles existants est proposée dans [SRT+11].

2.1.1 Modèle sténopé

Le modèle sténopé représente la formation de l'image d'une caméra à l'aide d'une projection perspective de la scène observée sur un plan. Il s'agit d'un modèle central où tous les rayons optiques s'intersectent en un même point appelé *centre optique* (ou centre de projection). Tout point 3D se projette selon la droite le reliant avec le centre optique de la caméra. Le point image s'obtient à l'intersection de cette droite et du plan image (situé à la distance focale f).

Le modèle sténopé (également appelé *modèle perspectif* ou *pin-hole*) est illustré par la Figure 2.1. Par convention, les axes du repère de la caméra sont choisis tels que X soit vers la droite, Y vers le bas et Z vers l'avant (même direction que l'axe optique). L'image se focalise à une distance $Z = -f$ derrière le centre optique sur le capteur (CCD ou CMOS). Par simple

symétrie centrale (de centre optique), on considère que le plan image est situé à $Z = f$. On définit le repère pixellique (u, v) dont l'origine est située dans le coin supérieur gauche de l'image. On appelle *plan normalisé* le plan tel que $Z = 1$. Les points 2D appartenant à ce plan sont exprimés dans le repère (x, y).

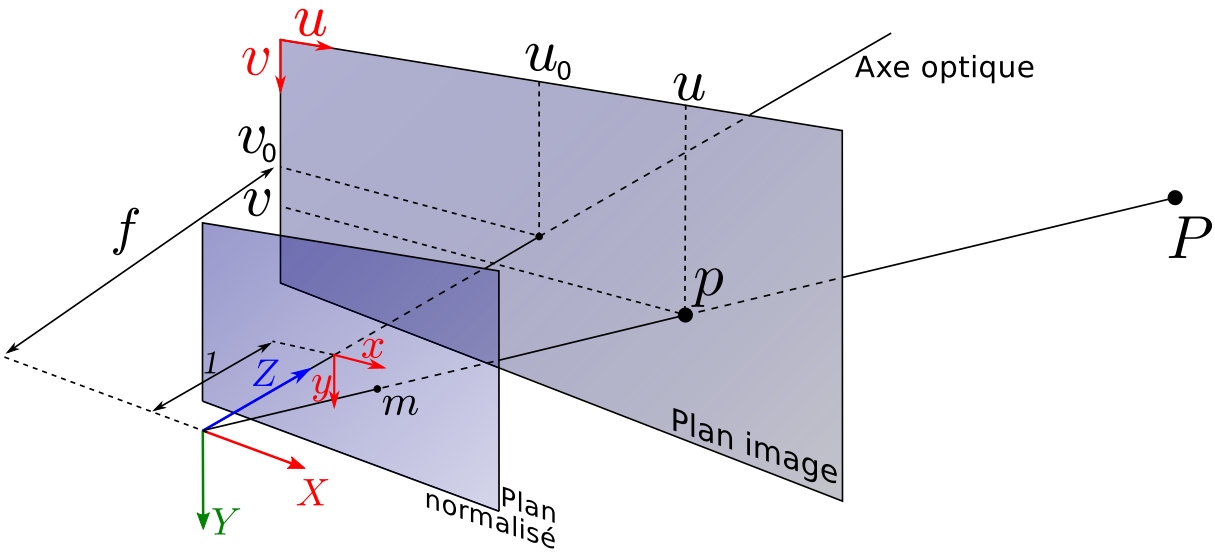


FIGURE 2.1 – Modèle perspectif d'une caméra. Un point 3D P est projeté dans le plan image de la caméra en un point p.

Soit P un point 3D. Le point P se projette dans le plan image en un point p. Soit $\overline{P}_c = (X_c\ Y_c\ Z_c\ 1)^\top$ les coordonnées homogènes du point P dans le repère de la caméra (si le point 3D est exprimé dans un repère monde, il suffit d'appliquer un changement de base comme expliqué en Annexe B.1). On note $\overline{m} = (x\ y\ 1)^\top \sim (X_c\ Y_c\ Z_c)^\top$ les coordonnées homogènes du point 2D dans le repère de la caméra. On note $\overline{p} = (u\ v\ 1)^\top \sim (X_p\ Y_p\ Z_p)^\top$ les coordonnées homogènes du point 2D dans le repère image.

La projection du point 3D P dans le plan image en un point p s'illustre en deux étapes, décrites ci-après. Par la suite, les étapes, permettant d'obtenir une entité b à partir d'une entité a, sont notées $a \quad \rightarrow b$ (*cf* équation 2.1).

$$\overline{P}_c = \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} \overset{1.}{\rightarrow} \overline{m} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} \overset{2.}{\rightarrow} \overline{p} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim \begin{pmatrix} X_p \\ Y_p \\ Z_p \end{pmatrix} \tag{2.1}$$

1. La projection d'un point 3D dans le plan normalisé de la caméra s'écrit en coordonnées homogènes :

$$\overline{m} = [I_3|0_{3\times 1}]\overline{P}_c \tag{2.2}$$

2. Pour passer du repère caméra au repère image (coordonnées en pixel), on utilise la transformation affine suivante :

$$\overline{p} = K\overline{m} \tag{2.3}$$

Où K est la matrice des paramètres intrinsèques définie par l'équation 2.4 :

$$K = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.4}$$

On note $f_x = f/d_x$ et $f_y = f/d_y$, avec f la distance focale de la caméra, d_x et d_y les dimensions élémentaires de la matrice CCD et (u_0, v_0) les coordonnées pixelliques du point principal dans l'image. En pratique, le rapport $\frac{f_x}{f_y} = \frac{d_y}{d_x}$ est proche de 1, et le point principal est proche du centre de l'image.

L'équation 2.3 s'écrit donc :

$$(2.3) \Leftrightarrow \begin{pmatrix} X_p \\ Y_p \\ Z_p \end{pmatrix} = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \begin{pmatrix} f_x X_c + u_0 Z_c \\ f_y Y_c + v_0 Z_c \\ Z_c \end{pmatrix} \tag{2.5}$$

$$\Leftrightarrow \begin{pmatrix} u \\ v \end{pmatrix} = \pi(\overline{p}) = \begin{pmatrix} f_x \frac{X_c}{Z_c} + u_0 \\ f_y \frac{Y_c}{Z_c} + v_0 \end{pmatrix} = \begin{pmatrix} f_x\, x + u_0 \\ f_y\, y + v_0 \end{pmatrix} \tag{2.6}$$

Où π est la fonction définie par l'équation 2.7, permettant de passer des coordonnées homogènes aux coordonnées cartésiennes d'un point 2D.

$$\begin{aligned} \pi :\ & \mathbb{P}^2 \longrightarrow \mathbb{R}^2 \\ \overline{p} = \begin{pmatrix} X_p \\ Y_p \\ Z_p \end{pmatrix} & \mapsto \pi(\overline{p}) = p = \begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{Z_p} \begin{pmatrix} X_p \\ Y_p \end{pmatrix} \end{aligned} \tag{2.7}$$

Remarque : Le repère pixellique (u, v) est situé dans le coin supérieur gauche de l'image. Il faut toutefois être attentif à la convention choisie, notamment si l'on travaille avec des données sous-pixelliques. En effet, selon les outils utilisés (programme de traitement d'image, bibliothèques C++, détecteur...), l'origine peut-être placée

- soit au centre du pixel supérieur gauche,
- soit dans le coin supérieur gauche de ce pixel,
- ou encore en dehors de l'image afin que le centre de ce pixel ait pour coordonnées $(1, 1)$.

2.1.2 Modèle sténopé et distorsions radiales

Les déformations géométriques engendrées par l'objectif peuvent être modélisées par des distorsions radiales et tangentielles. En optique, il existe également d'autres types d'aberration (géométriques, chromatiques, le vignetage, la diffraction... *cf* [Wel86]). Or, les distorsions

radiales étant prépondérantes, nous ne prendrons en compte que la composante radiale. Dans certains travaux [HK05], le point principal et le centre de distorsion (point du plan image où il n'y a pas de distorsion) sont distincts. Nous ne faisons pas cette hypothèse.

La composante radiale peut être modélisée à l'aide d'un polynôme. Dans la littérature, il permet de passer soit :

1. des coordonnées non-distordues m_u aux coordonnées distordues m_d. Ce modèle sera appelé *modèle direct* (*cf* Figure 2.2). Il est utilisé par OpenCV [Bra00] et la toolbox Matlab™ [Bou02].
2. des coordonnées distordues $m_d = (x_d \ y_d)^\top$ aux coordonnées non-distordues $m_u = (x_u \ y_u)^\top$. Ce modèle, qualifié de standard par [SRT+11, p45], sera appelé *modèle indirect* (*cf* Figure 2.3). Il est utilisé par [LVD99, LVD98].

2.1.2.1 Modèle direct

Le modèle direct permet de projeter le point 3D P, dans le plan image en un point distordu p_d, en trois étapes successives :

1. une projection perspective de P en un point m_u (*cf* équation 2.2),
2. une transformation polynomiale pour appliquer la distorsion (*cf* équation 2.10),
3. et une transformation affine qui utilise la matrice K des paramètres intrinsèques (*cf* équations 2.3 et 2.4).

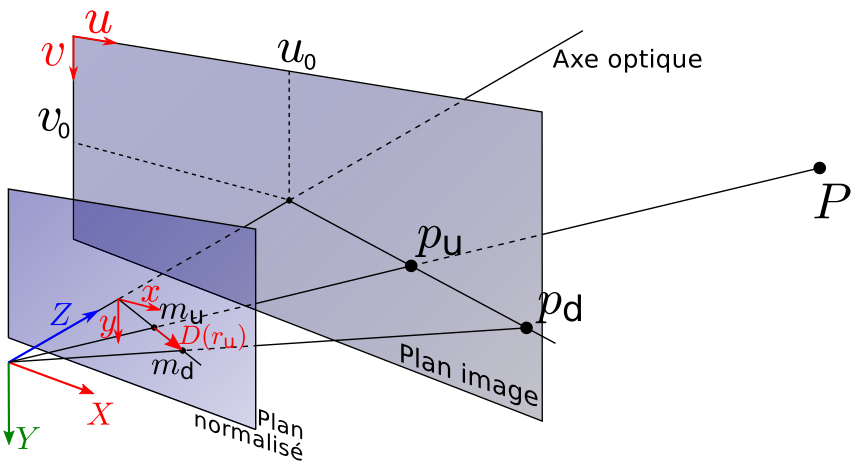


FIGURE 2.2 – Caméra perspective et distorsions directes. Un point 3D P est projeté dans le plan image de la caméra en un point distordu p_d, en appliquant successivement une projection perspective, une transformation polynomiale puis affine.

L'équation 2.8 et la Figure 2.2 illustrent ces trois étapes.

$$\overline{P}_c \overset{(2.2)}{\rightarrow} \overline{m}_u \overset{D(r_u)}{\rightarrow} \overline{m}_d \overset{K}{\rightarrow} \overline{p}_d \tag{2.8}$$

On note D le polynôme de degré n en r_u^2 (*cf* équation 2.9), qui permet de passer du point m_u sans distorsion au point m_d avec distorsions dans le plan normalisé *via* l'équation 2.10. Où $r_u = \sqrt{x_u^2 + y_u^2} = ||m_u||$ est la distance radiale entre l'axe optique et le point m_u dans le plan normalisé.

$$D(r_u) = \sum_{k=1}^{n} a_k r_u^{2k} \tag{2.9}$$

$$m_d = (1 + D(r_u))m_u \tag{2.10}$$

Pour plus de lisibilité, on note également la déformation d'un point non-distordu vers son point distordu $p_d = D(p_u)$ dans le repère image.

2.1.2.2 Modèle indirect

Comme son nom l'indique, le modèle indirect ne permet pas d'obtenir directement (*c.-à-d.* par des étapes successives) le point image distordu p_d. Par contre, le point image non-distordu p_u s'obtient par deux moyens :

- soit par projection perspective du point P en un point m_u (*cf* équation 2.2), suivi d'une transformation affine pour obtenir p_u (*cf* équation 2.3).
- soit avec une transformation polynomiale du point image distordu p_d (ramené dans le plan normalisé).

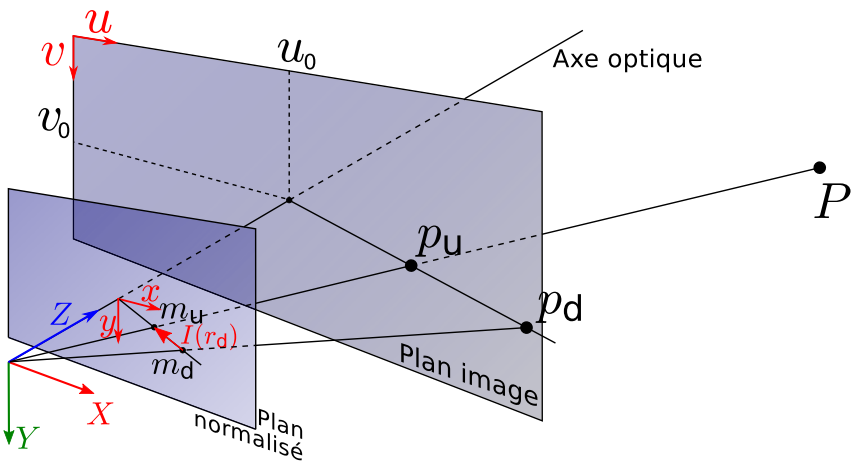


FIGURE 2.3 – Caméra perspective et distorsions indirectes. Un point 3D P est projeté dans le plan image de la caméra en un point non-distordu p_u. Il s'obtient également avec une transformation polynomiale à partir du point image distordu p_d.

Le système 2.11 et la Figure 2.3 illustrent les deux points précédents.

$$\begin{cases} \overline{P}_c \rightarrow \overline{m}_u \overset{K}{\rightarrow} \overline{p}_u \\ \overline{p}_d \overset{K^{-1}}{\rightarrow} \overline{m}_d \overset{I(r_d)}{\rightarrow} \overline{m}_u \overset{K}{\rightarrow} \overline{p}_u \end{cases} \tag{2.11}$$

On note I le polynôme de degré n en r_d (*cf* équation 2.12), qui permet de passer du point avec distorsion m_d au point sans distorsions m_u dans le plan normalisé *via* l'équation 2.13. Où $r_d = \sqrt{x_d^2 + y_d^2} = ||m_d||$ est la distance radiale entre l'axe optique et le point m_d dans le plan normalisé.

$$I(r_d) = \sum_{k=1}^{n} b_k r_d^{2k} \tag{2.12}$$

$$m_u = (1 + I(r_d))m_d \tag{2.13}$$

Pour plus de lisibilité, on note également la déformation d'un point distordu vers son point non-distordu $p_u = I(p_d)$ dans le repère image.

L'un des intérêts du modèle indirect est qu'il permet de corriger la distorsion des amers, juste après leur détection. On se ramène ainsi au simple modèle perspectif.

2.1.2.3 Discussions et remarques

En pratique, on choisit $n = 5$. Or un polynôme de degré 5 n'étant en général pas facilement inversible, il n'est pas possible de calculer l'inverse de D ou de I. Il serait possible d'utiliser simultanément les modèles direct et indirect selon les besoins, mais la relation $I \circ D = D \circ I = I_d$ ne serait pas vérifiée. Dans les chapitres suivants, nous n'utilisons qu'un seul modèle pour les distorsions : sauf mention contraire, nous choisissons de modéliser les paramètres intrinsèques des caméras par le modèle indirect.

Maximum de vraisemblance Considérons un problème de vision faisant intervenir une minimisation des erreurs de reprojection (par exemple, un calcul de pose ou un ajustement de faisceaux). Qu'en est-il du maximum de vraisemblance selon le modèle direct ou indirect ?

Si on suppose que les détections (dans l'image distordue) sont sujettes à un bruit gaussien, alors un estimateur utilisant le modèle direct sera maximum de vraisemblance. La fonction de coût que l'on minimise alors est de la forme :

$$\sum ||D(\pi(K[I_3|0_{3\times1}]\overline{P}_c)) - \hat{p}_i||^2 \tag{2.14}$$

Où $\hat{p}_i$ représente la détection d'un point dans le plan image.

Au contraire, si l'estimateur utilise le modèle indirect, on suppose alors que le bruit de détection est gaussien dans l'image non-distordue. La fonction de coût que l'on minimise alors est de la forme :

$$\sum ||\pi(K[I_3|0_{3\times1}]\overline{P}_c) - I(\hat{p}_i)||^2 \tag{2.15}$$

Or on peut supposer que le bruit de détection (de points d'intérêt sur une image) est gaussien. Dans le cas du modèle direct, on cherche bien à minimiser un bruit de mesure sur l'image distordue. Dans le cas du modèle indirect, pour minimiser du bruit de mesure, il faudrait détecter les points d'intérêts dans l'image non-distordue, sinon cela revient à supposer que les distorsions sont suffisamment faibles (et ne déforment pas ou peu la densité de probabilité gaussienne autour des points détectés dans l'image distordue).

Changement de résolution Quelle est l'influence d'un changement de résolution sur les paramètres intrinsèques ? Considérons des paramètres intrinsèques K obtenus avec des images de résolution $H \times L$. Un changement de résolution peut-être un rognage ou un ré-échantillonnage de l'image.

Premièrement, si l'image est rognée avec une marge de h sur la hauteur (h lignes sont rognées en haut de l'image), et une marge de l sur la largeur (l colonnes sont rognées à gauche de l'image), alors la focale reste inchangée ($f' = f$) et le point principal est translaté ($(u'_0, v'_0) = (u_0 - l, v_0 - h)$). Si l'image est ré-échantillonnée avec un rapport μ (*c.-à-d.* $H' = \mu H$ et $L' = \mu L$), alors la focale et le point principal sont multipliés par ce rapport (*c.-à-d.* $f'_x = \mu f_x$, $f'_y = \mu f_y$ et $(u'_0, v'_0) = (\mu u_0, \mu v_0)$).

Deuxièmement, qu'il s'agisse d'un rognage ou d'un ré-échantillonnage, les coefficients des polynômes de distorsion restent inchangés. C'est l'intérêt d'avoir défini ces polynômes dans le plan normalisé : ils ne dépendent ni de la focale, ni du repère image.

2.1.3 Modèle unifié

Le modèle unifié a été initialement proposé par [GD00] pour des capteurs catadioptriques centraux. Mais il permet de modéliser avec un même formalisme mathématique des caméras catadioptriques, perspectives, ou encore grand angle (fish-eye), comme montré dans [Bar06]. Pour cela, une projection sur une sphère permet de modéliser les non-linéarités induites par l'objectif.

On note avec un indice c les points exprimés dans le repère de la caméra (X_c, Y_c, Z_c), et avec un indice s les points exprimés dans le repère de la sphère (X_s, Y_s, Z_s).

Projection La projection du point 3D P dans le plan image en un point p s'effectue en trois étapes successives :

1. la projection du point P sur la sphère unitaire de centre $Z_c = \xi \geq 0$ en un point Q,

$$Q_s = \frac{P_s}{\rho} = \frac{1}{\rho} \begin{pmatrix} X_s \\ Y_s \\ Z_s \end{pmatrix} \tag{2.16}$$

   avec $\rho = \sqrt{X_s^2 + Y_s^2 + Z_s^2}$

2. la projection perspective de Q sur le plan normalisé $Z_c = 1$ en un point m,

$$\overline{m}_c = \begin{pmatrix} \frac{X_s}{Z_s + \rho\xi} \\ \frac{Y_s}{Z_s + \rho\xi} \\ 1 \end{pmatrix} \sim \begin{pmatrix} \frac{X_s}{\rho} \\ \frac{Y_s}{\rho} \\ \frac{Z_s}{\rho} + \xi \end{pmatrix} \tag{2.17}$$

3. et la transformation affine (qui utilise la matrice $K = \begin{pmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{pmatrix}$) pour obtenir le point p.

$$\overline{p}_c = K\overline{m}_c \tag{2.18}$$

L'équation 2.19 et la Figure 2.4 illustrent les trois étapes de cette projection.

$$\overline{P}_s \overset{(2.16)}{\rightarrow} Q_s \overset{(2.17)}{\rightarrow} \overline{m}_c \overset{K}{\rightarrow} \overline{p}_c \tag{2.19}$$

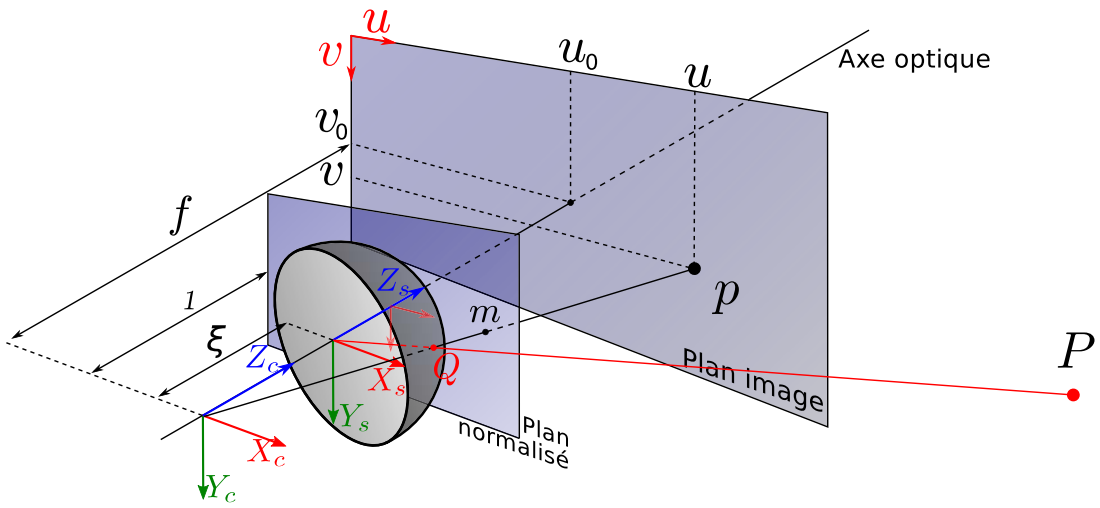


FIGURE 2.4 – Modèle de caméra unifié. Un point 3D P est projeté dans le plan image de la caméra en un point distordu p.

Certains articles [MSA05] rajoutent une distorsion (radiale et tangentielle) au point m. Or, la non-linéarité induite par la sphère unité est suffisante pour modéliser les distorsions. Ceci a été vérifié dans [Cou09] et [MCMar] à l'aide de plusieurs objectifs dont les champs de vue varient entre 94°et 190°. [CMEM07] prouve également des équivalences entre différents modèles (pour des caméras catadioptriques ou fish-eyes) et le modèle unifié.

Lancer de rayon Contrairement aux modèles précédents modélisant la distorsion par un polynôme, le modèle unifié est "inversible" au sens où il est possible de retrouver analytiquement le point sur la sphère (et donc le rayon (QP)) à partir du point image p. Le lancer de rayon s'illustre par l'équation 2.20. Ceci également est expliqué dans [Bar03] et [Cou09].

$$\overline{p}_c \overset{(2.21)}{\rightarrow} \overline{m}_c \overset{(2.22)}{\rightarrow} Q_s \tag{2.20}$$

En effet si la caméra est étalonnée,

1. on retrouve le point m sur le plan image normalisé à partir du point image p,

$$\overline{m}_c = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = K^{-1}\overline{p}_c \tag{2.21}$$

2. puis le point Q sur la sphère :

$$Q_s = \eta \begin{pmatrix} x \\ y \\ 1 - \eta^{-1}\xi \end{pmatrix} \tag{2.22}$$

avec

$$\eta = \frac{\xi + \sqrt{1 + (x^2 + y^2)(1 - \xi^2)}}{x^2 + y^2 + 1} \tag{2.23}$$

Démonstration : Pour retrouver l'équation 2.22, il suffit d'imposer que le point Q, qui intersecte la droite reliant $\overline{m}$ et l'origine du repère caméra, soit de norme 1 dans le repère de la sphère. Plus précisément, il faut résoudre $||Q_s|| = 1$ avec η un réel positif tel que :

$$\overline{m}_c = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \Rightarrow Q_c = \begin{pmatrix} \eta\, x \\ \eta\, y \\ \eta \end{pmatrix} \Rightarrow Q_s = \begin{pmatrix} \eta\, x \\ \eta\, y \\ \eta - \xi \end{pmatrix} \tag{2.24}$$

On obtient donc :

$$\begin{aligned} ||Q_s|| = 1 &\Leftrightarrow \eta^2 x^2 + \eta^2 y^2 + (\eta - \xi)^2 = 1 \\ &\Leftrightarrow \eta^2(x^2 + y^2 + 1) - 2\eta\xi + (\xi^2 - 1) = 0 \end{aligned} \tag{2.25}$$

On prenant la solution positive de cette équation du second degré en η, on retrouve bien les équations 2.22 et 2.23. □

Plan image non-distordu Le modèle unifié permet d'obtenir aisément les coordonnées d'un point non-distordu à partir d'un point distordu, et inversement. Un observateur (ou une caméra perspective) qui serait situé au centre de la sphère verrait un monde sans distorsion. Il reste à choisir l'orientation et la focale f' de cette caméra perspective. On peut donc définir un plan image non-distordu (qui est ici parallèle au plan focal de la Figure 2.4) en choisissant f'. Pour passer du plan normalisé au plan image non-distordu, on utilise la transformation affine avec la matrice $K' = \begin{pmatrix} f' & 0 & u_0 \\ 0 & f' & v_0 \\ 0 & 0 & 1 \end{pmatrix}$. Le choix de f' permet de déterminer la taille de l'image non-distordue.

L'équation 2.26 illustre le passage des coordonnées du point $\overline{p}$ de l'image distordue aux coordonnées du point $\overline{p}'$ dans l'image sans distorsion. On note U_{undist} cette fonction corrigeant la distorsion d'un point pour le modèle unifié.

$$U_{undist} : \qquad \overline{p}_c \xrightarrow{K^{-1}} \overline{m}_c \xrightarrow{(2.22)} Q_s \xrightarrow{\pi} \overline{m}'_s \xrightarrow{K'} \overline{p}'_s \tag{2.26}$$

Inversement, l'équation 2.27 illustre le passage des coordonnées du point $\overline{p}'$ de l'image non-distordue aux coordonnées du point $\overline{p}$ dans l'image d'origine (*c.-à-d.* l'image présentant des distorsions). On note U_{dist} cette fonction appliquant la distorsion à un point pour le modèle unifié.

$$U_{dist} : \qquad \overline{p}'_s \xrightarrow{K'^{-1}} \overline{m}'_s \xrightarrow{(2.16)} Q_s \xrightarrow{(2.17)} \overline{m}_c \xrightarrow{K} \overline{p}_c \tag{2.27}$$

2.2 Mire d'étalonnage

Considérons quelques méthodes d'étalonnage intrinsèque existantes basées sur une mire d'étalonnage. Certaines nécessitent un damier, comme OpenCV [Bra00] et la toolbox Matlab™ [Bou02]). L'utilisation d'un tel damier est assez souple, car OpenCV inclut une détection rapide de ce dernier. D'autres méthodes utilisent une mire noire sur laquelle sont collées des pastilles réfléchissantes (*cf* [LVD99, LVD98]). De bonnes conditions de prise de vue, avec un éclairage annulaire, permettent une détection précise de ces pastilles. Toutefois, l'utilisation de cet outil est fastidieuse, car la détection et la labellisation des pastilles nécessitent une intervention manuelle sur chaque image. Dans cette section, nous proposons une mire qui allie la souplesse d'utilisation à la précision de détection de la mire.

2.2.1 Cibles

Une procédure d'étalonnage souple doit être la plus automatique possible. Pour ce faire, nous avons créé une *cible*, son détecteur automatique et son processus de labellisation automatique. Cette cible circulaire, similaire à [LdInMH02], que nous avons proposée dans [1, §VII] est composée d'un disque noir, entouré d'une couronne noire, elle-même entourée par un code-barre circulaire (*cf* Figures 2.9 et 2.10). Nous allons décrire les raisons de cette conception, puis les algorithmes mis en place pour la détection, le raffinement sous-pixellique et la labellisation automatique des cibles.

2.2.1.1 Pourquoi utiliser des cercles concentriques ?

En vision par ordinateur, plusieurs articles utilisent :
- des cercles non-concentriques et coplanaires :
  - pour étalonner intrinsèquement une caméra perspective [Tar94],
  - pour déterminer la structure euclidienne d'une scène à partir d'une image de plusieurs cercles [GSW06].
- des cercles non-concentriques et parallèles :
  - pour étalonner intrinsèquement une caméra perspective [WZHW04].
- des cercles concentriques :
  - [KGK05] et [ACV04] étudient leurs propriétés en géométrie projective, pour étalonner une caméra. Dans le même but, [KKK02] et [JQ05] utilisent plusieurs jeux de cercles concentriques, identifiés avec un invariant projectif (birapport).
  - [KGK05], [CGC+11] et [YZ07] proposent des détecteurs supportant une occultation partielle. Une procédure de vote est souvent utilisée (transformée de Hough). Pour calculer la pose d'une caméra, [CGC+11] propose un ajustement de faisceaux minimisant des distances entre les points de contour détectés et les coniques estimées.
  - [NPD08] propose une mire originale détectable à différentes échelles.

Considérons un ensemble de cercles concentriques coplanaires. Il est équivalent de considérer que ces cercles appartiennent soit :
- à un plan 2D, puis qu'on leurs applique une homographie,
- à un plan de l'espace, puis sont projetés dans un autre plan (dans notre cas, le plan image d'une caméra).

Chaque cercle $\mathcal{C}_i$ (défini par son centre c et son rayon r_i) est alors transformé en ellipse $\mathcal{E}_i$ (défini par son centre e_i, un grand axe a_i, un petit axe b_i et une orientation θ_i).

Rappelons quelques propriétés vérifiées par ces cercles (illustrées par la Figure 2.5) :

Propriété 1. *Le projeté $proj(c)$ du centre d'un cercle n'est pas le centre e_i de la projection d'un cercle (sauf en fronto-parallèle : cas affine).*

$$proj(c) \neq e_i \tag{2.28}$$

Propriété 2. *Les centres e_i de la projection des cercles concentriques sont alignés.*

Propriété 3. *Plus le rayon d'un cercle diminue, plus le centre de la projection du cercle tend vers le projeté du centre du cercle.*

$$proj(c) = \lim_{r_i \to 0} e_i \tag{2.29}$$

Bien que chaque ellipse ait un centre e_i, on peut donc définir $proj(c)$ comme étant *le* centre des ellipses concentriques. Sur la Figure 2.5, une homographie a été appliquée à une image de cercles concentriques. Chaque ellipse $\mathcal{E}_i$ a été détectée et dessinée de la même couleur que son centre e_i (bleu, vert et rouge). Le point jaune représente le centre $proj(c)$ des 3 ellipses concentriques. Le centre des 2 plus petites ellipses a également été détecté au même endroit que le point jaune (avec une erreur de $3.3 * 10^{-3}$ pix). La méthode de détection utilisée est décrite dans les parties 2.2.1.2 et 2.2.1.3.

Par la suite, nous souhaitons utiliser des amers visuels ponctuels détectés avec précision. La méthode décrite dans les deux sous-sections suivantes permet de détecter précisément le centre des ellipses concentriques.

2.2.1.2 Détection automatique

Afin d'automatiser au maximum la procédure, nous avons développé une procédure de détection automatique de disques elliptiques (qu'ils soient blancs ou noirs). En entrée, le détecteur n'a besoin que d'une image. Il fournit les caractéristiques de chaque région elliptique présente dans l'image (position, grand axe, petit axe). La détection est réalisée automatiquement comme suit :
- Recherche des extremums locaux des niveaux de gris dans toute l'image, par opérations morphologiques (*cf* [Soi10, p170-171]).
- Segmentation en régions de l'image binaire.

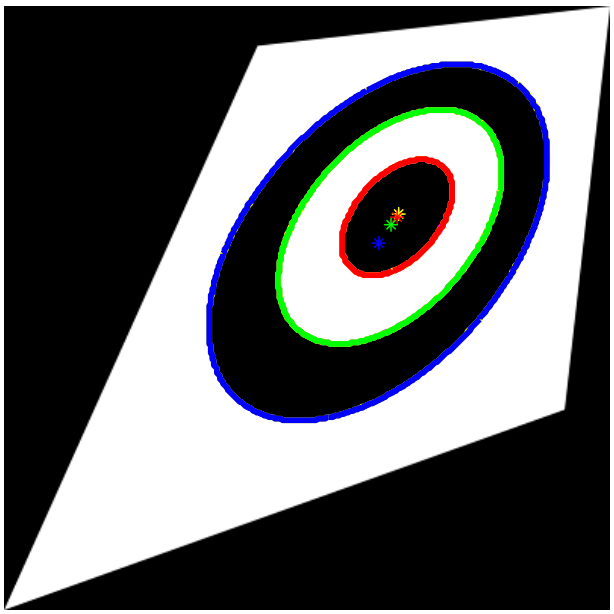


FIGURE 2.5 – Cercles concentriques auxquels une homographie a été appliquée. Les ellipses sont dessinées de la même couleur que leur centre. Le point jaune représente *le* centre des ellipses concentriques (défini comme étant le centre des cercles concentriques auquel l'homographie a été appliquée). La résolution de l'image est de 400x400 pixels.

– Classification des régions elliptiques.

Une région est considérée elliptique, si :

$$\frac{\mathcal{A}}{\pi ab} > 0.99 \tag{2.30}$$

où $\mathcal{A}$, a et b sont respectivement l'aire de la région considérée (*c.-à-d.* nombre de pixels de la région), le demi-grand axe et le demi-petit axe de l'ellipse ayant les mêmes moments (d'ordre inférieur à deux) que la région.

2.2.1.3 Raffinement sous-pixellique automatique

Afin d'obtenir une détection sous-pixellique, nous utilisons un algorithme qui superpose au mieux le motif connu sur l'observation (*pattern matching*). Une optimisation non-linéaire sous contrainte optimise à la fois un modèle de transformation photométrique de la luminance et la géométrie de la cible (via une homographie). Un algorithme similaire a été proposé précédemment dans [LVD99, Bra95], mais avec uniquement une transformation affine d'une seule ellipse (entrainant un biais systématique étudié par [Bra95]) et une optimisation avec l'algorithme de Levenberg-Marquardt. Notre approche est plus adaptée aux données réelles, notamment car les cas divergents sont évités.

Transformation photométrique L'intensité lumineuse $\mathcal{L}(r)$ d'une cible (*cf* Figure 2.6) est paramétrée par : une valeur haute, une valeur basse, et pour la transition, une pente p pour le segment et deux arcs de cercles de rayon r_1 et r_2. Afin de modéliser une éventuelle surexposition, un sixième paramètre h est introduit. Les pentes montantes sont alors centrées sur $1-h$ et $3-h$, et la pente descendante sur $2+h$.

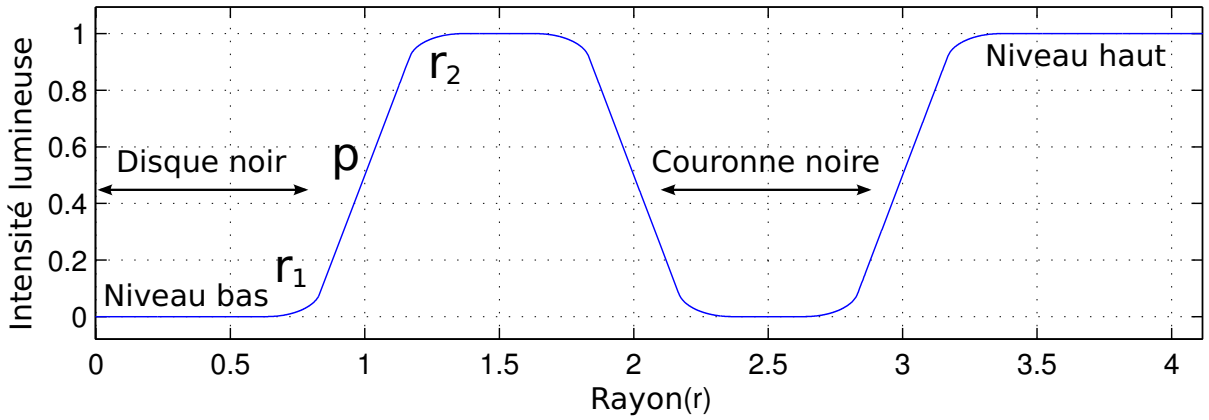


FIGURE 2.6 – Intensité lumineuse $\mathcal{L}(r)$ d'une cible, en fonction du rayon r dans le modèle.

Transformation géométrique On souhaite retrouver la transformation géométrique (l'homographie H) entre le modèle fronto-parallèle de la cible (en adéquation avec le modèle de luminance) et le contenu de l'image au voisinage de la cible (*cf* Figure 2.7). Concernant le modèle de la cible, nous utilisons le disque entouré de la couronne noire.

L'homographie représente la matrice de passage homogène entre le repère de l'image (identique au repère du modèle de la cible) et le repère de la cible dans l'image acquise (*cf* Figure 2.7). D'après les équations A.1 et A.7 de l'annexe, l'homographie H vérifie la relation suivante :

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim H \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.31}$$

Pour initialiser les paramètres de la transformation géométrique, nous approximons la transformation projective H par une affinité H_{affine} (*cf* Annexe A.2.5, Figure A.1 et équation 2.32). Pour cela, on utilise les caractéristiques de l'ellipse obtenue lors de la détection du disque noir : son centre de coordonnées $\mathbf{t} = (t_u, t_v)^\top$, son grand axe et son petit axe de longueurs respectives a et b et l'angle de rotation θ entre l'horizontale et le grand axe. Cette transformation affine initiale conserve l'orthogonalité des axes (*c.-à-d.* le shear r est nul).

$$H \approx H_{affine} = H_e\, H_a = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & t_u \\ \sin(\theta) & \cos(\theta) & t_v \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.32}$$

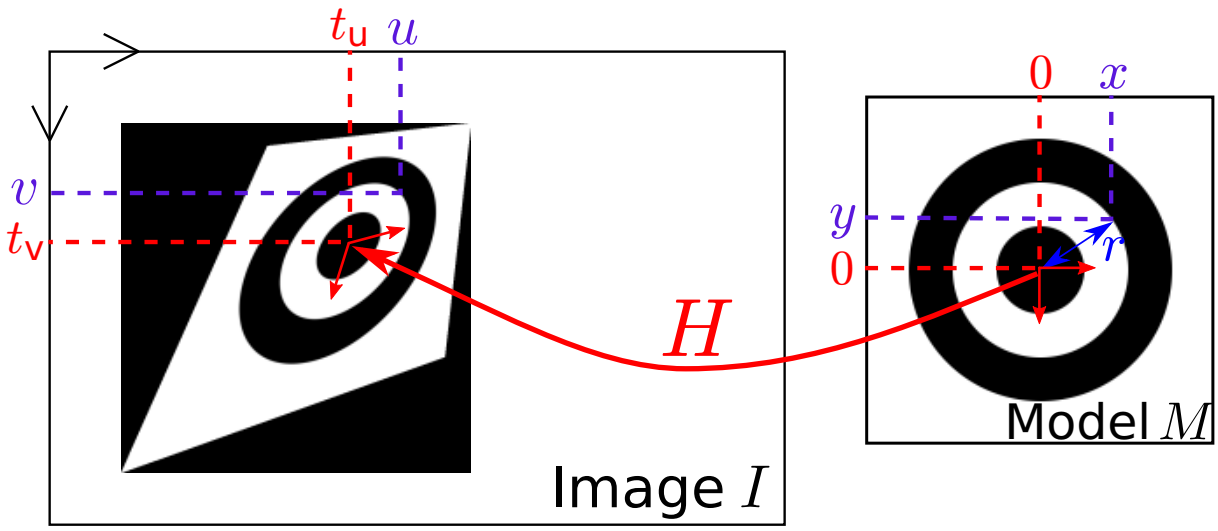


FIGURE 2.7 – Homographie entre le modèle fronto-parallèle d'une cible et son acquisition dans une image. L'homographie H s'interprète comme la matrice de passage homogène entre le repère du modèle et le repère de la cible acquise.

La Figure 2.8 montre l'initialisation du détecteur sur la première ligne, et le résultat après convergence sur la deuxième ligne. La première colonne correspond à une cible issue d'une image, la deuxième représente le modèle généré (selon les transformations photométriques et géométriques estimées) et la troisième est la valeur absolue de la différence entre l'image et son modèle estimé. Les histogrammes des images de la dernière colonne sont égalisés pour pouvoir observer l'erreur résiduelle. Pour l'optimisation, l'homographie H est paramétrée par 8 paramètres (θ, t_u, t_v, λ_1, λ_2, r, v_1 et v_2), comme détaillé en Annexe A.2.5.

Optimisation L'ensemble des paramètres photométriques et géométriques est raffiné à l'aide d'une optimisation non-linéaire sous contraintes (avec l'algorithme *active-set* de la fonction *fmincon* de Matlab™). On évite ainsi de s'approcher de paramètres aberrants : chaque paramètre est optimisé dans un intervalle adéquat (à la différence du LM utilisé dans [LVD99]). Des contraintes inégalitaires sont imposées, comme :

- les rayons r_1 et r_2 doivent être positifs.
- le centre des ellipses concentriques doit être à l'intérieur de l'ellipse détecté au paragraphe 2.2.1.2.
- le niveau haut (de l'intensité lumineuse) doit être supérieur au niveau bas à une marge près.

De plus, on se limite à la région d'intérêt $\mathcal{R}$ dont les pixels sont à l'intérieur de *l'ellipse englobante*. Cette ellipse englobante est définie comme étant p fois plus grande que l'ellipse initiale (avec p =nombre d'ellipses $+0.6 = 3.6$).

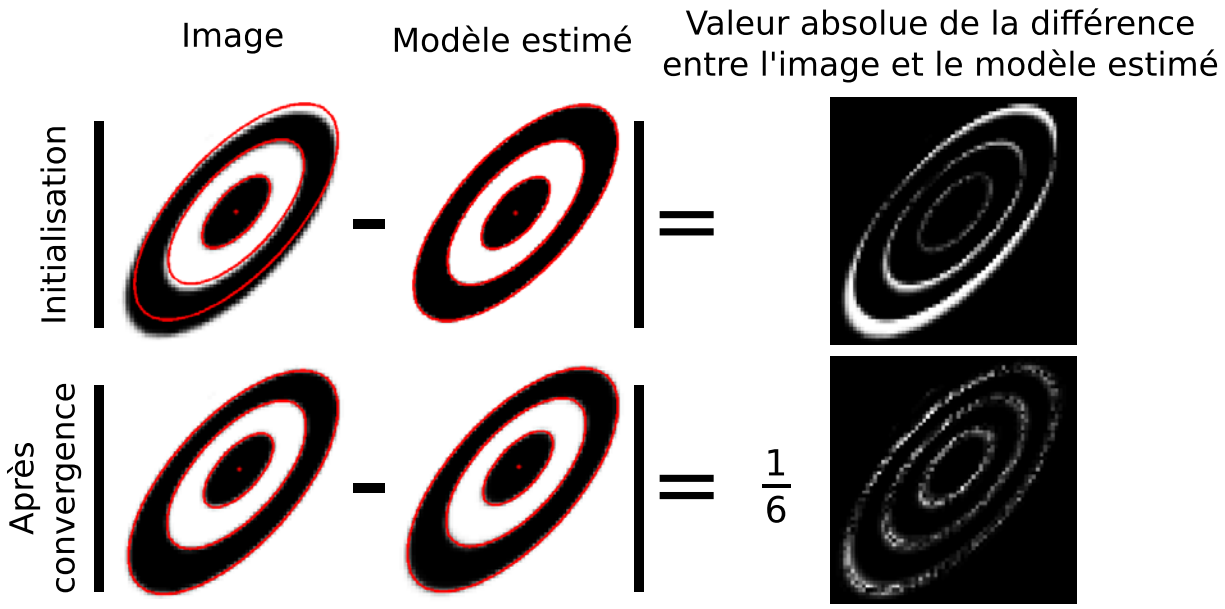


FIGURE 2.8 – Détection sous-pixellique d'ellipses concentriques, avant et après convergence (en déterminant la transformation homographique et photométrique). Sur chaque image, les ellipses (correspondant au modèle estimé) sont superposées en rouge.

La fonction de coût f_{cible} que l'on cherche à minimiser est égale à la somme des carrés de la différence entre l'image acquise I de la cible, et son modèle M estimé (*cf* Figure 2.8). En d'autres termes,

$$f_{cible} = \sum_{(u,v)\in\mathcal{R}} \left(I(u,v) - M(x,y)\right)^2 \tag{2.33}$$

En utilisant les formalisations de la luminance $\mathcal{L}$ et de la transformation géométrique, on obtient :

$$f_{cible} = \sum_{(u,v)\in\mathcal{R}} \left(I(u,v) - \mathcal{L}(\sqrt{x^2+y^2})\right)^2 \tag{2.34}$$

où, x et y sont définis par l'équation 2.31.

Une étape robuste suit l'optimisation : en cas de fausse détection, le candidat est rejeté si l'erreur moyenne quadratique est supérieure à 0.07 (pour une image dont les niveaux de gris sont compris entre 0 et 255).

A cette étape, pour une caméra étalonnée, la pose de la cible peut être déterminée à l'aide des équations (4.4), à une rotation près autour de la normale à la cible passant par son centre.

2.2.1.4 Labellisation automatique

Une cible est labellisée automatiquement grâce à sa géométrie asymétrique. De plus, la labellisation est également possible que la cible soit vue directement ou par son reflet sur un miroir (ce qui se révèle être pratique pour le Chapitre 4). Un code-barre circulaire est composé de trois parties : un entête `H` commun à toutes les cibles, un label `L` et un bit de parité `P` (*cf* Figure 2.9). Nous avons choisi de coder l'entête sur 7 bits et le label sur 8 bits selon le code de Gray. Le bit de parité vaut 1 si la somme des bits de l'entête et du label est paire.

Le code doit pouvoir être lu à la fois si la cible est vue directement, ou indirectement (pour les cibles observées via un reflet et ayant donc subi une symétrie axiale. *cf* Figure 2.10). L'entête n'a donc pas été choisi aléatoirement. Pour trouver sa valeur la plus appropriée, nous avons parcouru exhaustivement les valeurs d'un code-barre. Ainsi, un code-barre circulaire est valide (pour une vue directe) si et seulement si :

- l'entête n'est retrouvé qu'une seule fois dans le sens direct (lecture de l'entête dans le sens contraire des aiguilles d'une montre), par corrélation circulaire du code-barre $[\texttt{HLP}]$ avec l'entête `H`.
- et si l'entête n'est jamais retrouvé dans le sens indirect, par convolution circulaire entre du code-barre $[\texttt{HLP}]$ avec l'entête `H`.

Plusieurs entêtes vérifient ces deux conditions. On choisit comme meilleur entête celui qui permet d'avoir un maximum de codes valides. On sélectionne finalement l'entête $H = 104$ en décimale (*c.-à-d.* 1011100 en code Gray). Le nombre de labels (et donc de codes) valides est de 212 sur un total de $2^8 = 256$.

Grâce à ce choix de l'entête, 212 codes barres circulaires sont potentiellement utilisables aussi bien en vue directe qu'en vue indirecte. A cette étape, pour une caméra étalonnée, la pose de la cible peut maintenant être complètement déterminée à l'aide des équations (4.4).

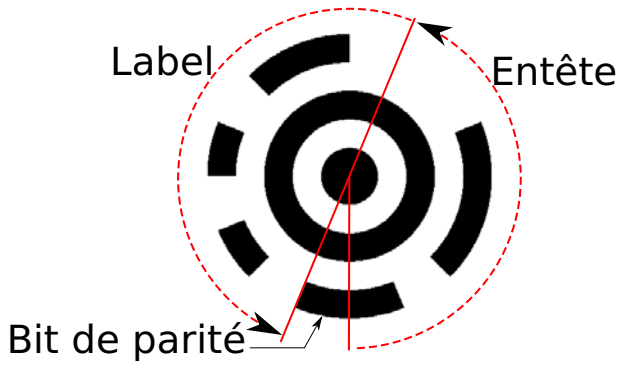


FIGURE 2.9 – Exemple de cible. Son code-barre circulaire est composé d'un entête $\texttt{H} = 104$, d'un label $\texttt{L} = 76$ et d'un bit de parité $\texttt{P} = 1$.

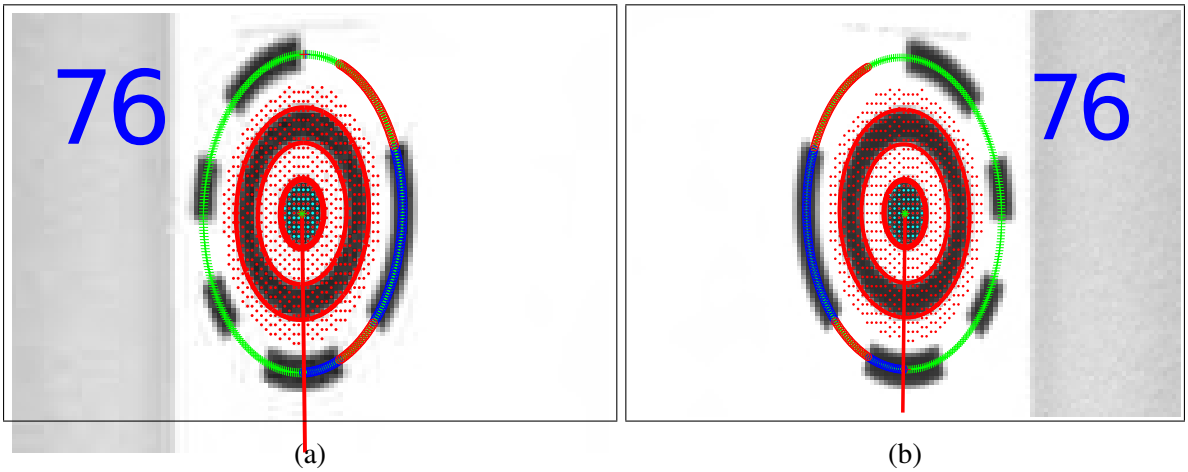


FIGURE 2.10 – Cible identifiée (avec un label $\mathtt{L} = 76$) pour une vue directe (a) ou indirecte (par son reflet sur un miroir) (b). Les pixels appartenant au disque initialement détecté et à l'ellipse englobante sont respectivement distingués par des points cyans et rouges. Les 3 ellipses intérieures détectées sont dessinées en rouge. L'ellipse extérieure, servant à la lecture du code-barre, est dessinée en bleu et rouge pour l'entête, et en vert pour le label et le bit de parité. Le segment rouge représente l'orientation de la cible en délimitant le début de l'entête.

2.2.2 Mire de cibles et protocole d'étalonnage

D'après les conclusions de [LVD99, LVD98], un étalonnage intrinsèque précis ne repose pas sur la construction précise d'une mire, mais plutôt sur la précision de la détection de cette mire. C'est pourquoi nous proposons ici une mire d'étalonnage composée des cibles décrites dans 2.2.1 (*cf* Figure 2.11).

De plus, pour obtenir un étalonnage précis, il est primordial de respecter quelques règles lors de l'acquisition. La mire utilisée doit être rigide. On ne peut pas étalonner précisément une caméra en utilisant une mire collée sur un carton souple... Afin d'avoir la meilleure précision, il est préférable de choisir la plus haute résolution d'image disponible et tout le champ de vue admissible. De bonnes conditions d'illumination sont souhaitées, en adéquation avec l'ouverture de l'objectif et le temps d'exposition (*shutter*). Si la mire ou la caméra est déplacée manuellement lors de l'acquisition des images, un faible temps d'exposition permet de limiter le flou de mouvement.

Pour l'étalonnage intrinsèque, nous conseillons d'utiliser au minimum les images respectant les conditions de prises de vue suivantes :

- 4 vues fronto-parallèles en tournant la mire de $90°$ entre chaque vue (Figure 2.11a),
- une vue proche de la mire dans les 4 coins de l'image (Figure 2.11b),
- une vue éloignée de la mire dans les 4 coins de l'image (Figure 2.11c),
- 4 vues en inclinant la mire vers la gauche, la droite, le haut et vers le bas (Figure 2.11d).

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

FIGURE 2.11 – 16 prises de vue conseillées pour étalonner intrinsèquement une caméra.

Voici une interprétation non-exhaustive de l'utilité de ce protocole d'acquisition :
- Les vues fronto-parallèles permettent d'estimer le ratio f_x/f_y.
- Les vues de la mire sur les bords de l'image permettent de modéliser les distorsions induites par l'objectif.
- Les vues proches et lointaines permettent de mieux estimer la focale. En effet, si la mire était toujours à la même distance de la caméra, il y aurait une ambiguïté entre le zoom et la distance caméra/mire. Cette ambiguïté est d'ailleurs utilisée au cinéma dans l'effet Vertigo.
- Les vues inclinées permettent de contraindre la hauteur des amers de la mire dans son repère (car ces hauteurs sont également optimisées lors de l'étalonnage proposé en Section 2.3.1).

Pour vérifier le bon déroulement de l'étalonnage, on calcule la pose de la mire avec une autre image (n'ayant pas servi à l'étalonnage) et les paramètres intrinsèques fraichement déterminés. On vérifie alors que l'ordre de grandeur de l'écart-type des erreurs de reprojection est le même que pour l'étalonnage.

2.3 Etalonnage intrinsèque et cibles

Les outils d'étalonnage à disposition sont souvent restreints à une seule modélisation intrinsèque. Par exemple, OpenCV [Bra00] et la toolbox Matlab™ [Bou02]) fonctionnent uniquement avec le modèle direct (*cf* Section 2.1.2.1). Quant à [LVD99, LVD98], il n'utilise que le modèle indirect (*cf* Section 2.1.2.2). Les algorithmes d'étalonnage intrinsèque les plus couramment utilisés sont répertoriés et décrits sous un même formalisme dans [ASB02]. Récemment, [CE11] propose un outil pour étalonner un système multi-caméra à champs recouvrants avec un modèle intrinsèque pouvant être adapté et choisi pour chaque caméra. Que ce soit pour [LVD99] ou pour [CE11], une intervention manuelle est souvent nécessaire pour sélectionner sur chaque image au moins un sous-ensemble des amers de la mire. Dans cette section, nous proposons une toolbox permettant un étalonnage intrinsèque d'une caméra avec la mire présentée en Section 2.2.2 et un modèle choisi par l'utilisateur (parmi les 3 modèles présentés dans la section 2.1).

2.3.1 Principe

Considérons une caméra à étalonner (pouvant être modélisée par un modèle de la section 2.1), et une série d'images de la mire enregistrées selon le protocole décrit dans la section 2.2.2. Le but est de déterminer les paramètres intrinsèques du modèle.

Pour chaque image, les cibles de la mire sont détectées de manière robuste (*cf* Figure 2.12). En pratique, les occultations de la mire sont supportées : toutes les cibles n'ont pas forcément besoin d'être observées ou détectées dans une image. Une première phase initialise le système avec des paramètres intrinsèques approximatifs (en calculant les poses de la mire, avec la méthode rappelée en Section 4.2.3 pour une mire 2D, ou la méthode de [DD95] pour une mire 3D). Puis, une optimisation non-linéaire (de type Levenberg-Marquardt) raffine ensuite les paramètres intrinsèques, la géométrie et les poses de la mire en minimisant les erreurs de reprojection.

L'étalonnage intrinsèque est récapitulé par l'algorithme 1. Dans lequel, Int représente le vecteur des paramètres intrinsèques pour le modèle choisi. $\underset{C}{\mathrm{proj}}()$ est la fonction de projection d'un point 3D (exprimé dans le repère de la caméra) pour ce modèle. m_l^j est le $l^{ème}$ amer de la mire détecté sur la $j^{ème}$ image.

Algorithme 1: Étalonnage intrinsèque d'une caméra

Entrées : Paramètres intrinsèques approximatifs Int et images de la mire
Sorties : Paramètres intrinsèques optimisés Int, poses et géométrie de la mire

$1^{ère}$ *étape : Initialisation*

Choisir le modèle intrinsèque.
Pour chaque *image* j **:**
Détecter les M_l points d'amers de la mire.
Calculer la pose T_C^j de la mire dans le repère de la caméra C.

$2^{ème}$ *étape : Optimisation non-linéaire*

$$\underset{\left(Int,\, M_l,\, T_C^j\right)}{\operatorname{argmin}} \left(\sum_j \sum_l \| m_l^j - \underset{C}{\mathrm{proj}}\left(T_C^j \overline{M}_l\right) \|^2\right) \tag{2.35}$$

La géométrie d'une mire, qu'elle soit élaborée manuellement (par exemple avec des pastilles réfléchissantes) ou issue d'une imprimante (qui ne conserve pas les longueurs), n'a pas besoin d'être parfaitement connue si cette mire peut être détectée avec précision. En effet, tout comme dans [LVD99, LVD98], nous optimisons également la géométrie de la mire. Les points M_l d'amers de la mire sont définis dans un repère lié à celle-ci. Pour éviter une sur-modélisation redondante avec la pose de la mire, sept paramètres de la géométrie de la mire restent constants. Pour cela, considérons trois points non-colinéaires de la mire. L'origine du repère de la mire est

fixée sur le premier point (3 paramètres fixés). L'axe x passe par le second point (de coordonnées constantes $(d_x, 0, 0)$, fixant 3 autres paramètres) et le troisième a sa coordonnée nulle selon l'axe z (1 paramètre fixé). Chacun des autres points d'amer de la mire a ses 3 coordonnées optimisées.

2.3.2 Avantages/inconvénients par rapport à l'existant

Comme l'illustre la Figure 2.12, les méthodes "classiques" incluent une méthode robuste lors de la phase d'optimisation. Il s'agit de l'étape où les points mal détectés de la mire doivent être classés comme outliers. Les points bien détectés sont censés être classés comme inliers. L'inconvénient des approches "classiques" est qu'elles sont sensibles aux mauvaises initialisations : des points bien détectés peuvent avoir de fortes erreurs de reprojection et être classés comme outliers. Ceci se vérifie notamment sur les bords de l'image fortement distordue avec de mauvais coefficients de distorsion. Ces points, les plus sujets aux distorsions, risquent d'être rejetés à tort. Avec l'outil développé par [LVD99, LVD98], il n'est pas rare de devoir augmenter successivement le degré du polynôme modélisant la distorsion (équation 2.12) pour pallier ce problème.

A contrario, notre approche inclut la méthode robuste lors de la détection. Aucune cible bien détectée ne risque d'être éliminée. Ceci permet de pouvoir initialiser plus grossièrement les paramètres intrinsèques, et donc d'avoir un bassin de convergence plus large. De plus, contrairement à la toolbox Matlab™ [Bou02], et à la méthode [LVD99, LVD98], l'utilisateur n'a pas besoin d'indiquer où sont les cibles : nul besoin de cliquer fastidieusement sur chaque image.

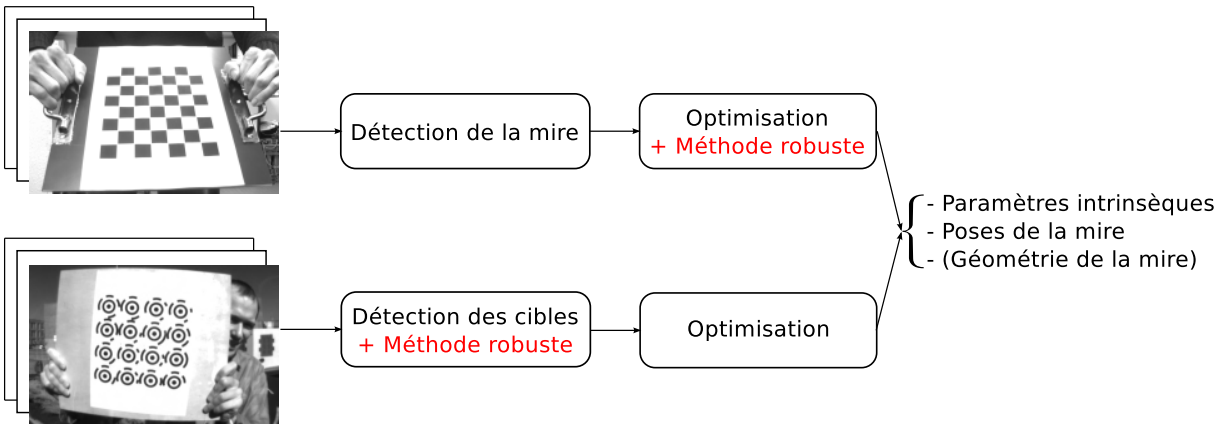


FIGURE 2.12 – Comparaison entre les méthodes "classiques" d'étalonnage intrinsèque basés mire et notre méthode. Seules les cibles détectées avec précision sont utilisées par notre algorithme.

En pratique, la détection est rapide pour un damier, et assez lente pour la mire avec les cibles à cause de la minimisation non-linéaire qui optimise leurs géométries et leurs réponses

photoniques. Notons que cette lenteur est directement liée à l'implémentation sous Matlab™. Une version utilisant C++ (et openGL) devrait permettre une détection en temps réel de la mire. La version temps réel et affine de notre détecteur (allégée du raffinement de l'homographie présenté en Section 2.2.1.3) a été utilisée dans [BALM] pour des vues fronto-parallèles.

Concernant les conditions d'éclairement de la mire : en cas de surexposition, les zones noires sont rognées par les zones blanches. Pour un damier, ses coins intérieurs ne s'intersectent plus, et sont donc détectés avec moins de précision. Nos cibles restent détectables avec suffisamment de précision, car le rognage est en partie modélisé par le paramètre h (*cf* explication sur la transformation photométrique p19). Une amélioration possible de la détection initiale de la cible (Section 2.2.1.2), serait de considérer au moins une seconde ellipse pendant ou après la détection de la région elliptique noire (par exemple avec le détecteur utilisé dans [CGC+11]). On pourrait ainsi remplacer l'initialisation affine du pattern par une initialisation perspective. Une autre amélioration du détecteur de cible serait de prendre en compte la distorsion ou de détecter les ellipses sur des images corrigées. Notons que cela ne s'est pas avéré nécessaire dans nos expérimentations (ce qui revient à supposer que les distorsions soient localement négligeables). D'autre part, bien qu'en pratique, la paramétrisation utilisée permet à l'optimisation de converger, seuls 7 paramètres suffisent à modéliser la transformation géométrique. En effet, le motif (disque et couronne) est invariant par rotation autour de lui-même (*c.-à-d.* toute homographie de la forme $HR_z(\alpha) = H \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}$ est équivalente à H pour modéliser la transformation géométrique du motif). Pour avoir cette paramétrisation minimale, on peut soit rajouter une contrainte à la solution proposée, soit utiliser l'équation A.9, avec une rotation R' constante (et redéfinir l'intervalle de chaque paramètre à optimiser). Alternativement, un motif ne présentant cette symétrie centrale (par exemple en rajoutant l'entête du code-barre circulaire) permettrait une paramétrisation minimale de la géométrie avec 8 paramètres. Il serait également intéressant d'étudier plus en détail l'influence du choix de la fonction de transition (pour la réponse photométrique des cibles). On pourrait par exemple la remplacer par la fonction d'erreur de Gauss $erf(\frac{x}{\sigma})$, en paramétrant la pente par l'écart-type σ (*cf* [Bra95]).

2.3.3 Résultats

Cette toolbox d'étalonnage intrinsèque (développée durant le projet VIPA) a été éprouvée sur de nombreuses caméras. C'est le cas de tous les étalonnages intrinsèques nécessaires aux parties suivantes (Chapitres 4 à 6). Nous présentons les résultats obtenus sur une caméra (de marque Dolphin™) de focale 5.6 mm, avec les trois modèles intrinsèques décrits en Section 2.1.

On peut vérifier le bon déroulement de l'étalonnage *quantitativement* (par les erreurs de reprojection). Après convergence d'un étalonnage, la moyenne des erreurs de reprojection est inférieure à un millième de pixel selon les axes u et v (ce qui signifie juste que le barycentre des résidus sont bien centrés sur $(0, 0)$). Mais l'information pertinente est surtout attribuée aux écarts-types des erreurs de reprojection (dans le plan image, selon u et v), donnés par la Table 2.1. Leur ordre de grandeur est d'environ 0.03 pix pour les modèles direct et indirect.

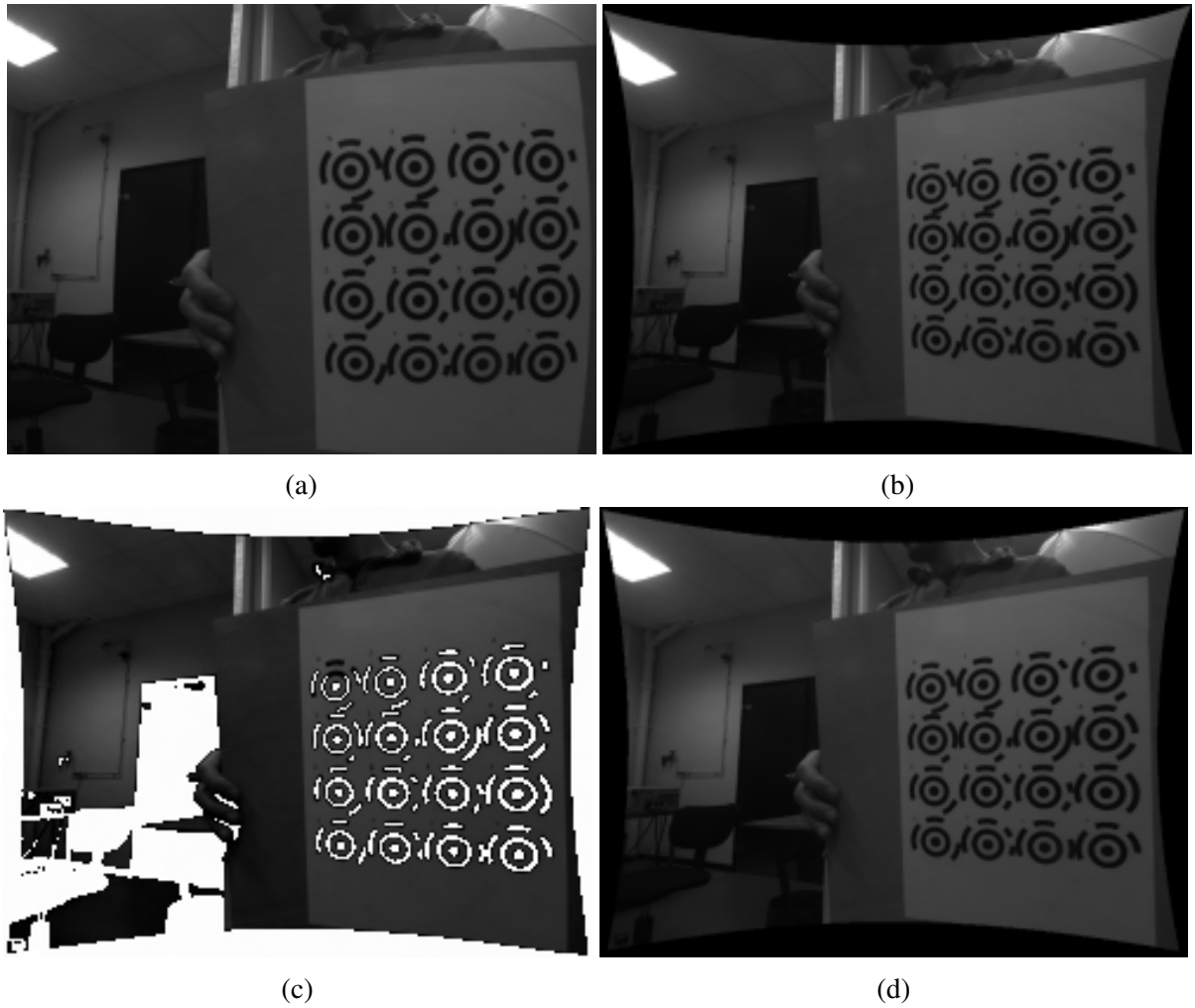
On remarque que pour le modèle unifié, qui a le moins de paramètres pour modéliser la distorsion, les écarts-types sont légèrement plus élevés d'environ $0.02\ pix$ par rapport aux deux autres modèles.

Mais on peut aussi vérifier *qualitativement* le bon déroulement de l'étalonnage. Il suffit de corriger la distorsion d'une image avec les paramètres intrinsèques calculés (*cf* Figure 2.13 qui illustre la correction de la distorsion après les trois étalonnages distincts). Il suffit alors de vérifier que les lignes droites dans l'image corrigée sont belles et bien droites. Pour générer l'image corrigée, on parcourt chaque pixel de cette image, puis on calcule les coordonnées de son point correspondant dans l'image originale (en appliquant à ces coordonnées la fonction appliquant la distorsion spécifique au modèle [1]). Comme les coordonnées obtenues ne sont pas entières dans l'image originale, l'image corrigée est finalement obtenue par interpolation.

Modèle direct	Modèle indirect	Modèle unifié
$f_x = 1107.1$ $f_y = 1107.2$ $u_0 = 788.1$ $v_0 = 618.4$	$f_x = 1106.6$ $f_y = 1106.7$ $u_0 = 787.9$ $v_0 = 618.3$	$f_x = 2191.5$ $f_y = 2192.4$ $u_0 = 787.8$ $v_0 = 618.5$
$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} -0.2536 \\ 0.1246 \\ -4.357\ .\ 10^{-2} \end{pmatrix}$	$\begin{pmatrix} b_1 \\ b2 \\ b_3 \\ b_4 \\ b_5 \end{pmatrix} = \begin{pmatrix} 0.2465 \\ 0.1538 \\ -0.4363 \\ 0.8522 \\ -0.5934 \end{pmatrix}$	$\xi = 0.9810$
Ecart-type des erreurs de reprojection après étalonnage		
$\sigma_u = 0.027$ $\sigma_v = 0.028$	$\sigma_u = 0.030$ $\sigma_v = 0.032$	$\sigma_u = 0.052$ $\sigma_v = 0.046$
Ecart-type des erreurs de reprojection pour une autre image		
$\sigma_u = 0.059$ $\sigma_v = 0.064$	$\sigma_u = 0.059$ $\sigma_v = 0.067$	$\sigma_u = 0.087$ $\sigma_v = 0.099$

TABLE 2.1 – Résultats de trois étalonnages intrinsèques qui utilisent le modèle direct, indirect ou bien unifié. La caméra est de marque Dolphin™, de focale 5.6 mm. 304 points sont répartis sur les 19 images utilisées. La résolution des images est 1600×1200.

1. Pour le modèle indirect, la fonction de correction de distorsion n'est pas analytique, mais est obtenue par une table de correspondance entre la distance radiale corrigée r_u et distordue r_v



(a) (b)

(c) (d)

FIGURE 2.13 – Correction de la distorsion après l'étalonnage intrinsèque (algorithme 1). Image originale (a). Correction avec le modèle direct (b), indirect (c), ou unifié (d).

2.4 Correction automatique d'images distordues

Nous proposons dans cette section, une méthode alternative à l'étalonnage intrinsèque réalisé dans la section précédente (2.3). Le but est de corriger automatiquement la distorsion à partir d'une image d'un environnement structuré, en déterminant certains paramètres intrinsèques. Il s'agit d'estimer automatiquement les paramètres qui permettront de corriger l'image d'origine. Pour cela, on fait l'hypothèse que l'on observe un environnement structuré présentant des segments rectilignes (ce qui peut être intéressant pour un calibrage en ligne dans un environnement urbain). A partir d'une image distordue (où ces segments ne sont pas rectilignes), on souhaite obtenir une image non-distordue (où ces segments seront rectilignes), en déterminant les paramètres de distorsion.

2.4.1 Etat de l'art

Dans la communauté photogrammétrique, [Bro71] est l'un des premiers à exploiter le fait que les droites observées dans une image devraient avoir une courbure nulle. Il propose une méthode analytique avec des droites issues d'une grille plane. Dans une approche souvent citée [DF01], chaque contour est détecté avec une précision sous-pixellique, chainé puis approximé par des segments successifs. Les paramètres intrinsèques modélisant la distorsion sont optimisés itérativement, en réévaluant à chaque itération l'approximation polygonale des contours et en minimisant les erreurs associées à chaque segment. [TSR06] modélise la distorsion en considérant que la distance focale varie en fonction de la distance radiale. A l'aide de points sélectionnés comme appartenant à une droite, une résolution algébrique répétée itérativement permet d'estimer les paramètres de la focale. Il propose ensuite d'extraire ces points non-pas d'une seule image, mais à partir de plusieurs images en contraignant le déplacement de la caméra : soit à des mouvements composés de plusieurs translations pures (ou bien de rotations pures) en suivant des points 3D ; soit à des mouvements en observant un objet plan (les points suivis sont initialisés sur la première image, sur une droite passant par le point principal). [CF05] modélise les paramètres intrinsèques par une fonction rationnelle : un point (u, v) d'abord projeté sans distorsion via le modèle sténopé classique (*cf* Section 2.1.1), et ensuite distordu en un point (p, q) tel que $(p,\ q)^\top = \pi\left(A\,(u^2\ uv\ v^2\ u\ v\ 1)^\top\right)$, avec $A \in \mathbb{R}^{3\times 6}$. Les droites de l'espace se projettent dans le plan image en des coniques. Après avoir détecté puis chainé les pixels des contours, une estimation linéaire de la matrice A est proposée. Une optimisation non-linéaire permet de raffiner A (avec uniquement 2 paramètres) et les paramètres des droites, en minimisant la distance entre les coniques (la projection des droites estimées) et leurs points de contours. Une approche similaire est proposée dans [YHZ06] pour des caméras fisheye : après avoir identifié des droites (représentées par un plan passant par le centre optique), leurs points de contours sont projetés sur une sphère unitaire centrée sur la caméra (proche du modèle intrinsèque unifié de la Section 2.1.3). Les erreurs minimisées sont les distances entre les points sur la sphère aux plans correspondants.

Un des inconvénients des approches précédentes est qu'une même droite peut être traitée comme un ensemble disjoint de segments (par exemple en cas d'occultation de la droite). L'information sur la distorsion est traitée localement sur chaque segment, alors qu'il serait plus contraignant que tous les segments d'une même droite apportent une même contribution. C'est ce que fait [RL11], où chaque point de contour est représenté par son orientation. En effet, les contours issus d'une courbe ont différentes orientations, alors que les contours issus d'une même droite ont la même orientation. En minimisant l'entropie de l'histogramme des orientations, les pics dans cet histogramme sont accentués et les contours redressés, tout en estimant les paramètres de distorsion. L'inconvénient principal de cette approche est que si différentes droites de l'image ont des orientations légèrement différentes, l'algorithme aura alors tendance à fusionner leur orientation. C'est par exemple le cas avec une image quasi-fronto parallèle d'un échiquier : les droites parallèles dans l'espace ne le sont plus dans l'image. Un simple histogramme des orientations n'est pas suffisamment discriminant. Cette approche ne fonctionne donc qu'avec des vues parfaitement fronto-parallèles, ou bien avec des vues où les droites présélectionnées ont des orientations suffisamment éloignées. D'autres moyens pour pallier le problème d'occultation de droites sont présentés dans [FP01] qui exploite des corrélations dans le domaine fréquentielle d'une image distordue, ou bien dans [BA06, BA03]. Cette dernière approche, se limitant aux caméras catadioptriques avec un miroir parabolique, permet d'approximer des droites de l'image par des coniques. Pour le modèle unifié, cela signifie que ξ reste constant à 1 (*cf* [Bar06]), et l'article estime alors itérativement les paramètres restants (*c.-à-d.* la matrice des paramètres intrinsèques). Les points appartenant à chaque droite sont vraisemblablement sélectionnés manuellement. [CGPV03] utilise une méthode assez proche de celle que nous avons retenue : basée sur une transformée de Hough mais uniquement appliquée sur une région de l'image (contenant une droite) sélectionnée manuellement, et un modèle polynomial de distorsion radiale.

2.4.2 Notre approche

Afin de modéliser la distorsion induite par l'objectif de la caméra, on se base sur les modèles intrinsèques décrits dans la section 2.1, plus particulièrement le modèle indirect (Section 2.1.2.2) et le modèle unifié (Section 2.1.3). L'approche proposée permet de déterminer certains paramètres du modèle. En ce sens, l'algorithme d'auto-correction des distorsions présenté ci-après s'interprète également comme un auto-étalonnage intrinsèque partiel d'une caméra. Le caractère automatique est lié notamment à l'absence de mire d'étalonnage.

Il s'agit d'une approche paramétrique basée primitives. Une primitive est constituée des coordonnées d'un point image (appartenant aux contours), associées à une orientation (qui définit la direction du contour). Les primitives sont représentées par un ensemble de petites flèches sur les figures 2.14b et 2.14c. Une optimisation non-linéaire (de type méthode de Nelder-Mead [LRWW99]) optimise itérativement les paramètres de distorsions Υ. Afin d'avoir des primitives déformées (*cf* Figure 2.14c), on ne déforme pas l'image d'entrée à chaque itération, pour ensuite détecter de nouvelles primitives. Au lieu de cela, les primitives sont tout d'abord extraites

sur l'image d'entrée (*cf* Figure 2.14a), puis déformées. Elles sont les seules données servant à déterminer les paramètres du modèle. Finalement, pour estimer les paramètres de déformation, on emprunte le chemin (a $\rightarrow$ b $\rightarrow$ c) de la Figure 2.14. Une fois qu'ils sont estimés, on les utilise pour rectifier l'image d'entrée (*cf* Figure 2.14(a $\rightarrow$ d)). Notons que contrairement à certaines approches présentées dans l'état de l'art, il est ici inutile de chainer les edgels pour les identifier comme appartenant à un même segment.

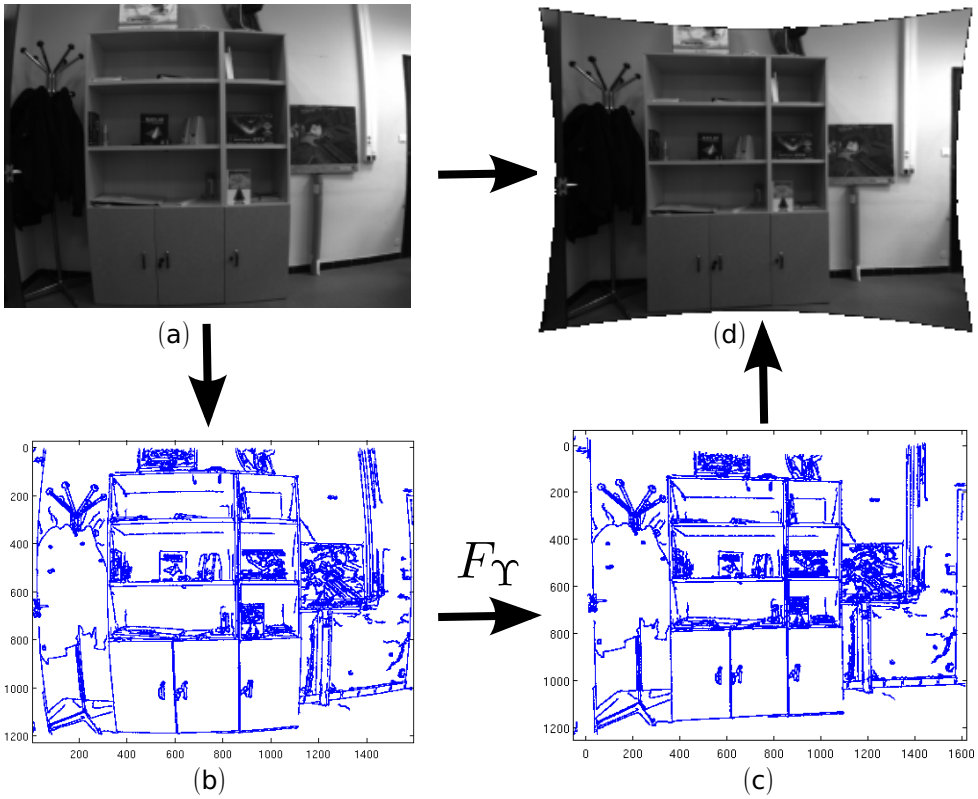


FIGURE 2.14 – Correction automatique d'images distordues : à chaque itération, on ne déforme pas l'image d'entrée (a), mais uniquement les primitives de cette image (b $\rightarrow$ c). Une fois les paramètres de distorsions estimés (c), on corrige l'image entière (a $\rightarrow$ d).

L'algorithme de correction automatique des distorsions est récapitulé dans l'Algorithme 2 et illustré dans la Figure 2.15.

Pour la représentation polaire, on n'utilise pas les coordonnées polaires du point (u'_i, v'_i), mais plutôt les coordonnées de la droite portant la primitive déformée (*cf* Figure 2.16). Il s'agit d'une paramétrisation des droites identique à la transformée de Hough [2].

2. Seule la paramétrisation est identique à la transformée de Hough. Pour celle-ci, un pixel vote comme appartenant potentiellement à un ensemble de droites. Dans notre approche, chaque primitive portée par une droite (dont la paramétrisation (ρ'_i, θ'_i) est connue) contribue à augmenter la densité de probabilité qu'il existe une droite dans l'image portant ces primitives.

Algorithme 2: Correction automatique des distorsions à partir d'une image

1. Détecter des primitives :
   - Détecter des contours (avec l'algorithme de Canny [Can86]).
   - Calculer les gradients en u et v de l'image (g_u et g_v).
   - Déduire une image de l'orientation des contours (*c.-à-d.* la perpendiculaire au gradient : $atan2(g_v, g_u) - \frac{pi}{2}$).
   - Conserver les M points des contours et leur orientation.
2. Définir le vecteur initial Υ_0 des paramètres de distorsion.
3. Déterminer le vecteur Υ en maximisant la fonction de coût suivante :
   - Appliquer la déformation F_Υ aux primitives u_i, v_i, θ_i.
   - Passer à une représentation polaire ρ'_i, θ'_i.
   - Estimer à l'aide d'un noyau gaussien discret la carte de densité de probabilité des ρ'_i, θ'_i.
   - Calculer le coût (faisant la somme des carrés de chaque valeurs de la carte).
4. Corrigée la distorsion de l'image d'origine avec le vecteur des paramètres Υ.

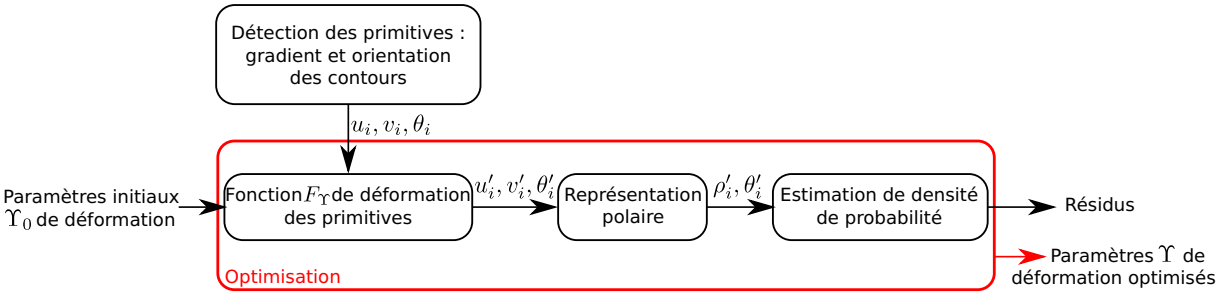


FIGURE 2.15 – Principe de correction automatique des distorsions d'une image.

Ainsi, $\rho'_i \in \mathbb{R}$ représente la distance orthogonale du milieu de l'image à cette droite et $\theta'_i \in]0,\ \pi]$ représente l'angle entre l'horizontal et cette droite (*c.-à-d.* l'orientation du contour déformé). La représentation des angles étant restreinte à un intervalle de largeur π, les effets de bords sont gérés lors de la création de la carte de densité de probabilité (plus précisément, lors de la pondération par un noyau gaussien). On cherche donc à augmenter les maximums globaux de la carte : plus les densités de probabilité s'agglomèrent en maximums locaux, plus les primitives corrigées sont portées par un ensemble de droites.

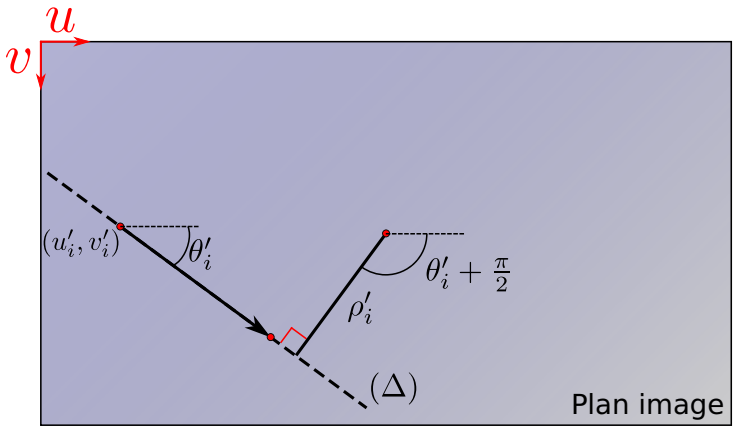


FIGURE 2.16 – Paramétristion d'une droite portée par la primitive (u'_i, v'_i, θ'_i).

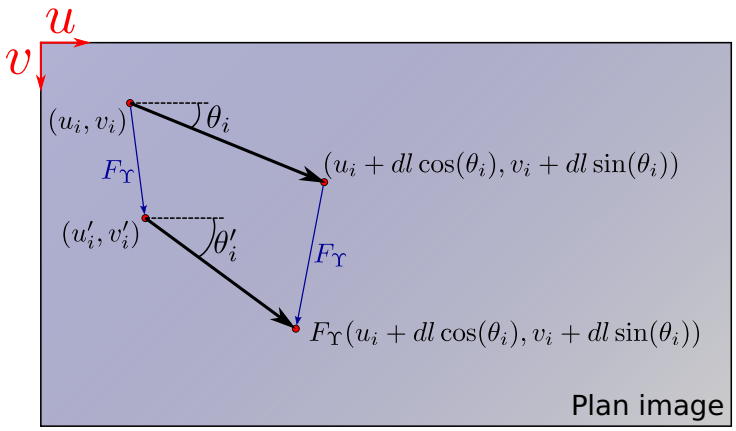


FIGURE 2.17 – Principe du calcul d'une primitive (u'_i, v'_i, θ'_i) déformée par la fonction F_Υ, à partir de la primitive détectée (u_i, v_i, θ_i).

Afin de calculer une primitive déformée (u'_i, v'_i, θ'_i), il suffit d'appliquer la fonction de déformation F_Υ à chaque extrémité de la flèche (comme l'illustre la Figure 2.17), puis d'en déduire

son orientation θ'_i. Pour cela, les flèches ont une taille dl infinitésimale. En effet, la fonction F_Υ s'applique aux coordonnées (u_i, v_i) d'un point image. Pour le modèle indirect, $F_\Upsilon = I$ (*cf* équation 2.10) et $\Upsilon = (b_1, b_2, b_3, b_4, b_5)$. Pour le modèle unifié, $F_\Upsilon = U_{dist}$ (*cf* équation 2.27) et $\Upsilon = \xi$. On fait l'hypothèse que la focale est constante. Pour le modèle unifié, il faut être vigilant par rapport au domaine de définition de l'équation 2.22. On choisit une focale qui respectera ce domaine de définition au cours de l'algorithme. Pour le modèle unifié, il faut choisir une valeur cohérente de la focale du plan image non-distordu. En effet, si on conserve la focale initiale, les primitives sont à la fois étirées vers l'extérieur (distorsion radiale), mais aussi fortement zoomées. Afin que l'image non-distordue ait une taille proche de l'image d'origine, on impose qu'à mi-hauteur, l'image de sortie ait la même demi-largeur que l'image d'origine (*c.-à-d.* que la largeur entre la correction des points $(\frac{L}{4}, \frac{H}{2})$ et $(\frac{3L}{4}, \frac{H}{2})$ reste constante). Pour la carte de probabilité, on fixe la résolution en distance polaire ρ'_i pour qu'elle ne varie pas lors de l'optimisation.

2.4.3 Résultats

On exécute l'algorithme proposé sur l'image 2.14a. Avec le modèle unifié, on initialise $\Upsilon_0 = \xi = 0$ pour obtenir $\xi = 0.400$ après convergence. Ce qui correspond au minimum de la fonction de coût donnée en Figure 2.18. Les primitives déformées et l'image corrigée sont illustrées respectivement par les Figures 2.14(c) et 2.14d. Les cartes de densité de probabilité des ρ_i, θ_i sont représentées avec les paramètres initiaux (*cf* Figure 2.19a) et après convergence (*cf* Figure 2.19b).

L'algorithme d'auto-correction des distorsions est de nouveau exécuté avec le modèle indirect, initialisé avec les coefficients de distorsion b_k à zéro. On obtient après convergence $\Upsilon = (b_1, b_2, b_3, b_4, b_5) = (0.2308, 0.07963, -6.295\ .\ 10^{-3}, 5.904\ .\ 10^{-5}, -1.222\ .\ 10^{-4})$. Les primitives déformées et l'image corrigée sont illustrées respectivement par les Figures 2.20a et 2.20b. Pour chaque modèle intrinsèque, on vérifie qualitativement que les primitives sont belles et bien droites.

2.4.4 Perspectives

Nous venons de présenter une méthode assez générique pour corriger automatiquement les distorsions d'une image. La méthode est dite générique, car on peut *a priori* choisir tout modèle intrinsèque possédant une fonction de correction de distorsion des primitives. Plus de tests sont à effectuer, notamment pour des caméras dont le champ de vue est proche des 180°. Pour ces caméras, il n'est plus possible de rectifier l'image entière (il serait d'ailleurs intéressant d'effectuer plusieurs rectifications perspectives, en regardant des directions différentes que l'axe optique). Il serait alors surement plus pertinent de remplacer la représentation des primitives en distance polaire ρ'_i dans le plan image, par l'angle entre l'axe optique et la normale à la primitive ramenée sur la sphère unitaire (*c.-à-d.* remplacer ρ'_i par $\tan^{-1}(\frac{\rho'_i}{f'})$).

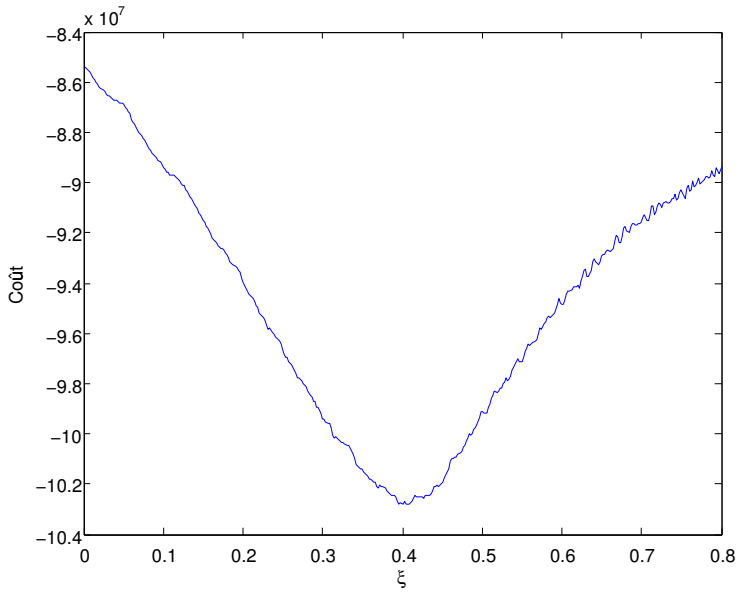
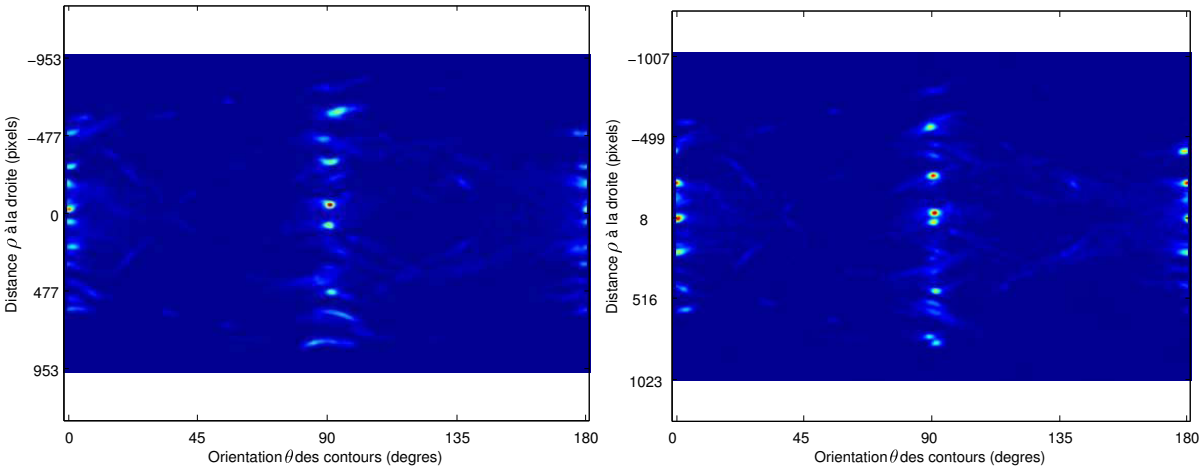


FIGURE 2.18 – Fonction de coût pour l'image 2.14a et le modèle unifié en fonction du paramètre ξ.

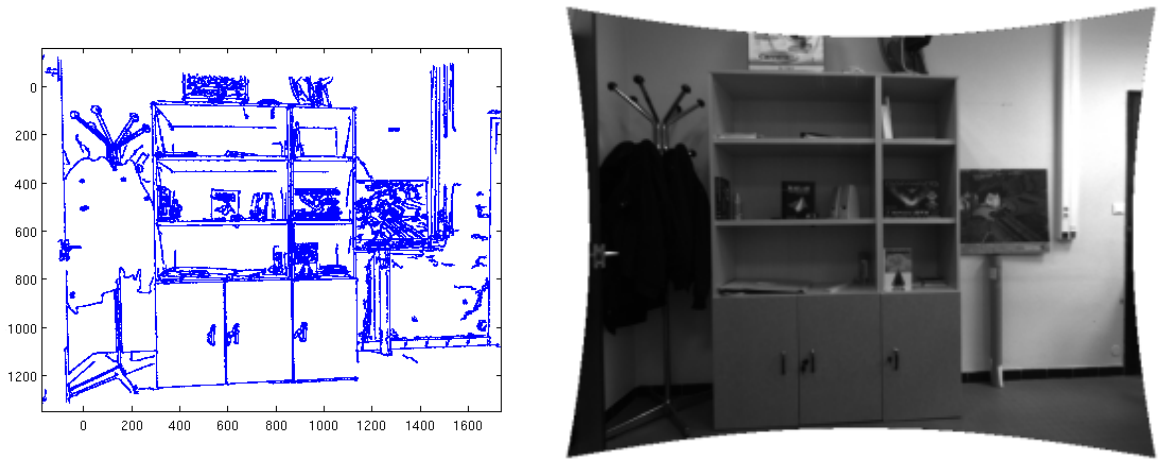
L'estimation de la densité de probabilité utilisée discrétise l'espace de représentation des droites (espace polaire). Il serait intéressant de la remplacer par une estimation continue (par exemple une version continue de l'estimation de densité de probabilité par noyau), ou bien de modifier la méthode pour ne plus faire intervenir cette discrétisation. Avec une meilleure paramétrisation des droites, le bruit de détection des gradients (sur la position et l'orientation) serait mieux modélisé. Le but étant d'obtenir un estimateur du maximum de vraisemblance avec ce bruit. Un détecteur de contours plus précis améliorerait les résultats, car l'orientation initiale des tangentes aux contours serait moins bruitée. Il serait également possible de rejeter automatiquement des points n'appartenant pas à des droites (ce qui lisserait la fonction de coût). Ces améliorations énoncées dans ce paragraphe sont proposées avec une approche probabiliste dans [4].

Cette méthode d'auto-correction présente l'avantage d'être très simple à mettre en œuvre du point de vue de l'utilisateur. On pourrait rajouter une fonctionnalité de sélection automatique de l'image à traiter (ou une sélection de primitives sur plusieurs images, tout comme dans [CF05]). Un étalonnage simple pourrait venir compléter cette méthode afin de déterminer les paramètres intrinsèques restants (en utilisant par exemple les points de fuite). Il faudrait toutefois quantifier si ce gain en simplicité ne se fait pas au détriment de la précision, et savoir qu'elle en serait l'influence sur l'application visée (par exemple, une reconstruction).



(a) (b)

FIGURE 2.19 – Cartes de densité de probabilité des ρ_i, θ_i avec (a) les paramètres initiaux $\Upsilon_0 = \xi = 0$ et (b) après convergence avec $\Upsilon = \xi = 0.400$ (modèle unifié).



(a) (b)

FIGURE 2.20 – (a) Primitives issues de l'image 2.14a ayant subi la déformation avec le modèle indirect après convergence de l'algorithme. (b) Image corrigée à partir des paramètres du modèle indirect estimés par l'algorithme d'auto-correction des distorsions.

Chapitre 3

Etat de l'art : modélisation et utilisation des systèmes multi-caméra à champs non-recouvrants

Dans ce chapitre, nous nous penchons sur comment modéliser un ensemble de caméras. Puis nous balayons les différentes utilisations de caméras à champs non-recouvrants, qu'elles soient statiques (cas de la vidéo-surveillance [JRAS03]), ou bien en mouvement (si elles sont par exemple embarquées sur un véhicule). Dans les chapitres 4 et 5, nous reviendrons plus en détail sur chaque type d'étalonnage, à travers un état de l'art spécifique à la méthode considérée.

3.1 Modélisations multi-caméra

Alors que les systèmes multi-caméra sont fréquemment utilisés (comme le montre [SRT+11, §2.4] qui dresse un historique des systèmes multi-caméra remontant jusqu'à l'année 1884) et que la littérature regorge d'étalonnages de caméras à champs recouvrants, les travaux portant sur des caméras à champs de vue disjoints sont plus récemment explorés (*cf* Figure 3.1). Tout d'abord, nous étudions les modélisations de systèmes multi-caméra utilisés dans la littérature. Puis, partant de ce constat, nous décrirons le modèle que nous avons choisi.

Considérons un ensemble de N caméras. Comme expliqué dans le glossaire (p187), on appelle *paramètres extrinsèques* les paramètres de la transformation rigide entre deux caméras (*c.-à-d.* leurs poses relatives). Dans la littérature, certains auteurs optent pour la même définition, alors que d'autres désignent une autre entité avec ce même terme. C'est pourquoi il est vivement conseillé de lire les définitions de "paramètres extrinsèques", de "pose" et de "caméra" du glossaire, afin d'éviter toute ambigüité.

(a) (b) (c)

FIGURE 3.1 – Systèmes multi-caméra à champs non-recouvrants ou partiellement recouvrants. (a) et (b) sont respectivement utilisés par [KHK08] et [KD06, KD10]. Le capteur Ladybug® (c) est notamment utilisé par [LHK08, PFF+10].

3.1.1 Modèles existants

Différents types de modèles de système multi-caméra sont disponibles dans la littérature. Nous les avons regroupés en trois catégories.

Modèle redondant Certains optent pour la redondance des paramètres extrinsèques. C'est le cas de l'algorithme multi-caméra basé sur un EKF (*cf* glossaire p187) et proposé par [PW]. Ainsi, il n'y a pas de notion de caméra maître ou esclave : les $N(N-1)/2$ transformations entre chaque couple de caméras sont modélisées afin de pouvoir leur associer une incertitude liée aux mesures.

Caméras génériques Le modèle générique [GN01b] peut modéliser tout aussi bien une ou plusieurs caméras, qu'elles soient centrales ou non. Ce modèle est notamment utilisé dans [MLD+09, DHLH, RLS06] pour des applications de reconstruction 3D. On appelle *image générique*, la concaténation de l'ensemble des images acquises par la ou les caméras à un instant donné. A chaque pixel de cette image générique est associé un unique rayon optique défini dans un repère donné. L'étalonnage consiste donc à déterminer les coordonnées des rayons de tous les pixels dans un repère unique. Dans [SR03], des expérimentations sont menées avec des capteurs non-centraux et une mire plane. Des caméras panoramiques ont ensuite été calibrées dans [SRL05], mais d'autres expérimentations sont nécessaires pour savoir si cette méthode est adaptée pour les caméras à champs disjoints. Il faudrait alors utiliser plusieurs mires pour étalonner à la fois intrinsèquement et extrinsèquement ces caméras (considérées comme une seule caméra non-centrale). Sachant que cette méthode nécessite au moins 3 vues d'une mire dans chaque zone de l'image générique, elle semble être peu adaptée pour les caméras embarquées sur nos véhicules (*cf* Figure 1.3).

Plusieurs caméras, un seul centre optique Dans certains travaux [CCPB09, CI01, KHK10, KHK08], la baseline entre les caméras est supposée nulle. Tout se passe comme si l'ensemble du système multi-caméra était une seule caméra centrale (avec un seul centre optique). Cette hypothèse simplificatrice permet d'appliquer des algorithmes pour les caméras centrales (comme pour le calcul de pose), ou encore de calculer les homographies entre les plans images des caméras perspectives. Avec cette hypothèse d'un centre optique commun à toutes les caméras, certains modèles intrinsèques centraux peuvent être appliqués à un système multi-caméra. C'est le cas du modèle d'*image sphérique* où, pour un système mono ou multi-caméra central, le plan image est remplacé par une image sphérique sur laquelle sont projetés les points de la scène. [LBL10, Lim10, LB10] utilisent ce modèle d'image sphérique et la notion de points antipodaux (points diamétralement opposés sur la sphère). Ces points alimentent un flot optique afin d'estimer le mouvement du système en déplacement (egomotion). C'est également l'approche et le but de [HC09], qui s'inspire de la morphologie animale (de certains oiseaux), pour estimer le mouvement grâce à une paire stéréo dont les caméras latérales sont diamétralement opposées. D'après cet auteur, cette approche est moins sensible aux défauts d'étalonnage que les méthodes classiques de SfM.

3.1.2 Modèle retenu

Il est toutefois possible d'opter pour une autre modélisation qui formalise les paramètres extrinsèques et intrinsèques de manière différente (*cf* [SMP05]). Dans les chapitres suivants, nous avons choisi de modéliser un système multi-caméra par un ensemble de N caméras distinctes (dont les centres optiques diffèrent), de la manière suivante :

1. Le modèle indirect est choisi pour les paramètres intrinsèques de chaque caméra.
2. On choisit de paramétrer un minimum de transformations entre les couples de caméras (ce sont les paramètres extrinsèques).

Nous avons fait le choix n°1, car comme expliqué en Section 2.1.2.2, il suffit de corriger la distorsion des primitives détectées pour se ramener à un simple modèle perspectif pour chaque caméra. Le choix n°2 est préférable pour les systèmes d'équations algébriques et les méthodes de types ajustement de faisceaux, car il vaut mieux éviter une sur-paramétrisation. L'une des caméras est choisie arbitrairement comme *caméra maître*, les $N-1$ autres sont appelées *caméras esclaves*. Seules les matrices de passage entre le repère de la caméra maître et le repère de chaque caméra esclave sont définies (plus de détails sont donnés dans le Chapitre 5). Si besoin est, des matrices de passage entre deux caméras esclaves peuvent s'exprimer simplement par des combinaisons de celles déjà définies.

3.2 Utilisation des systèmes multi-cameras

Dans cette section, nous dressons un état de l'art sur l'utilisation des systèmes multi-caméra, qu'ils soient à champs recouvrants ou disjoints, statiques ou embarqués.

3.2.1 A champs recouvrants

Lorsque les champs des caméras sont recouvrants, il est aisé d'utiliser un même objet observé simultanément par chaque caméra pour déterminer leurs poses relatives. Ce problème a été largement traité (notamment pour le cas stéréo-caméra), et publié en de nombreuses variantes. Certains utilisent une mire d'étalonnage [PST99, BD08b], d'autres utilisent en pointeur laser [SMP05], ou encore des points d'intérêt. Il en résulte l'utilisation de différents outils comme le calcul d'homographie entre le plan image et une mire plane, des calculs de poses pour des mires non-planes ou encore des ajustements de faisceaux. Dans la communauté photogrammétrique, [LNCS10] propose un ajustement de faisceaux en contraignant la baseline entre les caméras. Récemment, [BD08a, Baj10] propose un livre sur l'étalonnage d'un ensemble de caméras statiques (et à champs recouvrants) basé sur la théorie des graphes. Les caméras forment les nœuds d'un graphe, dont les branches sont pondérées par la précision de leurs poses relatives. Le graphe est parcouru selon le coût minimal, pour déterminer (de proche en proche) les poses des caméras dans un repère global.

3.2.2 A champs non-recouvrants

3.2.2.1 Statiques

Des caméras statiques à champs disjoints peuvent être calibrées en suivant un objet mobile passant successivement plusieurs fois devant chaque caméra. Grâce à des *a priori*, l'information manquante sur la trajectoire de l'objet est alors estimée dans les zones non observables [ATC, RDD04, WLS11] (pour des applications de vidéo-surveillance). [AP09] considère un ensemble de caméras à champs disjoints et un objet rigide suffisamment large pour qu'il soit vu simultanément par toutes les caméras. En suivant des points indépendamment dans chaque caméra (supposée affine), l'objet mobile est reconstruit par une méthode de factorisation multi-caméra.

Dans [Mou07], l'auteur évoque l'utilisation une caméra mobile supplémentaire se déplaçant à proximité des caméras statiques à étalonner. La scène observée par la caméra mobile est reconstruite. Comme cette scène est partiellement observée par les caméras statiques, il suffit de calculer leurs poses par rapport à cette scène pour obtenir l'étalonnage extrinsèque.

3.2.2.2 Embarquées

Des caméras à champs disjoints peuvent également être embarquées sur un mobile. [Ple03] étudie quelle est la meilleure disposition d'un ensemble de caméras rigidement liées pour l'estimation du mouvement (*egomotion*). En se basant sur des observations du flot optique dans chaque caméra, il cherche à remonter au mouvement du système multi-caméra. Pour un ensemble de dispositions préalablement choisies, il détermine quelles sont les corrélations existantes (et donc les ambigüités) entre les paramètres de rotation et de translation modélisant le mouvement du système. Il utilise pour cela des outils statistiques comme la matrice d'information de Fischer. Il en conclut que la meilleure disposition de deux caméras est de les mettre tête-bêche. [CWYO07] arrivent expérimentalement à la même conclusion avec un système binoculaire fixé sur un casque pour suivre les mouvements d'une tête. C'est également la disposition que nous avons choisie pour les véhicules du projet VIPA. En pratique, cette disposition présente d'autres avantages pour des applications de localisation, comme la symétrie du véhicule et la robustesse face aux conditions extérieures d'illumination. Dans [GdVLL10], quatre caméras calibrées et à champs disjoints sont utilisées pour estimer le mouvement à partir du flot optique. [RWCC07, RWCC] s'interroge sur l'influence du nombre de caméras sur la précision d'une approche EKF pour le suivi de pose. Il conclut, qu'en terme de précision, une paire stéréo à champs-recouvrants est généralement équivalente à deux paires stéréos à champs-recouvrants fixées dos à dos, aucune étude n'est faite sur une paire stéréo à champs non-recouvrants.

Lorsque des caméras à champs disjoints ne sont pas synchronisées, l'approche de [CXF06] permet d'aligner temporellement les séquences d'images de chaque caméra (en se basant sur les matrices fondamentales). [SI03] estime le mouvement à l'aide de caméras actives (aux champs potentiellement disjoints). Chaque caméra est dite "active", car elle tourne sur elle-même afin de suivre un point 3D.

Les systèmes multi-caméra embarqués sont également utilisés à des fins de cartographie et de trajectographie. Chaque orientation des différents capteurs utilisés dans [RHK$^+$11] est déterminée par l'intermédiaire d'une centrale inertielle. Des caméras (grands-angles) peuvent également être embarquées pour informer le conducteur en éliminant les angles morts. [HGJD09] présente à la fois des aspects techniques et législatifs (dans différents pays) pour ces outils d'aide à la conduite.

Une fois les paramètres extrinsèques estimés, l'estimation minimale de la pose est possible [CKF$^+$08, LHK08, RW, Kim08], pour localiser le système multi-caméra [FKK04, KHFP07, KHK08, SIY04, PFF$^+$10]. [KHK08] et [LHK08, PFF$^+$10] estiment le mouvement avec respectivement le système de la Figure 3.1a et Figure 3.1c. D'autres peuvent alors utiliser des algorithmes de reconstruction à partir du mouvement (Structure From Motion) basés sur un filtre de Kalman étendu EKF [KCPL08, KC06, KJCTC10, RWCC07], du SLAM [CAD11a], des ajustements de faisceaux [SIY04, SÅ04, SOÅ05], ou sur une approche probabiliste [KD06, KD10] (qui utilise le dispositif de la Figure 3.1b). Certains de ces articles déclarent utiliser un étalonnage extrinsèque manuel. Cependant, des mesures approximatives des angles ou des distances (avec un mètre) peuvent être trop imprécises.

Chapitre 4

Étalonnage extrinsèque multi-caméra et miroir

Ce chapitre présente un étalonnage extrinsèque multi-caméra en ayant recours à un miroir plan. La principale contribution consiste à utiliser une scène de géométrie inconnue et de créer un champ de vision commun entre les différents capteurs statiques ou embarqués. De plus, l'influence de la réfraction induite par le miroir est étudiée. Une méthode d'estimation de pose d'un objet réfracté est proposée. Les méthodes sont validées par des simulations et une expérience métrologique.

4.1 Introduction et état de l'art

En vision, l'utilisation de miroir plan n'est pas rare. L'auteur de [NJ04] place un objet entre la caméra et des miroirs plans pour étalonner le système et reconstruire l'objet. Pour cela, il suppose que les points sont à la fois visibles directement et indirectement via le miroir. Dans des conditions expérimentales similaires, [MSMP09] utilise une mire pour déterminer la pose relative de deux miroirs plans et calculer la pose de la caméra grâce à une approche multi-vue (*cf* Figure 4.1a).

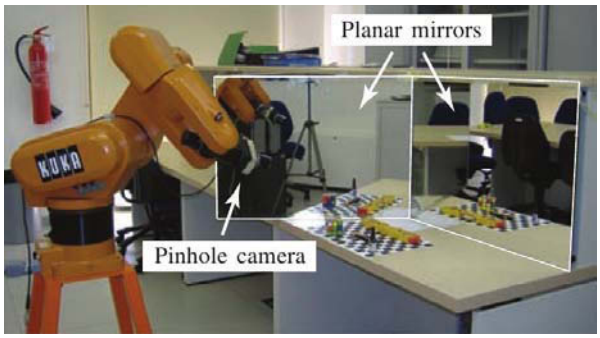
Une approche originale est présentée dans [AmS11] pour détecter la présence de réflexions (induites par un miroir plan) dans une image. Une autre utilisation des miroirs plans est faite dans [GN98, GN01a] pour réaliser une tête stéréo à l'aide d'une seule caméra. [NN98] étudie la géométrie épipolaire pour des systèmes stéréo en utilisant des miroirs plans, ellipsoïdaux, hyperboliques et paraboliques. Dans [JSO10], les auteurs proposent un système d'acquisition composé d'une caméra fish-eye et de 4 miroirs plans rigidement liés (*cf* Figure 4.1e).

Récemment [SB06], plusieurs miroirs plans ont été utilisés pour déterminer la pose d'une mire d'étalonnage, par rapport à une caméra (*cf* Figure 4.1b). Dans sa thèse [Bon06], Bonfort effectue également des calculs de pose via des surfaces réfléchissantes (qu'il cherche à reconstruire). Des travaux similaires [RBN10] ont été menés avec un miroir plan mobile et une mire

statique : à partir des poses des caméras virtuelles, l'auteur estime les plans des miroirs et la pose de la caméra réelle par résolution d'un système linéaire.

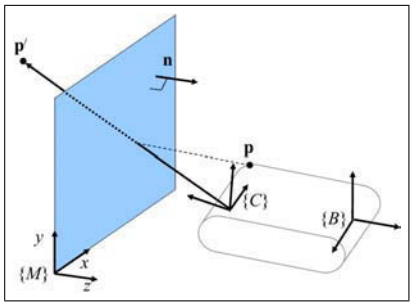
Les auteurs de [HMR08a, HMR08b, HMR10] utilisent un robot mobile s'approchant d'un miroir plan statique. Le but est de déterminer analytiquement la pose du robot par rapport à la caméra. Pour cela, une mire d'étalonnage est fixée sur le robot (*cf* Figure 4.1c).

Enfin, un miroir peut être utilisé pour obtenir des champs de vision recouvrants. Des travaux ont été effectués sur l'étalonnage extrinsèque d'un système multi-caméra, via un miroir et une mire plane connue [KIFP] (*cf* Figure 4.1d). Ces derniers articles sont les plus proches de nos travaux.

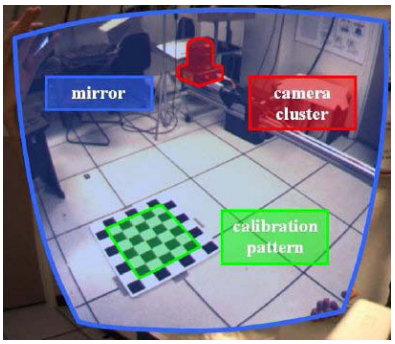


(a) [MSMP09]

(b) [SB06, Bon06]



(c) [HMR08a, HMR08b, HMR10]



(d) [KIFP]

(e) [JSO10]

FIGURE 4.1 – Quelques utilisations de miroirs plans en vision artificielle.

Dans ce chapitre, nous proposons une méthode d'étalonnage extrinsèque qui ne nécessite pas d'*a priori* sur la géométrie de la scène. C'est la principale originalité de notre approche par rapport à l'état de l'art existant. Des amers visuels sont collés sur le miroir afin d'initialiser les calculs non linéaires. Rappelons que le but est de déterminer la transformation rigide entre chaque caméra (*c.-à-d.* les paramètres extrinsèques) à partir d'appariements 2D/2D. Après une

formalisation du problème, nous présenterons deux méthodes distinctes pour estimer les poses relatives entre différentes caméras rigidement liées. Leurs champs de vision peuvent être totalement disjoints. Nous verrons ensuite comment appliquer ces méthodes à un robot mobile. Dans la section 4.4, nous étudions et traitons l'impact de la réfraction induite par le miroir. Enfin, la section 4.5 valide notre approche avec des résultats issus de simulations et d'expériences métrologiques.

4.2 Formalisation mathématique

4.2.1 Vue d'ensemble

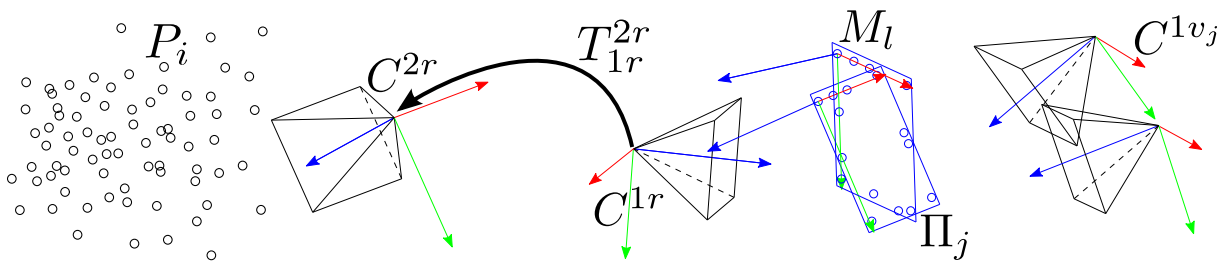


FIGURE 4.2 – Schéma du système multi-caméra / miroir.

$S = \{P_i\}$	Scène de points 3D (appartenant éventuellement à une mire)
C^{1r}, C^{2r}	Première et deuxième caméras réelles
Π_j	Miroir observé par la $j^{ème}$ image de C^{1r}
C^{1v_j}	$j^{ème}$ caméra virtuelle de C^{1r} par rapport à Π_j
M_l	$l^{ème}$ marqueurs collés sur le miroir
$p_i^{1v_j}$	Projection du point P_i dans le plan image de C^{1v_j}
p_i^{2r}	Projection du point P_i dans le plan image de C^{2r}
m_l^{1r}	Projection d'un marqueur M_l dans le plan image de C^{1r}
T_{1r}^{2r}	Transformation homogène entre C^{1r} et C^{2r}
$T_{1r}^{\Pi_j}$	Transformation homogène entre C^{1r} et Π_j
T_{2r}^{S}	Transformation homogène entre C^{2r} et S

TABLE 4.1 – Récapitulatif des notations utilisées dans ce chapitre.

Considérons une scène $S = \{P_i\}$ de points 3D, et au moins deux caméras rigidement liées C^{1r} et C^{2r}, aux paramètres intrinsèques connus. Le système est illustré par la Figure 4.2, et les notations récapitulées dans le Tableau 4.1. La caméra C^{2r} observe directement la scène. La caméra C^{1r} observe différentes vues de la scène via le miroir mobile (en déplaçant le miroir entre chaque vue). Soit T_{1r}^{2r} la transformation homogène permettant de passer du repère de C^{1r} à celui de C^{2r}. De même $T_{1r}^{\Pi_j}$ désigne la transformation homogène entre le repère de C^{1r} et celui de Π_j, où Π_j désigne le miroir observé par la $j^{ème}$ image de C^{1r}.

On appelle *caméra virtuelle*, la caméra obtenue par la symétrie par rapport au plan du miroir. On note C^{1v_j} la $j^{ème}$ caméra virtuelle de C^{1r} par rapport à Π_j. On suppose que l'on a plusieurs images prises par C^{1r} et une autre image de la scène prise par C^{2r}. $p_i^{1v_j}$ et p_i^{2r} sont les projections respectives du point P_i dans le plan image de C^{1v_j} et de C^{2r}. La Figure 4.2 illustre l'ensemble de notre système. Les caméras sont représentées par leur repère, direct pour les caméras réelles, et indirect pour les caméras virtuelles. En effet, la symétrie de l'espace induite par le miroir transforme une base orthonormée directe en une base orthonormée indirecte.

Un ensemble de 9 marqueurs $\{M_l\}$ (représentés par les cercles bleus dans la Figure 4.2) sont collés à la surface du miroir pour estimer précisément les poses du miroir Π_j par rapport à C^{1r} (*cf* Section 4.2.3). Leur disposition n'a besoin que d'être approximativement connu dans le repère attaché au miroir. On note m_l^{1r} la projection d'un marqueur M_l dans le plan image de C^{1r}. Les marqueurs sont des pastilles blanches réfléchissantes sur un fond noir (*cf* Figure 4.3 ou Figure 4.10b). Leur détection automatique ainsi que leur raffinement géométrique et photométrique sont similaires à ceux que nous avons développés dans les sections 2.2.1.2 et 2.2.1.3. Après l'étape de détection automatique, un pré-filtrage est appliqué pour conserver les 9 marqueurs les plus similaires. Ils sont ensuite labellisés automatiquement en utilisant l'asymétrie de leur disposition. En cas d'erreur (mauvaise labellisation), l'image est rejetée. L'utilisation de marqueurs présente plusieurs intérêts :

- L'ajout de contraintes géométriques supplémentaires.
- L'estimation de la pose du miroir permet de déduire la relation entre une caméra réelle et sa caméra virtuelle. Ainsi, contrairement à [KIFP] il n'est pas nécessaire de résoudre un système linéarisé (sous-optimal), mais plutôt un système "exact". De plus, contrairement au même article, qui présente des configurations singulières (par exemple, le miroir ne peut pas être orthogonal à l'axe optique de la caméra), notre méthode n'interdit aucune pose au miroir. La tache qui incombe à l'utilisateur est donc plus souple.

4.2.2 Paramètres intrinsèques d'une caméra virtuelle

On souhaite modéliser la projection des points de la scène dans le plan image d'une caméra virtuelle. Le but étant de pouvoir utiliser des algorithmes classiques (comme le calcul de pose) avec une caméra virtuelle. Pour cela, nous allons voir que le concept d'une matrice des paramètres intrinsèques peut être étendu aux caméras virtuelles. Certains articles [RBN10] ont fait le choix d'attribuer à une caméra virtuelle les paramètres intrinsèques de la caméra réelle dont

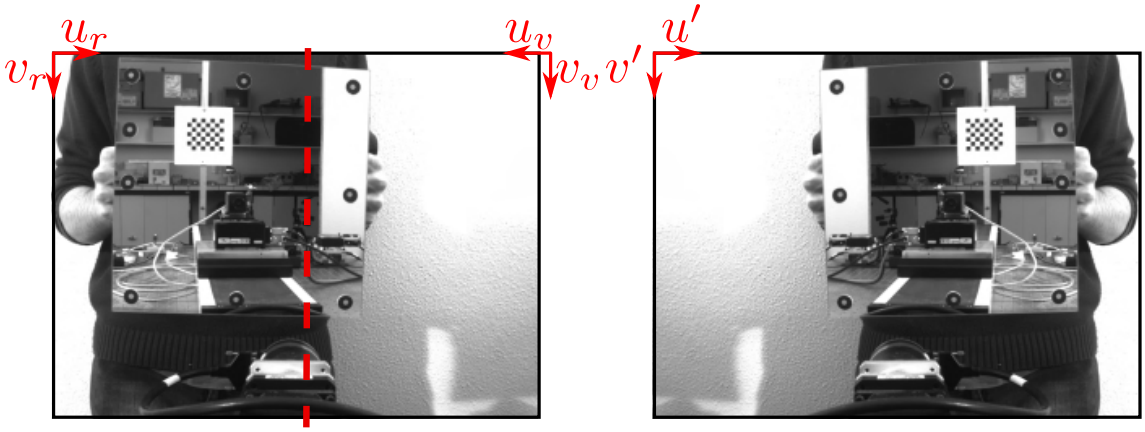
elle est issue. Mais nous proposons de les définir légèrement différemment. Considérons les paramètres intrinsèques K_r d'une caméra réelle dans le cas du modèle sténopé (*cf* équation 4.2 et Section 2.1.1).

L'image de la caméra virtuelle s'obtient par une symétrie axiale de notre choix. Afin d'obtenir une matrice intrinsèque virtuelle simple, on choisit l'axe vertical passant par le point principal [1]. En notant respectivement (u_r, v_r) et (u_v, v_v) les coordonnées dans le plan image de la caméra réelle et de la caméra virtuelle, et (x, y) les coordonnées dans le plan normalisé de la caméra réelle, on a alors :

$$\begin{cases} u_r = f_x\, x + u_0 \\ v_r = f_y\, y + v_0 \end{cases} \Leftrightarrow \begin{cases} u_v = 2u_0 - u_r = -f_x\, x + u_0 \\ v_v = v_r = f_y\, y + v_0 \end{cases} \tag{4.1}$$

La matrice des paramètres intrinsèques K_v d'une caméra virtuelle s'écrit alors :

$$K_r = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow K_v = \begin{pmatrix} -f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.2}$$



(a) (b)

FIGURE 4.3 – Image issue d'une caméra réelle (a) et image virtuelle (b) obtenue par symétrie par rapport à l'axe vertical passant par le point principal.

Imposer que l'axe de symétrie soit vertical (en rouge dans la Figure 4.3), contrairement à [AmS11] qui impose seulement que l'axe de symétrie passe par le point principal, est un choix assez pragmatique.

1. Cette problématique est également évoquée dans [FMW03].

Ainsi, lorsque nous souhaitons travailler avec des primitives issues des images virtuelles, nous pouvons soit :

- détecter les primitives dans l'image réelle puis appliquer une symétrie sur les coordonnées (avec l'équation 4.1).
- détecter directement les primitives de coordonnées (u', v') dans l'image de la Figure 4.3b (en ayant permuté la gauche et la droite sur l'image réelle), puis appliquer une translation pour obtenir les coordonnées virtuelles (*cf* Equation 4.3).

$$\begin{cases} u' = 2\frac{L}{2} - u_r = L - u_r \\ u_v = 2u_0 - u_r \end{cases} \Rightarrow \begin{cases} u_v = u' - L + 2u_0 \\ v_v = v' = v_r \end{cases} \tag{4.3}$$

4.2.3 Calcul de la pose du miroir

Nos algorithmes d'étalonnage commencent avec le calcul des poses du miroir par rapport à C^{1r}. A partir d'une connaissance approximative de la disposition des marqueurs (mesurés grossièrement avec une règle), on calcule l'homographie H entre l'objet et le plan image de la caméra l'observant. H est estimée linéairement (avec l'algorithme "normalized Direct Linear Transformation" [HZ04, p109]), puis raffinée avec une optimisation non-linéaire (avec l'algorithme [HZ04, p114]). La pose $(R, \mathbf{t})$ des objets plans est ensuite obtenue par une décomposition de l'homographie (*cf* [Zha00]) via les équations :

$$\begin{cases} \lambda = \|K^{-1}\mathbf{h}_1\|^{-1} = \|K^{-1}\mathbf{h}_2\|^{-1} \\ \mathbf{t} = \lambda K^{-1}\mathbf{h}_3 \end{cases} \text{ et } \begin{cases} \mathbf{r}_1 = \lambda K^{-1}\mathbf{h}_1 \\ \mathbf{r}_2 = \lambda K^{-1}\mathbf{h}_2 \\ \mathbf{r}_3 = \mathbf{r}_1 \wedge \mathbf{r}_2 \end{cases} \tag{4.4}$$

Avec $R = [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{r}_3]$, $H = [\mathbf{h}_1\ \mathbf{h}_2\ \mathbf{h}_3]$ et K la matrice des paramètres intrinsèques de la caméra considérée. Une fois que les poses du miroir sont calculées, elles sont ensuite raffinées en même temps que la géométrie des marqueurs par un ajustement de faisceaux [LVD99, LVD98].

4.3 Étalonnage de caméras à champs disjoints

Dans cette section, nous proposons deux types d'étalonnage de caméras à champs disjoints : le premier utilise une mire connue, le second n'en a pas besoin. Puis, nous présentons comment appliquer la procédure d'étalonnage pour un robot mobile.

4.3.1 Étalonnage avec une mire connue

Supposons que la géométrie de la scène est connue (*c.-à-d.* les points P_i appartiennent à une mire). L'algorithme 3 présente la procédure d'étalonnage.

Algorithme 3: Étalonnage extrinsèque avec une mire connue

Entrées : p_i^{2r}, $p_i^{1v_j}$, m_l^{1r}, M_l et P_i
Sorties : T_{1r}^{2r},$T_{1r}^{\Pi_j}$, et M_l

$1^{ère}$ *étape : Initialisation*

Pour chaque *image* j **:**
Calculer la pose du miroir Π_j dans le repère de C^{1r}.

Raffiner la géométrie des marqueurs M_l et les poses du miroir Π_j par un ajustement de faisceaux.

Pour chaque *image j* **:**
Déduire la pose de la caméra virtuelle C^{1v_j} par symétrie par rapport à Π_j.
Calculer la pose de la mire $S = \{P_i\}$ par rapport à C^{1v_j}.

Calculer la transformation T_{2r}^{S} entre la caméra C^{2r} et la scène S.
En déduire une transformation T_{1r}^{2r} initiale en "moyennant" les différentes estimations [Gra01, Moa02, Pen98].

$2^{ème}$ *étape : Optimisation non-linéaire*

$$\underset{\left(T_{1r}^{2r},T_{1r}^{\Pi_j},T_{2r}^{S},M_l\right)}{\operatorname{argmin}} \Big(\sum_l \sum_j \| m_l^{1r} - \underset{C^{1r}}{\operatorname{proj}}(M_l) \|^2 + \sum_i \left[\| p_i^{2r} - \underset{C^{2r}}{\operatorname{proj}}(P_i) \|^2 + \sum_j \| p_i^{1v_j} - \underset{C^{1v_j}}{\operatorname{proj}}(P_i) \|^2 \right] \Big) \tag{4.5}$$

Avec $\underset{C}{\operatorname{proj}}(P)$, la fonction projetant le point P, exprimé dans le repère de C, dans le plan image de cette caméra. Avec $\pi([XYZ]^T) = \frac{1}{Z}(X\ Y)^T$ (*cf* Equation 2.7), et avec la composition des transformations homogènes (équation A.2), on a :

$$\begin{aligned} \underset{C^{1r}}{\operatorname{proj}}(M_l) &= \pi(K_{1r} T_{1r}^{\Pi_j} M_l) \\ \underset{C^{2r}}{\operatorname{proj}}(P_i) &= \pi(K_{2r} T_{2r}^{S} P_i) \\ \underset{C^{1v_j}}{\operatorname{proj}}(P_i) &= \pi(K_{1v_j} T_{1v_j}^{1r} T_{1r}^{2r} T_{2r}^{S} P_i) \end{aligned} \tag{4.6}$$

où la symétrie entre C^{1v_j} et C^{1r} est représentée par la matrice homogène $T_{1v_j}^{1r}$ (équation 4.7).

$$T_{1v_j}^{1r} = \begin{pmatrix} I_3 - 2n_j n_j^T & 2d_j n_j \\ 0_{1\times 3} & 1 \end{pmatrix} \tag{4.7}$$

Ici, n_j est le vecteur normal au miroir exprimé dans le repère de C^{1r}. d_j est la distance entre C^{1r} et Π_j.

La mire peut éventuellement être plane. Dans la phase d'initialisation, la pose d'un objet plan est obtenue par le système d'équations 4.4. Si la mire d'étalonnage n'est pas plane, la pose d'un objet 3D est estimée par la méthode décrite dans [DD95]. Dans les deux cas, la pose est raffinée par une optimisation non-linéaire. La dernière étape de l'algorithme 3 optimise l'ensemble des paramètres du système (la pose relative des caméras réelles, les poses du miroir, la pose de la mire par rapport à la caméra réelle C^{2r} et la géométrie des marqueurs) avec un algorithme de Levenberg Marquardt, en minimisant toutes les erreurs de reprojection.

En pratique, l'utilisateur doit déplacer le miroir afin que les points de la mire soient vus simultanément par la caméra C^{2r} en direct et par la caméra C^{1r} dans le reflet du miroir.

4.3.2 Etalonnage sans mire connue

Nous proposons maintenant une méthode d'étalonnage (Algorithme 4) qui suppose que la géométrie des points de la scène S est inconnue : aucune mire d'étalonnage n'est nécessaire. Ainsi, on ne restreint plus le champ de vision à la seule mire. On peut utiliser toute l'intersection du champ d'observation fourni par le miroir avec le champ de vue de C^{2r}. Le but est de rendre la procédure d'étalonnage plus souple et plus aisée pour l'utilisateur. En effet, il n'est pas nécessaire que toute la scène soit visible à chaque observation via le miroir (*c.-à-d.* les occultations de la scène sont supportées). Pour cela, nous utilisons les cibles (cf Section 2.2.1) qui sont détectées précisément et automatiquement. Les cibles sont automatiquement labellisées, que la vue soit directe, ou indirecte (*cf* Section 2.2.1.4 et Figure 2.10).

Ici, contrairement à l'algorithme 3, la dernière étape optimise également chaque point de la scène. Il est uniquement nécessaire de connaître précisément la distance entre deux de ces points, afin de déterminer le facteur d'échelle [2] de l'ensemble du système. Dans le schéma proposé, C^{1r} et C^{2r} ne jouent pas un rôle symétrique. En effet, la quantité de données issues de C^{1r} est d'autant plus grande que le nombre de poses du miroir est élevé, alors que l'information issue de C^{2r} ne provient que d'une seule image. Les méthodes sont décrites ainsi pour des raisons de clarté. Cependant, il est très simple de les rendre symétriques en prenant en compte, dans le processus d'optimisation, de nouvelles images obtenues en plaçant le miroir devant C^{2r}.

De plus, il est inutile d'utiliser un processus d'élimination des points aberrants. En effet, en utilisant des cibles (*cf* Section 2.2.1) comme points de la scène, les fausses détections sont éliminées lors de la détection ou lors de la labellisation.

2. inhérent au processus d'étalonnage.

Algorithme 4: Etalonnage extrinsèque sans mire connue

Entrées : p_i^{2r}, $p_i^{1v_j}$, m_l^{1r} et M_l
Sorties : T_{1r}^{2r},$T_{1r}^{\Pi_j}$, M_l et P_i

$1^{ère}$ *étape : Initialisation :*

Pour chaque *image* j **:**
Calculer la pose du miroir Π_j dans le repère de C^{1r}.

Raffiner la géométrie des marqueurs M_l et les poses du miroir Π_j par un ajustement de faisceaux.

Pour chaque *image j* **:**
Déduire la pose de la caméra virtuelle C^{1v_j} par symétrie par rapport à Π_j.

Reconstruire les points P_i de la scène S dans le repère de la caméra C^{1r} : à partir des C^{1v_j} et $p_i^{1v_j}$, les points sont triangulés (par exemple avec une factorisation projective [HZ04, p 444]), puis optimisés par un ajustement de faisceaux.
Calculer la pose $(T_{2r}^S)^{-1}$ de la caméra C^{2r} par rapport à la scène S. Comme la scène est exprimée dans le repère de C^{1r}, on obtient alors la pose initiale $T_{1r}^{2r} = (T_{2r}^S)^{-1}$.

$2^{ème}$ *étape : Optimisation non-linéaire :*

$$\underset{\left(T_{1r}^{2r},T_{1r}^{\Pi_j},P_i,M_l\right)}{\operatorname{argmin}} \Big(\sum_l \sum_j \| m_l^{1r} - \underset{C^{1r}}{\operatorname{proj}}(M_l) \|^2 + \sum_i \left[\| p_i^{2r} - \underset{C^{2r}}{\operatorname{proj}}(P_i) \|^2 + \sum_j \| p_i^{1v_j} - \underset{C^{1v_j}}{\operatorname{proj}}(P_i) \|^2 \right] \Big) \tag{4.8}$$

4.3.3 Etalonnage appliqué à un robot mobile

Jusqu'à présent, le système multi-caméra était statique. Toutefois, il est possible de l'embarquer sur un robot mobile, ce qui modifie légèrement l'algorithme 4. La procédure d'étalonnage décrite ci-après tire parti du mouvement du robot.

Procédure : Etalonnage extrinsèque pour un robot mobile

Déplacer le véhicule en enregistrant $K + 1$ images de la scène.
Arrêter le véhicule de manière à ce qu'une des caméras observe la scène. On la note C^{2r}. Dans notre cas, il s'agit de la caméra avant.
Déplacer le miroir mobile face à la caméra arrière, notée C^{1r} en enregistrant $K + 1$ images.
Appliquer la méthode d'étalonnage sans mire connue (algorithme 4).

Contrairement à l'algorithme 4, après l'étape de reconstruction on estime une pose pour les $K + 1$ observations de la scène. Elles sont ensuite prises en compte lors de l'optimisation non-linéaire. L'ensemble des observations supplémentaires de la scène permet :

– une meilleure reconstruction de la scène 3D.
– d'avoir le même nombre d'images pour chaque caméra.

4.4 Influence de la réfraction

Cette section a pour but d'étudier l'influence de la réfraction — induite par la couche de verre d'un miroir aluminé sur la face arrière — sur l'estimation de la pose d'un objet vu par son reflet sur le miroir. Dans [LRL03, LRL00] la réfraction est prise en compte pour des caméras immergées, tout comme dans [TSS08, MGP93] où l'interface eau-verre-air est modélisée. D'autres travaux, associant réfraction et vision par ordinateur, ont été publiés récemment. Dans [CS09], les bases de la géométrie multi-vue sont revisitées dans le cas où une caméra et une scène observée ne sont pas dans le même milieu. [IKL$^+$10] dresse un état de l'art des méthodes de reconstruction d'objets transparents (et réfractant). [CWM$^+$11] estime la profondeur d'un objet à partir de deux images : une vue directe et une vue à travers un milieu réfractant (en intercalant une plaque de verre entre l'objet et la caméra).

En effet, si la réfraction est ignorée, alors les algorithmes d'étalonnage peuvent converger vers des résultats biaisés. Nous allons voir comment modéliser la réfraction et la prendre en compte lors de l'étalonnage. On peut tout aussi bien utiliser un miroir aluminé sur la face avant, pour lequel le problème de la réfraction ne se posera pas.

4.4.1 Caméra perspective et réfraction induite par le miroir

Ce paragraphe a pour but d'étudier l'influence de la réfraction d'un miroir sur l'observation d'un objet par une caméra centrale (ici, perspective). On ramène l'observation d'un objet par miroir réfractant à un problème équivalent : l'observation d'un objet virtuel à travers une lame de verre. Par symétrie, la lame de verre équivalente est deux fois plus épaisse que celle du miroir. D'après la première loi de Snell-Descartes, on peut se rapporter au plan contenant le centre optique de la caméra C, le point P' observé et la normale au miroir. Le phénomène de réfraction est illustré par la Figure 4.4. Pour le cas du miroir, le point virtuel P' est le symétrique du point réel P par rapport au plan réfléchissant du miroir. e correspond à l'épaisseur du miroir (et donc $2e$ à l'épaisseur de la lame de verre équivalente), i_c est l'angle incident lorsque la réfraction est prise en compte et i_{c_0} l'angle entre la droite (CP') et la normale au miroir.

L'objectif est de déterminer le chemin optique minimal entre P' et C. Ici, le chemin optique total vaut :

$$\mathcal{C} = n_{air}(a + c) + n_{glass}b \tag{4.9}$$

avec $n_{air} = 1$, $n_{glass} = 1.5$ et les distances a, b et c sont illustrées en rouge sur la Figure 4.4. En optique, ce problème est connu comme un principe de moindre action : le principe de Fermat. Il

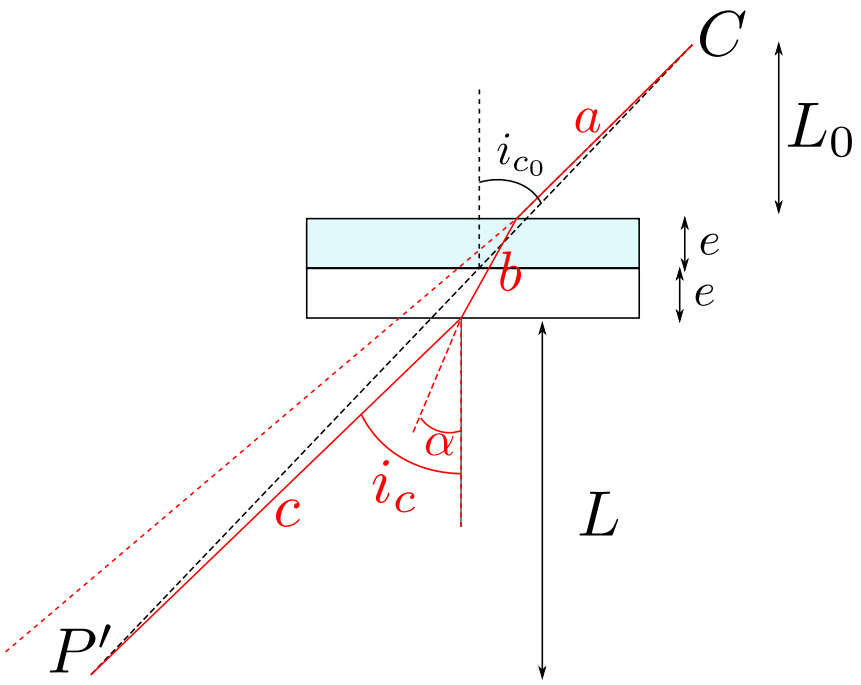


FIGURE 4.4 – Réfraction d'un miroir plan observé par une caméra centrale

peut être résolu par une technique de lancer de rayon, ou bien analytiquement. L'équation (4.10) doit être vérifiée.

$$2e\tan\left(\arcsin\left(\frac{n_{air}}{n_{glass}}\sin(i_c)\right)\right) = (L_0 + L + 2e)\tan(i_{c_0}) - (L_0 + L)\tan(i_c) \qquad (4.10)$$

Pour cela, i_{c_0} est tout d'abord calculé en négligeant la réfraction. Puis, l'algorithme de Gauss-Newton est utilisé pour déterminer i_c vérifiant l'équation (4.10). Elle est obtenue à partir de la seconde loi de Snell-Descartes (reliant les angles incident et réfracté), en remarquant que les projetés des chemins (CP') et (a, b, c) sur la surface du miroir font la même distance.

4.4.2 Influence de la réfraction sur le plan image

Si on néglige la réfraction, on tente alors d'observer le point P' sur la droite rouge en pointillés émanant du centre optique de la caméra C (*cf* Figure 4.4), alors que P' est en fait sur la ligne noire en pointillés. Cela engendre donc une erreur angulaire de $\delta i_c = i_c - i_{c_0}$. Si on observe un ensemble de points peu éloignés les uns des autres, leurs projections dans le plan image de la caméra sont biaisées (*cf* Figure 4.5 qui illustre ces erreurs de reprojection pour une mire plane (un échiquier). La largeur et longueur d'une case de l'échiquier font environ $2\ cm$). Dans cette section, nous avons choisi un miroir plan d'une épaisseur de $e = 2.80\ mm$, et une caméra munie d'un objectif $8\ mm$. La largeur d'un pixel vaut $4.4\ \mu m$.

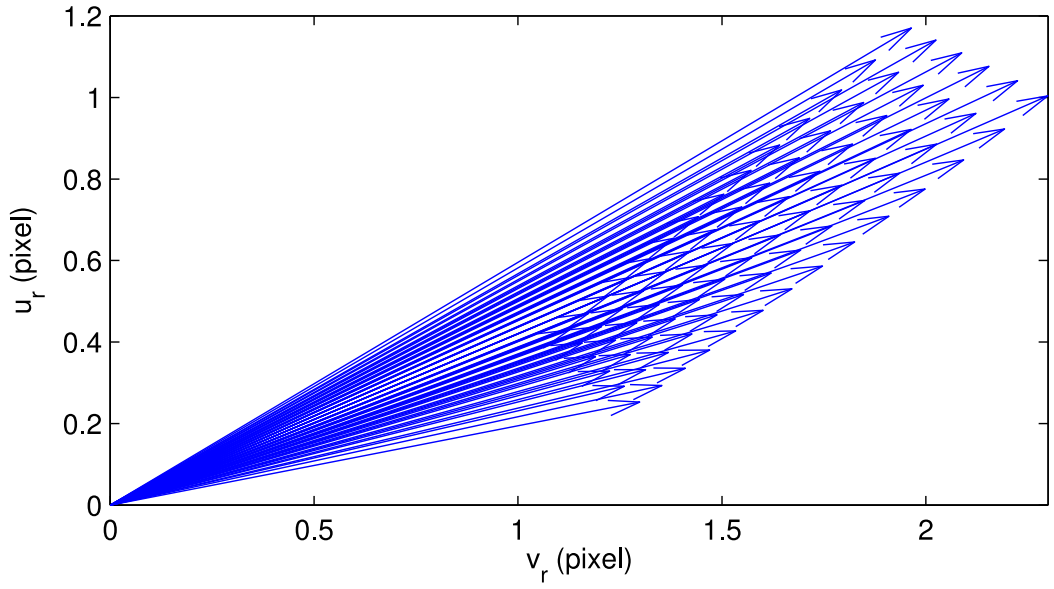


FIGURE 4.5 – Erreur de reprojection dans le plan image si la réfraction est négligée. Un échiquier (plan) est observé à une distance d'environ $1.2\ m$ avec un angle d'incidence moyen d'environ $29°$.

Nous pouvons donc quantifier l'influence de la réfraction en terme d'erreur angulaire δi_c, en fonction de l'angle d'incidence i_{c_0} et de la distance CP' entre la caméra et le point. Cependant, on interprète mieux cette influence en étudiant l'erreur engendrée dans le plan image de la caméra (qui dépend directement de l'erreur angulaire δi_c). La Figure 4.6 représente l'erreur pixellique théorique err_{pix} commise si l'on négligeait la réfraction. La procédure ayant permis d'obtenir cette figure est décrite ci-après. Pour chaque i_{c_0} et chaque CP', on calcule $L + L_0 = CP' \cos(i_{c_0}) - 2e$ et on résout l'équation (4.10) pour avoir δi_c. Enfin, l'erreur pixellique est obtenue avec : $err_{pix} = f_y sin(\delta i_c)$, en considérant que P' est porté par l'axe optique de la caméra.

On retrouve alors des résultats intuitifs : la réfraction est d'autant plus importante que l'angle incident est élevé, et que le point observé est proche. Est-il donc nécessaire de prendre en compte la réfraction ? La réponse n'est pas si simple, car elle dépend de la précision désirée, de la distance focale, de l'épaisseur du miroir, de l'angle d'incidence et de l'éloignement du point observé.

Par exemple, pour un point observé à une distance d'environ $1\ m$ avec un angle incident de $16°$, l'erreur commise dans le plan image vaut alors 1 pixel (*cf* Figure 4.6). La réfraction induit donc aisément une erreur pixellique. Or on dispose d'une détection sous-pixellique des

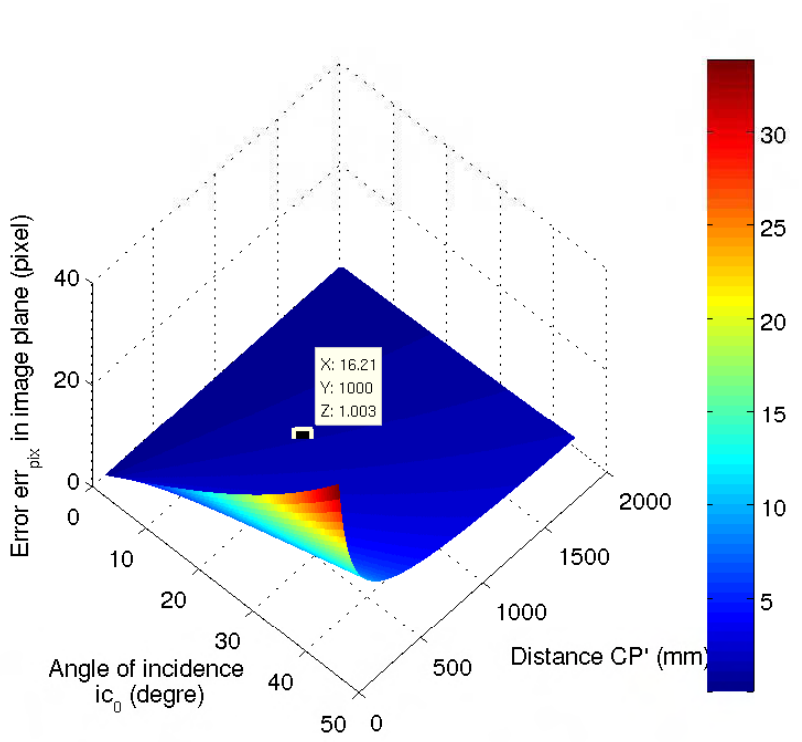


FIGURE 4.6 – Erreur err_{pix} dans le plan image si l'on négligeait la réfraction (en fonction de l'angle d'incidence i_{c_0} et de la distance CP' entre la caméra et le point).

amers : la précision est de l'ordre du dixième, voire de quelques centièmes de pixel. Dans ces conditions, la réfraction n'est donc pas négligeable.

4.4.3 Estimation de pose et réfraction

Considérons une caméra centrale et une lame réfractante. Le but est d'estimer la pose d'un objet observé à travers ce milieu réfractant. Ce calcul peut se faire en trois étapes :

- Calculer la pose de l'objet en négligeant la réfraction (ex. via l'algorithme de Dementhon [DD95]).
- Pour chaque point de l'objet, initialiser l'angle incident i_c à i_{c_0}(pour le miroir, on l'initialise à partir de sa pose : avec la normale au miroir et le rayon reliant le centre optique de la caméra et le point 3D estimé).
- Minimiser les erreurs de reprojection en prenant en compte la réfraction (calcul du chemin optique optimal, *cf* résolution de l'équation 4.10).

Concernant les algorithmes d'étalonnage, la réfraction est simplement incluse dans la fonction de projection $\underset{C^{1}v_j}{\mathrm{proj}}()$ des caméras virtuelles.

4.5 Résultats

Dans cette section, des expériences synthétiques et réelles valident les algorithmes proposés.

4.5.1 Avec des données synthétiques

Les données synthétiques fournissent une vérité terrain et permettent d'évaluer statistiquement les algorithmes. Le protocole expérimental est le suivant : tout d'abord, on génère des points P_i de la scène. On génère également la pose de chaque caméra, ainsi que des poses aléatoires du miroir à environ $150\ mm$ de la caméra C^{1r}. Cette vérité terrain sert à calculer les points image 2D p_i^{2r}, $p_i^{1v_j}$ et m_l^{1r}. Puis, on leur applique un même bruit blanc additif suivant une loi gaussienne d'écart type σ. La pose relative entre les deux caméras est ensuite estimée à partir des données bruitées. On prend 6 images du miroir mobile. Par ailleurs, on suppose que tous les points de la scène sont dans les champs de vue des caméras virtuelles. Les paramètres intrinsèques sont supposés parfaitement connus. Pour ces simulations, on cherche à estimer la transformation T_{1r}^{2r} correspondant à une translation de $(0.3m\ -0.5m\ -1m\ )^T$ et à une rotation déterminée par les angles de roulis, tangage, lacet de (15°, 180°, -20°). Pour chaque niveau de bruit, 20 mesures ont été effectuées. On contrôle la précision de l'étalonnage extrinsèque à l'aide de l'erreur commise lors de l'estimation de T_{1r}^{2r}. On l'évalue par la norme $\|dT\|$ de la différence entre la translation réelle et estimée, et par l'erreur angulaire dR définie par :

$$dR = d(\hat{R}, R^*) = \arccos\left(\frac{trace(\hat{R}^T R^*) - 1}{2}\right) \tag{4.11}$$

Où R^* représente la rotation réelle et $\hat{R}$, la rotation estimée. dR s'interprète comme l'angle positif [3] de la matrice de rotation $\hat{R}^T R^*$ permettant de passer de $\hat{R}$ à R^*. D'autre métriques pour comparer des matrices de rotations sont définies dans [DTL+10].

La Figure 4.7 indique les performances de l'algorithme 3, utilisant une mire plane de 84 points (d'environ $12\ cm$ par $30\ cm$), connue avec précision et placée à un mètre de C^{2r}. Cette méthode sert de base de comparaison pour l'algorithme 4 dans lequel on observe une scène 3D de 84 points, générés aléatoirement à environ un mètre de C^{2r}. On observe que pour un même nombre de points, l'estimation de la translation est similaire pour les deux méthodes. Alors que l'estimation de la rotation est en moyenne trois fois plus précise avec la deuxième méthode proposée. En effet les points de la mire sont restreints à une zone relativement limitée de l'espace ; or nos expériences montrent que plus les amers visuels de la scène sont dispersés, plus l'estimation des paramètres extrinsèques est précise. Cependant, le bruit de détection des amers naturels peut compenser la précision apportée par la dispersion. La précision obtenue dépend alors d'un compromis entre le nombre, la dispersion et le bruit de détection des amers.

3. Angle de la représentation axe/angle des matrices de rotations

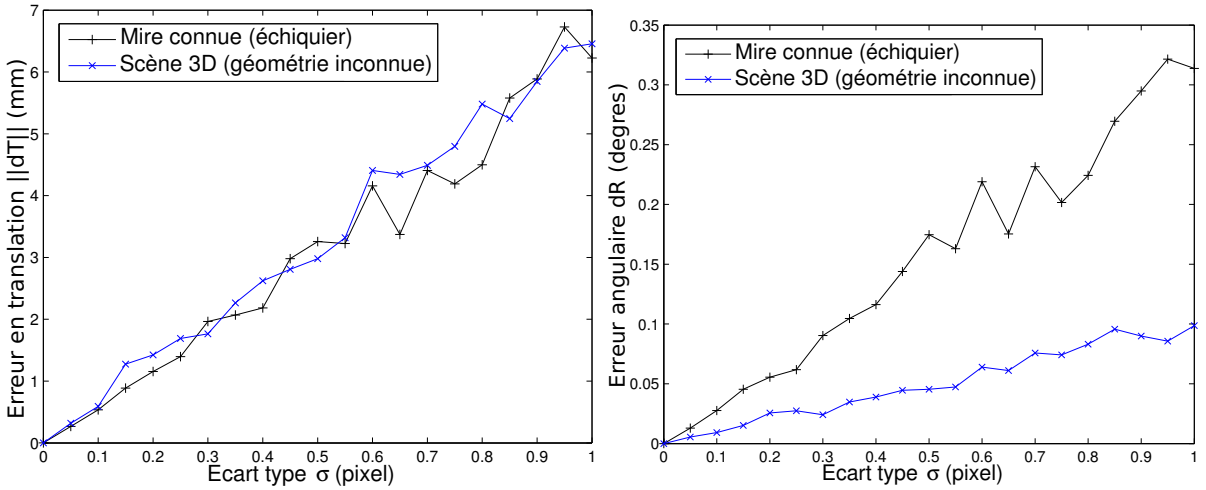


FIGURE 4.7 – Résultats avec des données synthétiques : erreur moyenne sur l'estimation de pose en fonction du niveau de bruit

4.5.2 Avec des données réelles

On utilise une caméra munie d'un objectif $8\ mm$. La largeur d'un pixel est de $4.4\ \mu m$. Lors de nos premières expériences, nous avions utilisé un miroir bas-coût avec une qualité inconnue. Ces résultats n'étaient pas satisfaisants (l'écart type des erreurs de reprojection était de $0.6\ pix$) à cause du manque de planéité du miroir et des déformations liées à sa flexion (dues à la faible épaisseur du miroir $e = 2.8\ mm$). Nous utilisons donc un miroir aluminé sur la face avant d'une épaisseur de $6mm$. La précision de sa surface est comprise entre 4λ et 6λ par pouce, λ étant la longueur d'onde.

Afin d'obtenir une vérité terrain, on place une caméra sur un banc de métrologie. Son schéma cinématique est illustré par la Figure 4.8. Un dispositif, permettant de réaliser une rotation avec une précision de l'ordre de $0.01°$, est fixé sur une plate-forme translatée. La translation s'effectue avec une précision micrométrique. On suppose que l'axe de rotation et l'axe de translation sont parfaitement orthogonaux. On note L la norme du vecteur de translation, et ψ l'angle de rotation mesuré. On fixe rigidement la caméra sur le rotor. Le centre optique est placé sur l'axe de rotation en minimisant manuellement la parallaxe.

Le banc de métrologie permet de réaliser un système multi-caméra fictif avec précision. Chaque caméra est obtenue pour un couple rotation/translation (ψ, L) donné.

Pour vérifier la précision de la rotation entre les caméras C^{1r} et C^{2r}, on utilise une mesure de l'erreur angulaire :

$$dR = |d(I_3, \hat{R}) - (\psi_1 - \psi_2)| \tag{4.12}$$

où la fonction $d()$ est définie par l'équation 4.11. Des paramètres inconnus sont introduits par le système expérimental : le défaut δ de parallaxe, et la matrice de rotation entre le rotor et le

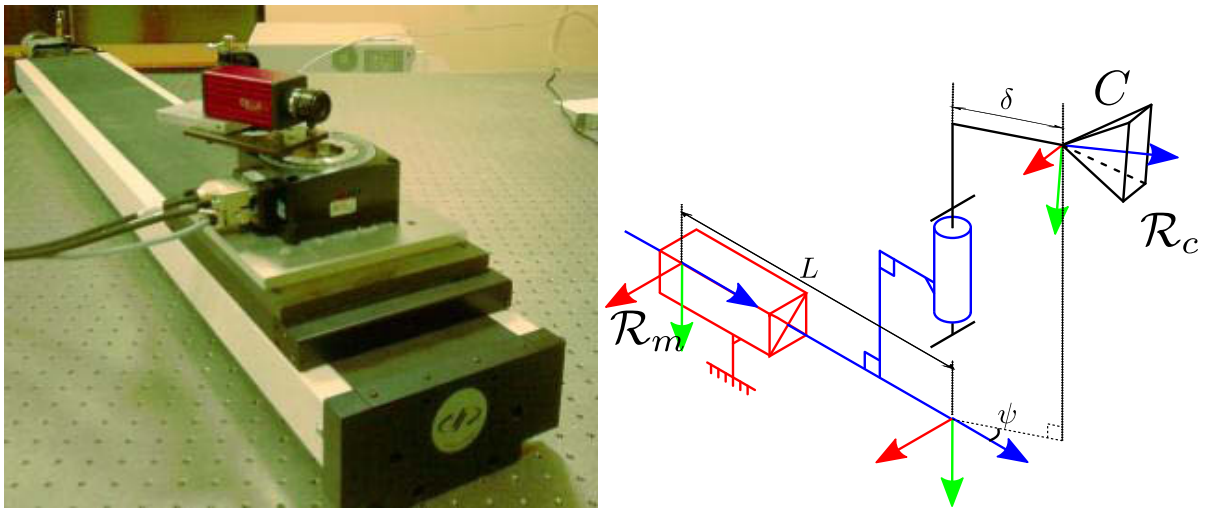


FIGURE 4.8 – Banc de métrologie et son schéma cinématique

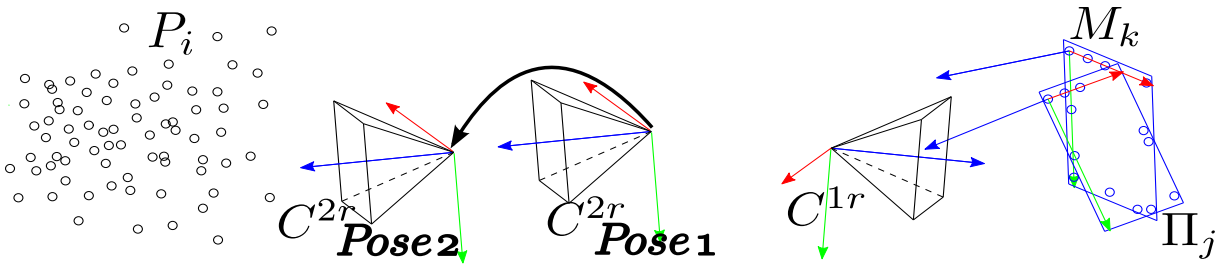


FIGURE 4.9 – Expérience avec des poses relatives - translations pures

repère caméra $\mathcal{R}_c$. On ne fait aucune hypothèse sur ces inconnues, on utilise uniquement les mesures précises fournies par le banc de métrologie. Pour ce faire, on applique la procédure d'étalonnage appliqué à un robot mobile (de la Section 4.3.3) avec des translations pures (*cf* Figure 4.9). On prend plusieurs photos du miroir que l'on déplace devant la caméra C^{1r} qui reste fixe. Une translation pure est appliquée à la caméra C^{2r}. Une image est prise tous les $5\ cm$, pour L allant de 0 à $100\ cm$. On obtient alors un ensemble de poses de la caméra C^{2r} qui sert de vérité terrain.

On estime la droite Δ passant au mieux par les centres optiques des poses alignées de C^{2r}. La qualité de l'algorithme est évaluée à partir de différents critères :

- La distance d entre cette droite et les centres optiques,
- L'erreur angulaire ϕ entre chaque pose de C^{2r} et la pose moyenne [Gra01, Moa02, Pen98],
- L'erreur angulaire dR entre C^{1r} et les poses de C^{2r} (*cf* Equation 4.11),
- L'erreur τ sur la norme du vecteur translation entre 2 poses successives.

	Moyenne	Ecart-type
d (mm)	0.04	0.03
ϕ (°)	0.009	0.005
dR (°)	0.02	0.002
τ (mm)	0.02	0.02
$\|dT\| + 2\delta$ (mm)	4.7	0.07

TABLE 4.2 – Résultats expérimentaux de la procédure d'étalonnage appliquée à un robot mobile (*cf* Section 4.3.3)

(a) (b)

FIGURE 4.10 – Expérience avec une scène inconnue en vue directe (a), et indirecte (b). La résolution des images est de 1600x1200.

Le tableau 4.2) liste les valeurs de ces critères, évalués lorsqu'une rotation de $180°$est appliquée entre C^{1r} et C^{2r} (pour $N = 21$). La Figure 4.10 illustre deux images issues de cette expérience.

Comme le montre la Table 4.2, les poses de C^{2r} reconstruites sont alignées avec précision (d est inférieur à $0.1\ mm$ et ϕ vaut environ $0.01°$). La rotation entre C^{1r} et C^{2r} est retrouvée avec une précision inférieure à $0.03°$ (dR). L'erreur en translation τ, d'environ $0.02\ mm$, se cumule entre les différentes poses successives : une translation d'un mètre engendre une erreur de $0.3\ mm$. Cette erreur est due à l'estimation du facteur d'échelle. L'erreur $\|dT\|$ commise sur la norme du vecteur translation entre C^{1r} et C^{2r} est biaisée, ce qui est principalement causé par l'erreur de parallaxe δ. Après la minimisation, l'écart-type des erreurs de reprojection est de $0.14\ pix$. Ces résultats sont satisfaisants pour des applications de robotique mobile.

Nous avons comparé les résultats précédents avec l'Algorithme 4 (*c.-à-d.* avec une seule

pose pour C^{2r}). La distance entre les deux rotations extrinsèques obtenues est inférieure à 0.02°. La norme de la différence entre les deux translations extrinsèques obtenues est de 2.6 mm. En effet, les poses supplémentaires de C^{2r} permettent d'obtenir une meilleure estimation de la profondeur de la scène, et donc une translation extrinsèque plus précise.

4.5.3 Réfraction et étalonnage

Considérons une vérité terrain créée en simulation en prenant en compte la réfraction. Le protocole expérimental est similaire à celui décrit précédemment dans la section 4.5.1. Avant de bruiter les observations, il suffit de s'assurer que les rayons vérifient l'équation 4.10. La Figure 4.11 montre l'influence de la réfraction sur l'Algorithme 3, appelé deux fois : en modélisant, ou non, la réfraction. Lors des simulations, pour T_{1r}^{2r} la translation extrinsèque vaut $(0m\ 0m\ -1m\ )^T$, et les angles (roulis, tangage, lacet) de la rotation extrinsèque sont (5°, 40°, -20°). Lorsque la réfraction est négligée pour des faibles niveaux de bruit, la Figure 4.11 montre que l'estimation de la translation et de la rotation peut être biaisée. Dans un contexte de localisation d'un robot mobile, ce biais en translation ($2\ mm$) est acceptable, mais le biais en rotation (0.07°) est plus grand que la précision obtenue avec des données réelles (*cf* Section 4.5.2).

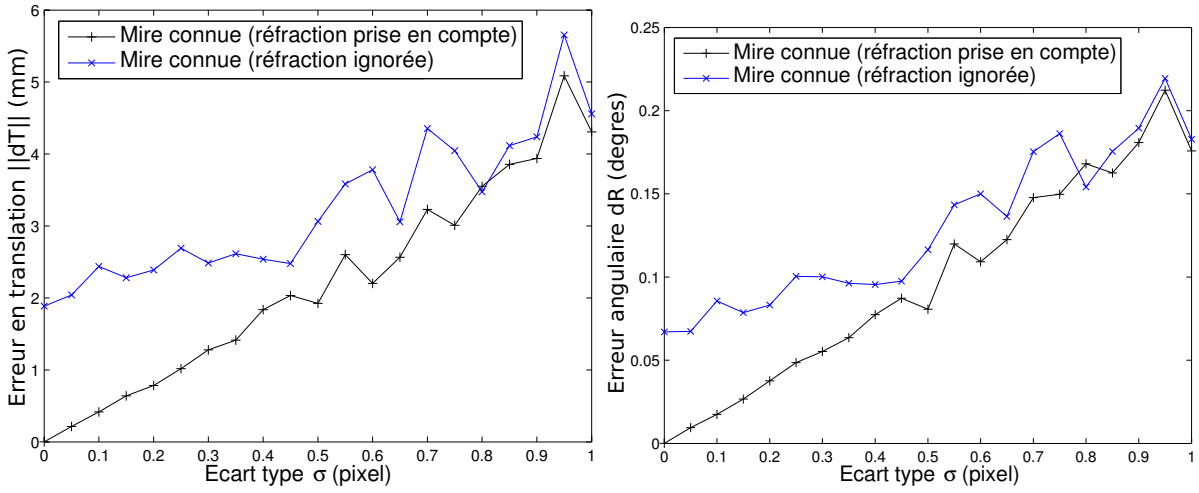


FIGURE 4.11 – Impact de la réfraction sur la précision de l'Algorithme 3 (données synthétiques).

4.6 Conclusions et perspectives

4.6.1 Conclusions

Une méthode d'étalonnage extrinsèque, d'un ensemble de caméras, statiques ou embarquées, et à champs disjoints, a été présentée, puis validée avec des données synthétiques et réelles. L'étude de la réfraction permet de savoir si elle peut être négligée, en fonction de différents critères expérimentaux : la précision désirée, la distance focale, l'épaisseur du miroir, l'angle d'incidence et la distance au point observé. De plus, les résultats ont montré que la méthode d'étalonnage sans mire connue, plus souple à mettre en œuvre, présente des performances similaires à une méthode d'étalonnage avec une mire connue, moyennant une dispersion suffisante des primitives de la scène. Enfin, l'expérience métrologique montre que la précision obtenue est suffisante pour des applications de localisation centimétrique de robot mobile.

4.6.2 Perspectives

Certains aspects de la méthode proposée sont améliorables. Par exemple, la sélection des images est faite manuellement. Pour l'automatiser, il faudrait estimer la netteté d'une image afin de ne pas sélectionner des images floues. Concernant les marqueurs collés sur le miroir et compte tenu de la propriété 1 (p17), on pourrait remplacer l'erreur de reprojection de leurs centres par une distance entre les coniques du modèle et les coniques détectées. Il serait également possible d'utiliser ces marqueurs pour faire simultanément l'étalonnage intrinsèque.

Nous avons démontré la faisabilité de la méthode d'étalonnage sans mire connue (*cf* Section 4.3.2) en reconstruisant des points d'une scène 3D. On pourrait tout aussi bien remplacer les cibles (*cf* Section 2.2.1) par des points d'intérêt. Sachant que la scène virtuelle a une vitesse apparente démultipliée par rapport à la vitesse du miroir, il faudra bien entendu gérer le flou.

Les points de la scène sont perçus plus éloignés s'ils sont vus par le miroir que s'ils sont vus directement. En pratique, quand des cibles sont utilisées, il faut qu'elles soient suffisamment grandes pour être détectées à la fois par les caméras réelles et virtuelles. Toujours sur l'aspect pratique, si le système est équipé sur un véhicule, celui-ci peut masquer une partie du champ de vue que fournit le miroir. De plus comme le miroir est déplacé manuellement, il faut veiller à ce que l'utilisateur ne lui applique pas de flexion (c'est pourquoi nous avons choisi un miroir épais pour augmenter sa rigidité, et donc sa planéité).

Afin de tendre vers un protocole d'étalonnage encore plus souple, il est intéressant d'explorer d'autres pistes qui ne nécessitent aucun miroir. C'est ce que nous proposons dans le chapitre suivant.

Chapitre 5

Etalonnage extrinsèque multi-caméra grâce aux mouvements

Dans ce chapitre, nous proposons deux approches pour étalonner extrinsèquement le système multi-caméra grâce à son mouvement. La première procédure d'étalonnage consiste à manœuvrer le véhicule pendant que chaque caméra observe une scène statique composée de cibles. Dans la deuxième approche, nous montrons que l'étalonnage extrinsèque peut être obtenu simultanément à la reconstruction 3D (par exemple lors de la phase d'apprentissage), en utilisant des points d'intérêt comme amers visuels.

Les principales contributions sont l'étude des mouvements singuliers, et un ajustement de faisceaux multi-caméra qui optimise les scènes, les poses du système multi-caméra, et étalonne extrinsèquement les caméras. Nous étudions comment traiter les mouvements singuliers, comme les mouvements plans. Les méthodes proposées sont validées avec des données synthétiques et réelles.

5.1 Introduction et état de l'art

Avant d'introduire nos travaux et nos algorithmes d'étalonnage extrinsèque, dressons tout d'abord un état de l'art sur l'étalonnage extrinsèque multi-caméra à champs de vue disjoints. Nous avons regroupé différentes approches des travaux existants en plusieurs catégories.

Scène panoramique : Un moyen pour étalonner extrinsèquement un système multi-caméra est de calculer la pose de chaque caméra par rapport à une seule scène (ou mire). Si les caméras sont à champs de vue disjoints, il faut préalablement avoir défini dans un même repère un ensemble d'amers visuels observables simultanément par toutes les caméras (*c.-à-d.* on obtient alors une sorte de mire panoramique). Cette approche est simple si on dispose déjà de la géométrie de cette scène. C'est le cas de [RW07, annexe B], où deux paires stéréo placées dos à dos sont calibrées en plaçant devant chaque paire une mire plane. Les deux mires planes sont

parallèles et une mesure de la distance les séparant permet de connaitre leurs poses relatives. Autre exemple, [CCPB09] réalise une image panoramique (à l'aide d'un système rotatif sur trépied) puis localise chaque caméra à l'aide de cette image (seule les rotations extrinsèques sont estimées, car les centres optiques des caméras sont supposés confondus).

[Pag] utilise également cette approche via des motifs comparables aux cibles (présentées en Section 2.2.1). Ce dernier a d'ailleurs été publié en même temps que notre article [3]. Il parsème ces motifs devant chaque caméra (et conseille de les placer autour du système multi-caméra, pour que deux motifs consécutifs soient visibles simultanément à un instant donné). La première étape consiste à trouver la pose relative des différents motifs (en observant successivement les motifs deux à deux, avec une même caméra). Dans la seconde étape, les caméras observent des motifs différents (devant leurs champs de vue respectifs) pour déterminer les paramètres extrinsèques. A chaque étape, des erreurs de reprojection sont minimisées pour optimiser soit les poses relatives des motifs, soit les poses relatives des caméras. En résumé, l'auteur construit donc incrémentalement une scène panoramique, sans remettre en cause cette reconstruction lors de l'étalonnage extrinsèque (contrairement à notre approche, comme on le verra plus loin).

[ISY03] utilise une approche "hybride" pour étalonner intrinsèquement et extrinsèquement les caméras à champs partiellement recouvrants de la Ladybug®2 (*cf* Figure 3.1c) restant statique : une mire est déplacée devant chaque caméra et à chaque pose de la mire, les coordonnées 3D de ses points sont déterminées par un tachéomètre. Les points 3D étant exprimés dans un même repère, il suffit de remonter aux poses de chaque caméra pour déterminer les paramètres extrinsèques. En fait pour ce capteur, les champs de vue sont partiellement recouvrants. On peut donc étalonner extrinsèquement avec des correspondances directes (d'une même scène, entre des paires de caméras) comme évoqué dans [KCL05] pour un capteur similaire doté de 10 caméras.

Observations successives d'amers dans des caméras différentes : [LRFK07] étalonne un système multi-caméra en mouvement. Les champs des caméras sont disjoints et un objet doit être suivi successivement dans chaque caméra. Une connaissance *a priori* sur la vitesse du système est nécessaire. [BHS11a] [BHS11b] disposent deux caméras de la même manière que les VIPA (l'une à l'avant et l'autre à l'arrière) sur une voiture pour détecter les lignes horizontales délimitant les voies sur la chaussée. Le but est de prévenir le conducteur d'un franchissement de ligne. Pour ce faire, les auteurs appliquent une rectification perspective pour fusionner les images avant et arrière et obtenir une vue de dessus virtuelle. Cette méthode est sensible à la hauteur des caméras et aux paramètres extrinsèques (les caméras sont supposées parfaitement opposées à 180°). Les auteurs estiment donc la hauteur de chaque caméra à l'aide d'un *a priori* sur la distance séparant deux lignes blanches, puis raffinent la translation extrinsèque : l'écart latéral entre les caméras est optimisé en observant en même temps une ligne dans les deux caméras ; la distance entre les deux caméras est estimée en observant successivement un même motif (marquage au sol au milieu de la voie) dans la caméra avant puis arrière (en supposant que le véhicule avance en ligne droite à vitesse constante).

Du mouvement des caméras vers leurs poses relatives : Une autre approche [EWK07a] consiste à utiliser la contrainte de rigidité entre les différents capteurs du système multi-caméra en mouvement. Cette approche a été initialement proposée par [LF93] pour le cas stéréo. L'évolution temporelle de la pose de chaque caméra est déterminée. Puis, on déduit la pose relative entre les différentes caméras (paramètres extrinsèques), qui ne varie pas au cours du temps. [KHK08] l'évoque également pour plusieurs caméras ayant un centre optique commun (seules les rotations extrinsèques sont estimées). [NHA10] se focalise sur le nombre minimal de mouvements nécessaires pour étalonner extrinsèquement les caméras : les rotations extrinsèques sont estimées à partir de deux rotations pures, puis les translations extrinsèques sont estimées avec deux mouvements 3D. Cet article à également été publié en même temps que le nôtre [3], l'auteur tire notamment des conclusions similaires à nos travaux (comme le fait que les mouvements plans forment un cas singulier). Il se base sur une résolution de contraintes multi-linéaires (comme proposé dans [Ste05] et l'article [Ste03], dont la formulation est proche du tenseur trifocal [Hey00]), écrites à partir de points suivis dans 3 vues successives. Le papier prétend qu'il est suffisant de suivre les points, sans remonter jusqu'à leurs coordonnées 3D. En fait, il semble qu'un algorithme SfM soit tout de même nécessaire. Des travaux supplémentaires ont été proposés par [DTL$^+$10] pour moyenner des matrices de rotation, afin d'étalonner extrinsèquement des caméras. [FKK04] évoque de manière annexe une méthode en théorie similaire, mais sans rentrer dans les détails.

Nous verrons que l'étalonnage extrinsèque multi-caméra est en partie similaire à l'étalonnage bras-œil, car on retrouve des équations identiques (*cf* Section 5.2.2.1). Une caméra et un bras articulé, sur lequel elle est fixée, fournissent respectivement des mesures proprioceptives et extéroceptives de leur mouvement. Mais en généralisant, on peut trouver d'autres étalonnages faisant apparaitre ces équations : il s'agit d'étalonnage *capteur vers capteur*, où chaque capteur dispose d'une estimation locale de son mouvement. Par exemple, on peut ranger dans cette catégorie l'étalonnage extrinsèque entre une caméra et une centrale inertielle [LD07].

Concernant les publications les plus récentes, une méthode similaire à notre approche a été proposée par [CAD11b] (simultanément à notre article [2]). Plusieurs caméras sont embarquées sur un robot qui effectue une série de mouvements pré-enregistrés (dont un tour sur lui-même qui est obligatoire, contrairement à notre méthode d'initialisation). Pour chaque caméra, un EKF (filtre de Kalman étendu) puis un ajustement de faisceaux sont appliqués. Lors de son tour sur lui-même, chaque caméra a pu observer des points vus par une autre caméra. L'auteur les utilise (via des appariements SURF) pour recaler les reconstructions 3D monoculaires. Cela sert d'initialisation à un ajustement de faisceaux multi-caméra (optimisant les points 3D, les poses et les paramètres extrinsèques) peu détaillé mais similaire à celui que nous proposons dans la Section 5.4. [PW] propose un outil de type EKF pour étalonner extrinsèquement les caméras. Les paramètres extrinsèques initiaux (approximatifs) ont d'abord une grosse incertitude qui tend à diminuer au fur et à mesure que les données sont filtrées. Chaque caméra estime localement son mouvement, qui est ensuite propagé aux autres caméras, pour finalement fusionner les estimations. Cet article présente uniquement les aspects théoriques de l'algorithme sans fournir d'expérimentation prouvant sa faisabilité. [LZWS11] étalonne des caméras statiques à champs

disjoints en déplaçant une mire connue devant chaque caméra. Ces mires sont rigidement liées par une barre pouvant être longue de quelques mètres (une variante est proposée dans [MH03] avec deux sphères reliées par une barre). Il s'agit d'une méthode duale à celles proposées dans [EWK07a] et dans la Section 5.2, où les objets statiques deviennent mobiles et inversement.

Systèmes articulés : Dans un contexte assez proche, il existe des approches où des caméras sont embarquées sur un système sans être rigidement liées. Par exemple, [KTS11] effectue un ajustement de faisceaux pour un système stéréo à champs recouvrant dont la baseline reste constante, mais l'orientation de chaque caméra est variable. Ce type de système d'acquisition est notamment utilisé au cinéma. L'auteur optimise également en même temps la distance focale commune aux deux caméras. Notons également les travaux [SBSV10], où les yeux d'un robot iCub sont partiellement étalonnés (en déterminant certains angles de rotation). Ce sont des caméras à champs recouvrants qui pivotent sur elles-mêmes selon 2 axes de rotation.

D'autre part, [CW] étalonne des systèmes stéréo articulés : les caméras sont fixées rigidement sur leurs supports respectifs qui sont reliés par une liaison pivot. Le but est d'estimer où est située la liaison pivot par rapport à chaque caméra. Pour cela, une fois que les poses de chaque caméra sont déterminées, des outils de résolution de système linéaire puis non-linéaire sont appliqués sur les équations matricielles. Toutefois, aucun ajustement de faisceaux, ni de minimisation des erreurs de reprojection ne sont appliqués. Ce qui aurait *a priori* amélioré la précision de l'étalonnage (comme nous le verrons pour le cas de caméras rigidement liées dans la Section 5.2).

Nos travaux : Dans ce chapitre, nous proposons une méthode flexible pour obtenir un étalonnage extrinsèque précis d'un ensemble mobile de caméras rigidement liées. La flexibilité est liée à la simplicité de la mise en œuvre : pas besoin de miroir, ni d'*a priori* sur la vitesse du système multi-caméra, ni de parfaitement connaitre au préalable la géométrie d'une mire.

Les travaux de ce chapitre appartiennent à la catégorie "Du mouvement des caméras vers leurs poses relatives". Ils peuvent être vus comme :

- une amélioration de l'algorithme proposé par Esquivel *et al* [EWK07a].
- une extension de l'algorithme classique d'ajustement de faisceaux, décrit par Triggs *et al* [TMHF00], au cas de plusieurs caméras rigidement liées (avec des champs éventuellement disjoints) dans l'optique d'un étalonnage extrinsèque (*c.-à-d.* en optimisant la pose relative des caméras).

On remarque que [DTL$^+$10] et [EWK07a] considèrent tous deux que les poses de chaque caméra (estimées indépendamment) sont suffisamment précises pour obtenir des paramètres extrinsèques globalement cohérents. Nous ne faisons pas cette hypothèse. Afin d'augmenter la précision, des poses calculées dans l'étape d'initialisation sont également raffinées simultanément dans la dernière étape.

Dans ce chapitre, nous proposons deux approches pour étalonner extrinsèquement le système multi-caméra en mouvement. La première est présentée dans la Section 5.2. Elle consiste à manœuvrer le véhicule pendant que chaque caméra observe une scène statique composée de cibles. Une initialisation linéaire estime les paramètres extrinsèques dans le cas de mouvements quelconques ou singuliers (Section 5.2.2). Dans la deuxième approche (Section 5.3), nous montrons que l'étalonnage extrinsèque peut être obtenu simultanément à la reconstruction 3D (par exemple lors de la phase d'apprentissage), en utilisant des points d'intérêt. Ces deux approches utilisent un outil commun : l'ajustement de faisceaux multi-caméra (MCBA, détaillé dans la Section 5.4), qui optimise les paramètres extrinsèques, la scène et les poses du système multi-caméra. Nous verrons en détail comment exploiter la structure creuse des matrices du MCBA. La Section 5.5 illustre les résultats obtenus pour les deux approches (avec des données synthétiques et réelles), mais aussi des reconstructions faites après avoir étalonné les caméras.

5.2 Manœuvres et cibles

5.2.1 Formalisation du problème

Soient $N \geq 2$ caméras rigidement liées, dont les paramètres intrinsèques sont connus, avec des champs de vue non-recouvrants. Les caméras C_j sont synchronisées et effectuent K mouvements. Ainsi, $K+1$ est le nombre de poses du système multi-caméra. C_j^k représente la caméra n°j à l'instant k. Chaque pose de la caméra C_j^k est exprimée relativement à sa première pose à l'instant $k = 0$. Soit T_j^k la transformation homogène passant du repère de C_j de l'instant 0 à l'instant k, quelque soit $k \in [\![1..K]\!]$ et $j \in [\![1..N]\!]$ (*cf* Figure 5.1). De même, ΔT_j est la transformation homogène passant du repère de C_1 à celui de C_j (il s'agit de la pose relative entre ces deux caméras, également appelée *paramètres extrinsèques*). Lorsque le système multi-caméra se déplace, chaque caméra C_j observe une scène statique S_j de points 3D. p_j^k sont les points détectés dans le plan image de C_j^k, correspondants aux points de la scène S_j.

Chaque transformation homogène T est exprimée à l'aide de la rotation R et de la translation $\mathbf{t}$, comme expliqué en Annexe A.2.1. Plus précisément, on exprime T_j^k et ΔT_j telles que :

$$T_j^k = \begin{pmatrix} R_j^k & \mathbf{t}_j^k \\ 0_{1\times 3} & 1 \end{pmatrix} \tag{5.1}$$

R_j^k désigne la rotation entre les repères de C_j^0 et C_j^k. Le vecteur $\mathbf{t}_j^k$ désigne la translation appliquée à la caméra C_j (exprimée dans le repère de C_j^0) entre les instant 0 et k.

$$\Delta T_j = \begin{pmatrix} \Delta R_j & \Delta \mathbf{t}_j \\ 0_{1\times 3} & 1 \end{pmatrix} \tag{5.2}$$

ΔR_j désigne la rotation extrinsèque entre les repères de C_1 et de C_j. Le vecteur $\Delta \mathbf{t}_j$ désigne la translation extrinsèque (exprimée dans le repère de C_1) entre la caméra C_1 et C_j.

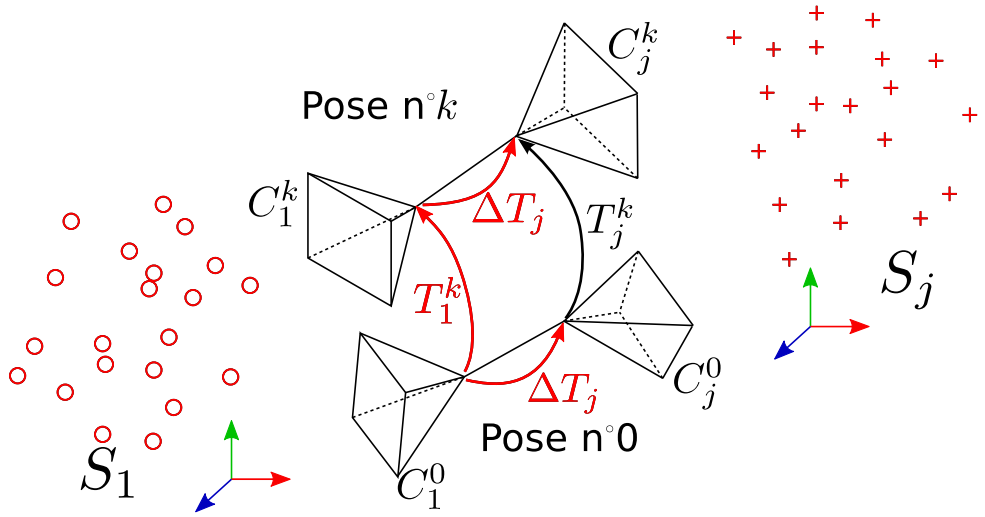


FIGURE 5.1 – Système multi-caméra à champs non-recouvrants se déplaçant dans un environnement statique.

La Figure 5.2 illustre une vue d'ensemble de l'algorithme d'étalonnage proposé. Il consiste à tout d'abord calculer les poses T_j^k de chaque caméra (*cf* ci-dessous : estimation des trajectoires). Ensuite, les paramètres extrinsèques ΔT_j sont initialisés linéairement (Partie 5.2.2), puis raffinés en même temps que les scènes et les poses du système multi-caméra grâce à l'ajustement multi-caméra (Section 5.4).

Estimation des trajectoires Premièrement, les trajectoires de chaque caméra sont calculées indépendamment. On appelle *trajectoire* d'une caméra, l'ensemble de ses $K+1$ poses. Pour cela, si la scène est de géométrie inconnue, un algorithme de structure from motion est utilisé (comme décrit dans [HZ04]). *A contrario*, si la géométrie de la scène est approximativement connue, alors les poses de chaque caméra sont initialisées par la méthode décrite par Dementhon [DD95]. Ensuite, pour chaque caméra C_j, un ajustement de faisceaux classique [TMHF00] optimise à la fois les poses des caméras T_j^k et la scène S_j. Par la suite, nous supposons qu'au moins une mesure de distance entre 2 points est disponible pour chaque scène S_j. Pour chaque caméra, ses poses et la scène estimée qu'elle observe sont ramenées à l'échelle réelle. Pour cela, on calcule la médiane des rapports entre chaque distance mesurée et estimée. Ainsi, l'ensemble du système a le même facteur d'échelle. Les facteurs d'échelle relatifs δ_j entre les scènes S_1 et S_j sont donc fixés à 1. Si ces mesures ne sont pas disponibles, les δ_j peuvent être ajoutés aux paramètres à retrouver lors de l'initialisation.

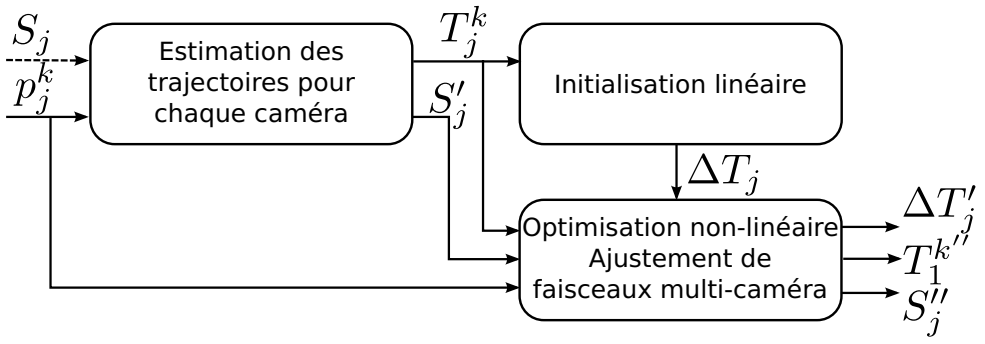


FIGURE 5.2 – Vue d'ensemble de l'étalonnage extrinsèque. A partir des observations des scènes (et d'une éventuelle information sur leurs géométries), on estime les poses de chaque caméra. Ensuite, ces poses permettent une estimation linéaire des paramètres extrinsèques (poses relatives des caméras). Enfin, les paramètres extrinsèques, les poses du système multi-caméra et la géométrie des scènes sont raffinées non-linéairement. Le symbole apostrophe distingue les estimations successives.

5.2.2 Initialisation linéaire et cas singuliers

5.2.2.1 Mouvements 3D : cas général

Il est possible d'estimer linéairement les poses relatives ΔT_j des caméras (paramètres extrinsèques) à partir des trajectoires T_1^k et T_j^k. L'hypothèse de rigidité entre les caméras et un changement de base permettent d'obtenir :

$$\forall j \in [\![2..N]\!], \forall k \in [\![1..K]\!],\ T_1^k \Delta T_j = \Delta T_j T_j^k \tag{5.3}$$

Ces équations sont de la forme $AX = XB$, où X est la matrice inconnue [1]. Ce sont les mêmes équations que l'on retrouve pour les problèmes d'étalonnage bras-œil [TL89, Dho09], qui sont résolues en représentant les rotations soit par des quaternions unitaires [EWK07a, LF93], soit par des matrices orthogonales 3×3 [AHE01].

Rappelons tout d'abord la propriété 5.4, où A, B et C sont des matrices, $\otimes$ représente le produit de Kronecker (*cf* Annexe A.3.2) et vec renvoie un vecteur colonne en concaténant la transposée des lignes d'une matrice. Le lecteur peut se référer à [SS97, §1.8, §2.11] pour plus de détails.

$$vec(ABC) = (A \otimes C^\top) vec(B) \tag{5.4}$$

A l'aide des équations (5.1), (5.2) et de la propriété (5.4), l'équation (5.3) se scinde en deux parties :

1. Il s'agit d'un cas particulier de l'équation de Sylvester (également appelée l'équation de Lyapunov) du type $AX + XB = C$, où A, B, C et X sont des matrices complexes de taille $n \times n$.

$\forall j \in [\![1..N]\!], \forall k \in [\![1..K]\!]$,

$$(5.3) \Leftrightarrow \begin{cases} R_1^k \Delta R_j = \Delta R_j R_j^k & (5.5a) \\ R_1^k \Delta \mathbf{t}_j + \mathbf{t}_1^k = \Delta R_j \mathbf{t}_j^k + \Delta \mathbf{t}_j & (5.5b) \end{cases}$$

$$\Leftrightarrow \begin{cases} (I_9 - R_1^k \otimes R_j^k) vec(\Delta R_j) = 0_{9\times 1} & (5.6a) \\ (I_3 - R_1^k) \Delta \mathbf{t}_j = \mathbf{t}_1^k - \Delta R_j \mathbf{t}_j^k & (5.6b) \end{cases}$$

Où I_n est la matrice identité de taille n. L'équation 5.6a est obtenue en remarquant que $vec(\Delta R_j) = vec(R_1^k \Delta R_j (R_j^k)^\top) = (R_1^k \otimes R_j^k) vec(\Delta R_j)$.

Comme suggéré dans [AHE01], nous avons choisi la représentation matricielle des rotations (ce qui permet d'exprimer de nouvelles équations dans la partie 5.2.2.2) et une solution en deux étapes : on estime premièrement les rotations ΔR_j, puis les translations $\Delta \mathbf{t}_j$. Dans un premier temps, pour chaque caméra C_j, il suffit de concaténer les équations (5.6a) obtenues pour différents mouvements. La rotation ΔR_j est alors estimée à l'aide de l'équation (5.7) :
$\forall j \in [\![2..N]\!]$,

$$(5.6a) \Rightarrow \underbrace{\begin{pmatrix} I_9 - R_1^1 \otimes R_j^1 \\ \vdots \\ I_9 - R_1^k \otimes R_j^k \\ \vdots \\ I_9 - R_1^K \otimes R_j^K \end{pmatrix}}_{=L_j} vec(\Delta R_j) = 0_{9K\times 1} \quad (5.7)$$

En dehors des cas singuliers (*c.-à-d.* pour des mouvements 3D quelconques), les $N-1$ équations (5.7) admettent chacune une solution unique. Afin de le vérifier (en étudiant le rang des matrices L_j), intéressons-nous à la propriété suivante.

Propriété 4. *Les matrices $I_9 - R_1^k \otimes R_j^k$ sont au plus de rang 6.*

Démonstration : D'après la propriété 10 (p160) sur les matrices de rotations, et comme R_1^k et R_j^k sont semblables (car (5.5a) $\Rightarrow R_1^k = \Delta R_j R_j^k {\Delta R_j}^{-1}$), elles ont les mêmes valeurs propres : 1, $e^{\mathtt{i}\theta_k}$ et $e^{-\mathtt{i}\theta_k}$. Or, d'après la propriété 13 (p161) sur les valeurs et vecteurs propres d'un produit de Kronecker, les valeurs propres de $R_1^k \otimes R_j^k$ sont :

- 1, dont l'ordre de multiplicité est $m_1 = 3$,
- $e^{\mathtt{i}\theta_k}$, dont l'ordre de multiplicité est 2,
- $e^{-\mathtt{i}\theta_k}$, dont l'ordre de multiplicité est 2,
- $e^{2\mathtt{i}\theta_k}$, dont l'ordre de multiplicité est 1,
- $e^{-2\mathtt{i}\theta_k}$, dont l'ordre de multiplicité est 1.

Si $\theta_k \equiv \pi[2\pi]$ alors l'ordre de multiplicité de la valeur propre 1 vaut $m_1 = 5$. Si $\theta_k \equiv 0[2\pi]$ alors l'ordre de multiplicité de la valeur propre 1 vaut $m_1 = 9$. Plaçons-nous dans le cas général

où $\theta_k \not\equiv \pi[\pi]$. Les 3 vecteurs propres unitaires et deux-à-deux distincts [2] de $R_1^k \otimes R_j^k$ associés à la valeur propre 1 sont $\mathbf{n}_1^k \otimes \mathbf{n}_j^k$, $\mathbf{u}_1^k \otimes \mathbf{v}_j^k$ et $\mathbf{v}_1^k \otimes \mathbf{u}_j^k = (\mathbf{u}_1^k)^* \otimes (\mathbf{v}_j^k)^*$.

Notons $\{\wedge_l\}_{[\![1..3]\!]}$ ces 3 vecteurs, on a :

$$\forall l \in [\![1..3]\!],\ (R_1^k \otimes R_j^k)\wedge_l = 1 \wedge_l \tag{5.8}$$

$$\Leftrightarrow \forall l \in [\![1..3]\!],\ (I_9 - R_1^k \otimes R_j^k)\wedge_l = 0_{9\times 1} \tag{5.9}$$

$$\Rightarrow \dim(\mathrm{Ker}(I_9 - R_1^k \otimes R_j^k)) \geq 3 \tag{5.10}$$

Dans le meilleur des cas, $I_9 - R_1^k \otimes R_j^k$ est donc de rang 6. □

Dans le cas général (mouvements 3D du système multi-caméra), le rang de L_j vaut 8 (avec au minimum deux mouvements avec des rotations indépendantes, *cf* [AHE01]). La résolution de l'équation 5.7 se fait alors comme suit. Soit v_j un vecteur du noyau de L_j et soit V_j la matrice 3×3 telle que $v_j = vec(V_j)$, on a alors :

$$\Delta R_j = V_j \, \frac{sign(det(V_j))}{\sqrt[3]{|det(V_j)|}} \tag{5.11}$$

Démonstration : Soit v_j et V_j tel que définis ci-dessus. Il existe $\varrho \in \mathbb{R}$ tel que $\Delta R_j = \varrho V_j$. Or ΔR_j est une matrice orthogonale de taille 3×3. En prenant le déterminant, on obtient : $det(\Delta R_j) = det(\varrho V_j) \Leftrightarrow 1 = \varrho^3 det(V_j)$. □

Dans un deuxième temps, chacune des $N-1$ translations extrinsèques $\Delta\mathbf{t}_j$ est estimée grâce à l'équation de rang plein (5.12), obtenue pour chaque mouvement k grâce aux équations (5.6b) : $\forall j \in [\![2..N]\!]$,

$$\begin{pmatrix} I_3 - R_1^1 \\ \vdots \\ I_3 - R_1^k \\ \vdots \\ I_3 - R_1^K \end{pmatrix} \Delta\mathbf{t}_j = \begin{pmatrix} \mathbf{t}_1^1 - \Delta R_j \mathbf{t}_j^1 \\ \vdots \\ \mathbf{t}_1^k - \Delta R_j \mathbf{t}_j^k \\ \vdots \\ \mathbf{t}_1^K - \Delta R_j \mathbf{t}_j^K \end{pmatrix} \tag{5.12}$$

En présence de bruit (*cf* remarque p79), l'erreur d'initialisation de ΔR_j se répercute donc sur l'initialisation de $\Delta\mathbf{t}_j$. D'autre part, si les facteurs d'échelle relatifs δ_j (entre les scènes S_1 et S_j) ne sont pas connus, il est possible de les estimer linéairement en même temps que les translations extrinsèques. C'est ce qui est proposé dans [EWK07a] pour les mouvements non-singuliers (mais aussi dans [AHE01]).

2. On peut vérifier que les 3 vecteurs unitaires sont bel et bien 2 à 2 distincts : d'une part le premier est réel, tandis que les deux autres sont complémentaires avec des parties imaginaires non nulles. D'autre part, les 3 vecteurs sont orthogonaux.

5.2.2.2 Mouvements singuliers

Cette partie met en exergue les séquences de mouvements critiques pour lesquels les équations (5.7) et (5.12) deviennent singulières et ne peuvent plus être utilisées telles quelles. Premièrement, les translations pures sont des mouvements singuliers et ont déjà été étudiées par [EWK07a]. Cependant, il existe d'autres singularités. Comme ce problème d'étalonnage extrinsèque peut être formulé avec les mêmes équations que l'étalonnage bras-œil, nous pouvons déduire les cas singuliers à partir de travaux antérieurs [AHE01, FL05]. Ce dernier fournit notamment une interprétation géométrique des cas singuliers.

Par conséquent, les cas singuliers ont lieu lorsque les axes $\mathbf{n}_1^k$ des rotations R_1^k sont parallèles pour tout $k \in [\![1..K]\!]$. Autrement dit, les mouvements singuliers sont l'orbite de la première pose (pour $k = 0$) sous l'action du groupe G, où G est composé des vissages d'axes colinéaires.

Pour information, d'autres travaux analysent les mouvements singuliers pour des sujets similaires, comme :

- la reconstruction 3D et l'estimation du mouvement à partir d'une paire stéréo calibrée sans correspondance stéréo [KC06, KJCTC10]. Dans ces articles, la reconstruction 3D est basée sur un EKF dont la baseline entre les caméras fait partie du vecteur d'état. Les paramètres extrinsèques sont déterminés au préalable, et seule l'échelle entre les 2 caméras est également optimisée.
- le calcul de pose d'un système multi-caméra à champs disjoints [CKF$^+$08, Cli10], calibré extrinsèquement. Cette approche utilise un nombre minimal de points mis en correspondance entre deux poses consécutives (5 points vus par une caméra et 1 point vu par une autre caméra), deux RANSAC successifs permettent d'estimer la pose. La condition que les mouvements doivent remplir est formulée afin que l'échelle soit calculable. Sinon le mouvement est singulier pour le calcul de pose : c'est le cas d'une translation pure ou bien si les caméras se déplacent le long de cercles concentriques.
- l'auto-étalonnage et la reconstruction euclidienne en utilisant la notion de conique absolue, avec des caméras calibrées (ou à focale inconnue) [KT99], ou non-calibrées [Stu97]. Pour ce deuxième cas, [PVG00] représente la notion abstraite de conique absolue à l'aide de formes géométriques réelles, pour interpréter géométriquement les séquences de mouvements singuliers.
- l'auto-étalonnage d'une caméra ou d'une tête stéréo avec une focale variable [Stu99].

Les mouvements singuliers sont soit des rotations et des vissages autour d'un même axe (lorsque les axes $\mathbf{n}_1^k$ sont confondus), soit des translations pures (selon un même axe ou des axes différents), soit des vissages avec des axes parallèles (lorsque les axes $\mathbf{n}_1^k$ sont différents). Pour tous ces mouvements singuliers, l'étalonnage est partiel : seulement certains paramètres peuvent être estimés.

La Table 5.1 énumère le nombre de degrés de liberté observables pour les paramètres extrinsèques. Pour le cas n°1), illustré par la Figure 5.3, on estime les paramètres extrinsèques $\widehat{\Delta T_j}$ à un vissage d'axe $\mathbf{a}$ près (*cf* équation 5.13). On note que si l'angle et l'axe de vissage sont

Mouvements	Axes de rotation	ΔR_j	$\Delta \mathbf{t}_j$
1) Rotations et vissages [3] selon un axe $\mathbf{a}$	confondus ($\mathbf{a} = \mathbf{n}_1^k$)	2	0
2) Translations pures selon un axe $\mathbf{a}$	pas de rotation	2	0
3) Translations pures selon plusieurs axes	pas de rotation	3	0
4) Mouvements plans et vissages selon différents axes	différents et parallèles	3	2
5) 3D (cas général)	différents	3	3

TABLE 5.1 – Nombre de degrés de liberté observables des paramètres extrinsèques avec K mouvements du système multi-caméra. L'échelle est supposée connue.

déterminés, alors deux degrés de liberté de la translation extrinsèques sont observables [3].

$$\text{Cas n°1) : } \widehat{\Delta T_j} = \begin{pmatrix} R(\mathbf{a}) & s\mathbf{a} \\ 0_{1\times 3} & 1 \end{pmatrix} \Delta T_j \text{ , avec } s \in \mathbb{R} \tag{5.13}$$

$$\Leftrightarrow \widehat{\Delta R_j} = R(\mathbf{a})\Delta R_j \text{ et } \widehat{\Delta \mathbf{t}_j} = R(\mathbf{a})\Delta \mathbf{t}_j + s\mathbf{a} \tag{5.14}$$

Notons que pour le cas 1), si l'axe $\mathbf{a}$ passe par les centres optiques de C_1 et C_j, alors la translation extrinsèque $\Delta \mathbf{t}_j$ est observable à un facteur d'échelle près (car il existe $\nu \in \mathbb{R}$ tel que $\Delta \mathbf{t}_j = \nu\mathbf{a}$ et donc $\widehat{\Delta \mathbf{t}_j} = (\nu + s)\mathbf{a}$). Pour les mouvements 1) et 2), si on applique un vissage selon $\mathbf{a}$ à S_j et aux paramètres extrinsèques, alors on ne change pas les observations (comme le montre la Figure 5.3b, dont les éléments rouges ont subi un vissage d'axe $\mathbf{a}$).

Les mouvements plans (cas n°4) peuvent être vus comme plusieurs vissages dont les axes $\mathbf{n}_1^k$ sont parallèles, et dont les translations selon ces axes sont nulles. Pour un système multi-caméra suivant des mouvements plans, chaque caméra est associée à un plan dans lequel évolue son centre optique.

Propriété 5. Translation et mouvements plans : *Comme illustré dans la Table 5.1 et la Figure 5.4, l'une des dimensions de la translation extrinsèque $\Delta \mathbf{t}_j$ n'est pas observable selon la normale au plan $\mathbf{n}$ (qui est parallèle aux axes de vissage). On peut seulement affirmer que $\Delta \mathbf{t}_j$ est de la forme :*

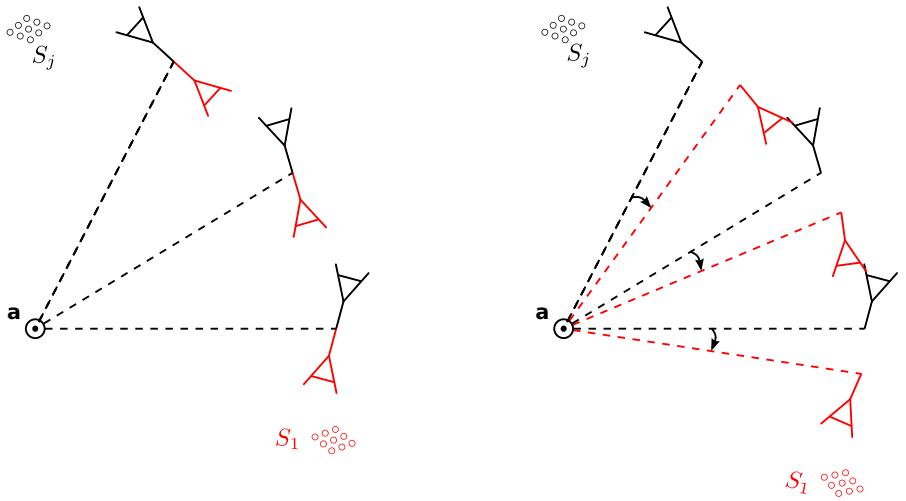
$$\Delta \mathbf{t}_j = \Delta \mathbf{t}_j^{\perp} + (h + \alpha_j)\mathbf{n}_1 \tag{5.15}$$

où $\Delta \mathbf{t}_j^{\perp}$ est la projection de $\Delta \mathbf{t}_j$ dans le plan du mouvement, h est un réel quelconque, α_j est la hauteur relative entre C_1 et C_j et $\mathbf{n}_1$ est la normale au plan exprimée dans le repère de la caméra maître.

Démonstration : D'après la propriété 9 (p160), on a :

$$\forall k \in [\![1..K]\!],\ R_1^k \mathbf{n}_1^k = \mathbf{n}_1^k \tag{5.16}$$

3. Pour le cas singulier n°1, les degrés de liberté de la translation sont couplés à ceux de la rotation.



(a) (b)

FIGURE 5.3 – Mouvements de vissage (rotation et translation) selon l'axe **a** (cas singulier n°1) appliqués au système multi-caméra (vue de dessus). Les centres optiques des caméras sont portés par des cylindres concentriques (**a** est leur axe de révolution). Les observations ne changent pas entre le système réel (a) et estimé (b). En effet, les ΔT_j sont bien identiques à chaque instant pour la Figure (a) d'une part et la Figure (b) d'autre part. Les observations sont invariantes à tout vissage d'axe **a** (*c.-à-d.* rotation d'axe **a** et translation selon **a**) appliqué aux points et aux poses de la caméra (dessinés en rouge).

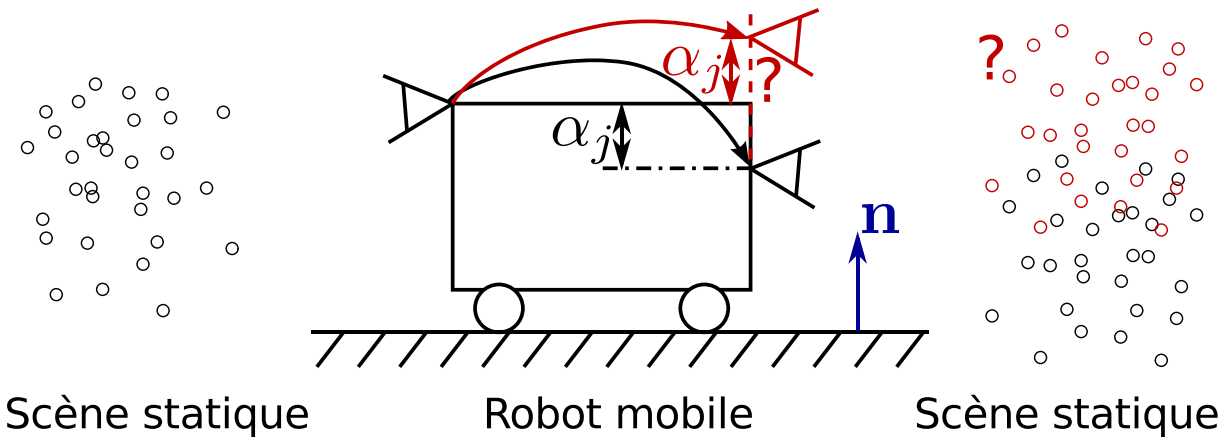


FIGURE 5.4 – Mouvement plan (cas singulier n°4) : la hauteur relative α_j entre les caméras (par rapport au plan du mouvement) n'est pas observable. Les observations ne changent pas entre le système réel (en noir) et estimé (en rouge).

Or, comme les axes $\mathbf{n}_1^k$ sont parallèles, chaque vecteur directeur (exprimé dans le repère de C_1^0) vaut $\mathbf{n}_1$. On retrouve donc bien que pour tout $k \in [\![1..K]\!]$, $\mathbf{n}_1$ appartient au noyau de $I_3 - R_1^k$. Tout vecteur de la forme donné par l'équation 5.15 vérifie donc l'équation 5.12. □

5.2.2.3 Solutions pour traiter les cas singuliers

Cette partie traite le cas des mouvements plans, qui est le cas singulier le plus fréquent pour les robots mobiles. Une estimation linéaire initiale des rotations extrinsèques est proposée. Puis nous montrons un protocole expérimental pour que les translations extrinsèques soient entièrement observables (y compris les hauteurs relatives des caméras). Nous développons des équations algébriques pour les initialiser linéairement. Enfin, comme la singularité des mouvements plan est contournée, l'ajustement de faisceaux multi-caméra (détaillé en Section 5.4) est appliqué pour raffiner l'ensemble des paramètres.

Estimation des rotations extrinsèques (pour des mouvements plans) Ce cas ne peut pas être résolu ni par la méthode d'Esquivel *et al* pour des mouvements quelconques [EWK07a, §4.1], ni par notre initialisation linéaire (de la Section 5.2.2.1), où la première étape d'estimation de la rotation (équation 5.7) échoue, car le système linéaire n'est pas de rang plein.

Avant d'énoncer puis de prouver cette affirmation via la propriété 7, rappelons la propriété 6 sur la codiagonalisation.

Propriété 6. Codiagonalisation : *Soit A et B deux matrices de $\mathbb{R}^{n\times n}$. A et B commutent et sont diagonalisables dans $\mathbb{K}$ ($\mathbb{R}$ ou $\mathbb{C}$), si et seulement si elles sont diagonalisables dans la même base de vecteurs propres de $\mathbb{K}^n$.* On dit alors que A et B sont codiagonalisables dans $\mathbb{K}$.

Une démonstration est disponible dans le livre [HJ, p51-53].

Propriété 7. *Pour les mouvements plans (et plus généralement les mouvements dont les axes de vissage sont parallèles), les matrices L_j (définies par l'équation 5.7) sont au plus de rang 6.*

Démonstration : Considérons K mouvements de vissage dont les axes sont parallèles (cas singulier n°4). Soient $j \in [\![2..N]\!]$ et $k \in [\![2..K]\!]$, on a :
- R_1^1 et R_1^k commutent et sont diagonalisables dans $\mathbb{C}$.
- R_j^1 et R_j^k commutent et sont diagonalisables dans $\mathbb{C}$.

D'après la propriété 6, R_1^1 et R_1^k sont codiagonalisables dans une base B_1 de vecteurs propres ($B_1 = (n_1, u_1, v_1)$, dont les vecteurs sont associés, dans l'ordre, aux valeurs propres 1, $e^{\mathtt{i}\theta_k}$ et $e^{-\mathtt{i}\theta_k}$). De même, R_j^1 et R_j^k sont codiagonalisables dans une base $B_j = (n_j, u_j, v_j)$ dont les vecteurs sont associés aux mêmes valeurs propres.

Ainsi, pour tout $k \in [\![2..K]\!]$, les 3 vecteurs $n_1 \otimes n_j$, $u_1 \otimes v_j$ et $v_1 \otimes u_j$ sont les vecteurs propres (deux à deux distincts) communs aux matrices $R_1^k \otimes R_j^k$. Il s'agit d'une *codiagonalisation généralisée* (également appelée *diagonalisation simultanée généralisée*). Tout comme dans la démonstration de la propriété 4 (p72), ces 3 vecteurs propres sont associés à la valeur

propre 1. Ces 3 vecteurs (deux à deux distincts) appartiennent donc au noyau de chaque matrice $I_9 - R_1^k \otimes R_j^k$. Or L_j concatène verticalement les K matrices $I_9 - R_1^k \otimes R_j^k$. Par conséquent, dans le cas d'une séquence de mouvements dont les axes de vissages sont parallèles, les matrices L_j sont donc au maximum de rang 6.

Remarque : Il suffit qu'au moins 2 mouvements aient des axes de vissage non-parallèles pour que les propriétés de codiagonalisation ne soient pas vérifiées. Dans ce cas, L_j est de rang 8. □

Par conséquent, dans le cas singulier n°4, les équations précédentes (présentées pour le cas général d'un mouvement 3D en Section 5.2.2.1) ne sont pas suffisantes pour estimer tous les degrés de liberté des rotations extrinsèques. Lors de mouvements plans, de nouvelles contraintes doivent être ajoutées à l'initialisation précédente (équation 5.7) afin de pouvoir estimer les rotations ΔR_j, comme expliqué par la propriété 8. Sans perte de généralité, on choisit comme premier mouvement T_1^1 un mouvement avec une rotation et une translation non-nulles.

Propriété 8. *Pour les mouvements plans, les rotations extrinsèques sont estimables. En notant* $\mathbf{t}_j'^k = (I_3 - R_j^k)\mathbf{t}_j^1 - (I_3 - R_j^1)\mathbf{t}_j^k$ *et en supposant que* $\mathbf{t}_j'^k \neq 0_{3\times1}$*, on résout le système de rang plein suivant :*
$\forall j \in [\![2..N]\!], \forall k \in [\![2..K]\!]$,

$$\begin{pmatrix} I_3 \otimes (\mathbf{t}_j'^k)^\top \\ I_3 \otimes (\mathbf{n}_j^k)^\top \\ I_3 \otimes (\mathbf{t}_j'^k \times \mathbf{n}_j^k)^\top \end{pmatrix} vec(\Delta R_j) = \begin{pmatrix} \mathbf{t}_1'^k \\ \mathbf{n}_1^k \\ \mathbf{t}_1'^k \times \mathbf{n}_1^k \end{pmatrix} \tag{5.17}$$

Démonstration : Primo, en partant des équations (5.6b) pour $k = 1$ et $k \neq 1$, on a :

$$(I_3 - R_1^1)\Delta\mathbf{t}_j = \mathbf{t}_1^1 - \Delta R_j \mathbf{t}_j^1 \tag{5.18}$$
$$(I_3 - R_1^k)\Delta\mathbf{t}_j = \mathbf{t}_1^k - \Delta R_j \mathbf{t}_j^k \tag{5.19}$$

Secondo, comme R_1^1 et R_1^k commutent car leurs axes de rotation sont parallèles, on a :
$\forall k \in [\![1..K]\!]$,

$$(I_3 - R_1^k)(I_3 - R_1^1) - (I_3 - R_1^1)(I_3 - R_1^k) = 0_{3\times3} \tag{5.20}$$

En développant l'expression $(I_3 - R_1^k)(5.18) - (I_3 - R_1^1)(5.19)$, elle se simplifie (grâce à l'équation (5.20)) en (5.21) :

$$(5.18) \text{ à } (5.20) \Rightarrow \quad (I_3 - R_1^k)(\mathbf{t}_1^1 - \Delta R_j \mathbf{t}_j^1) - (I_3 - R_1^1)(\mathbf{t}_1^k - \Delta R_j \mathbf{t}_j^k) = 0_{3\times1} \tag{5.21}$$
$$(5.21) \Leftrightarrow \quad (I_3 - R_1^k)\Delta R_j \mathbf{t}_j^1 - (I_3 - R_1^1)\Delta R_j \mathbf{t}_j^k = (I_3 - R_1^k)\mathbf{t}_1^1 - (I_3 - R_1^1)\mathbf{t}_1^k \tag{5.22}$$
$$(5.22)\&(5.5a) \Leftrightarrow \quad \Delta R_j(I_3 - R_j^k)\mathbf{t}_j^1 - \Delta R_j(I_3 - R_j^1)\mathbf{t}_j^k = (I_3 - R_1^k)\mathbf{t}_1^1 - (I_3 - R_1^1)\mathbf{t}_1^k \tag{5.23}$$
$$\Leftrightarrow \quad \Delta R_j \mathbf{t}_j'^k = \mathbf{t}_1'^k \tag{5.24}$$

D'autre part, pour le $k^{ème}$ mouvement du système multi-caméra, l'axe $\mathbf{n}^k$ de rotation est commun à toutes les caméras. Cet axe de rotation $\mathbf{n}^k$ est simplement exprimé dans un repère donné : chaque axe $\mathbf{n}_j^k$ des rotations R_j^k est exprimé dans le repère de la caméra C_j à l'instant $k = 0$. Avec un changement de base (*cf* Annexe A.1) entre les repères de C_1^k et de C_j^k, on vérifie que :

$$\forall j \in [\![1..N]\!], \forall k \in [\![1..K]\!],\ \Delta R_j \mathbf{n}_j^k = \mathbf{n}_1^k \tag{5.25}$$

De plus, $\mathbf{t}_j'^k$ et $\mathbf{n}_j^k$ sont linéairement indépendants car le mouvement est plan. Par conséquent, les vecteurs $\mathbf{t}_j'^k$, $\mathbf{n}_j^k$ et le produit vectoriel $\mathbf{t}_j'^k \times \mathbf{n}_j^k$, forment une famille linéairement indépendante. Finalement, en utilisant la propriété 5.4 (p71), on obtient le système de rang plein (5.17). □

Remarque : En pratique, les systèmes (5.7) et (5.17) sont concaténés pour estimer ΔR_j. De plus, comme les conditions de planéité ne sont pas parfaites en présence de bruit, les équations (5.20) sont quasiment vérifiées. Il est donc préférable d'orthogonaliser les matrices de rotation obtenues via une SVD.

Protocole pour contourner la singularité des mouvements plans Pour les mouvements plans, les translations extrinsèques demeurent partiellement estimables. Si on conserve uniquement l'initialisation proposée pour le cas général (en Section 5.2.2.1), les hauteurs relatives des caméras α_j restent inconnues. Ainsi, au lieu d'utiliser des valeurs aberrantes, on les initialise à une éventuelle valeur approximativement connue *a priori* ou 0 m sinon. En pratique, au moins une pose non-coplanaire suffit à s'affranchir de la singularité. Par exemple, si le système multi-caméra est embarqué, il suffirait d'avoir un dos-d'âne sur la trajectoire du robot mobile.

Une autre solution (pour estimer totalement les translations extrinsèques avec des mouvements plans) consiste à déplacer le système multi-caméra, afin qu'au moins un point de la scène S soit vu par chaque caméra à différents instants. Pendant la phase d'acquisition, le système multi-caméra est déplacé afin que les scènes observées soient permutées (*c.-à-d.* la scène observée par l'une des caméras à un instant donné, peut ensuite être vue par une autre caméra.). Avec des correspondances connues entre les images, les hauteurs relatives α_j sont calculables. En résumé, pour des mouvements plans respectant ce protocole, il est possible d'estimer à la fois les rotations ΔR_j et les translations $\Delta \mathbf{t}_j$.

Nous allons montrer comment il est possible de compléter l'initialisation de la Section 5.2.2.1 (du cas général mouvements 3D du système multi-caméra) : on ajoute de nouvelles contraintes (grâce à la permutation des scènes) pour pouvoir estimer les hauteurs relatives entre les caméras pour des mouvements plans. Notons que ces contraintes restent valables, que le mouvement soit plan ou bien 3D. Soit σ une permutation de l'ensemble $[\![1..N]\!]$. Soit $T^k_{j\to\sigma(j)}$ la transformation homogène entre la caméra C_j à l'instant 0 et la caméra $C_{\sigma(j)}$ à l'instant k.

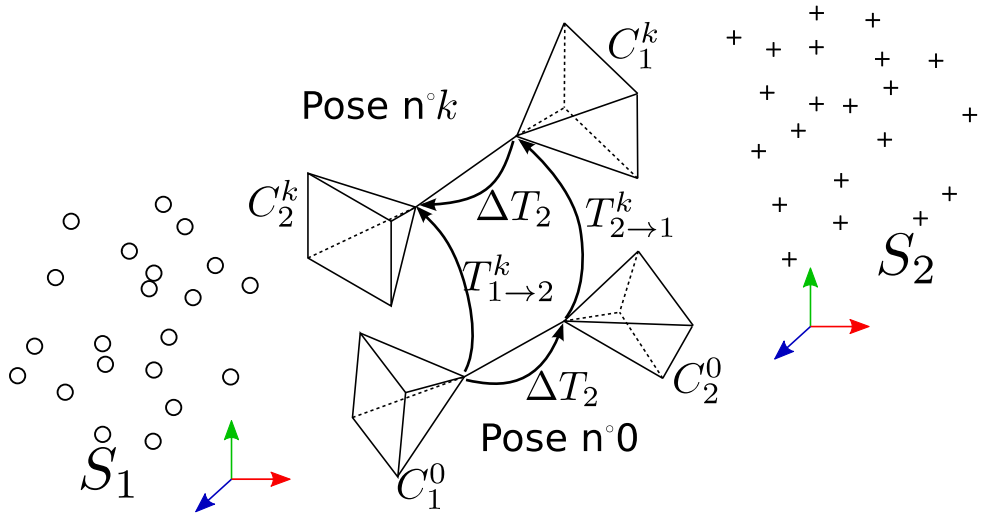


FIGURE 5.5 – Système multi-caméra à champs non-recouvrants ayant subit une permutation entre les poses n°0 et n°k. Pour la pose n°0, C_1 observe S_1 et C_2 observe S_2, et inversement pour la pose n°k.

Estimation des translations extrinsèques, pour des mouvements plans et plus de deux caméras : Pour $N > 2$, en cas de permutation circulaire des scènes observées, on obtient un système d'équations couplées (non-détaillées dans le manuscrit). Plus de travaux seraient nécessaires pour résoudre ces équations. Cependant, un moyen simple de les contourner est d'imposer que, lors de l'expérience à $N > 2$ caméras, plusieurs transpositions $\sigma = \tau$ soient effectuées. Les scènes observées sont donc transposées deux-à-deux (*c.-à-d.* $j' = \tau(j)$ et $j = \tau(j')$). On se ramène ainsi au cas plus simple de deux scènes (résolu ci-dessus). En représentant les caméras comme les nœuds d'un graphe dont les transpositions sont les branches, il suffit alors de s'assurer qu'il existe au moins un arbre couvrant passant par tous les nœuds. On peut alors estimer la hauteur relative des N caméras de proche en proche, en appliquant le cas ci-dessous pour les $N-1$ branches de l'arbre couvrant. Pour être plus précis, pour chaque branche (et donc chaque appel de la méthode ci-dessous), l'une des deux caméras est alors temporairement identifiée comme caméra maître. En parcourant l'arbre à partir de C_1, on détermine l'ensemble des $N-1$ transformations extrinsèques ΔT_j. Si toutefois, la disposition des caméras et la restriction aux mouvements plans font qu'il existe une caméra dont la scène observée ne peut être permutée avec celle d'une autre caméra, alors la hauteur de cette caméra isolée (n'appartenant pas à un graphe couvrant) ne sera pas estimable (car non-observable).

Estimation de la translation extrinsèque, pour des mouvements plans et deux caméras : Pour $N = 2$, on a $\sigma(1) = 2$ et $\sigma(2) = 1$, comme l'illustre la Figure 5.5, on obtient les équations suivantes :

$$T_{1\to2}^k (\Delta T_2)^{-1} = \Delta T_2 T_{2\to1}^k \tag{5.26}$$

$$\Leftrightarrow \begin{pmatrix} R^k_{1\to2} & \mathbf{t}^k_{1\to2} \\ 0_{1\times3} & 1 \end{pmatrix} \begin{pmatrix} \Delta R_2{}^\top & \Delta R_2{}^\top \Delta \mathbf{t}_2 \\ 0_{1\times3} & 1 \end{pmatrix} = \begin{pmatrix} \Delta R_2 & \Delta \mathbf{t}_2 \\ 0_{1\times3} & 1 \end{pmatrix} \begin{pmatrix} R^k_{2\to1} & \mathbf{t}^k_{1\to2} \\ 0_{1\times3} & 1 \end{pmatrix} \tag{5.27}$$

$$\Leftrightarrow \begin{cases} R^k_{1\to2}(\Delta R_2)^\top = \Delta R_2 R^k_{2\to1} & \text{(5.28a)} \\ R^k_{1\to2}(\Delta R_2)^\top \Delta \mathbf{t}_2 + \mathbf{t}^k_{1\to2} = \Delta R_2 \mathbf{t}^k_{2\to1} + \Delta \mathbf{t}_2 & \text{(5.28b)} \end{cases}$$

$$\Leftrightarrow \begin{cases} R^k_{1\to2}(\Delta R_2)^\top (R^k_{2\to1})^\top = \Delta R_2 & \text{(5.29a)} \\ \Delta R_2 R^k_{2\to1} \Delta \mathbf{t}_2 + \mathbf{t}^k_{1\to2} = \Delta R_2 \mathbf{t}^k_{2\to1} + \Delta \mathbf{t}_2 & \text{(5.29b)} \end{cases}$$

$$\Leftrightarrow \begin{cases} (I_9 \quad (R^k_{1\to2} \otimes R^k_{2\to1})\Phi_9) vec(\Delta R_2) = 0_{9\times1} & \text{(5.30a)} \\ (I_3 + \Delta R_2 R^k_{2\to1}) \Delta \mathbf{t}_2 = \mathbf{t}^k_{1\to2} \quad \Delta R_2 \mathbf{t}^k_{2\to1} & \text{(5.30b)} \end{cases}$$

où Φ_9 est la matrice de permutation 9×9 telle que $vec(M^\top) = \Phi_9 vec(M)$.

Les équations (5.30a) peuvent être ajoutées aux équations (5.7), mais cela n'augmente pas suffisamment le rang du système dans le cas de mouvements plans (contrairement à l'équation (5.17), prévue à cet effet). Finalement, les équations (5.12) et (5.30b) sont concaténées pour estimer linéairement la translation $\Delta \mathbf{t}_2$ en totalité.

Interprétation : Ci-après, nous proposons une interprétation plus intuitive montrant que la permutation des scènes permet de contraindre l'estimation de la hauteur. En effet, si le véhicule repasse au même endroit, les caméras sont toujours à la même hauteur par rapport au sol (et par rapport aux scènes qu'elles observent).

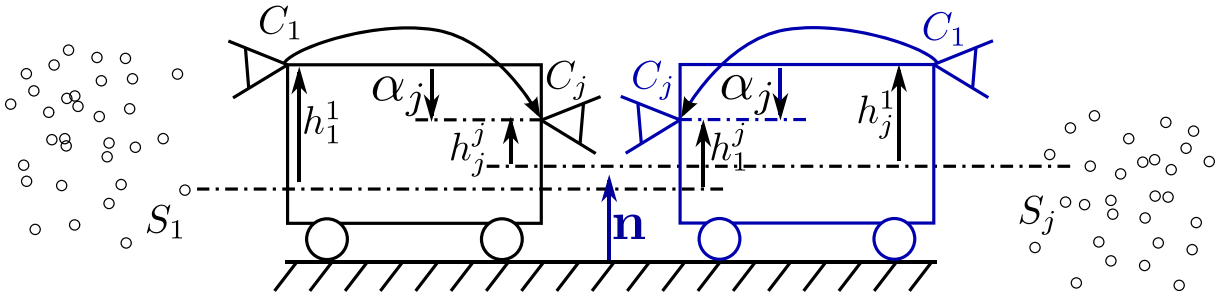


FIGURE 5.6 – La permutation des scènes observées (via un demi-tour) permet de contraindre la hauteur relative des caméras. Sur cette vue de côté, le demi-tour (rotation d'environ 180° autour de la normale au plan du sol) revient à appliquer une symétrie axiale au véhicule.

Comme chaque caméra peut estimer sa hauteur par rapport à une scène qu'elle observe, la hauteur relative entre les scènes reste constante avant et après la permutation. En s'appuyant sur la Figure 5.6, on a :

$$h^1_1 + \alpha_j \quad h^j_j = h^j_1 \quad \alpha_j \quad h^1_j \tag{5.31}$$

$$\Leftrightarrow \alpha_j = \frac{1}{2}(h^j_j + h^j_1 \quad h^1_1 \quad h^1_j) \tag{5.32}$$

Pour prouver l'unicité de l'estimation de la hauteur relative α_j grâce à la permutation des scènes, considérons que $\alpha'_j = \alpha_j + h$. Comme le montre la Figure 5.7, la caméra C_j et sa scène S_j sont donc translatées de h vers le haut (*c.-à-d.* selon la normale au plan). La caméra C_1 permutée estime correctement sa hauteur h_j^1 par rapport à la scène S_j. La hauteur relative entre les caméras après le demi-tour est toujours $\alpha_j + h$. La hauteur de la caméra Cj permutée par rapport à la scène S_1 est donc $h_1^j + 2h$ d'une part. D'autre part, la caméra C_j permutée estime correctement sa hauteur h_1^j par rapport à la scène S_1. Par conséquent, $h_1^j + 2h = h_1^j$, d'où $h = 0$ et $\alpha'_j = \alpha_j$.

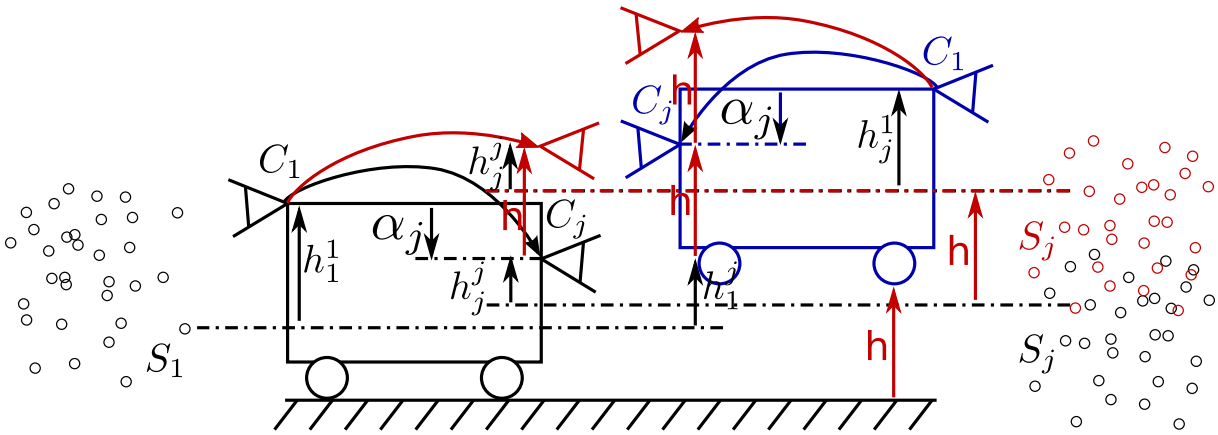


FIGURE 5.7 – Schéma prouvant intuitivement que la permutation des scènes (via un demi-tour) permet d'estimer une unique hauteur relative des caméras. On suppose qu'il y a un offset h sur la hauteur relative estimée, pour prouver que cet offset est nul.

5.3 Structure from motion et auto-étalonnage extrinsèque

Dans cette section, nous allons montrer qu'il est possible de réaliser l'étalonnage extrinsèque conjointement à une reconstruction 3D de type SFM. L'idée est de pouvoir s'affranchir d'une procédure dédiée à l'étalonnage extrinsèque en se servant directement de la séquence d'apprentissage à la fois comme la référence à suivre et comme données d'étalonnage.

5.3.1 Formalisation du problème

Tout comme dans la section précédente, on étudie au moins $N \geq 2$ caméras rigidement liées, synchronisées et à champs disjoints. Les caméras se déplacent maintenant le long d'un parcours en observant des points d'intérêt. Le formalisme reste donc sensiblement identique à la Section 5.2, à quelques nuances près. En effet, dans cette section on utilise un repère global. T_j^k

représente alors la transformation homogène entre ce repère monde et le repère de la $k^{ème}$ pose de la caméra C_j (*cf* Figure 5.8). Ainsi, T_1^k correspond à la $k^{ème}$ pose de la *caméra maître* C_1, qui est définie comme étant la $k^{ème}$ pose du système multi-caméra. Enfin, à la précédente notion de scènes (indexées par l'indice de la caméra l'observant), on préfère dans cette section la notion d'un ensemble statique de M points 3D observés par le système multi-caméra en mouvement. On note $\{\overline{P}_i\}_{i\in[\![1..M]\!]}$ les M points 3D exprimés dans le repère global.

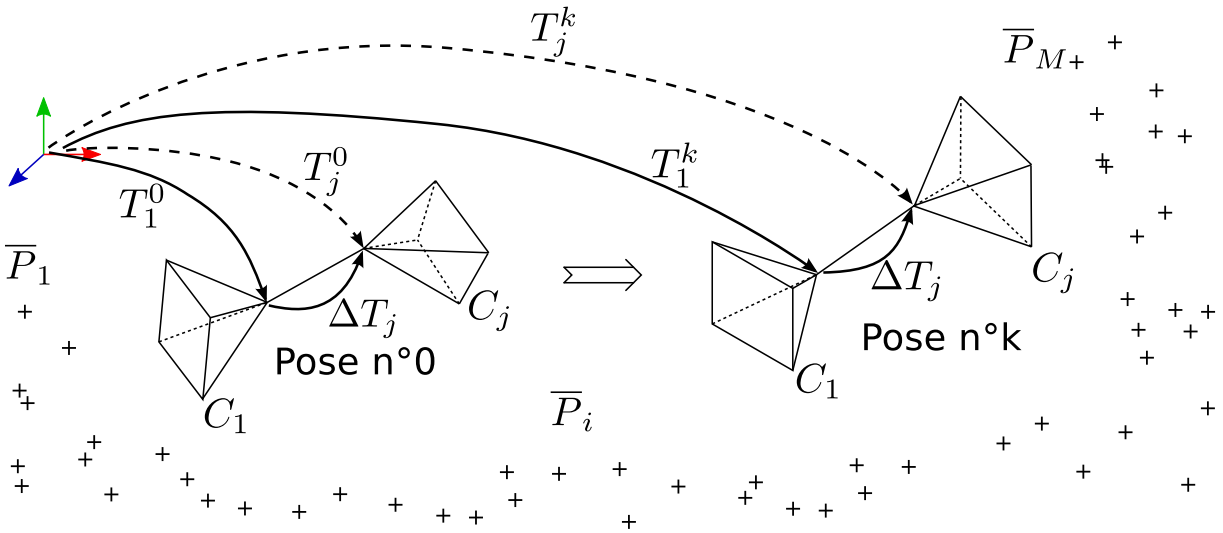


FIGURE 5.8 – Caméras $C_1..C_j..C_N$ à champs disjoints se déplaçant le long d'une scène statique. Les flèches pleines indiquent les transformations optimisées par le MCBA (Section 5.4). Les points 3D $\overline{P}_1..\overline{P}_i..\overline{P}_M$ sont également optimisés.

5.3.2 Algorithme proposé

Considérons une séquence d'images enregistrées par N caméras synchronisées, lors d'un déplacement du système le long d'un parcours. L'algorithme décrit ci-après permet d'obtenir conjointement à l'étalonnage extrinsèque du système multi-caméra, ses poses et la reconstruction d'une carte 3D d'amers visuels.

1. A partir d'une connaissance *a priori* grossière sur les paramètres extrinsèques $\widehat{\Delta T_j}$, ou d'une estimation linéaire 5.2.2 de ces paramètres, on applique un ajustement de faisceaux hiérarchique pour des caméras rigidement liées (un algorithme de *Structure From Motion* généralisé à partir de [RLDL07] au cas multi-caméra, *cf* Section 5.3.3). Le SFM se termine avec un ajustement de faisceaux multi-caméra global dont les paramètres extrinsèques $\widehat{\Delta T_j}$ restent constants. On obtient alors une reconstruction grossière.

2. Puis, on recoupe les observations du robot s'il est revenu dans une zone déjà empruntée. Cette étape permet de fermer les boucles présentes sur la trajectoire. Nous détaillons l'algorithme développé dans la Section 5.3.4.
3. Enfin, les paramètres extrinsèques, les poses du système multi-caméra et les points 3D sont optimisés par l'ajustement de faisceaux multi-caméra (MCBA, Section 5.4).

Remarques : Comme les différentes caméras sont séparées par une distance non-nulle, la reconstruction est obtenue à un facteur d'échelle près (global, avec peu de dérive par rapport au cas monoculaire). De plus, l'approche présente les mêmes cas singuliers que dans la Section 5.2.2.2.

5.3.3 Structure from motion

L'algorithme de structure from motion (SFM) repose sur une mise en correspondance de points d'intérêt entre des images successives. Pour faire cela, des points d'Harris [HS88] sont détectés, puis mis en correspondance à l'aide d'une corrélation de type ZNCC (Zero-mean Normalized Cross Correlation).

Dans un premier temps, les *poses clés*[4] sont sélectionnées. La première pose du système multi-caméra est toujours une pose clé. Puis, une pose $\mathcal{P}$ de la séquence devient une pose clé avant que le nombre de points mis en correspondance entre $\mathcal{P}$ et la dernière pose clé passe sous un seuil. La géométrie épipolaire est ensuite calculée sur les trois premières images clés pour chaque caméra. On détermine les trois premières poses du système multi-caméra et un ensemble de points 3D. Cette géométrie initiale est raffinée par un ajustement de faisceaux aux paramètres extrinsèques constants. Pour reconstruire les séquences, nous utilisons un ajustement de faisceaux hiérarchique similaire à celui publié dans [RLDL07] et adapté au cas du multi-caméra. Cette méthode est détaillée par l'algorithme 5. En bref, la séquence originale est scindée en sous-séquences plus courtes qui se recouvrent sur quelques poses. La géométrie de chaque sous-séquence est calculée, et les points 3D sont triangulés. Les reconstructions de chaque sous-ensemble sont optimisées à l'aide d'un ajustement de faisceaux aux paramètres extrinsèques constants $\widehat{\Delta T_j}$. Les sous-séquences sont fusionnées et l'algorithme se termine avec un ajustement de faisceaux global (*c.-à-d.* sur toutes les poses clés) aux paramètres extrinsèques constants.

D'autre part, une méthode robuste de rejet des points aberrants est incluse dans l'ajustement de faisceaux : si l'erreur de reprojection d'un point 3D est supérieure à un seuil, le point 2D est rejeté. Les points d'intérêt sont itérativement classés en inlier ou outlier jusqu'à la convergence de l'ajustement de faisceaux.

4. Pour du multi-caméra, la $k^{ème}$ pose clé est liée aux images clés de chaque caméra C_j^k.

Algorithme 5: Structure From Motion Hiérarchique

fonction SFM(Séquence $\mathcal{S}$)
  si Nombre de poses clés dans $\mathcal{S} \geq 4$ **alors**
    Scinder $\mathcal{S}$ en $\mathcal{S}_1, \mathcal{S}_2$ avec un recouvrement d'au moins 2 poses.
    Appeler SFM($\mathcal{S}_1$).
    Appeler SFM($\mathcal{S}_2$).
    $\mathcal{S} \leftarrow$ fusion($\mathcal{S}_1, \mathcal{S}_2$).
    Optimiser $\mathcal{S}$ avec des paramètres extrinsèques constants.
  sinon
    Dans ce cas $\mathcal{S} = \{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3\}$.
    si $\mathcal{S}$ = 3 premières poses **alors**
      Calculer la géométrie épipolaire pour chaque caméra.
      Trianguler les points 3D.
    sinon
      Dans ce cas, la géométrie de la pose $\mathcal{P}_1$ et $\mathcal{P}_2$ a été calculée précédemment.
      Calculer la pose $\mathcal{P}_3$ avec les points 2D de $\mathcal{P}_3$ mis en correspondance avec les points triangulés depuis $\mathcal{P}_1$ et $\mathcal{P}_2$.
      Trianguler des points supplémentaires mis en correspondance entre $\mathcal{P}_2$ et $\mathcal{P}_3$.
    finsi
    Optimiser $\mathcal{S}$ avec des paramètres extrinsèques constants.
  finsi
fin de la fonction

5.3.4 Fermeture de boucles

Lorsque le système multi-caméra repasse par une zone déjà empruntée, nous proposons d'utiliser l'algorithme 6 pour ajouter de nouvelles observations 2D aux points 3D déjà reconstruits. Ainsi, par *boucle*, on n'entend pas uniquement une trajectoire globale commençant et se terminant sur un même endroit, mais plutôt une portion de la trajectoire qui emprunte la même zone entre deux instants distincts. A partir de la reconstruction avec les paramètres extrinsèques approximatifs, un simple critère de proximité permet de présélectionner les poses voisines des caméras. Grâce à un processus de re-localisation, les meilleurs points 2D inliers sont ajoutés comme les nouvelles observations.

Comme la permutation des scènes observées est supportée, une caméra peut très bien reconnaitre une zone observée préalablement par une autre caméra. Cette étape de fermeture de boucle permet de contraindre l'estimation des hauteurs relatives des caméras, notamment dans le cas de mouvements plans (*cf* Section 5.2.2.2) et de permutation des scènes observées. Elle permet aussi d'éviter les dernières dérives dans la reconstruction 3D.

La Figure 5.9 illustre l'algorithme 6 en fonctionnement sur une trajectoire synthétique. Les correspondances entre images sont représentées par des segments verts, les points inliers lors d'un nouveau passage sont illustrés en jaune.

Pour aller plus loin, on peut se poser la question de l'influence des fermetures de boucles sur les points 3D, les poses du système multi-caméra et les paramètres extrinsèques.

Les points 3D : Comme de nouvelles observations des points 3D sont ajoutées, les points 3D ne peuvent qu'être mieux reconstruits.

Les poses du système multi-caméra : Si une dérive résiduelle (par exemple en altitude) s'est accumulée entre deux poses k_1 et k_2, le bouclage permet de la répartir sur toutes les poses intermédiaires (pour $k \in [\![k_1..k_2]\!]$).

Les paramètres extrinsèques : Si les scènes observées sont permutées, la hauteur relative entre deux caméras est observable. Cela a été démontré dans la Section 5.2.2.3.

Au contraire, l'écart latéral relatif des caméras n'est pas forcément contraint par une fermeture de boucle. Cela dépend de la configuration des caméras (*c.-à-d.* des paramètres extrinsèques) et des positions du système multi-caméra. Considérons qu'une zone soit empruntée lors d'un premier passage dans un sens, puis réempruntée mais dans le sens opposé lors d'un deuxième passage, comme illustrées en vue de dessus par la Figure 5.10a. Si la rotation extrinsèque entre les caméras C_1 et C_j est d'exactement $180°$ autour de la normale $\mathbf{n}$ au plan, et si le demi-tour définit la même matrice de rotation, alors réemprunter la même direction avec ce demi-tour ne contraindra pas l'écart latéral. Dans ce cas particulier, il est possible d'avoir un écart relatif erroné : les observations ne seront pas modifiées en translatant le système permuté (*cf* Figure 5.10b). Nous allons prouver ci-après la non-unicité de l'écart latéral Δx_j entre la caméra C_1 et C_j dans ce cas particulier. Considérons sa valeur erroné $\Delta x'_j = \Delta x_j + x$, induisant à C_j une translation du vecteur x. Comme chaque caméra peut estimer sont écart-latéral par rapport à la scène qu'elle observe, la scène S_j est également translatée de x. Pour la pose permutée, la caméra C_1 estime donc sa pose latérale par rapport à S_j translatée. En s'appuyant sur la Figure 5.10b, et sur le fait que l'écart latéral entre les scènes estimées doit être constant

Algorithme 6: Détection des boucles

Définir les seuils d_{max} et α_{max} du critère de proximités.
Pour chaque *pose* $k_1 \in [\![0..K]\!],\ \forall k_2 \in [\![0..K]\!],\ k_1 < k_2$ **:**
  Pour chaque *cameras* $j_1 \in [\![2..N]\!], \forall j_2 \in [\![2..N]\!]$ **:**
    si $C_{j_1}^{k_1}$ est proche de $C_{j_2}^{k_2}$ (*c.-à-d.* $\|\mathbf{t}_{j_1}^{k_1} \quad \mathbf{t}_{j_2}^{k_2}\| < d_{max}$ et $d(R_{j_1}^{k_1}, R_{j_2}^{k_2}) < \alpha_{max}$)
    alors
      Mettre en correspondance les primitives de $C_{j_1}^{k_1}$ et $C_{j_2}^{k_2}$.
      Calculer la pose relative de $C_{j_1}^{k_1}$ par rapport à $C_{j_2}^{k_2}$ avec ces appariements.
      si le nombre d'inliers et supérieur à un seuil **alors**
        Ajouter les nouveaux points 2D inliers liés aux points 3D vus précédemment par $C_{j_1}^{k_1}$.
      finsi
    finsi

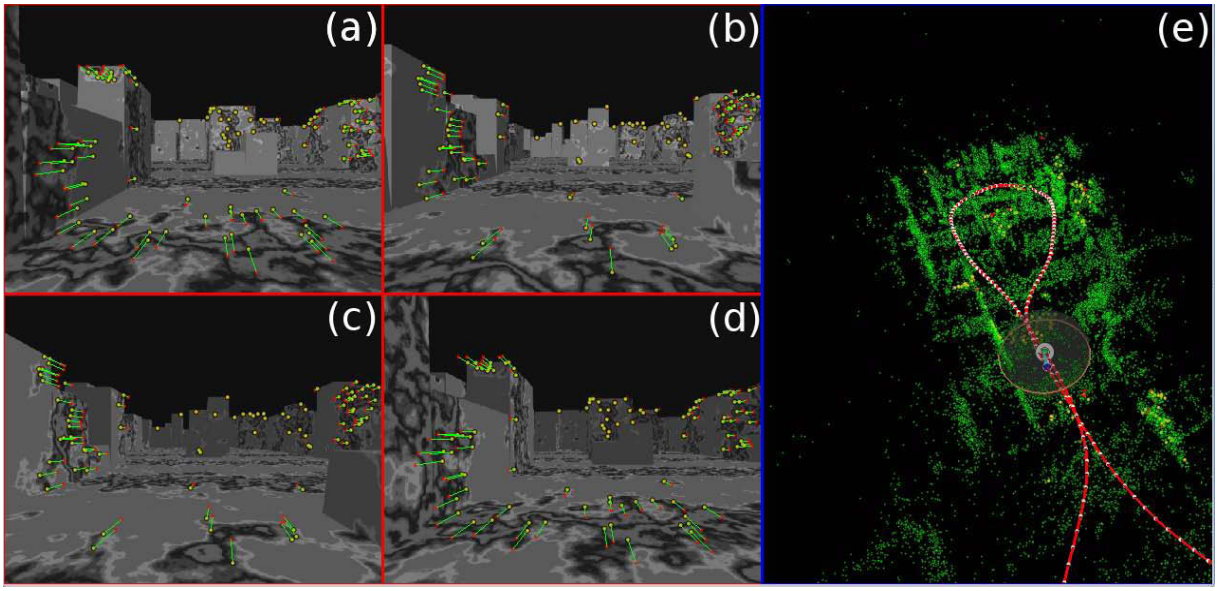


FIGURE 5.9 – Algorithme de fermeture de boucles multi-caméra en fonctionnement sur une trajectoire synthétique. (a) et (b) sont respectivement les images avant et arrière lors du premier passage. (c) et (d) sont respectivement les images avant et arrière lors du deuxième passage. Ici, les images observées par différentes caméras sont mises en correspondance (*c.-à-d.* (a) avec (d), et (b) avec (c)). (e) est une vue plongeante de la reconstruction (avec la trajectoire 3D du système multi-caméra en rouge, les points 3D en vert, la pose courante $C_{j_1}^{k_1}$ en blanc et la pose ré-estimée de $C_{j_2}^{k_2}$) en bleu.

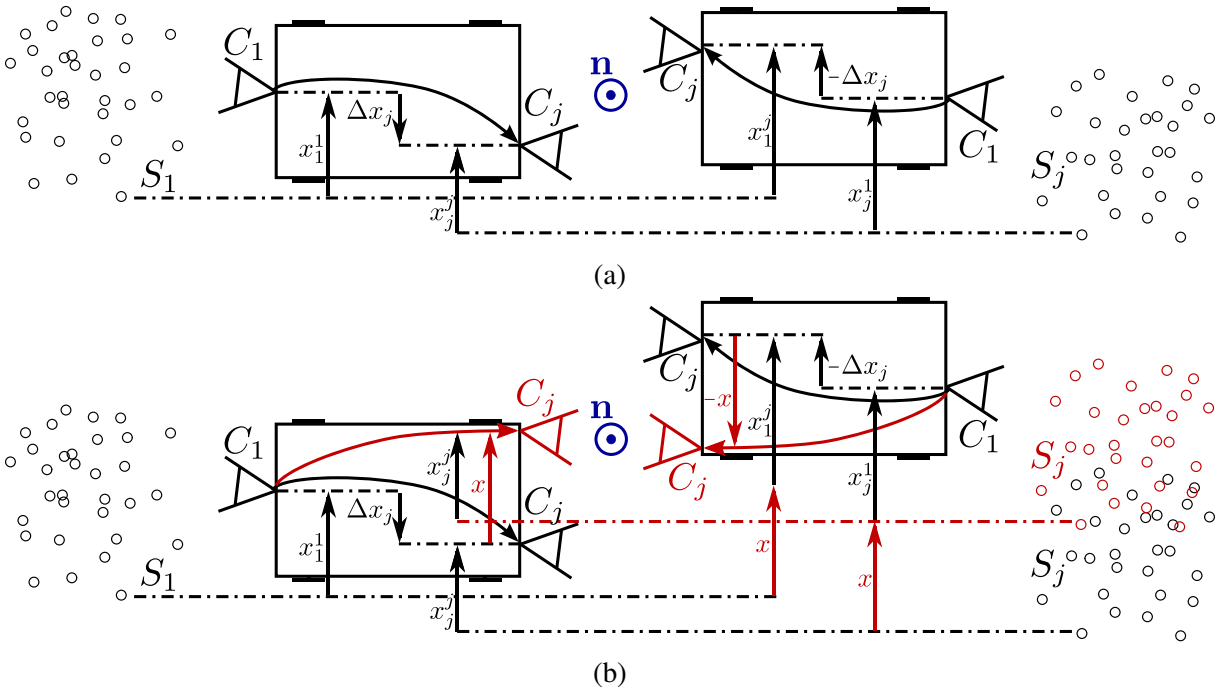


FIGURE 5.10 – Schéma prouvant que la permutation des scènes (via un demi-tour) ne permet pas forcément de contraindre l'écart latéral des caméras. Ici, la rotation extrinsèque ΔR_j est de 180° autour de la normale $\mathbf{n}$ au plan. En vue de dessus, le demi-tour (rotation d'exactement 180°autour de la normale au plan) revient à appliquer une symétrie centrale au véhicule. Les figures (a) et (b) illustrent respectivement la configuration réelle et une configuration équivalente (dans laquelle les observations sont identiques).

avant et après permutation, on obtient l'équation 5.33 (où les termes Δx_j se simplifient) :

$$x_1^1 + \Delta x_j + x \quad x_j^j = \quad x_j^1 + \Delta x_j + x + x_1^j \tag{5.33}$$

$$\Leftrightarrow \qquad x_1^1 \quad x_j^j = x_1^j \quad x_j^1 \tag{5.34}$$

On conclut que ce demi-tour (associé à cette configuration de caméra) ne permet pas de contraindre la translation latérale Δx_j. Dès que l'angle de la rotation extrinsèque ou bien l'angle du demi-tour ne vaut plus exactement 180°, on ne rentre plus dans ce cas particulier.

En poussant ce raisonnement, on peut généraliser à partir de l'équation 5.30b que si pour tout k, $\Delta R_j R^k_{j\to 1}$ est une matrice de rotation $R(\mathbf{a}, 180°)$ de 180°autour de l'axe $\mathbf{a}$ (avec $\mathbf{a}$ respectivement égal à $(1,0,0)^\top$, $(0,1,0)^\top$ et $(0,0,1)^\top$), alors $det(I_3 + \Delta R_j R^k_{j\to 1}) = 0$. On ne peut alors qu'observer partiellement la translation extrinsèque $\Delta\mathbf{t}_j$ (grâce à la permutation) respectivement selon sa première, sa deuxième (c'est le cas du demi-tour pour le véhicule où seule la hauteur est estimable) et sa troisième composante.

Un résultat corollaire est que pour que l'écart latéral (et longitudinal) entre les caméras C_1 et C_j soit contraint par une pose d'un deuxième passage après un demi-tour, il faut que l'équation 5.35 soit vérifiée.

$$R^k_{j\to 1} \neq \Delta {R_j}^\top R(\mathbf{n}, 180^\circ) \tag{5.35}$$

Pour le bouclage (Algorithme 6), des observations ajoutées entre une pose k_1 lors d'un premier passage et une pose k_2 contraignent l'écart latéral (et longitudinal), si l'équation 5.36 est vérifiée.

$$R^{k_1\to k_2}_{j\to 1} \neq \Delta {R_j}^\top R(\mathbf{n}, 180^\circ) \tag{5.36}$$

Ceci est d'autant plus vrai si les membres de l'équation 5.36 diffèrent. Finalement, on retrouve que pour des rotations extrinsèques de 180°, plus le virage est serré, plus l'écart latéral et longitudinal sont contraints (qu'un demi-tour soit appliqué ou non). Ce n'est donc pas la permutation des scènes en elle-même qui apporte une contrainte sur l'écart latéral (mais plutôt les virages effectués pendant la manœuvre d'étalonnage).

5.4 Ajustement de faisceaux multi-caméra (MCBA)

La dernière étape des algorithmes précédents (Sections 5.2 et 5.3) consiste à optimiser à la fois les paramètres extrinsèques ΔT_j, les points 3D $\overline{P}_i$ et les poses du système multi-caméra T^k_1. Nous appelons cet algorithme *ajustement de faisceaux multi-caméra* (MCBA pour *Multi-Camera Bundle Adjustment*) [5]. Notons que les similitudes avec l'étalonnage bras-œil ne sont plus valables dans cette partie.

5.4.1 MCBA et état de l'art

Cet algorithme n'a pas été proposé ni par [DTL+10], ni par [EWK07a], qui estiment les paramètres extrinsèques à l'aide des trajectoires des caméras supposées connues (ni par [Pag], qui ne ré-optimise pas la position de ses motifs lors de l'étalonnage extrinsèque). [Kin93] est l'un des premiers articles de photogrammétrie proposant d'incorporer les paramètres extrinsèques dans un ajustement de faisceaux pour une tête stéréo à champs recouvrants. Il compare un algorithme d'ajustement de faisceaux, qui a une représentation minimale des paramètres, à un algorithme avec une représentation redondante et des contraintes supplémentaires sur les paramètres extrinsèques. Sa variante des équations normales et leurs résolutions, basées sur [oPSTH80], sont qualifiées par [TMHF00] comme inadaptées aux problèmes en grandes dimensions. Nous avons tout d'abord présenté le principe du MCBA dans [3], que nous avons ensuite détaillé dans [2] (principe repris dans [CAD11b]). Publié quelques mois auparavant, [VA09] utilise un système multi-caméra totalement mobile, qu'il étalonne grâce à l'équivalent d'un MCBA décrit

5. Nous avons remplacé la sigle SBA (pour Specific Bundle Adjustment) que nous avions proposé dans [2], afin d'éviter d'éventuelle confusion avec le Sparse Bundle Adjustment proposé dans [LA09]

avec l'algèbre géométrique conforme (en anglais *Conformal Geometric Algebra* [6]). Appliqué à un espace euclidien de dimension 3, cette algèbre représente dans un même espace de dimension 5, aussi bien des primitives (points, droites, plans, cercles,...), que des transformations (rotations, translations,...). L'auteur sème un ensemble de petites sphères dans la scène. Leurs positions sont préalablement mesurées à l'aide d'un tachéomètre. Il minimise une fonction de coût dépendant à la fois d'une erreur angulaire et d'une erreur de reconstruction des points 3D (par rapport à la mesure métrologique). Seul le gradient de la fonction de coût est exprimé dans l'algèbre, sans décrire l'implémentation de l'optimisation.

5.4.2 Vue d'ensemble

La Figure 5.8 (p83) illustre géométriquement l'ensemble du système, ainsi que les paramètres optimisés. Les caméras étant rigidement liées, on optimise seulement les poses T_1^k, exprimées dans un repère global, de la *caméra maître* C_1 ($6(K+1)$ paramètres pour les $\mathbf{t}_1^k$ et R_1^k). Les poses des autres caméras sont exprimées relativement à la caméra maître ($6(N-1)$ paramètres pour $\Delta\mathbf{t}_j$ et ΔR_j, représentées localement par des angles d'Euler). Au final, avec les M points 3D, $6(K+N)+3M$ paramètres sont optimisés par l'algorithme de Levenberg Marquardt en minimisant les erreurs de reprojection. L'annexe B.1 formalise la projection des points de la scène dans le repère de chaque caméra (pour les différentes poses).

Tout comme pour les reconstructions monoculaires, une reconstruction multi-caméra est obtenue à une similitude directe de l'espace près (*c.-à-d.* à une rotation, une translation et une échelle près, *cf* Annexe A.2.2). Sept paramètres doivent donc être fixés. A cet effet, pour satisfaire cette contrainte, il est suffisant de fixer la première pose T_1^0 de la caméra maître (6 paramètres). Pour fixer l'échelle, on fixe la composante de la translation (1 paramètre) d'une autre pose (par exemple $\mathbf{t}_1^1$) qui a le plus changé [7]. Pour cela, il suffit de supprimer les lignes et les colonnes correspondantes dans la matrice jacobienne et hessienne.

Soit $\boldsymbol{\epsilon}_{ij}^k$ l'erreur de reprojection du point $\overline{P}_i$ observé par la caméra C_j à la $k^{ème}$ pose. La fonction de coût que l'on cherche à minimiser est :

$$f = \frac{1}{2}\sum_{k=0}^{K}\sum_{i=1}^{M}\sum_{j=1}^{N}\boldsymbol{\epsilon}_{ij}^{k\top}\boldsymbol{\epsilon}_{ij}^{k} = \frac{1}{2}\boldsymbol{\epsilon}^{\top}\boldsymbol{\epsilon} \tag{5.37}$$

où $\boldsymbol{\epsilon}$ est le vecteur qui concatène les erreurs de reprojection $\boldsymbol{\epsilon}_{ij}^k$ (également appelés *résidus*). Une expression détaillée de la fonction de coût est donnée en annexe B.2. Sous l'hypothèse d'un bruit gaussien des points 2D détectés, l'algorithme proposé est un estimateur de maximum de vraisemblance (dans l'espace des images non-distordues, *cf* remarques en Section 2.1.2.3, p12).

6. Pour plus d'informations, le lecteur peut notamment lire [BCLF04] qui réécrit le modèle unifié (de la Section 2.1.3) avec l'algèbre géométrique conforme

7. Dans le cas d'un ajustement de faisceaux multi-caméra où les paramètres extrinsèques ΔT_j sont constants, l'échelle est fixée s'il existe une translation extrinsèques $\Delta\mathbf{t}_j$ non-nulle.

Chaque point 3D est potentiellement observable par n'importe quelle caméra (à différents instants), donc les permutations des scènes sont possibles. De plus, le processus de rejet de points aberrants (expliqué à la fin de la Section 5.3.3) est intégré dans l'ajustement de faisceaux multi-caméra.

5.4.3 Algorithme LM et équations normales

On utilise l'algorithme de Levenberg-Marquardt (détaillé dans [PFTV92, p.683], et présentée en pseudo-code par l'Algorithme 7) pour minimiser la fonction de coût f. Il s'agit d'une optimisation non-linéaire itérative, qui résout des problèmes de moindres carrés, en combinant les avantages des optimisations de type descente de gradient et de Gauss-Newton. Pour chaque

Algorithme 7: Algorithme de Levenberg-Marquardt

$\lambda = 0.001$ et Update = true
Evaluer la fonction de coût f avec les paramètres initiaux x : coût $= f(x)$.
tantque Le nombre maximal d'itérations n'est pas atteint **faire**
  si Update **alors**
    Update=false
    Calculer la matrice hessienne H (Equation 5.39) et le vecteur g (Equation 5.40).
  finsi
  Appliquer le facteur d'amortissement λ à H.
  Résoudre les équations normales (Equation 5.38), avec la hessienne amortie, pour déterminer l'incrément δ.
  Evaluer : coût$' = f(x + \delta)$.
  si coût$' <$ coût **alors**
    $x = x + \delta$
    si 0.9999 coût $<$ coût$'$ (*c.-à-d.* la convergence est trop lente) **alors**
      Fin de l'algorithme.
    finsi
    $\lambda = \frac{\lambda}{10}$; coût $=$ coût$'$ et Update = true
  sinon
    $\lambda = 10\ \lambda$
  finsi
fin tantque

itération de l'algorithme de Levenberg-Marquardt, on cherche à annuler les variations de la fonction de coût. Grâce à une approximation du premier ordre, on obtient les *équations normales* (5.38). Le but est de calculer l'incrément δ qui mettra à jour les paramètres à optimiser.

$$J^\top J\delta = J^\top \epsilon \Leftrightarrow H\delta = g \tag{5.38}$$

Où J est la jacobienne des résidus $\boldsymbol{\epsilon}$ par rapport aux paramètres optimisés, H est une approximation de la matrice hessienne (que l'on appellera simplement *matrice hessienne*) et g est le vecteur unicolonne tels que :

$$H = J^\top J \tag{5.39}$$

$$g = J^\top \boldsymbol{\epsilon} \tag{5.40}$$

Plus précisément, les termes diagonaux de la matrice hessienne sont multipliés par $(1 + \lambda)$, où λ est le facteur d'amortissement (initialisé à 10^{-3}).

Soit les matrices suivantes :

- A, la jacobienne des résidus par rapport aux poses T_1^k du système multi-caméra,
- B, la jacobienne des résidus par rapport aux points $\overline{P}_i$,
- C, la jacobienne des résidus par rapport aux paramètres extrinsèques ΔT_j.

Sans perte de généralité, on suppose que le nombre de points 3D est grand devant le nombre de poses, lui-même élevé devant le nombre de caméras, on ordonne les paramètres à optimiser tel que la matrice jacobienne soit définie telle que $J = [C|A|B]$.

Soit les blocs matriciels suivants :

$$U = A^\top A \text{(diagonale par blocs } 6 \times 6) \tag{5.41}$$

$$V = B^\top B \text{(diagonale par blocs } 3 \times 3) \tag{5.42}$$

$$Q = C^\top C \text{(diagonale par blocs } 6 \times 6) \tag{5.43}$$

$$W = A^\top B \tag{5.44}$$

$$E = C^\top A \tag{5.45}$$

$$F = C^\top B \tag{5.46}$$

$$g_\Delta = C^\top \epsilon \tag{5.47}$$

$$g_T = A^\top \epsilon \tag{5.48}$$

$$g_P = B^\top \epsilon \tag{5.49}$$

Exprimons les équations normales (5.38) à l'aide des blocs matriciels précédents :

$$\left(\begin{array}{cc|c} Q & E & F \\ E^\top & U & W \\ \hline F^\top & W^\top & V \end{array}\right) \left(\begin{array}{c} \delta_\Delta \\ \delta_T \\ \hline \delta_P \end{array}\right) = \left(\begin{array}{c} g_\Delta \\ g_T \\ \hline g_P \end{array}\right) \tag{5.50}$$

et simplifions l'expression en concaténant certains blocs (équation 5.51) :

$$\Leftrightarrow \left(\begin{array}{c|c} \bar{\mathbf{U}} & \bar{\mathbf{W}} \\ \hline \bar{\mathbf{W}}^\top & V \end{array}\right) \left(\begin{array}{c} \delta_{cam} \\ \hline \delta_P \end{array}\right) = \left(\begin{array}{c} g_{cam} \\ \hline g_P \end{array}\right) \tag{5.51}$$

avec les blocs $\bar{\mathbf{U}} \overset{def}{=} \begin{pmatrix} Q & E \\ E^\top & U \end{pmatrix}$, $\bar{\mathbf{W}} \overset{def}{=} \begin{pmatrix} F \\ W \end{pmatrix}$ et l'incrément δ_{cam} sur les paramètres extrinsèques et les poses : $\delta_{cam} \overset{def}{=} \begin{pmatrix} \delta_\Delta \\ \delta_T \end{pmatrix}$.

On rappelle, à titre comparatif, que les équations normales d'un ajustement de faisceaux classique (mono-caméra, ou multi-caméra avec paramètres extrinsèques fixes) sont de la forme suivante :

$$\begin{pmatrix} U & W \\ W^\top & V \end{pmatrix} \begin{pmatrix} \delta_T \\ \delta_P \end{pmatrix} = \begin{pmatrix} g_T \\ g_P \end{pmatrix} \tag{5.52}$$

Avec l'ajustement de faisceaux multi-caméra, on ajoute donc aux équations normales les lignes et colonnes liées aux paramètres extrinsèques ΔT_j (*cf* équations (5.50) et (5.52)).

5.4.4 Exploitation de la structure creuse

5.4.4.1 Des équations normales jusqu'au Système Caméras/Poses

Nous allons expliquer comment obtenir l'algorithme 8, qui détaille une itération du LM dans sa forme creuse.

Comme la structure des équations normales diffère des travaux précédents, nous proposons une méthode spécifique qui permet de résoudre les équations en tirant parti de la structure creuse et de la symétrie de H. Compte tenu de ses dimensions, au lieu de calculer l'inverse de H, on applique uniquement un complément de Schur à l'équation (5.51) en multipliant chaque membre par $\left(\begin{smallmatrix} I & -\bar{\mathbf{W}}V^{-1} \\ 0 & I \end{smallmatrix}\right)$;

$$\left(\begin{array}{c|c} \bar{\mathbf{U}} - \bar{\mathbf{W}}V^{-1}\bar{\mathbf{W}}^\top & 0 \\ \hline \bar{\mathbf{W}}^\top & V \end{array}\right) \begin{pmatrix} \delta_{cam} \\ \delta_P \end{pmatrix} = \begin{pmatrix} g_{cam} - \bar{\mathbf{W}}V^{-1}g_P \\ \hline g_P \end{pmatrix} \tag{5.53}$$

En conservant la partie supérieure de l'équation (5.53), on obtient alors les équations (5.54) que l'on appelle le *Système Caméras/Poses*, noté CPS (pour *Camera-Pose System*) :

$$\bar{\mathbf{S}}\delta_{cam} = g'_{cam} \Leftrightarrow \begin{pmatrix} \bar{\mathbf{A}} & \bar{\mathbf{B}} \\ \bar{\mathbf{B}}^\top & \bar{\mathbf{D}} \end{pmatrix} \begin{pmatrix} \delta_\Delta \\ \delta_T \end{pmatrix} = \begin{pmatrix} g'_\Delta \\ g'_T \end{pmatrix} \tag{5.54}$$

Avec la matrice $\bar{\mathbf{S}}$ et le vecteur unicolonne g'_{cam} définis par :

$$\begin{aligned} \bar{\mathbf{S}} \overset{def}{=} \begin{pmatrix} \bar{\mathbf{A}} & \bar{\mathbf{B}} \\ \bar{\mathbf{B}}^\top & \bar{\mathbf{D}} \end{pmatrix} &\overset{def}{=} \bar{\mathbf{U}} - \bar{\mathbf{W}}V^{-1}\bar{\mathbf{W}}^\top \\ &= \begin{pmatrix} Q & E \\ E^\top & U \end{pmatrix} - \begin{pmatrix} FV^{-1}F^\top & FV^{-1}W^\top \\ WV^{-1}F^\top & WV^{-1}W^\top \end{pmatrix} \end{aligned} \tag{5.55}$$

$$\begin{aligned} g'_{cam} \overset{def}{=} \begin{pmatrix} g'_\Delta \\ g'_T \end{pmatrix} &= g_{cam} - \bar{\mathbf{W}}V^{-1}g_P \\ &= \begin{pmatrix} g_\Delta \\ g_T \end{pmatrix} - \begin{pmatrix} F \\ W \end{pmatrix} V^{-1}g_P \end{aligned} \tag{5.56}$$

Les matrices $\bar{\mathbf{A}}$ (de taille $6(N-1) \times 6(N-1)$) et $\bar{\mathbf{B}}$ (de taille $6(N-1) \times 6(K+1)$) sont pleines, car au moins un point est vu à chaque pose clé. Une fois que le CPS est résolu (*cf*

Section 5.4.5), on dispose de l'incrément δ_{cam} des paramètres extrinsèques et des poses. Il ne reste plus qu'à calculer l'incrément δ_P sur les points 3D, qui s'obtient avec une substitution-avant (forward-substitution) avec l'équation (5.58).

En effet, comme V est diagonale par bloc, et donc aisément inversible, la partie inférieure de l'équation (5.53) permet d'écrite :

$$(5.53) \Rightarrow \bar{\mathbf{W}}^\top \delta_{cam} + V\delta_P = g_P \tag{5.57}$$

$$\begin{aligned} \Leftrightarrow \delta_P &= V^{-1}(g_P - \bar{\mathbf{W}}^\top \delta_{cam}) \\ &= V^{-1}\left(g_P - \begin{pmatrix} F^\top & W^\top \end{pmatrix}\begin{pmatrix} \delta_\Delta \\ \delta_T \end{pmatrix}\right) \\ &= V^{-1}\left(g_P - F^\top\delta_\Delta - W^\top\delta_T\right) \end{aligned} \tag{5.58}$$

5.4.4.2 Expression creuse des équations normales

Pour de nombreuses applications, comme la navigation combinant caméras et systèmes électroniques embarqués, il est nécessaire d'avoir des algorithmes d'étalonnage temps réel qui utilisent un minimum de ressources mémoire. Dans cette section, nous présentons une implémentation rapide de l'ajustement de faisceaux multi-caméra qui utilise à la fois la structure creuse et l'expression analytique de la matrice jacobienne des erreurs de reprojection en fonction des paramètres à optimiser.

Les blocs élémentaires A_{ij}^k, B_{ij}^k et C_{ij}^k de la matrice jacobienne J (*cf* Figure 5.11), désignent les dérivées partielles du résidu $\boldsymbol{\epsilon}_{ij}^k$ par rapport à respectivement, la $k^{ème}$ pose de la caméra maître, au $i^{ème}$ point et par rapport aux paramètres extrinsèques de la $j^{ème}$ caméra. Ces définitions sont données par les équations (5.59) :

$$A_{ij}^k = \frac{\partial \boldsymbol{\epsilon}_{ij}^k}{\partial T_1^k},\ B_{ij}^k = \frac{\partial \boldsymbol{\epsilon}_{ij}^k}{\partial \overline{P}_i},\ C_{ij}^k = \frac{\partial \boldsymbol{\epsilon}_{ij}^k}{\partial \Delta T_j} \tag{5.59}$$

A_{ij}^k, B_{ij}^k et C_{ij}^k sont respectivement des matrices 2×6, 2×3 et 2×6. Un détail complet du calcul des expressions analytiques de chaque bloc est disponible en annexe B.3.

Une implémentation creuse d'un ajustement de faisceaux classique est détaillée dans [HZ04, A.6.7], et étendue dans [MLD+09] au cas des caméras génériques (*cf* 3.1.1). Dans notre cas, où l'on utilise des caméras rigidement liées, les paramètres à optimiser diffèrent. Par conséquent, la structure de la matrice jacobienne change également.

Comme toute la matrice jacobienne J n'a pas besoin d'être calculée afin de résoudre les équations normales (5.38), les blocs de H et g sont calculés incrémentalement.

Structure primaire Tout comme dans [TMHF00], nous utilisons la notion de *structure primaire* pour représenter la matrice jacobienne et hessienne (Figures 5.11 et 5.12). Cette structure

illustre les blocs en supposant que chaque point 3D est observé par toutes les caméras dans toutes les poses.

On note A_i^k la jacobienne des résidus par rapport à la pose T_1^k pour toutes les caméras, et B_i^k, la jacobienne des résidus par rapport au point $\overline{P}_i$ pour toutes les caméras (équations 5.60). On retrouve ces matrices dans la Figure 5.11.

$$A_i^k = \begin{pmatrix} A_{i1}^k \\ \vdots \\ A_{ij}^k \\ \vdots \\ A_{iN}^k \end{pmatrix}, \; B_i^k = \begin{pmatrix} B_{i1}^k \\ \vdots \\ B_{ij}^k \\ \vdots \\ B_{iN}^k \end{pmatrix} \tag{5.60}$$

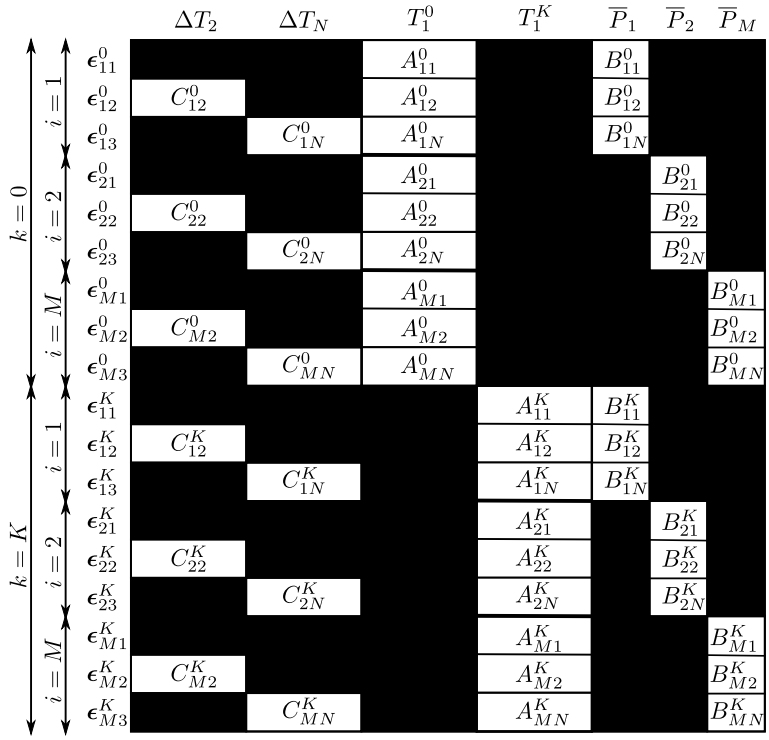


FIGURE 5.11 – Structure primaire de la matrice jacobienne J pour $N = 3$ caméras, $K + 1 = 2$ poses et $M = 3$ points. Les blocs non-nuls sont représentés en blanc.

Structure secondaire La structure primaire est utile pour des fins de représentations, mais en pratique, tous les points ne sont évidemment pas visibles à chaque instant dans toutes les caméras. C'est pourquoi la *structure secondaire* va chercher la structure creuse dans chaque bloc :

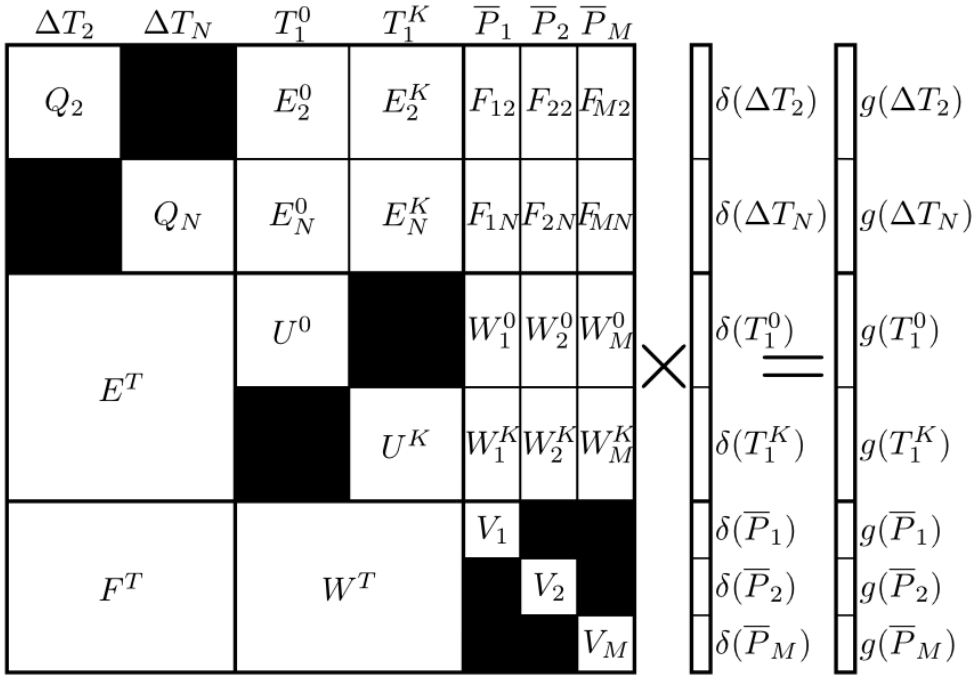


FIGURE 5.12 – Forme creuse des équations normales (5.38) de l'ajustement de faisceaux multi-caméra, pour $N = 3$ caméras, $K + 1 = 2$ poses et $M = 3$ points. La matrice hessienne $H = J^\top J$ est représentée par sa structure primaire.

si un point n'est pas visible par une caméra dans une pose donnée, alors les blocs correspondants sont nuls. Afin de tirer parti de la structure creuse, quelques index (*c.-à-d.* des ensembles d'indices) doivent être définis.

- Soit $\mathcal{I}_j$ l'index des points 3D qui sont vus au moins une fois par la $j^{ème}$ caméra durant toute la séquence.
- Soit $\mathcal{I}^k$ l'index des points 3D étant vus par le système multi-caméra pour la $k^{ème}$ pose.
- Soit $\mathcal{K}_i$ l'index des poses ayant vu le point $\overline{P}_i$.

$\mathcal{I}_j$ et $\mathcal{I}^k$ sont deux sous-ensembles de $[\![1..M]\!]$. Donc, $\mathcal{I}_j^k \overset{def}{=} \mathcal{I}_j \bigcap \mathcal{I}^k$ indexe le sous-ensemble des points 3D vus par la $j^{ème}$ caméra pour la $k^{ème}$ pose. $\mathcal{K}_i$ est un sous-ensemble de $[\![0..K]\!]$. Ces index sont illustrés par la Figure 5.13.

Pour obtenir la version la plus creuse possible de l'ajustement de faisceaux multi-caméra, il suffit de développer les équations précédentes en utilisant la structure secondaire. Les détails des calculs sont disponibles en annexe pour les équations normales (*cf* annexe B.4.1) et pour le Système Caméras/Poses (*cf* annexe B.4.2). Toutefois, l'implémentation de l'Algorithme 8 synthétise ces calculs utilisant la structure secondaire.

Algorithme 8: Ajustement de faisceaux multi-caméra (MCBA) : une itération de l'algorithme de Levenberg-Marquardt sous la forme creuse (structure secondaire)

Effacer les matrices.

pour tout point $\overline{P}_i$, vu par la caméra C_j à la $k^{ème}$ pose **faire**

Calculer les blocs élémentaires de la matrice jacobienne J : $A_{ij}^k \;\; B_{ij}^k \;\; C_{ij}^k$

Les blocs C_{ij}^k (et les blocs faisant intervenir C_{ij}^k) ne sont définis que pour $j \geq 2$.

Calculer les blocs des équations normales :

$$\begin{aligned} U^k &+= {A_{ij}^k}^\top A_{ij}^k & V_i &+= {B_{ij}^k}^\top B_{ij}^k \\ Q_j &+= {C_{ij}^k}^\top C_{ij}^k & W_i^k &+= {A_{ij}^k}^\top B_{ij}^k \\ E_j^k &+= {C_{ij}^k}^\top A_{ij}^k & F_{ij} &+= {C_{ij}^k}^\top B_{ij}^k \\ g(\Delta T_j) &+= {C_{ij}^k}^\top \boldsymbol{\epsilon}_{ij}^k & g(T_1^k) &+= {A_{ij}^k}^\top \boldsymbol{\epsilon}_{ij}^k \\ g(\overline{P}_i) &+= {B_{ij}^k}^\top \boldsymbol{\epsilon}_{ij}^k \end{aligned}$$

fin pour

Calculer les blocs intermédiaires :

$\forall j \in [\![2..N]\!],\ \forall i \in \mathcal{I}_j,\ Y_{ij} = F_{ij} {V_i}^{-1}$

$\forall k \in [\![0..K]\!],\ \forall i \in \mathcal{I}^k,\ Y_i^k = W_i^k {V_i}^{-1}$

Calculer les blocs du Système Caméras/Poses (CPS) :

$\forall j \in [\![2..N]\!],\ \bar{\mathbf{A}}_{jj} = Q_j$

$\forall j \in [\![2..N]\!],\ \forall k \in [\![0..K]\!],\ \bar{\mathbf{B}}_j^k = E_j^k$

$\forall k \in [\![0..K]\!],\ \bar{\mathbf{D}}^{kk} = U^k$

$\forall j_1 \in [\![2..N]\!], \forall j_2 \in [\![2..N]\!],\ j_1 \leq j_2, \forall i \in \mathcal{I}_{j_1} \bigcap \mathcal{I}_{j_2},\ \bar{\mathbf{A}}_{j_1 j_2} -= Y_{ij_1} {F_{ij_2}}^\top$

$\forall j \in [\![2..N]\!],\ \forall k \in [\![0..K]\!],\ \forall i \in \mathcal{I}_j \bigcap \mathcal{I}^k,\ \bar{\mathbf{B}}_j^k -= Y_{ij} {W_i^k}^\top$

$\forall k_1 \in [\![0..K]\!],\ \forall k_2 \in [\![0..K]\!],\ k_1 \leq k_2,\ \forall i \in \mathcal{I}^{k_1} \bigcap \mathcal{I}^{k_2},\ \bar{\mathbf{D}}^{k_1 k_2} -= Y_i^{k_1} {W_i^{k_2}}^\top$

$$\forall j \in [\![2..N]\!],\ g'(\Delta T_j) = g(\Delta T_j) - \sum_{i \in \mathcal{I}_j} Y_{ij}\, g(\overline{P}_i)$$

$$\forall k \in [\![0..K]\!],\ g'(T_1^k) = g(T_1^k) - \sum_{i \in \mathcal{I}^k} Y_i^k\, g(\overline{P}_i)$$

Calculer $\delta_{cam} = \left({\delta_\Delta}^\top {\delta_T}^\top\right)^\top$ à partir du CPS à l'aide d'une méthode de résolution creuse (*cf* Section 5.4.5).

Calculer l'incrément δ_P sur les points avec une substitution-avant :

$$\forall i \in [\![1..M]\!],\ \delta(\overline{P}_i) = (\delta_P)_i = {V_i}^{-1} \left(g(\overline{P}_i) - \sum_{\substack{j \in \mathcal{J}_i \\ j \geq 2}} {F_{ij}}^\top \delta(\Delta T_j) - \sum_{k \in \mathcal{K}_i} {W_i^k}^\top \delta(T_1^k) \right)$$

5.4.5 Résolution du Système Caméras/Poses

Le Système Caméra/Pose (5.54) est une équation matricielle linéaire, où $\bar{\mathbf{S}}$ est une matrice creuse symétrique de taille $6(K+1) \times 6(K+1)$ supposée définie positive. Nous proposons deux méthodes creuses pour le résoudre. La première est basée sur la factorisation de Cholesky (Section 5.4.5.1), la deuxième est le gradient conjugué préconditionné (Section 5.4.5.2).

5.4.5.1 Factorisation de Cholesky et complément de Schur

Comme expliqué dans [HZ04, p.614], si tous les points sont suivis sur au plus l_{max} poses consécutives, alors $WV^{-1}W^\top$ et $\bar{\mathbf{D}} = U - WV^{-1}W^\top$ sont des matrices bandes [8]. Dans ce cas, $\bar{\mathbf{S}}$ est donc une matrice flèche symétrique (*cf* Figure 5.14a).

Pour cette méthode, nous proposons de résoudre le Système Caméra/Pose avec un deuxième complément de Schur. On multiplie donc les deux membres de l'équation (5.54) par $\left(\begin{smallmatrix} I & -\bar{\mathbf{B}}\bar{\mathbf{D}}^{-1} \\ 0 & I \end{smallmatrix}\right)$:

$$(5.54) \Leftrightarrow \begin{pmatrix} \bar{\mathbf{A}}'' & 0 \\ \bar{\mathbf{B}}^\top & \bar{\mathbf{D}} \end{pmatrix} \begin{pmatrix} \delta_\Delta \\ \delta_T \end{pmatrix} = \begin{pmatrix} g''_\Delta \\ g'_T \end{pmatrix} \tag{5.61}$$

avec :

$$\begin{cases} \bar{\mathbf{A}}'' \stackrel{def}{=} \bar{\mathbf{A}} - \bar{\mathbf{B}}\bar{\mathbf{D}}^{-1}\bar{\mathbf{B}}^\top & (5.62a) \\ g''_\Delta \stackrel{def}{=} g'_\Delta - \bar{\mathbf{B}}\bar{\mathbf{D}}^{-1}g'_T & (5.62b) \end{cases}$$

$$(5.61) \Leftrightarrow \begin{cases} \bar{\mathbf{A}}''\delta_\Delta = g''_\Delta & (5.63a) \\ \delta_T = \bar{\mathbf{D}}^{-1}(g'_T - \bar{\mathbf{B}}^\top\delta_\Delta) & (5.63b) \end{cases}$$

Remarque : Notons que ce n'est par hasard que l'on a choisi le complément de Schur triangulaire par bloc supérieur. En effet, avec l'autre complément de Schur (triangulaire inférieur par bloc), il aurait fallu inverser une matrice pleine de taille $6(K+1) \times 6(K+1)$ et donc perdre l'intérêt de la structure creuse que l'on cherche à exploiter.

Comme $\bar{\mathbf{A}}''$ est une matrice pleine et symétrique, on calcule δ_Δ (équation (5.63a)) à l'aide d'une factorisation de Cholesky de $\bar{\mathbf{A}}''$. De plus, on ne calcule pas directement l'inverse de $\bar{\mathbf{D}}$, mais on utilise une factorisation de Cholesky pour les matrices bandes symétriques : $\bar{\mathbf{D}} = \bar{\mathbf{R}}^\top\bar{\mathbf{R}}$, où $\bar{\mathbf{R}}$ est une matrice triangulaire supérieure. Les vecteurs unicolonnes $\mathbf{x}$ vérifiant $\mathbf{x} = \bar{\mathbf{D}}^{-1}\mathbf{b}$ (*c.-à-d.* $\bar{\mathbf{R}}^\top\bar{\mathbf{R}}\mathbf{x} = \mathbf{b}$) sont calculés en résolvant $\bar{\mathbf{R}}^\top\mathbf{y} = \mathbf{b}$, puis $\bar{\mathbf{R}}\mathbf{x} = \mathbf{y}$. On peut donc calculer les expressions où $\bar{\mathbf{D}}^{-1}$ intervient en prenant comme vecteur $\mathbf{b}$ les $6(N-1)$ colonnes de $\bar{\mathbf{B}}^\top$ (équation 5.62a), le vecteur g'_T (équation 5.62b) et le vecteur $g'_T - \bar{\mathbf{B}}^\top\delta_\Delta$ (équation 5.63b).

Comme d'ordinaire pour les ajustements de faisceaux classiques, on suppose que les matrices $\bar{\mathbf{A}}''$ et $\bar{\mathbf{D}}$ sont définies positives, ce qui est garanti par le facteur d'amortissement de l'algorithme Levenberg-Marquardt. Enfin, une substitution-avant est appliquée pour obtenir δ_T

8. Le nombre de diagonales non-nulles au dessus de la diagonale principale est de $6l_{max} - 1$

(avec (5.63b)). δ_Δ et δ_T étant déterminés, le CPS est résolu. Le lecteur peut se référer à l'équation (5.58) qui permet de trouver le dernier incrément δ_P et ainsi de résoudre les équations normales.

Cette méthode est spécifiquement adaptée aux cas où la reconstruction 3D s'effectue le long d'un parcours (sans détection de fermeture de boucle), mais aussi au cas où les caméras se déplacent en observant le même ensemble de points (cas où la matrice $\bar{\mathbf{D}}$ est dense). Dans les autres cas (*c.-à-d.* parcours avec détection de fermeture de boucle), la solution proposée ci-dessus fonctionne. Elle sera simplement moins rapide, car non spécifique aux données. En effet, la matrice $\bar{\mathbf{D}}$ stocke un nombre de diagonales proportionnel à l_{max} qui est bien plus grand que l (le nombre moyen de poses ayant vu un point 3D) en cas de fermeture de boucle. Le nombre de valeurs nulles de $\bar{\mathbf{D}}$ stockées et traitées devient trop important dans ces cas. C'est pourquoi la méthode suivante (section 5.4.5.2) ne présente pas cet inconvénient et utilise la structure secondaire des matrices pour résoudre le Système Caméra/Pose (5.54).

5.4.5.2 Gradient conjugué préconditionné

Pour les séquences le long de parcours où des fermetures de boucles sont détectées, de nombreux blocs non-nuls apparaissent dans la matrice $\bar{\mathbf{D}}$. Par conséquent, la résolution précédente du CPS, qui utilise une factorisation de Cholesky pour une matrice bande (Section 5.4.5.1), n'est plus adaptée pour de longues séquences.

Afin de résoudre directement le CPS (5.54), nous utilisons l'algorithme du gradient conjugué préconditionné (PCG pour *Preconditioned Conjugate Gradient*), présenté dans [NW99] et rappelé par l'Algorithme 9. En effet, d'après l'étude comparative [JNS+10] sur des méthodes de résolution normales pour un ajustement de faisceaux classique, le PCG est le meilleur compromis. Bien que ses performances soient moins élevées qu'un algorithme dédié pour de petites séquences (ce qui reste tout de même acceptable), il a de bonnes performances avec des données importantes. Cette méthode itérative est bien adaptée à la résolution de systèmes linéaires creux de grande dimension du type $Ax = b$, où $A \in \mathbb{R}^{n\times n}$ est une matrice symétrique définie positive, $x \in \mathbb{R}^n$ et $b \in \mathbb{R}^n$ sont des vecteurs unicolonnes. Le Gradient Conjugué s'interprète comme la minimisation de la fonctionnelle $g(x) = \frac{1}{2}x^\top Ax - b^\top x$. A chaque itération, la direction d'optimisation $p^{it+1} \in \mathbb{R}^n$ (appelée *direction conjuguée*) est choisie comme une combinaison linaire entre la direction précédente p^{it} et la direction du gradient de g telle que les directions soient conjuguées (*c.-à-d.* ${p^{it+1}}^\top A p^{it} = 0$). La convergence est en théorie assurée avant un maximum de n itérations. Le préconditionement permet d'accélérer la convergence.

Du point de vue de l'implémentation, l'un des avantages du PCG est qu'il nécessite peu de mémoire vive, car les variables (comme les directions du gradients) sont mises à jour à chaque itération. Le coût calculatoire est principalement dû à la résolution du système linéaire $My^{it+1} = r^{it+1}$ et aux produits Ap^{it} et ${p^{it}}^\top (Ap^{it})$.

Nous avons donc implémenté nos propres structures de données et codé l'algorithme PCG pour résoudre le CPS (5.54). Il suffit donc d'appliquer le PCG avec $A = \bar{\mathbf{S}}$ et $b = g'_{cam}$. Pour chaque itération du PCG, on calcule donc $\bar{\mathbf{S}}p$ et $p^\top \bar{\mathbf{S}}p$ (on ôte l'exposant it pour alléger les

Algorithme 9: Algorithme de du gradient conjugué préconditionné (PCG), pour déterminer x qui minimise $\frac{1}{2}x^\top Ax - b^\top x$.

Entrées : A,b,x^0 et le préconditionneur M
Sorties : x^{it+1}

$r^0 = Ax^0 - b$
Résoudre $My^0 = r^0$
$p^0 = -y^0$
pour $it = 0;\ it < n;\ it++$ **faire**
  $\alpha^{it} = \frac{{r^{it}}^\top y^{it}}{{p^{it}}^\top A p^{it}}$
  $x^{it+1} = x^{it} + \alpha^{it} p^{it}$
  $r^{it+1} = r^{it} + \alpha^{it} A p^{it}$
  si La fonctionnelle a suffisamment diminuée (*c.-à-d.* ${r^{it+1}}^\top r^{it+1} \leq \epsilon\ {r^0}^\top r^0$) **alors**
    Fin de l'algorithme.
  finsi
  Résoudre $My^{it+1} = r^{it+1}$
  $\beta^{it+1} = \frac{{r^{it+1}}^\top y^{it+1}}{{r^{it}}^\top y^{it}}$
  $p^{it+1} = -y^{it+1} + \beta^{it+1} p^{it}$
fin pour

équations (5.64) et (5.65)) :

$$\bar{\mathbf{S}} p = \begin{pmatrix} \bar{\mathbf{A}} & \bar{\mathbf{B}} \\ \bar{\mathbf{B}}^\top & \bar{\mathbf{D}} \end{pmatrix} \begin{pmatrix} p_\Delta \\ p_T \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{A}} p_\Delta + \bar{\mathbf{B}} p_T \\ \bar{\mathbf{B}}^\top p_\Delta + \bar{\mathbf{D}} p_T \end{pmatrix} \tag{5.64}$$

$$p^\top \bar{\mathbf{S}} p = p_\Delta^\top \bar{\mathbf{A}} p_\Delta + p_\Delta^\top \bar{\mathbf{B}} p_T + p_T^\top \bar{\mathbf{B}}^\top p_\Delta + p_T^\top \bar{\mathbf{D}} p_T \tag{5.65}$$

Là encore, tout comme pour un ajustement de faisceaux classique, on suppose que la matrice $\bar{\mathbf{S}}$ est définie positive (ce qui est de nouveau garanti par le facteur d'amortissement du LM).

Pour la matrice $\bar{\mathbf{D}}$, nous ne gardons que les blocs non-nuls de sa partie triangulaire supérieure à l'aide d'un stockage compressé des blocs en ligne (*block row compressed storage*).

En pratique, on remarque qu'un préconditionneur est nécessaire pour assurer la convergence de l'algorithme à cause du conditionnement du CPS. Comme suggéré dans [JNS+10], nous utilisons le préconditionneur par bloc de Jacobi : la matrice diagonale par blocs 6×6 dont les blocs sont les blocs diagonaux de $\bar{\mathbf{S}}$ (pour l'inverser, on calcule l'inverse de chaque bloc avec une factorisation de Cholesky).

Pour le critère d'arrêt du PCG, nous utilisons une variante de la diminution relative des résidus proposée dans [JNS+10]. Le critère est :

$${r_\delta^{it+1}}^\top r_\delta^{it+1} \leq \epsilon\ {r_\delta^1}^\top r_\delta^1 \ \text{ et } \ {r_T^{it+1}}^\top r_T^{it+1} \leq \epsilon\ {r_T^1}^\top r_T^1 \tag{5.66}$$

où $\epsilon = 10^{-8}$. r_δ^{it} et r_T^{it} sont respectivement les résidus des paramètres extrinsèques et des poses après la $it^{ème}$ itération.

5.4.6 Complexité d'une itération du LM

On souhaite calculer la complexité d'une itération du Levenberg-Marquardt de l'ajustement de faisceaux multi-caméra (MCBA). On déclinera le calcul pour les deux méthodes précédentes de résolution des équations normales.

Cette complexité dépend naturellement des données que l'on utilise. C'est pourquoi il est nécessaire de commencer par la définition des variables suivantes :

- μ est le nombre moyen de points 3D points vus au moins une fois par une caméra. Pour des caméras à champs recouvrants : $\mu = M$. Pour des caméras à champs disjoints et des points 3D équitablement répartis : $\mu \simeq \frac{M}{N}$, sauf dans le cas de permutation des scènes (*cf* 5.2.2.3) ou de fermeture de boucle 5.3.4 où $\frac{M}{N} \lesssim \mu \lesssim M$.
- ρ le nombre moyen de points 3D vus par le système multi-caméra pour une pose donnée.
- l est le nombre moyen de poses clés ayant vu un même point 3D (*c.-à-d.* : si aucune fermeture de boucle n'est appliquée, l est la longueur moyenne du suivi d'un point 3D).
- l_{max} est le nombre maximal de poses clés séparant deux observations d'un même point 3D. Ce nombre augmente sensiblement en cas de fermeture de boucle.

Mathématiquement, ces définitions correspondent aux équations (5.67), (5.68), (5.69) et (5.70) :

$$\mu = \frac{1}{N}\sum_{j=1}^{N} card(\mathcal{I}_j) \in [\![0..M]\!] \tag{5.67}$$

$$\rho = \frac{1}{K+1}\sum_{k=0}^{K} card(\mathcal{I}^k) \in [\![0..M]\!] \tag{5.68}$$

$$l = \frac{1}{M}\sum_{i=1}^{M} card(\mathcal{K}_i) \in [\![0..K+1]\!] \tag{5.69}$$

$$l_{max} = \max_{i \in [\![1..M]\!]} (\max(\mathcal{K}_i) - \min(\mathcal{K}_i)) \ \leq K+1 \tag{5.70}$$

Remarque : En comptant les blocs non-nuls de la matrice W, on obtient la relation suivante :

$$Ml = (K+1)\rho \tag{5.71}$$

Pour la suite des calculs, on considère que les ensembles $\mathcal{I}_j$, $\mathcal{I}^k$ et $\mathcal{K}_i$ (définis dans la section 5.4.4.2) ont un cardinal respectivement constant à μ, ρ et l. Le lecteur peut se référer aux équations utilisant la structure creuse secondaire détaillée en Annexe B.4 afin de saisir en détail les calculs suivants. Le tableau 5.2 récapitule l'ensemble des index utilisés par le MCBA et la valeur moyenne de leur cardinal. La Figure 5.13 illustre l'utilisation des différents index. Le tableau 5.3 récapitule l'ensemble des calculs des complexités suivantes.

Index sur les	points			poses		caméras	
Index	$\mathcal{I}_j$	$\mathcal{I}^k$	$\mathcal{I}_j^k$	$\mathcal{K}_i$	$\mathcal{K}_{ij}$	$\mathcal{J}_i$	$\mathcal{J}_i^k$
Moyenne du cardinal	$\mu = M\frac{\eta}{N}$	ρ	$\iota = \rho\frac{\eta}{N}$	l	l	η	η

TABLE 5.2 – Récaptitulatif des index utilisés par le MCBA et la valeur moyenne de leur cardinal.

On note que les index $\mathcal{K}_i$ et $\mathcal{K}_{ij}$ ont le même cardinal, car un même point n'est pas successivement traqué dans des caméras différentes (dans le cas contraire, on aurait simplement $\mathrm{card}(\mathcal{K}_{ij}) \leq l$). De même, les index $\mathcal{J}_i$ et $\mathcal{J}_i^k$ ont en moyenne le même cardinal (sauf dans le cas de permutation de scène pour des caméras à champs disjoints, on a $\mathrm{card}(\mathcal{J}_i) \geq \mathrm{card}(\mathcal{J}_i^k)$). Pour être parfaitement rigoureux, on aurait pu définir un coefficient de ré-observation des points 3D (dans différentes caméras). Mais le calcul de la complexité est déjà bien assez compliqué...

Comme les blocs de la structure secondaire de $J^\top J$ sont calculés incrémentalement, la complexité du calcul de la hessienne est en $O(N_r)$, où $N_r = \rho(K+1)\eta$ est le nombre total de reprojection prises en compte. Les étapes les plus coûteuses sont le calcul de $WV^{-1}W^\top$ et la résolution du CPS.

Comme expliqué dans [MLD+09], W a $\rho(K+1)$ blocs non-nuls et V est diagonal par blocs, la complexité du produit $WV^{-1}W^\top$ est $O(\rho(K+1)^2)$. De même, pour $FV^{-1}W^\top$ la complexité du produit est $O(\iota(N-1)(K+1))$. La complexité de $FV^{-1}F^\top$ vaut seulement $O(\mu(N-1)^2)$. $\bar{\mathbf{W}}^\top\delta_{cam}$ et $\bar{\mathbf{W}}V^{-1}g_P$ ont la même complexité : $O(\mu(N-1)+\rho(K+1))$. Le produit faisant intervenir V^{-1} dans l'équation (5.58) a une complexité de $O(M)$.

Factorisation de Cholesky et complément de Schur La complexité d'une factorisation dense de Cholesky de $\bar{\mathbf{D}}$ est de $O((K+1)^3)$, mais si $\bar{\mathbf{D}}$ est une matrice bande, la complexité est réduite à $O(l_{max}(K+1)^2)$. La résolution des systèmes de la forme $\bar{\mathbf{R}}^\top\bar{\mathbf{R}}\mathbf{x} = \mathbf{b}$ (où $\mathbf{x}$ et $\mathbf{b}$ sont des vecteurs colonnes de $3(K+1)$ éléments) a une complexité de $O(l_{max}(K+1))$ (comme pour l'équation 5.63b). Par conséquent, les complexités des produits $\bar{\mathbf{D}}^{-1}\bar{\mathbf{B}}^\top$ et $\bar{\mathbf{B}}\bar{\mathbf{D}}^{-1}\bar{\mathbf{B}}^\top$ sont respectivement $O(l_{max}(N-1)(K+1))$ et $O(l_{max}(N-1)(K+1)^2)$ (équation 5.62a).

PCG Soit $N_{pcg} \leq 6(N+K)$ le nombre moyen d'itérations du PCG. La complexité d'un PCG dense est $O(N_{pcg}(K+1)^2)$, mais si $\bar{\mathbf{S}}$ est creux, la complexité diminue à $O(N_{pcg}l(K+1))$.

Au final, en utilisant le fait que $N << K$, la complexité d'une itération de l'ajustement multi-caméra (MCBA) est de :

- $O(M+(\rho+l_{max})(K+1)+(\rho+l_{max})(K+1)^2)$ pour la méthode basée sur la factorisation de Cholesky,
- $O(M+(\rho+N_{pcg}l)(K+1)+\rho(K+1)^2)$ pour la méthode basée sur le PCG.

Plus l_{max} est grand devant l, plus le PCG est rapide par rapport à la méthode basée sur la factorisation de Cholesky (ce qui est le cas pour des séquences bouclées). Il était donc pertinent

	Type de blocs ou d'opérations	Nombre de blocs ou d'opérations	Nombre maximal d'opérations par bloc	Complexité
Equation normale	U^k	$K+1$	$\mu\eta$	$O(N_r)$
	V_i	M	$l\eta$	$O(N_r)$
	Q_j	$N-1$	$(K+1)\iota = \mu l$	$O(N_r)$
	W_i^k	$(K+1)\mu$	η	$O(N_r)$
	E_j^k	$(N-1)(K+1)$	ι	$O(N_r)$
	F_{ij}	$(N-1)\mu$	l	$O(N_r)$
	$g(\Delta T_j)$	$N-1$	μl	$O(N_r)$
	$g(T_1^k)$	$K+1$	$\rho\eta$	$O(N_r)$
	$g(\bar{P}_i)$	M	$l\eta$	$O(N_r)$
CPS	$\bar{\mathbf{D}}^{k_1k_2}$ et $(WV^{-1}W^\top)^{k_1k_2}$	$\frac{(K+1)(K+2)}{2}$	ρ	$O(\rho(K+1)^2)$
	$\bar{\mathbf{B}}_j^k$ et $(FV^{-1}W^\top)_j^k$	$(N-1)(K+1)$	ι	$O(\iota(N-1)(K+1))$
	$\bar{\mathbf{A}}_{j_1j_2}$ et $(FV^{-1}F^\top)_{j_1j_2}$	$\frac{(N-1)N}{2}$	μ	$O(\mu(N-1)^2)$
	$(g'_\Delta)_j$ et $(FV^{-1}g_P)_j$	$N-1$	μ	$O(\mu(N-1))$
	$(g'_T)^k$ et $(WV^{-1}g_P)^k$	$K+1$	ρ	$O(\rho(K+1))$
	$(F^\top\delta_\Delta)_i$	M	η	$O(\eta M) = O(\mu(N-1))$
	$(W^\top\delta_T)_i$	M	l	$O(Ml)$
Cholesky	Factoriser $\bar{\mathbf{D}} = \bar{\mathbf{R}}^\top\bar{\mathbf{R}}$	$l_{max}(K+1)$	$(K+1)$	$O(l_{max}(K+1)^2)$
	Résoudre $\bar{\mathbf{R}}^\top\bar{\mathbf{R}}\mathbf{x} = \mathbf{b}$			
	pour $\mathbf{b} = (g'_T - \bar{\mathbf{B}}^\top\delta_\Delta)$	1	$l_{max}(K+1)$	$O(l_{max}(K+1))$
	Calculer $\bar{\mathbf{D}}^{-1}\bar{\mathbf{B}}^\top$	$6(N-1)$	$l_{max}(K+1)$	$O(l_{max}(N-1)(K+1))$
	Calculer $\bar{\mathbf{B}}\bar{\mathbf{D}}^{-1}\bar{\mathbf{B}}^\top$	$6(N-1)(K+1)$	$l_{max}(K+1)$	$O(l_{max}(N-1)(K+1)^2)$
PCG	Résoudre $My^{it+1} = r^{it+1}$	$N+K$	$N_{pcg}6^3$	$O(N_{pcg}(N+K))$
	$p_\Delta^\top\mathbf{A}p_\Delta$	$(N-1)^2$	N_{pcg}	$O(N_{pcg}(N-1)^2)$
	$p_\Delta^\top\mathbf{B}p_T$	$(N-1)(K+1)$	N_{pcg}	$O(N_{pcg}(N-1)(K+1))$
	$p_T^\top\mathbf{D}p_T$	$l(K+1)$	N_{pcg}	$O(N_{pcg}l(K+1))$

TABLE 5.3 – Complexité de chaque étape d'une itération du LM du MCBA implémenté avec la structure creuse secondaire.

de proposer le PCG, qui sera alors plus efficace que la factorisation de Cholesky pour des matrices bandes.

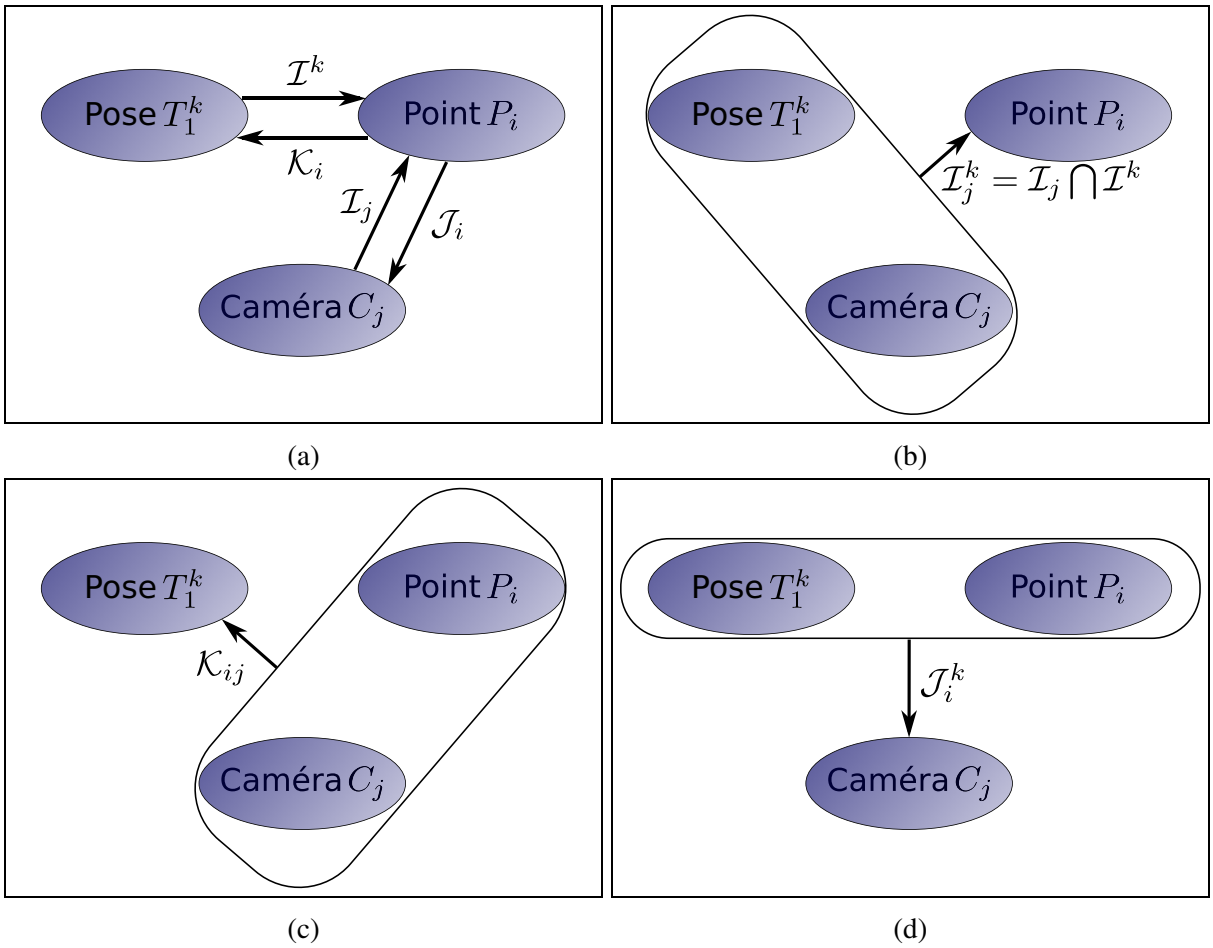


FIGURE 5.13 – Illustration des différents index utilisés entre les poses du système multi-caméra, les points 3D et les caméras. (a) Index à une entrée (les points $\{P_i\}_{i\in\mathcal{I}^k}$ sont vus à la $k^{ème}$ pose, les caméras $\{C_j\}_{j\in\mathcal{J}_i}$ ont vu le point P_i, ...). (b)(c)(d) Index à deux entrées (les points $\{P_i\}_{i\in\mathcal{I}_j^k}$ sont vus par la caméra C_j à la $k^{ème}$ pose,...).

5.4.7 Comparaison des implémentations

Nous allons montrer l'intérêt des implémentations creuses du MCBA, pour les deux protocoles expérimentaux que nous avons proposés.

5.4.7.1 Manœuvres

Tout d'abord, nous allons étudier l'apport d'une implémentation creuse en fonction du nombre de points 3D (qui peut rapidement être plus élevé que le nombre de poses). Pour cela, on génère des données synthétiques avec $N = 2$ caméras à champs non-recouvrants. On génère 30 poses aléatoirement avec un mouvement $3D$. Une scène S_j est créée devant chaque caméra C_j. Les points 3D sont ensuite projetés sans bruit afin d'avoir une vérité terrain. Pour estimer le gain obtenu grâce à une implémentation creuse, nous mesurons le temps moyen d'exécution d'une itération de l'algorithme de Levenberg-Marquardt du MCBA dans trois cas (*cf* Tableau 5.4) :

1. une implémentation naïve et dense : J est calculé analytiquement, puis on calcule $J^\top J$ et $J^\top \boldsymbol{\epsilon}$.
2. une implémentation dense : $J^\top J$ et $J^\top \boldsymbol{\epsilon}$ sont calculés analytiquement.
3. l'implémentation creuse utilisant la structure secondaire et la factorisation de Cholesky (*cf* Section 5.4.5.1) pour résoudre les équations normales.

Pour les deux premiers cas, les équations normales (5.38) sont résolues directement avec une factorisation de Cholesky pour les matrices denses. Toutes les structures de données ont été allouées avant le début des mesures.

Nombre de points (M)	Proportion de valeurs nulles de H	Cas 1 τ_1	Cas 2 τ_2	Cas 3 τ_3	Gain $\frac{\tau_1}{\tau_3}$	Gain $\frac{\tau_2}{\tau_3}$
38	49%	0,5 s	0,015 s	0,015 s	33	1
327	73%	51,0 s	0,37 s	0,07 s	728	7,4

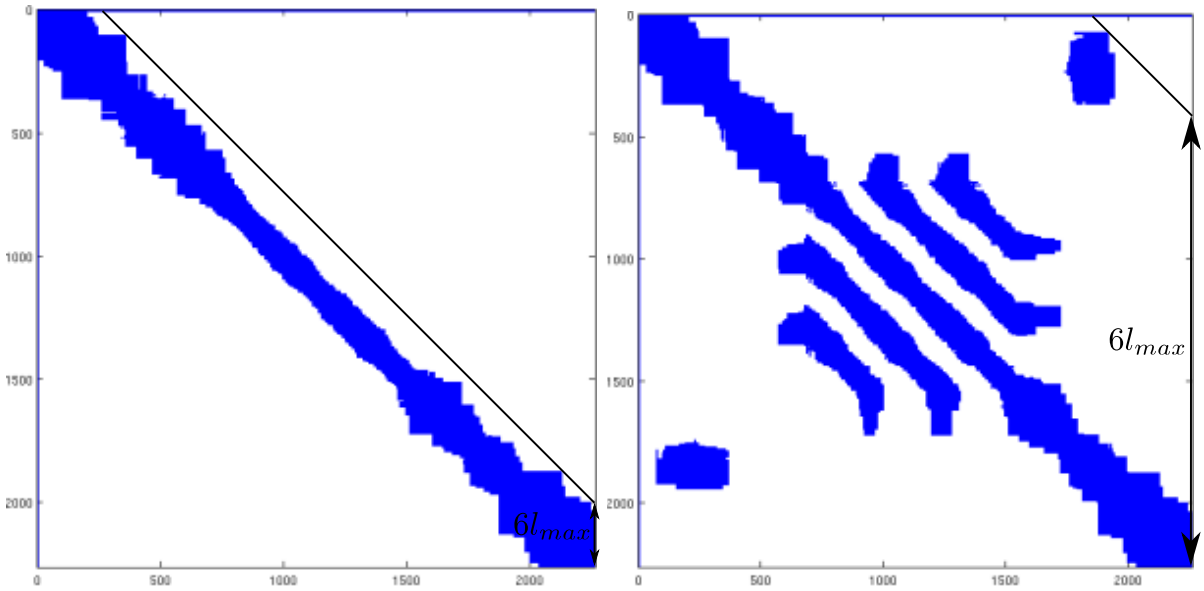
TABLE 5.4 – Gains en temps de calcul d'une implémentation creuse pour une itération du MCBA en fonction du nombre de points 3D et du nombre de zéros de la hessienne.

Les résultats du Tableau 5.4 montrent la diminution du temps de calcul qu'apporte une méthode creuse utilisant la structure secondaire. Stocker entièrement la matrice jacobienne n'est bien entendu pas recommandé. Le gain obtenu avec la méthode creuse (cas n°3) est satisfaisant pour permettre l'utilisation d'un nombre important de points 3D. On note que plus $N << K$ et $N << M$, plus la proportion de zéros de la matrice hessienne augmente. Lors des résultats expérimentaux pour la procédure de manœuvre (*cf* 5.5.1.2), H a environ 51% de valeurs nulles.

5.4.7.2 Parcours

L'algorithme de reconstruction est exécuté avec $(K+1) = 376$ poses clés, choisies sur un parcours d'environ 300 mètres. Environ 57000 points 3D sont reconstruits, avec environ 255000 amers 2D inliers. Sans fermeture de boucle, la matrice $\bar{\mathbf{S}}$ utilisée dans le CPS 5.54 et définie par l'équation 5.55) est une matrice flèche (*cf* Figure 5.14a). Après la fermeture de boucle, des blocs non-nuls apparaissent (*cf* Figure 5.14b). Compte tenu de la taille de cette matrice, les méthodes denses ne sont pas techniquement utilisables (trop d'espace mémoire est nécessaire). Nous comparons donc les deux méthodes creuses de résolutions du Système Camera/Pose.

La résolution du CPS basée sur la factorisation de Cholesky pour les matrices bandes dure en moyenne environ 3 s. La résolution du CPS avec le PCG converge en moyenne en 0.6 s. On rappelle que le PCG converge au maximum en $6(N-1)+6(K+1) = 2262$ itérations internes (c'est le nombre de colonnes de $\bar{\mathbf{S}}$). Ici, le PCG a convergé en moyenne en $N_{pcg} = 309$ itérations internes d'environ 2 ms chacune. Au final, l'algorithme de Levenberg-Marquardt du MCBA a convergé en 110 itérations pendant 4min 12s avec la méthode du PCG. La méthode basée sur Cholesky est environ 2.2 fois plus lente. Cet écart est d'autant plus grand que le nombre de poses augmente et que l est grand par rapport à l_{max} (comme l'a montré le calcul de complexité).



(a) (b)

FIGURE 5.14 – Matrice $\bar{\mathbf{S}}$ pour une séquence de 376 poses clés, dans le cas d'une reconstruction sans fermeture de boucle (a) ou bien dans le cas de fermeture de boucle (b). Les éléments non nuls sont représentés en bleu. Les $6(N-1)$ premières lignes et premières colonnes sont non nulles.

5.5 Résultats

Cette section présente les résultats des étalonnages extrinsèques multi-caméra en mouvement basés sur l'ajustement de faisceaux multi-caméra MCBA. On présente tout d'abord les résultats de la procédure d'étalonnage utilisant des manœuvres et les cibles (Section 5.5.1.1), puis de la procédure basée SFM (Section 5.5.2). Une fois le système multi-caméra étalonné, la Section 5.5.3 illustrera des reconstructions qu'il est possible d'obtenir.

5.5.1 Manœuvres et cibles

Nous présentons ici des résultats synthétiques et réels validant la procédure d'étalonnage de la Section 5.2.

5.5.1.1 Résultats avec des données synthétiques

Le protocole expérimental est le suivant, pour $N = 2$ caméras. 10 poses du système multi-caméra sont générées avec un mouvement 3D sans demi-tour. Les points de la scène S_j sont créés devant chaque caméra C_j. Cette vérité terrain sert à calculer les points images 2D. Puis, un bruit additif gaussien d'écart type σ leur est appliqué. Les mesures de distance entre des points 3D (utilisées pour fixer le facteur d'échelle) sont également soumises à un bruit additif gaussien d'écart type 0.5 mm. Enfin, l'algorithme est testé à partir des données bruitées. Pour cette expérience, la transformation ΔT_2 correspond à la translation (0.1m 0.1m -2m) et à la rotation déterminée par les angles de roulis, tangage, lacet de $(-179°, -4°, 171°)$.

Pour chaque valeur de σ, 10 mesures ont été faites. La précision de l'étalonnage extrinsèque est mesurable à la fois par la norme $\|dT\|$ de la différence entre la translation réelle et estimée (et aussi par $\|dT\|/\|\Delta T_2\|$ exprimé comme un pourcentage de la distance séparant les caméras), et par l'erreur angulaire $dR = dR_2$. Plus précisément :

$$dR_j = d(\widehat{\Delta R_j}, \Delta R_j) = \arccos\left(\frac{trace(\widehat{\Delta R_j}^{\,T} \Delta R_j) - 1}{2}\right) \tag{5.72}$$

où ΔR_j est la rotation parfaite et $\widehat{\Delta R_j}$ est la rotation estimée. La Figure 5.15 montre les performances de l'algorithme, avec une scène de 11 points 3D (contenus dans un volume d'environ $1.5\ m$ par $0.5\ m$ par $0.5\ m$) pour S_1 et S_2 à une distance d'environ $1\ m$ de chaque caméra C_j. Dans le cas d'un mouvement 3D, l'algorithme proposé dans [EWK07a] donne les mêmes résultats que notre initialisation linéaire (à des erreurs de quantification près). L'ajustement de faisceaux multi-caméra améliore les deux méthodes précédentes (l'estimation de la translation et de la rotation sont environ respectivement quatre et trois fois plus précises).

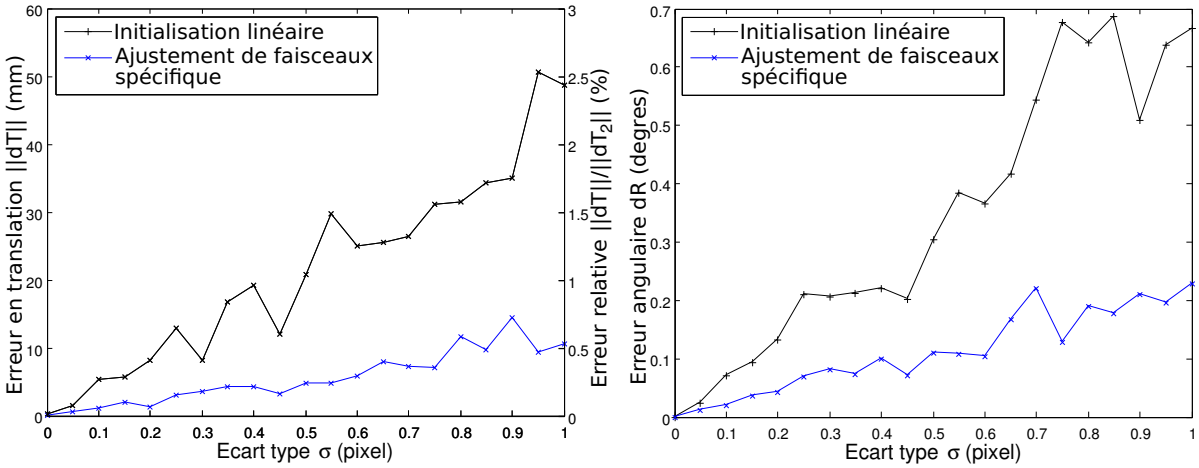


FIGURE 5.15 – Résultats avec des données synthétiques : erreur d'étalonnage extrinsèque en fonction du niveau de bruit. Avec des données réelles utilisant les cibles, l'écart type est en dessous de 0.1 pixel.

5.5.1.2 Résultats avec des données réelles

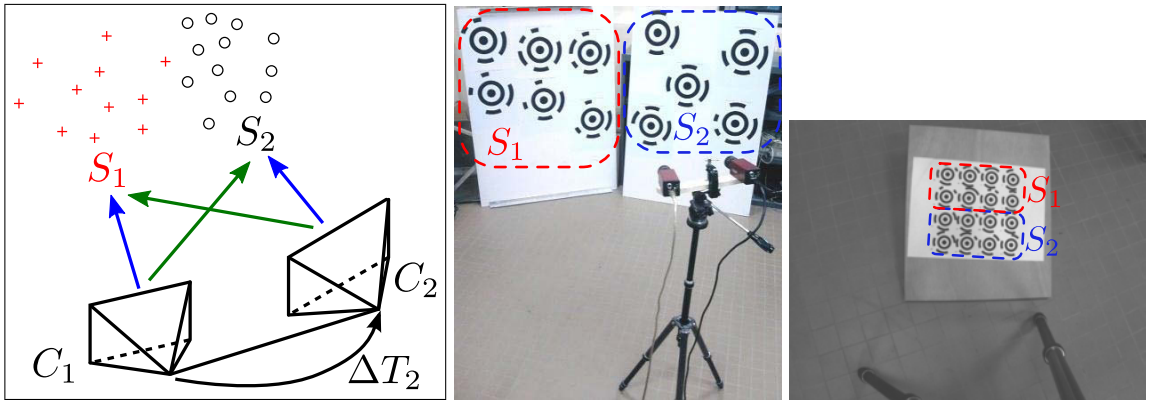
Dans un premier temps, la précision de l'étalonnage proposé est démontrée avec une paire stéréo dont les champs sont recouvrants (mais avec des ensembles disjoints de points 3D observés indépendamment par chaque caméra). Puis, la faisabilité de l'approche est illustrée à l'aide de caméras à champs non-recouvrants embarquées sur un véhicule. Pour les amers visuels de la scène, nous n'utilisons pas de points d'intérêt mais les cibles circulaires présentées dans la Section 2.2.1 (*cf* Figure 5.16). Elles sont détectées avec une précision sous-pixellique. Le détecteur trouve automatiquement le centre de la cible et son label. Par conséquent, les correspondances entre les images sont effectuées automatiquement. La résolution des images est 1600x1200. La distance focale des caméras est de $5.6\ mm$ et la taille d'un pixel est de $4.4\ \mu m$.

5.5.1.3 Paire stéréo

Dans ce paragraphe, une paire stéréo à champs recouvrant est étalonnée sous l'hypothèse de champs non-recouvrants (*c.-à-d.* chaque caméra observe uniquement sa propre scène), ce qui permet d'avoir une vérité terrain.

Afin d'étudier la précision et la répétabilité de l'algorithme, deux tableaux récapitulent quantitativement les résultats :

– Le tableau 5.5 montre la précision des étalonnages en fonction de l'hypothèse de champs de vue recouvrants, des mouvements (plans ou 3D) et de la permutation de scènes. Ils sont comparés à une référence : un étalonnage classique, dont les champs de vue sont recouvrants, en appliquant à la paire stéréo des mouvements 3D.



(a) (b) (c)

FIGURE 5.16 – Étalonnage d'une paire stéréo à champs recouvrants, sous l'hypothèse de champs de vue disjoints (*c.-à-d.* chaque caméra observe uniquement sa propre scène, comme l'illustre les flèches bleues de la Figure (a)).

– Le tableau 5.6 s'assure de la répétabilité de l'algorithme. Deux jeux de données différents sont utilisés. Pour le jeu de données n°1, les cibles reposent sur les deux panneaux (Figure 5.16b). Pour le jeu de données n°2, elles sont imprimées sur une feuille A4 (Figure 5.16c). Ce tableau récapitule la précision de l'initialisation (*cf* Section 5.2.2) et du MCBA, pour ces deux jeux de données, sous l'hypothèse de champs recouvrants ou disjoints.

Lorsque l'on fait l'hypothèse de champs disjoints, chaque caméra n'observe qu'une seule scène à la fois (*cf* flèches bleues de la Figure 5.16a, ou flèches vertes en cas de permutation). Pour le premier jeu de données, S_1 et S_2 sont respectivement sur le panneau de gauche et de droite (*cf* Figure 5.16b). Pour le deuxième jeu de données, S_1 et S_2 sont deux sous-ensembles disjoints composés de huit cibles chacun (*cf* Figure 5.16c).

Pour chaque étalonnage, on prend 15 images. La distance séparant les caméras est d'environ 22 cm. Tous les écarts-types des erreurs de reprojection valent en moyenne 0.03 pix.

Les résultats des tableaux 5.5 et 5.6 montrent que notre étalonnage avec des champs non-recouvrants est tout aussi précis qu'un algorithme classique dont les champs sont recouvrants. Lorsque les mouvements sont quasiment plans, et si les scènes ne sont pas permutées, alors la singularité induit une erreur sur la hauteur (*cf* $5^{ème}$ colonne du tableau 5.5, où il y a environ 6 cm d'erreur sur la hauteur relative). En effet, la translation extrinsèque n'est alors pas totalement observable (*cf* Figure 5.4). *A contrario*, lorsque les scènes sont permutées, on obtient la même précision qu'un algorithme classique (*cf* dernière colonne du tableau 5.5). On constate que sans l'équation 5.17 pour des mouvements plans, la rotation extrinsèque n'aurait pas été correctement déterminée puisque le rang du système 5.7 n'est pas plein (*cf* Section 5.2.2.3).

Hypothèse de champs	Recouvrants		Disjoints		
Mouvements	3D	Plan	3D	Plan	
Scènes permutées ?	n/a	n/a	non	non	oui
Erreur en translation $\|dT\|$ (mm)	0 (ref)	0.14	0.41	60.39	0.08
Erreur angulaire $dR(^\circ)$	0 (ref)	0.008	0.011	0.038	0.011

TABLE 5.5 – Comparaison de la précision des étalonnages extrinsèques en fonction de l'hypothèse de champs de vue recouvrants, du mouvement et de la permutation de scènes. La référence est l'étalonnage classique de la paire stéréo avec des champs recouvrants et en lui appliquant un mouvement 3D. Jeu de données : panneaux (Figure 5.16b).

		Panneaux (jeu n°1)		A4 (jeu n°2)	
		Hypothèse de champs			
	Algorithme	Recouvrants	Disjoints	Recouvrants	Disjoints
Erreur en translation $\|dT\|$ (mm)	Initialisation	7.7	2.3	0.97	2.0
	MCBA	0 (ref)	0.41	0.82	0.79
Erreur angulaire $dR(^\circ)$	Initialisation	0.29	0.11	0.24	0.29
	MCBA	0 (ref)	0.011	0.049	0.045

TABLE 5.6 – Comparaison de la précision et de la répétabilité des étalonnages extrinsèques en fonction de l'hypothèse de champs de vue recouvrants pour deux jeux de données différents. Les mouvements sont tous 3D. De nouveau, la référence est l'étalonnage classique de la paire stéréo avec des champs recouvrants.

D'autre part, si la permutation des scènes n'est pas prise en compte en rajoutant les équations 5.30b, alors la hauteur relative des caméras n'est pas observable (*cf* Section 5.2.2.3) : la hauteur obtenue dépend alors uniquement de l'initialisation du MCBA, du bruit et des défauts de planéité. Au contraire, la méthode traitant le cas singulier (proposé dans la Section 5.2.2.3) permet une initialisation cohérente des paramètres extrinsèques dans le cas des mouvements plans.

Le tableau 5.6 montre que la méthode proposée offre une meilleure précision que les travaux précédents (comme celui de [EWK07a]), qui sont utilisés comme initialisation de notre MCBA. On constate que les mesures sont répétables : on observe une erreur maximale de $0.8\ mm$ pour la translation (soit $\|dT\|/\|\Delta T_2\| < 0.4\%$) et une erreur inférieure à 0.05° pour la rotation.

5.5.1.4 Système multi-caméra embarqué

On embarque deux caméras sur un robot mobile : l'une à l'avant, l'autre à l'arrière (la distance les séparant est d'environ 22 cm, cf Figure 5.17). Le robot manœuvre sur un sol plat entre

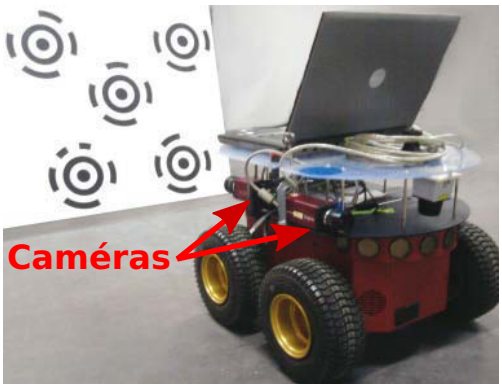


FIGURE 5.17 – Étalonnage d'une paire stéréo avec des caméras embarquées à champs non-recouvrants.

deux scènes. Pendant l'acquisition, un demi-tour est effectué afin de permuter les scènes observées. Comme les occultations de la scène sont supportées, on ajoute des prises de vue où les scènes sont vues de côté, pour améliorer la reconstruction des scènes 3D. L'algorithme renvoie les paramètres extrinsèques. L'écart type des erreurs de reprojection est d'environ $0.09\ pix$. On remarque que si l'équation (5.30b) n'était pas ajoutée pour initialiser la translation extrinsèque, on aurait une erreur d'environ $1.6\ m$ selon la normale au plan du mouvement.

Pour valider l'étalonnage, nous considérons une paire d'images stéréo qui n'a pas été utilisée lors de l'étalonnage (il s'agit du même principe utilisé pour l'étalonnage intrinsèque expliqué en 2.2.2). En utilisant l'estimation précédente des points 3D et des paramètres extrinsèques, la pose du système multi-caméra est calculée. Les erreurs de reprojection sont centrées et on obtient un écart type d'environ $0.05\ pix$ (même ordre de grandeur que précédemment).

5.5.2 Structure From Motion et auto-étalonnage extrinsèque

Nous présentons ici des résultats synthétiques et réels validant la procédure d'étalonnage de la Section 5.3. Dans les deux cas, le véhicule (équipé de $N = 2$ caméras à champs disjoints) se déplace le long d'un parcours. Puis, en utilisant les images enregistrées et une approximation des paramètres extrinsèques comme point de départ, on exécute l'algorithme proposé dans la Section 5.3.2.

5.5.2.1 Résultats avec des données synthétiques

On génère tout d'abord 201 poses clés dans un environnement 3D synthétique et texturé. Des images synthétiques sont obtenues pour les caméras avant et arrière (*cf* Figure 5.18a). La résolution des images est 512×768 pix. Le véhicule effectue une trajectoire non-plane en forme de O. Les paramètres extrinsèques à retrouver sont $\Delta R_j = diag(-1, 1, -1)$ et $\Delta\mathbf{t}_j = (0\ 0\ -1m\ )^\top$.

Ils correspondent à la vérité terrain. Pour initialiser notre algorithme, les paramètres extrinsèques approximatifs $\widehat{\Delta R_j}$ et $\widehat{\Delta \mathbf{t}_j}$ sont utilisés : $\widehat{\Delta R_j} = \rho(\xi)\Delta R_j$ avec les angles d'Euler $\xi = (5^\circ\ 5^\circ\ 5^\circ\ )^\top$ et $\widehat{\Delta \mathbf{t}_j} = (-104mm\ 59mm\ -968mm\ )^\top$.

De plus, comme on utilise des données synthétiques, une reconstruction servant de vérité terrain est réalisée en utilisant les paramètres extrinsèques restant constants. Environ 31000 points 3D sont reconstruits. On utilise l'erreur angulaire dR_j définie par l'équation 5.72 et le vecteur d'erreur $d\mathbf{t}_j$ en translation défini par l'équation 5.73. Cette erreur est invariante à l'échelle de la reconstruction obtenue par notre algorithme.

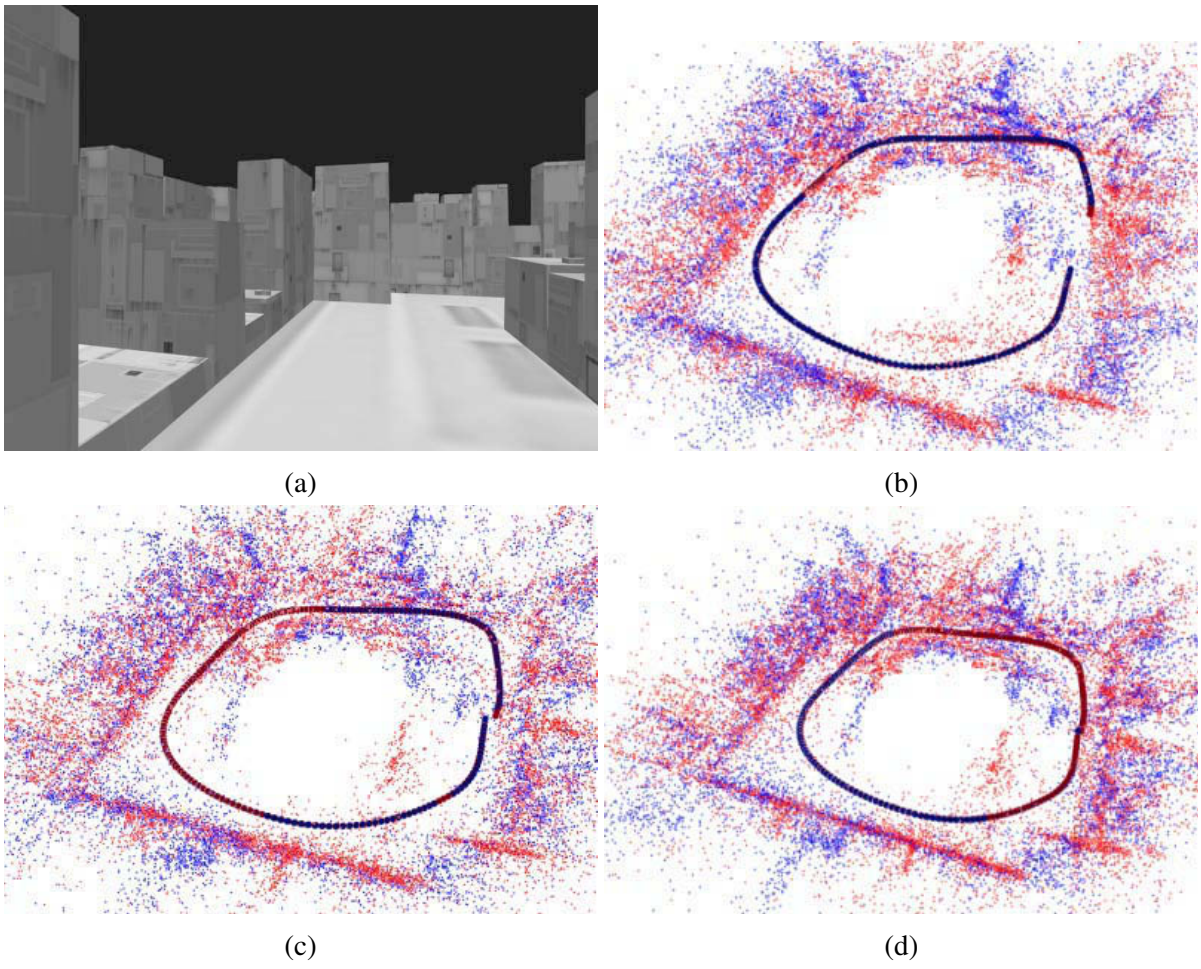
$$d\mathbf{t}_j = \Delta \mathbf{t}_j - \frac{\|\Delta \mathbf{t}_j\|}{\|\widehat{\Delta \mathbf{t}_j}\|}\widehat{\Delta \mathbf{t}_j} \tag{5.73}$$

Le tableau 5.7 montre que l'erreur de reprojection (RMS : $\sqrt{\frac{1}{N_r}\boldsymbol{\epsilon}^\top\boldsymbol{\epsilon}}$), l'erreur angulaire et l'erreur en translation diminuent grâce au MCBA (par rapport à la reconstruction avec $\widehat{\Delta T_j}$ constant). L'estimation de la translation extrinsèque peut être améliorée : en effet pour ce jeu de données, la courbure maximale de la trajectoire n'est pas très élevée. La translation latérale est donc moins bien conditionnée. Nous verrons comment améliorer cette estimation dans la seconde expérience synthétique, et dans le cas réel.

La Figure 5.18 illustre les reconstructions obtenues avec les différentes méthodes. Des dérives (notamment en altitude) apparaissent dans la trajectoire de la reconstruction 5.18b. Au contraire, ces dérives sont compensées pour la reconstruction 5.18c, et la trajectoire est totalement refermée pour la reconstruction 5.18d.

	Erreur de reprojection (pix)	Erreur en Translation $d\mathbf{t}_j$ (mm)	Erreur angulaire dR_2 (°)
SFM avec ΔR_j, $\Delta \mathbf{t}_j$	0.48	(0, 0, 0)	0°
SFM avec $\widehat{\Delta R_j}$, $\widehat{\Delta \mathbf{t}_j}$	0.61	(104, -59, 32)	8.5°
MCBA sans fermeture de boucle	0.47	(94, 0.2, -4)	0.45°
MCBA avec fermeture de boucle	0.46	(69, 5, -2)	0.23°

TABLE 5.7 – Comparaison de la précision obtenue avec plusieurs reconstructions : SFM avec des paramètres extrinsèques parfaitement connus, SFM avec des paramètres extrinsèques approximatifs constants, qui est bouclée ou non (avec l'algorithme 6), puis optimisée par le MCBA.



(a) (b)

(c) (d)

FIGURE 5.18 – (a)Une des images synthétiques utilisée comme entrée du SFM. Carte 3D reconstruite avec (b) le SFM aux paramètres extrinsèques approximatifs constants, puis optimisée par le MCBA (c), ou bien bouclée (avec l'algorithme 6) puis optimisée par le MCBA (d). Les points rouges et bleus sont respectivement observés par la caméra avant et arrière.

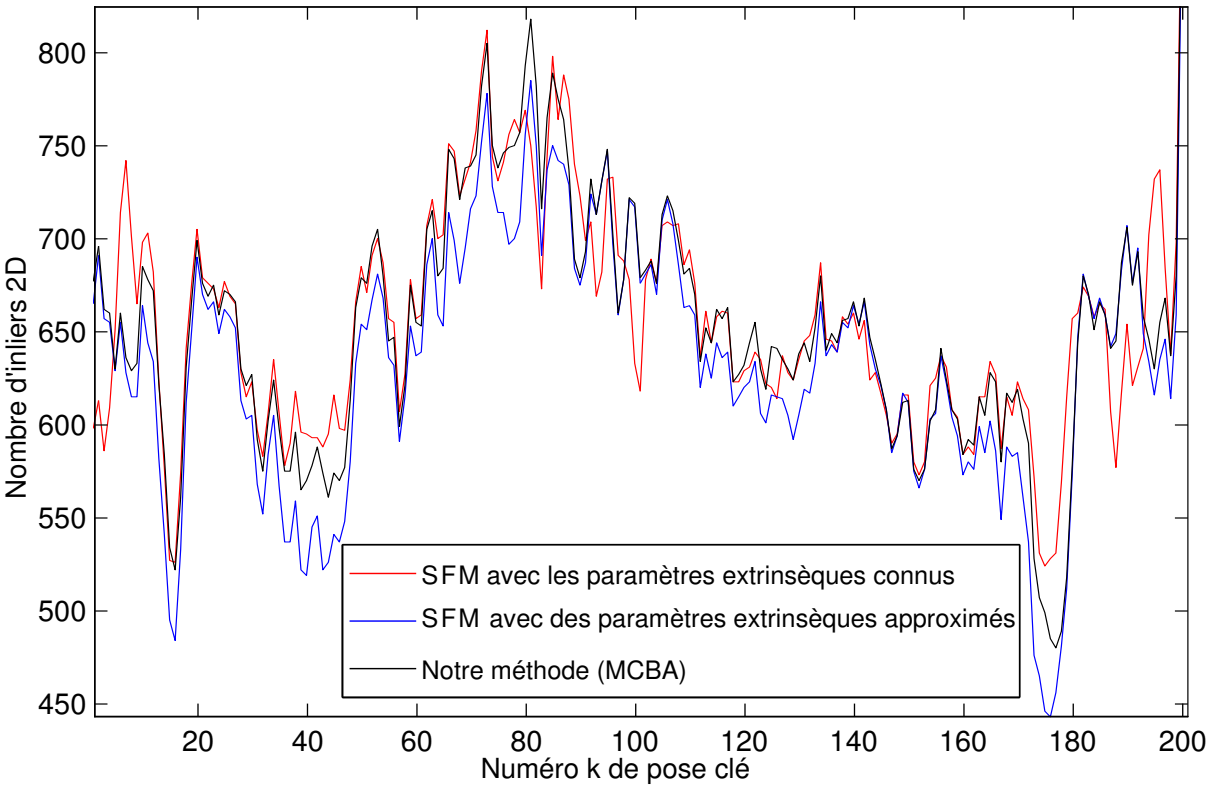


FIGURE 5.19 – Nombre d'inliers 2D pour les différentes reconstructions. La diminution du nombre d'inliers du SFM avec paramètres extrinsèques approximatifs constants (par rapport au SFM avec les vrais paramètres) est compensée par l'apport d'inliers du MCBA.

On rappelle que l'algorithme de reconstruction et le MCBA intègrent un processus de mise à jour des points 2D en inliers ou outliers. Lorsque les paramètres extrinsèques sont approximatifs, moins d'inliers 2D sont trouvés (*cf* Figure 5.19). En effet, de mauvais paramètres extrinsèques engendrent une géométrie épipolaire approximative, et entrainent donc plus de rejet de points aberrants (et donc une diminution du nombre d'inliers 2D). Le MCBA corrige la géométrie approximative et permet d'augmenter le nombre d'inliers. On note que l'intégralité des inliers n'est pas retrouvée par le MCBA, car les appariements potentiels ne sont pas tous réalisés lors de la première étape (à cause de la géométrie épipolaire approximative). Comme le MCBA inclut uniquement une classification des points en inliers ou outliers, il ne crée pas de nouveaux appariements. Le MCBA augmente cependant le nombre d'inliers : en moyenne, 20 inliers sont rajoutés pour chaque pose.

5.5.2.2 Résultats avec des données réelles

Pour cette expérience, un véhicule VipaLab est conduit manuellement dans PAVIN (*cf* Section 1.1.2). Il est équipé de deux caméras de marque Pixelink™. Les paramètres extrinsèques approximatifs sont $\widehat{\Delta R_j} = diag(-1, 1, -1)$ et $\widehat{\Delta \mathbf{t}_j} = (0\ 0\ -2m\ )^\top$. L'algorithme de reconstruction est exécuté avec 376 poses clés, choisies sur la trajectoire d'environ 300 mètres. Environ $57\,000$ points 3D sont reconstruits, avec environ $255\,000$ amers 2D inliers. L'écart type de l'erreur de reprojection est d'environ $0.6\ pix$. Les paramètres extrinsèques calculés par notre algorithme sont : $\Delta R_j = \rho(\xi) diag(-1, 1, -1)$ avec les angles d'Euler $\xi = (0.3^\circ\ 2.2^\circ\ -0.6^\circ\ )^\top$ (d'où, $dR_j = 2.3^\circ$) et $2\Delta \mathbf{t}_j / \|\Delta \mathbf{t}_j\| = (98mm\ -97mm\ -1995mm\ )^\top$. Les cartes reconstruites sont illustrées par la Figure 5.20. La reconstruction approximative 5.20d est raffinée par le MCBA (sans fermeture de boucle) pour obtenir la reconstruction 5.20e. On remarque que le MCBA seul (sans fermeture de boucle) permet déjà d'améliorer la reconstruction : les boucles de cette carte sont bien fermées comme l'illustre la vue de dessus de PAVIN 5.20f. Les nouvelles observations ajoutées par l'algorithme de fermeture de boucle sont tout de même complémentaires. Sur cette séquence, nous n'illustrons pas la vue de dessus de la carte obtenue après SFM, fermeture de boucle puis MCBA, qui est très proche de la Figure 5.20e. La Figure 5.21a illustre la dérive en altitude engendrée par les paramètres extrinsèques approximatifs. L'ajustement de faisceaux multi-caméra (MCBA) corrige pratiquement la dérive sans la fermeture de boucle (Figure 5.21b). Avec la fermeture de boucle et le MCBA, cette dérive est totalement annulée (Figure 5.21c).

D'autres expérimentations ont été menées, avec le VipaLab doté de deux caméras de marque AVT™(modèle Marlin). Nous les utilisons dans le chapitre 6. La Figure 5.22a illustre une reconstruction avec des paramètres extrinsèques approximatifs. En appliquant l'algorithme proposé (fermeture de boucle et MCBA) sur la même séquence, on obtient la reconstruction illustrée par la Figure 5.22b. Elle est constituée de 754 poses clés, d'environ $103\,000$ points 3D et d'environ $554\,000$ amers 2D inliers.

5.5.3 Résultats après étalonnage

Dans cette section, nous présentons d'autres résultats obtenus dans le cadre du projet VIPA. Une fois que les caméras sont étalonnées (intrinsèquement et extrinsèquement), on peut se contenter d'effectuer une reconstruction (SFM) multi-caméra sans ré-optimiser les paramètres extrinsèques. On peut alors valider qualitativement l'étalonnage, à l'aide de la reconstruction obtenue. Nous présentons différents jeux de données.

5.5.3.1 Reconstruction de quelques centaines de mètres

Les Figures 5.23a et 5.23c illustrent une reconstruction de 175 m sur PAVIN avec un système multi-caméra étalonné. On observe une faible dérive en altitude qui peut être compensée (*cf* Figures 5.23b et 5.23d) en appliquant une fermeture de boucle et un ajustement de faisceaux

multi-caméra (qui ne ré-optimise pas les paramètres extrinsèques).

5.5.3.2 Reconstructions de plusieurs kilomètres

Lors du challenge Bibendum 2011 à Berlin, deux VIPA ont fait la navette pour transporter les visiteurs entre deux sites pendant 5 jours. La trajectoire de référence est représentée sur la Figure 5.24. Sa longueur est d'environ 1.3 km, avec 1490 poses clés. La Figure 5.25 montre une localisation du véhicule sur cette trajectoire dans la carte 3D reconstruite lors de la phase d'apprentissage.

Un VipaLab a été conduit manuellement sur le campus des Cézeaux afin de reconstruire une trajectoire de plusieurs kilomètres (*cf* Figure 5.26). Elle est constituée de 5724 poses clés, d'environ $1\,150\,000$ points 3D et d'environ $5\,240\,000$ amers 2D inliers. Le parcours emprunté comprend toutes sortes de scénarios (sol plat, montées, descentes, virages serrés,...) pour une longueur totale d'environ 3.5 km. En ayant le même point de départ que d'arrivée, on constate une erreur de reconstruction d'environ 4 m. Si l'on répartit cette dérive sur l'ensemble du parcours, on obtient en moyenne une erreur d'environ 1 cm tous les 10 m.

5.5.3.3 Reconstruction multi-caméra et dérive du facteur d'échelle

Dans ce paragraphe, nous évoquons le problème de la dérive du facteur d'échelle qui nuit aux reconstructions monoculaires. Il est illustré par la reconstruction de la Figure 5.20b où seule la caméra avant est utilisée et le facteur d'échelle diminue. La Figure 5.20c illustre une reconstruction avec uniquement la caméra arrière, où le facteur d'échelle augmente. Pour pallier ce problème de dérive, [LBR+10] propose d'estimer l'échelle soit avec un calcul d'homographie de points de la route, soit en utilisant un système d'information géographique. Mais qu'en est-il de la dérive du facteur d'échelle pour des reconstructions multi-caméra ? Compte tenu de l'état de l'art et de nos résultats, nous proposons des éléments de réponse.

Dans [PFF+10, §4.2], les auteurs utilisent entre autres un capteur Ladybug®2 (un système multi-caméra pas plus grand qu'un verre d'eau, intégrant 6 caméras à champs partiellement recouvrants, *cf* Figure 3.1c) embarqué sur un véhicule. Comme les caméras sont très proches les unes des autres (moins de 4 cm), on peut considérer que les centres optiques des caméras sont quasiment confondus. Les auteurs sont donc confrontés au problème de dérive du "facteur d'échelle", comme pour le cas mono-caméra. Ils doivent donc estimer régulièrement l'échelle de la reconstruction. Ils affirment que cette estimation est possible avec une seule caméra, dans des virages pour un mouvement plan. Mais il faut que la caméra ne soit pas placée au dessus de l'essieu arrière, et que la distance essieu-caméra soit connue. Avec notre reconstruction (*cf* Section 5.3) et nos caméras séparées par une grande distance (baseline de plus d'un mètre), nous n'avons pas ce problème de dérive. Mais comment se propage l'échelle (et éventuellement sa dérive) lors d'une reconstruction ? Dans le cas mono ou multi-caméra, elle se propage notamment grâce aux points observés par différentes poses voisines. Dans le cas multi-caméra,

elle se propage également grâce aux translations extrinsèques dont les normes sont communes à chaque pose de la reconstruction.

Pour éviter le problème de dérive de l'échelle d'une reconstruction 3D multi-caméra, il faut trouver un bon compromis entre la distance des points observés, la baseline des caméras, et la précision de détection des points. Cette problématique d'estimation du mouvement est évoquée pour des caméras centrales ou non-centrales dans [SRT+11, §6.2]. En effet, si l'on suppose que la précision de détection des points est infinie, alors quelle que soit la distance des points observés et quelle que soit la baseline (non-nulle) des caméras, la reconstruction multi-caméra serait obtenue sans dérive du facteur d'échelle. En présence de bruit, pour chaque caméra, l'incertitude sur la position des points 3D observés (communs à différentes poses voisines) se propage en une incertitude sur ses poses. Les normes des translations extrinsèques sont non-nulles et communes aux différentes poses du système multi-caméra, elles assurent un facteur d'échelle commun aux poses voisines. De proche en proche, l'échelle globale de la reconstruction reste la même.

Avec un système multi-caméra étalonné, la reconstruction 3D est obtenue avec le facteur d'échelle fixé par la distance entre les caméras. En d'autres termes, la reconstruction (poses, points 3D) est directement à l'échelle réelle. On s'affranchit ainsi du phénomène de dérive du facteur d'échelle, qui est habituellement rencontré dans le cas monoculaire. Ce résultat reste valable même avec des caméras à champs disjoints.

5.6 Conclusions et perspectives

5.6.1 Conclusions

Deux approches flexibles pour l'étalonnage extrinsèque d'un système multi-caméra (à champs disjoints) ont été proposées et validées aussi bien avec des données synthétiques que des données réelles.

La première approche (*cf* Section 5.2) consiste à manœuvrer le véhicule pendant que chaque caméra observe une scène statique composée de cibles. Les paramètres extrinsèques sont initialisés linéairement à partir des trajectoires de chaque caméra. Les mouvements singuliers ont été mis en exergue et nous avons montré comment s'en affranchir (avec un protocole expérimental et des outils mathématiques) : pour le cas du mouvement plan, la permutation des scènes permet d'obtenir un étalonnage complet. De plus, les résultats montrent que notre étalonnage, sous l'hypothèse de champs de vue disjoints, est tout aussi précis qu'un étalonnage classique dont les champs de vue sont recouvrants.

Dans la deuxième approche, nous avons montré que l'étalonnage extrinsèque peut être obtenu simultanément à la reconstruction 3D (par exemple lors de la phase d'apprentissage), en utilisant des points d'intérêt. Nous avons détaillé un algorithme multi-caméra détectant des fermetures de boucle (*c.-à-d.* on détecte si le système réemprunte un chemin déjà emprunté). Nous avons montré que le MCBA (seul ou après avoir détecté des fermetures de boucle) permet

d'améliorer les reconstructions qui sont obtenues avec des paramètres extrinsèques constants et approximatifs. Dans le cadre du projet VIPA, nous avons également illustré des reconstructions (pouvant atteindre quelques kilomètres) qu'il a été possible d'obtenir avec un système multi-caméra étalonné (intrinsèquement et extrinsèquement).

L'outil commun aux deux approches (l'ajustement de faisceaux multi-caméra (MCBA)) a été proposé. Il optimise à la fois les points 3D, les poses du système multi-caméra et les paramètres extrinsèques (il en est un estimateur de maximum de vraisemblance). La jacobienne de la somme des erreurs de reprojection (en fonction des paramètres optimisés) a été exprimée analytiquement. Nous avons aussi expliqué en détail comment exploiter la structure creuse des matrices du MCBA, notamment en cas de fermeture de boucle. Les équations normales du MCBA ont été posées, réduites (pour obtenir un système de plus petite dimension : le *Système Caméras/Poses* - CPS) et résolue de deux façons (avec une factorisation de Cholesky ou bien un algorithme du gradient conjugué préconditionné). La complexité du MCBA a également été détaillée en fonction de la méthode de résolution du CPS. L'algorithme 8 synthétise de manière détaillée l'implémentation creuse de l'ajustement de faisceaux multi-caméra (MCBA). Il peut se placer comme une référence (au même titre que l'algorithme [HZ04, A.6.7] pour le monoculaire).

5.6.2 Perspectives

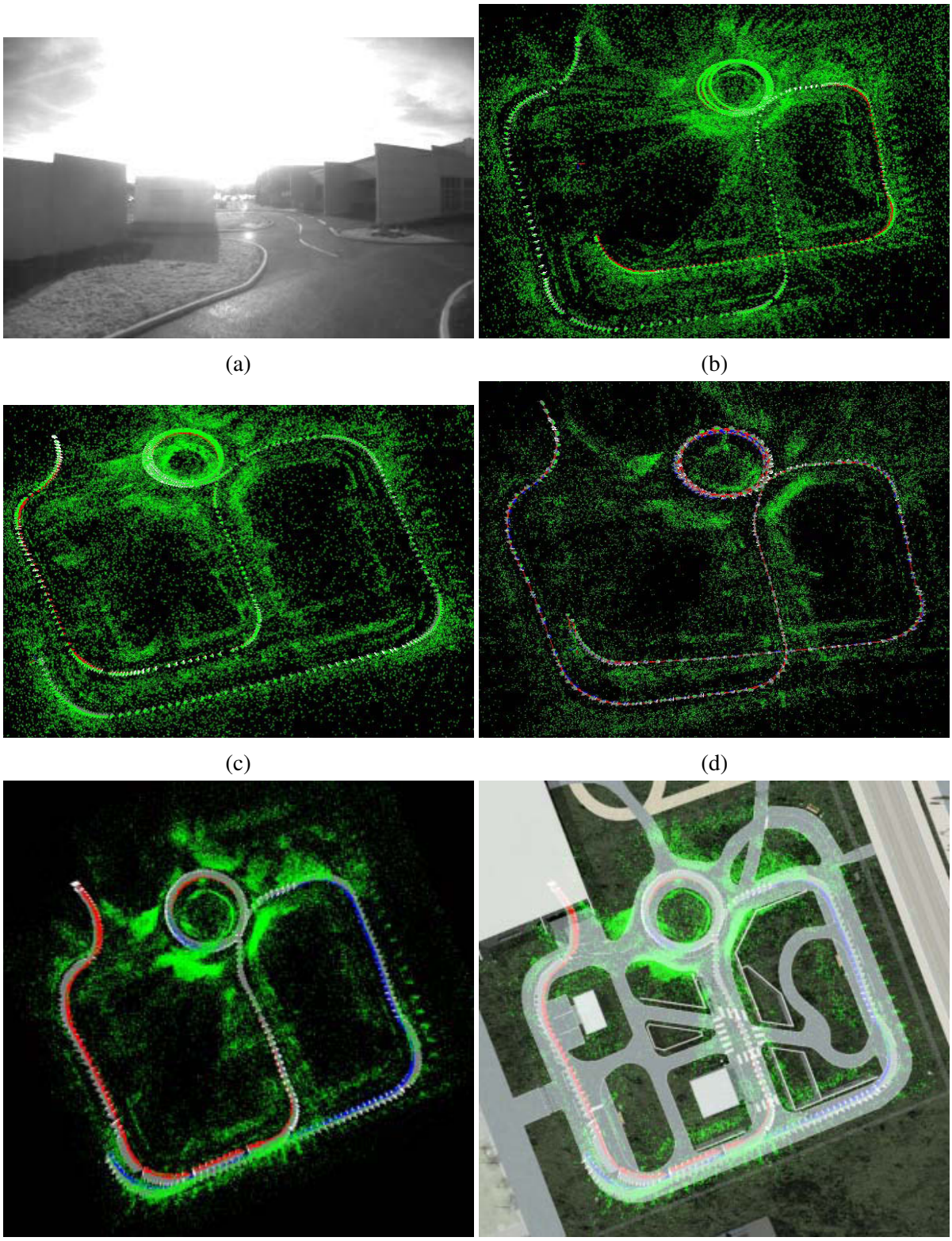
- Après qu'un étalonnage utilisant des cibles soit effectué, la géométrie de la scène est connue dans un même repère. Pour étalonner un autre système multi-caméra, il suffirait de calculer la pose de chaque caméra par rapport aux cibles qu'elle observe, puis d'en déduire leurs poses relatives. Ce procédé fournit toutefois des résultats moins précis que le MCBA (comme le montrent les résultats dans la Section 5.5.1). Il serait également possible (avec une scène connue) d'utiliser plusieurs poses du système multi-caméra : l'étalonnage se terminerait par un MCBA simplifié (qui n'optimiserait par les points 3D, mais uniquement les poses du système multi-caméra et les paramètres extrinsèques).
- Bien qu'elle soit suffisamment rapide, il serait possible d'accélérer la convergence de l'ajustement de faisceaux multi-caméra de la fermeture de boucle. En effet, après avoir parcouru puis détecté une fermeture de boucle entre la pose $k = 0$ et la pose $k = K$, [DES10] propose de répartir algébriquement la dérive calculée entre ces deux poses sur l'ensemble des poses intermédiaires. Dans notre cas, il faudrait généraliser cette méthode au cas de plusieurs fermetures de boucle qui ne concernent pas uniquement la première et la dernière pose de la trajectoire.
- Dans le cas d'une mauvaise initialisation manuelle des paramètres extrinsèques, la carte 3D est fortement distordue et il peut être nécessaire de recommencer l'ensemble de l'algorithme (SFM, fermeture de boucles, MCBA) pour améliorer la reconstruction. Cette solution est facile à implémenter, mais une solution plus rapide serait de mettre à jour les correspondances de points après modification des paramètres extrinsèques, ou bien de réaliser et conserver les mises en correspondances indépendamment pour chaque caméra

(lors du SFM), afin qu'elles ne dépendent pas des paramètres extrinsèques.

- Il serait également pertinent qu'au lieu de faire une reconstruction approximative globale (avec des paramètres extrinsèques constants), on optimise plutôt dès que possible certains ou tous les paramètres extrinsèques. En d'autres termes, dès que le mouvement le permet (*c.-à-d.* dès que l'on n'est pas dans un cas singulier), on estime les paramètres extrinsèques observables. Lors de la reconstruction, il est envisageable de n'optimiser un paramètre extrinsèque que lorsque le mouvement le permet. Ce procédé de reconstruction est détaillé dans [8]. Les reconstructions mises en œuvre peuvent tout aussi bien être hiérarchiques qu'incrémentales. Une méthode statistique basée sur la variance des paramètres extrinsèques est proposée dans [MAHW11] afin d'estimer quels paramètres peuvent être optimisés, avec quelle précision et si les *a priori* sur les paramètres extrinsèques sont cohérents avec les mouvements réalisés. La méthode est décrite pour un système bi-stéréo, sans recouvrement entre les deux paires stéréos. Il serait intéressant de comparer ces différentes approches.
- Lors de la phase de détection de boucle, les critères de proximité sont définis manuellement par des seuils. Il serait plus pertinent qu'ils soient définis automatiquement en prenant en compte la propagation des erreurs dans les ajustements de faisceaux (comme analysé dans [EL09]).
- Pour la résolution du Système Caméras/Poses, et d'après l'étude comparative [JNS+10] des méthodes de résolution normales pour un ajustement de faisceaux classique, nous avons codé l'algorithme du Gradient Conjugué Préconditionné en implémentant notre propre structure de données. Afin d'étendre l'étude comparative au cas de l'ajustement de faisceaux multi-caméra, il pourrait être adéquat :
  - d'utiliser une librairie récemment publiée [Lou10] qui inclut de nombreux algorithmes creux pour le LM.
  - de stocker la matrice symétrique $\bar{\mathbf{S}}$ de manière redondante tel que l'accès en lecture soit le plus contigüe possible. Ceci augmenterait le besoin de mémoire vive, mais devrait en contrepartie limiter les dépassements de cache L2 du CPU et donc améliorer les performances.

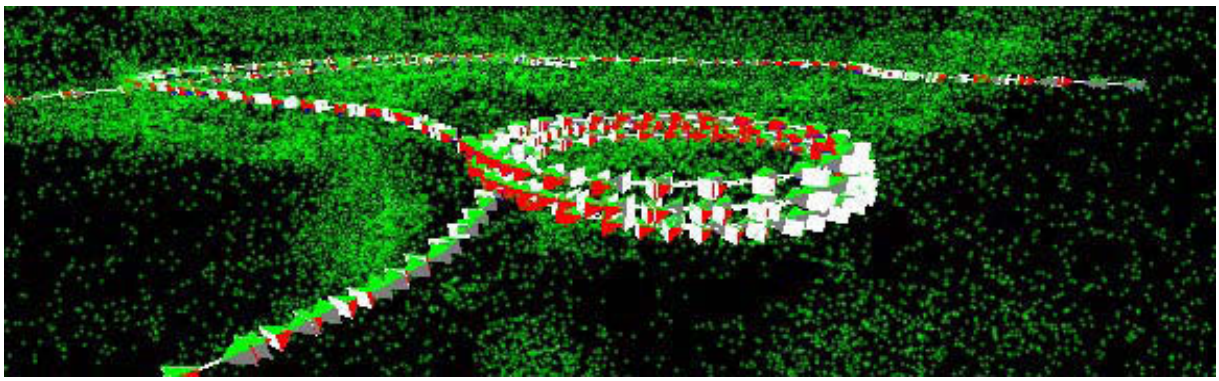
  Il serait également intéressant d'adapter la méthode mettant à jour les paramètres à optimiser des articles [Kae08, KRD08, KJR+11] (basés SLAM) au cas de l'étalonnage multi-caméra. Dans ces articles, des équations normales sont résolues via une factorisation QR. En réordonnant les variables, un système équivalent plus creux est résolu.
- L'ajustement de faisceau multi-caméra (MCBA) étant massivement parallélisable, il serait envisageable de l'implémenter sur GPU (comme cela était fait dans [CGN10] ou dans [WACS11] pour un ajustement de faisceaux classique).
- Pouvoir utiliser d'autres modèles intrinsèques comme le modèle unifié (*cf* Section 2.1.3) ou les caméras génériques (*cf* Section 3.1.1). Étudier si l'optimisation des paramètres intrinsèques, conjointe à la reconstruction, permettrait d'obtenir de bonnes reconstructions euclidiennes tout en garantissant plus de souplesse de mise en œuvre (affranchissement de l'étape d'étalonnage intrinsèque). On peut se référer aux travaux [Stu97] et [PVG00]

pour le cas monoculaire.
- Proposer une variante de l'ajustement de faisceaux multi-caméra qui intègre le calcul de la covariance des paramètres estimés dans chaque itération du Levenberg-Marquardt.
- Proposer une autre variante de l'ajustement de faisceaux multi-caméra en relatif : où la pose d'une caméra est définie relativement à une pose voisine, tout comme dans [SMRN09, SMRN10, MSC+10].
- Avec quelle périodicité faut-il ré-étalonner le système multi-caméra ? Pour répondre à cette question, il serait pertinent de mesurer l'intégrité du système multi-caméra au cours du temps. Ainsi, en cas d'un déplacement non-désiré d'une caméra sur son support, il serait possible de le détecter et proposer une solution (avertissement, déclenchement d'un arrêt de sécurité si nécessaire, ré-étalonnage automatique).
  Considérons un ensemble de points 3D bien reconstruits à échelle réelle, et des paramètres extrinsèques ΔT_j qui ne sont plus à jour. La caméra C_j pénalise alors la localisation (calcul de pose) du système multi-caméra : la pose que seule C_j permettrait d'obtenir serait différente de celle calculée avec les autres caméras. En mesurant l'écart entre ces poses, on pourrait en déduire un critère de non-intégrité des paramètres extrinsèques. Le critère peut aussi être défini en fonction du nombre d'inliers lors de différents calculs de pose avec les mêmes images, mais en ne prenant en compte que certaines caméras (ce qui revient à faire plusieurs calculs de la pose du système multi-caméra en masquant artificiellement certaines caméras). Ainsi, le nombre total d'inliers M_{multi} (somme des inliers dans chaque caméra) pour un calcul de pose multi-caméra est inférieur au nombre total d'inliers M_{ind} lorsque les caméras sont considérées indépendamment (N calculs de pose indépendants). Ces deux nombres (calculés pour une pose ou sur une portion de trajectoire) permettent de déterminer l'existence d'une caméra mal-étalonnée (si $\frac{M_{multi}}{N} < pM_{ind}$, avec $p \in]0,\ 1]$ fixé).

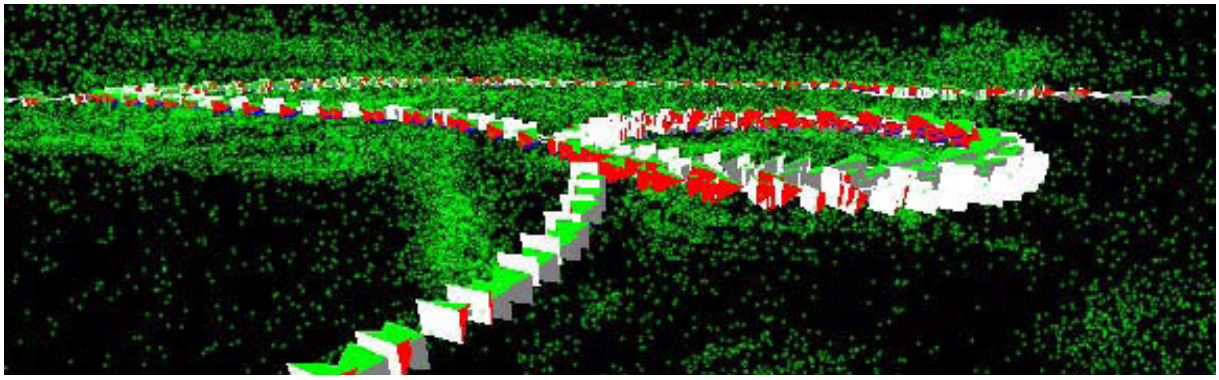


(a) (b) (c) (d) (e) (f)

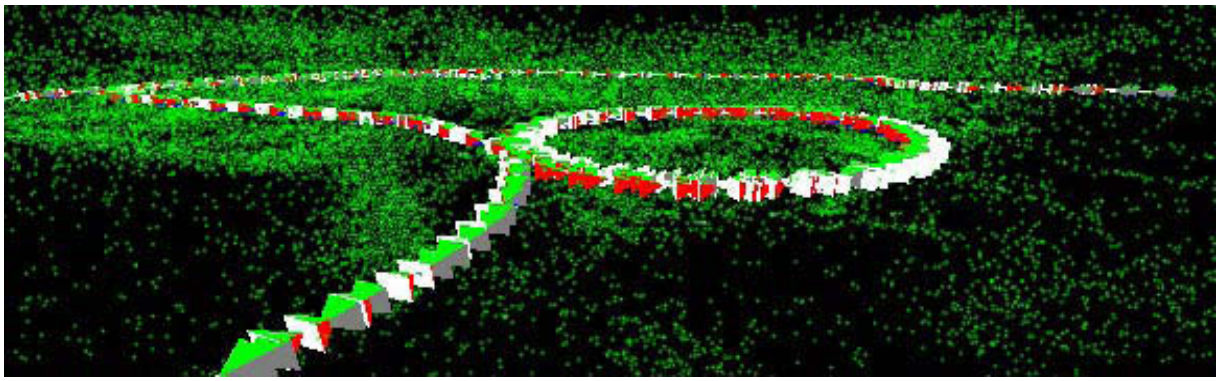
FIGURE 5.20 – (a) Image issue d'une séquence réelle (caméras Pixelink™). Vue de dessus de la carte reconstruite par le SFM avec : (b) uniquement la caméra avant, (c) uniquement la caméra arrière, (d) les deux caméras et des paramètres extrinsèques constants et approximatifs. En appliquant le MCBA à (d), on obtient la carte illustrée par (e) et (f).



(a)

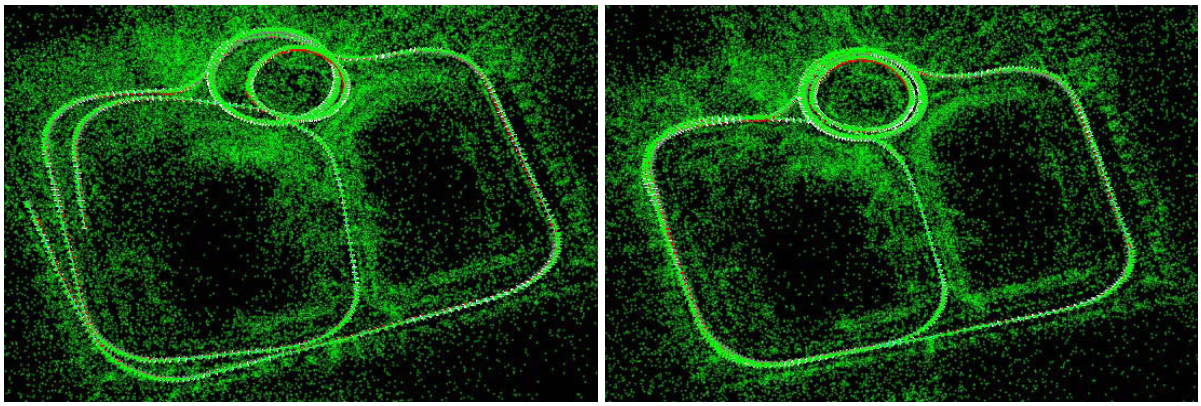


(b)



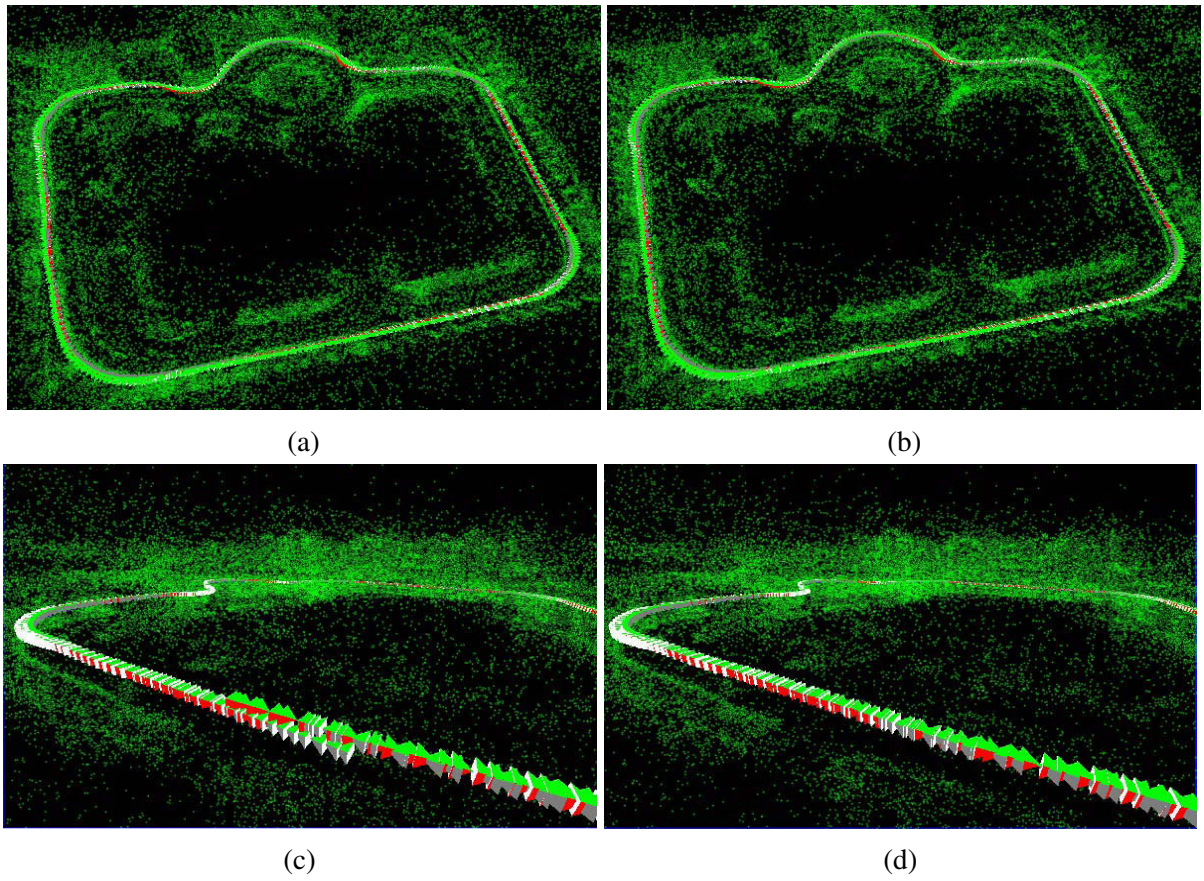
(c)

FIGURE 5.21 – Vue de côté de la carte reconstruite par le SFM avec des paramètres extrinsèques approximatifs constants (a), puis optimisée par le MCBA sans (b) ou avec (c) fermeture de boucle. Pour que la dérive soit bien visible, trois tours du rond-point ont été faits avec le véhicule.



(a) (b)

FIGURE 5.22 – Carte reconstruite par le SFM avec des paramètres extrinsèques approximatifs et constants (a), puis optimisée par le MCBA avec fermeture de boucle (b). Elle est composée de 754 poses clés, d'environ $103\,000$ points 3D et d'environ $554\,000$ amers 2D inliers.



(a) (b)

(c) (d)

FIGURE 5.23 – Reconstruction sur PAVIN d'une séquence de 175 m après avoir étalonné les caméras du VipaLab. Aucune fermeture de boucle n'est appliquée pour les Figures (a) et (c). Une faible dérive en altitude est observée (c). Une fermeture de boucle et un ajustement de faisceaux multi-caméra (qui ne ré-optimise pas les paramètres extrinsèques) permettent de corriger cette dérive (b) et (d).

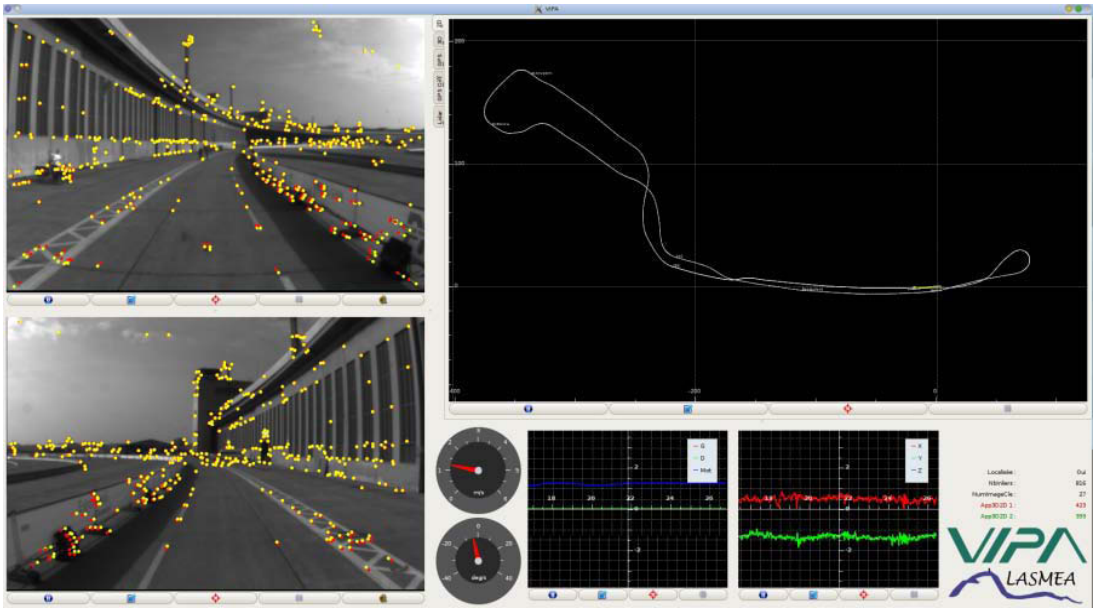


FIGURE 5.24 – Interface logicielle du VIPA se localisant sur une trajectoire d'environ 1.3 km au challenge Bibendum de Berlin.

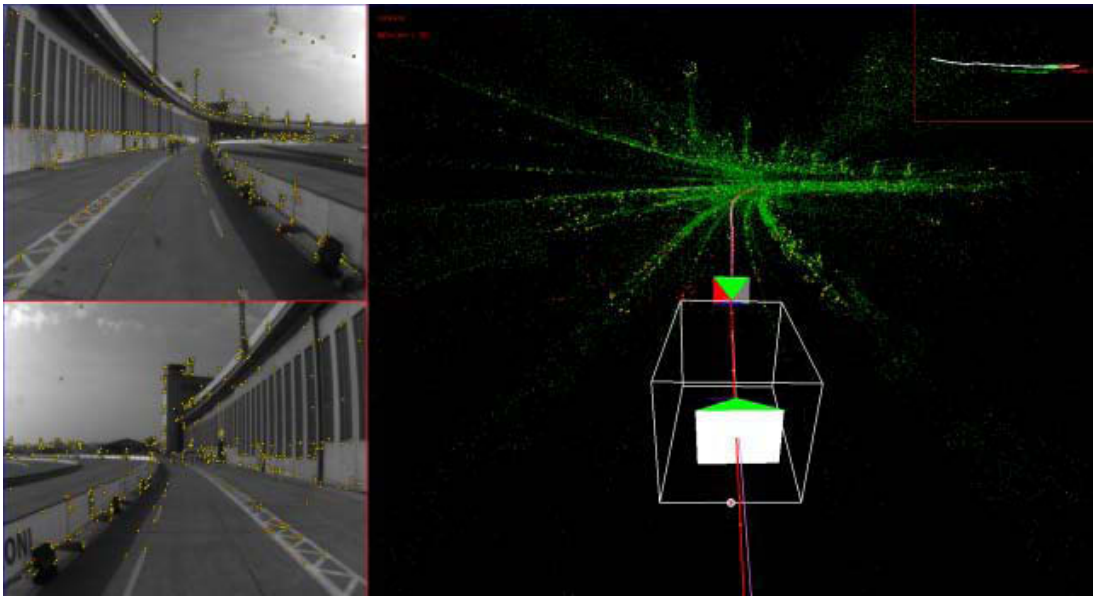


FIGURE 5.25 – Localisation bi-caméra du véhicule dans la carte 3D apprise au préalable.

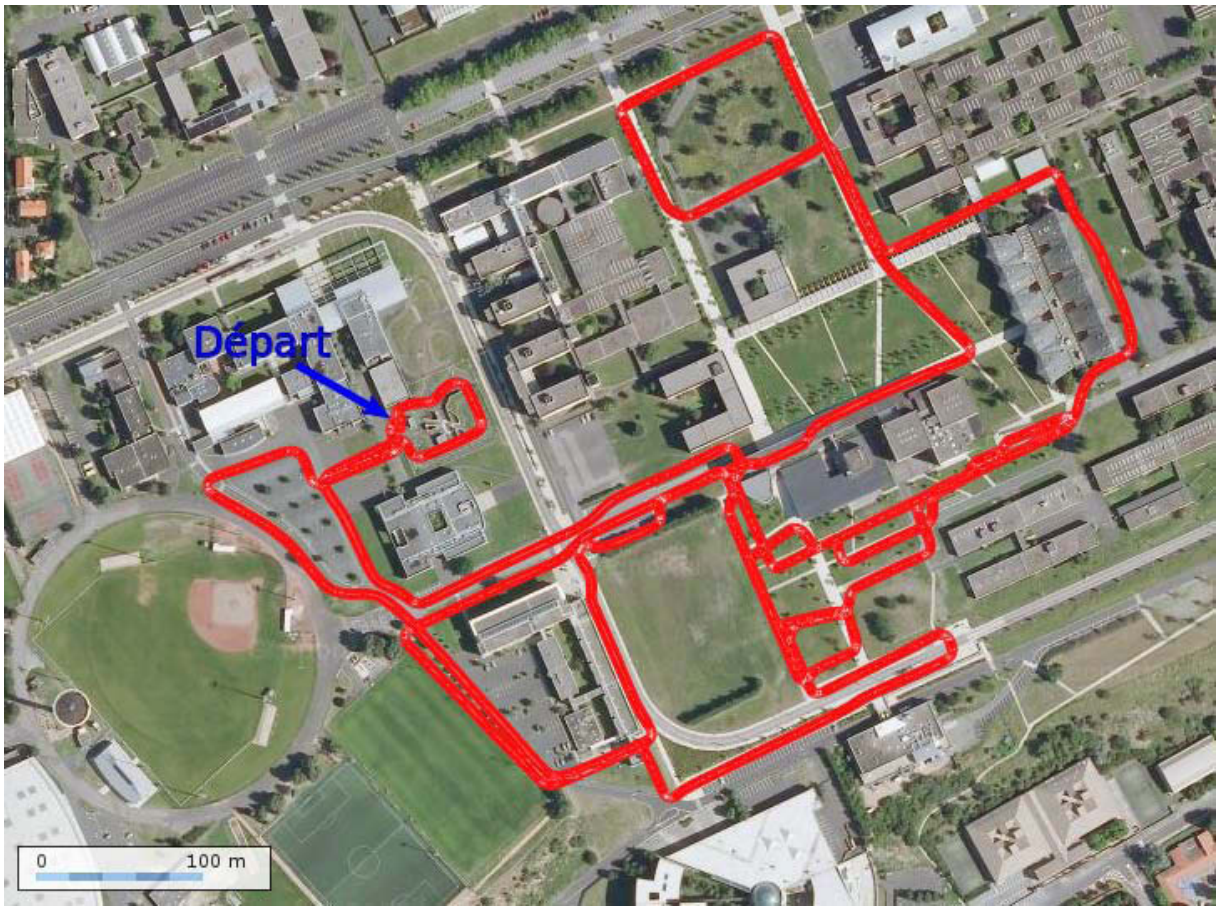


FIGURE 5.26 – Trajectoire d'environ 3.5 km sur le campus des Cézeaux. En revenant au point de départ, on constate une dérive d'environ 4 m, sans qu'aucun algorithme de fermeture de boucle ne soit appliqué. Image satellite du Géoportail de l'IGN.

Chapitre 6

Etalonnage du véhicule

Ce chapitre présente des algorithmes d'étalonnage afin de déterminer la pose (ou l'orientation) de la caméra par rapport au repère véhicule. Pourquoi a-t-on besoin de connaitre précisément cette pose ? C'est nécessaire en localisation et surtout en navigation. En effet, la commande du véhicule s'exprime dans son repère cinématique. Si on se localise à l'aide d'une caméra, il faut donc pouvoir effectuer le changement de repère de la caméra vers le repère du véhicule. De plus, lors d'un convoi, ou plus généralement lors du déploiement d'une flotte de véhicules, la carte 3D est apprise avec l'un d'eux. Elle est ensuite rejouée par les autres véhicules ayant des caméras positionnées différemment par rapport au véhicule d'apprentissage. L'étalonnage présenté dans ce chapitre est alors primordial afin que les véhicules puissent emprunter au mieux le même chemin.

Deux algorithmes sont proposés. La première approche impose que le véhicule se déplace en ligne droite. Au contraire, la deuxième méthode s'affranchit de cette contrainte. Les méthodes sont validées par des simulations et expériences réelles.

6.1 Etat de l'art

Plusieurs articles [NVMG06] [BHS11a, BHS11b] déterminent la pose d'un banc stéréo par rapport au véhicule en détectant les lignes blanches sur les routes. Les lignes blanches sont également utilisées dans le cas monoculaire [CPRR06] dans le même but.

Dans la thèse [Ste05] et l'article [Ste03], des contraintes multi-linéaires sont construites à partir du modèle cinématique d'un véhicule et des observations (suivi de points dans l'image et données odométriques). Ce système multi-linéaire est résolu algébriquement avec des bases de Gröbner. L'étalonnage se déroule en deux temps, avec une première étape imposant au véhicule de se déplacer parfaitement en ligne droite. Bien qu'ils soient mathématiquement élégants et qu'ils ne nécessitent pas une reconstruction 3D des points observés, ces travaux théoriques fournissent des résultats uniquement simulés et biaisés. Ce biais est causé par cette approche en deux étapes et une sensibilité au bruit lors de calculs de dérivées. D'autres travaux théoriques

sont proposés pour un étalonnage en ligne dans [MWS06].

[Mar09] se base sur un EKF dans le cas 2D pour déterminer la pose de la caméra par rapport au repère cinématique du véhicule (en prenant éventuellement en compte l'odométrie).

[RLPK10] suppose que la hauteur et l'orientation de la caméra sont connues, et détermine les deux dernières composantes de la translation. Le mouvement est considéré plan (tout autre mouvement est considéré comme aberrant). Les poses du véhicule sont obtenues par les mesures odométriques et les poses de la caméra par égo-motion. A partir des mêmes données, les mêmes auteurs [RPK] déterminent l'orientation (uniquement les angles de roulis et de tangage) d'une caméra fixée sur un véhicule, et retrouvent aussi la circonférence des roues. C'est également l'approche de [KGB11], qui utilise plutôt un laser (à la place de la caméra). Il estime dynamiquement le rayon de chaque roue (qui peut varier au cours du temps si la charge n'est pas équitablement répartie sur les roues) et la pose 2D du laser.

6.2 Première approche : ligne droite et foyer d'expansion

Dans cette section, on cherche à déterminer l'orientation (étalonnage partiel) d'une caméra se déplaçant en ligne droite. La caméra se translate et est orientée vers l'avant. Ces travaux illustrent la première approche que nous avons développée. Cette étude de faisabilité ne présente pas particulièrement de nouveauté, mais constitue un bon exercice de vision qui a abouti à une expérience métrologique.

6.2.1 Principe

Soit une caméra perspective se déplaçant rectilignement en observant un ensemble statique de points P_i. Il est équivalent de considérer que la caméra reste statique, avec les points 3D se déplaçant rectilignement (*cf* Figure 6.1). On note $\mathbf{v}$ le vecteur vitesse des points P_i.

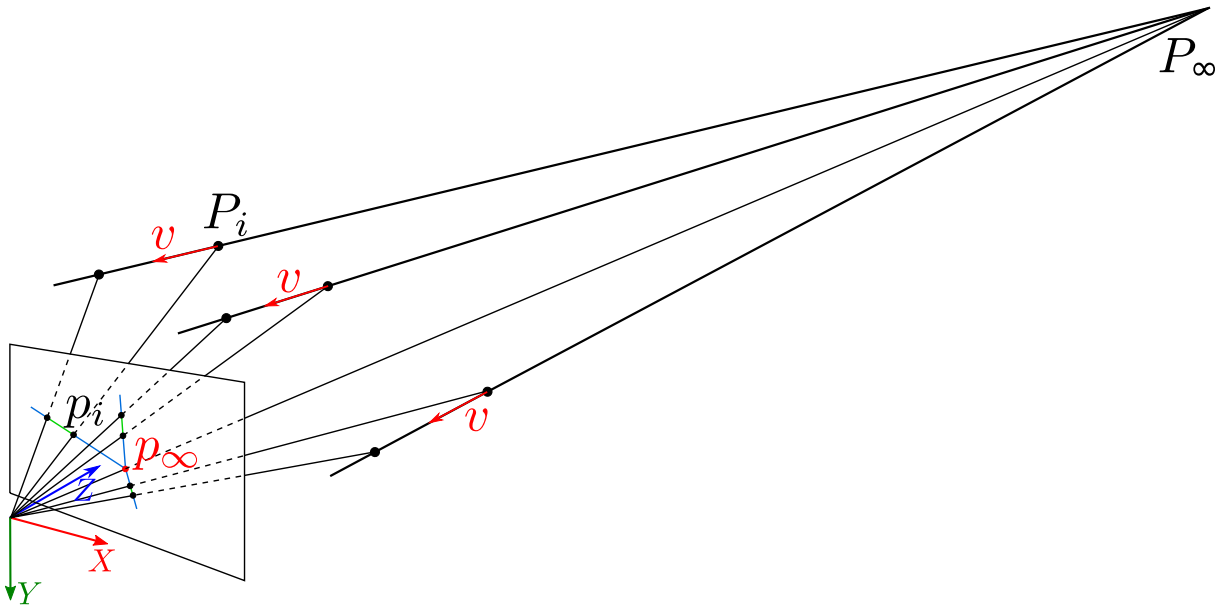


FIGURE 6.1 – Caméra statique observant des points 3D se déplaçant rectilignement selon un même vecteur vitesse $\mathbf{v}$.

La Figure 6.1 illustre une propriété de la géométrie projective : les droites parallèles s'intersectent à l'infini au point P_∞. En effet, si on note $(X_i\ Y_i\ Z_i)^\top$ les coordonnées du point P_i exprimées dans le repère caméra, et $\mathbf{v} = (A\ B\ C)^\top$, un point appartenant à la droite $P_i + \lambda\mathbf{v}$ (avec $\lambda \in \mathbb{R}$) se projette dans le plan image en un point de coordonnées (u_i, v_i) :

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \pi \left(K \begin{pmatrix} X_i + \lambda A \\ Y_i + \lambda B \\ Z_i + \lambda C \end{pmatrix} \right) \tag{6.1}$$

$$\Leftrightarrow \begin{cases} u_i = f_x \dfrac{X_i + \lambda A}{Z_i + \lambda C} + u_0 \\ v_i = f_y \dfrac{Y_i + \lambda B}{Z_i + \lambda C} + v_0 \end{cases} \tag{6.2}$$

L'ensemble des droites de l'espace se projette dans le plan image en un faisceau de droites, qui s'intersectent en un point. Ce point, noté $p_\infty = (u_\infty \; v_\infty)^\top$, est appelé foyer d'expansion comme expliqué dans[BB, p.199]. En faisant tendre λ vers $-\infty$, on obtient ses coordonnées : $p_\infty = (\frac{A}{C} + u_0 \; \frac{B}{C} + v_0)^\top$. Si la caméra pointe grossièrement dans la direction de déplacement, le foyer d'expansion (FOE) pourra être dans l'image.

Pour détecter avec précision le foyer d'expansion, on utilise l'algorithme décrit par la Figure 6.2. Des points sont tout d'abord détectés sur la première image de la séquence. Ils sont ensuite suivis dans les images suivantes (avec une méthode de flot optique). On appelle *track d'un point* l'ensemble de ses points 2D détectés et suivis. Pour chaque track, une droite 2D est estimée de manière robuste (grâce au M-estimateur de Welsh). En présence de bruit, les droites ne s'intersectent pas parfaitement. On initialise l'estimation du foyer d'expansion p_∞ avec un moindre carrés, puis on affine l'estimation avec l'algorithme itératif des moindres carrés repondérés (IRLS : Iteratively Reweighted Least Squares, expliqué dans [MM05]) et le M-estimateur de Welsh ρ. Le poids w_i associé à chaque droite est défini comme le produit de deux poids : le premier w_{i1} est défini comme étant le rapport de la longueur du track par la longueur du plus long track, le deuxième $w_2(r_i)$ est la fonction de poids de Welsh (*cf* [Zha97], le résidu r_i étant la distance entre la droite et l'estimation précédente de p_∞). On cherche donc à minimiser la fonction de coût $\sum_i w_{i1}\rho(r_i)$. Ce qui est équivalent, d'après [Zha97, §9.4], à minimiser $\sum_i w_{i1} w_2(r_i^{[k-1]}) r_i^2$ (à chaque itération k du IRLS). En pratique, on utilise une caméra étalonnée (avec le modèle indirect, *cf* Section 2.1.2.2) pour corriger la distorsion des points 2D.

Quelle est la précision nécessaire pour la détection du FOE pour avoir une précision angulaire donnée ? En d'autres termes, pour avoir un cap (angle de lacet) avec une précision donnée, quelle doit être la précision pour détecter le FOE ? Sur la Figure 6.3, on représente une caméra perspective en vue de dessus, le FOE théorique p_∞ (d'abscisse u_∞) et le FOE détecté $\hat{p}_\infty$ dans l'image (d'abscisse $u_\infty + \epsilon$). ϵ est l'erreur (en pixel) de détection du FOE. Soit φ l'angle réel de lacet par rapport au déplacement, et $\hat{\varphi}$ son estimation. On obtient alors :

$$\begin{cases} u_\infty - u_0 + \epsilon = f_x \tan(-\hat{\varphi}) \Rightarrow \epsilon = -f_x(\tan(\hat{\varphi}) - \tan(\varphi)) \\ u_\infty - u_0 = f_x \tan(-\varphi) \qquad\quad = f_x \tan(\varphi - \hat{\varphi})(1 + \tan(\hat{\varphi})\tan(\varphi)) \end{cases} \tag{6.3}$$

$$\Rightarrow \epsilon \simeq f_x(\varphi - \hat{\varphi})(1 + \tan^2(\varphi)) \tag{6.4}$$

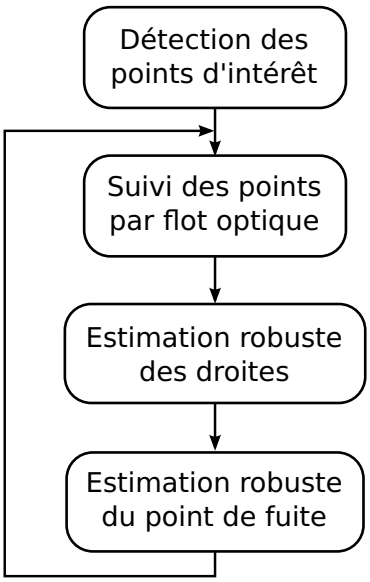


FIGURE 6.2 – Principe de détection du foyer d'expansion. Des points d'intérêt sont détectés sur la première image, puis suivis dans les images suivantes. Chaque point suivi permet d'estimer une droite. Enfin, avec les droites on estime la position du foyer d'expansion.

Plus l'angle de lacet est important, moins le FOE doit être détecté précisément. Faisons une application numérique afin d'avoir un ordre de grandeur de la précision de détection nécessaire. Si on souhaite une précision de 0.01° sur l'angle de lacet (avec la distance focale $f_x = 1836.8$ pix de la caméra utilisée pour l'expérience réelle), il faut que le FOE soit détecté à $1836.8 * 0.01 * \pi/180 = 0.32$ pix près (si la caméra est orientée vers l'avant) et à $1836.8 * 0.01 * \pi/180 * (1 + \tan^2(10 * \pi/180)) = 0.33$ pix près (si l'angle de lacet vaut $+/-10^\circ$).

6.2.2 Résultats

Nous avons validé cette approche avec le banc de métrologie illustré par la Figure 4.8 (que nous avons déjà utilisé dans le chapitre 4). On enregistre trois séquences où la caméra avance en ligne droite. Avec le rotor du banc de métrologie, on fixe l'angle de lacet φ qui vaut respectivement $-10^\circ + \varphi_0$, φ_0 et $+10^\circ + \varphi_0$. Seule la constante φ_0 n'est pas fournie par le banc.

On détermine l'angle de lacet pour chacune des séquences, pour ensuite comparer leurs valeurs. Pour cela, on estime le foyer d'expansion avec l'algorithme précédent, pour en déduire l'angle de lacet (connaissant la matrice des paramètres intrinsèques, et en supposant que la caméra est horizontale, $\hat{\varphi} = -\arctan((u_\infty - u_0)/f_x)$). En pratique, on utilise la séquence inversée (*c.-à-d.* comme si l'on reculait la caméra), afin que les points soient détectés sur l'image la plus proche de la scène, et que la plupart puisse être suivi sur toute la séquence.

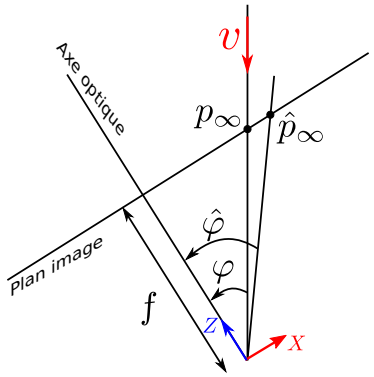


FIGURE 6.3 – Quelle précision de détection du foyer d'expansion est nécessaire pour une caméra perspective, avec une précision angulaire et un angle de lacet φ donnés ?

Trois images, issues de chaque séquence, sont illustrées par les Figures 6.4a,(c),(e). La première ligne de la Figure 6.4 correspond à la séquence où la caméra est orientée vers la gauche. La deuxième correspond à la séquence pour laquelle l'axe optique de la caméra est quasiment aligné avec l'axe de translation. Pour la troisième ligne, la caméra est orientée vers la droite. Les images 6.4b,(d),(f) illustrent les faisceaux de droites permettant d'estimer le foyer d'expansion. L'algorithme proposé renvoie les angles de lacet de $-9.64°$, $0.37°$ et $10.37°$. Avec le banc de métrologie, on conclue donc que l'angle est relativement estimé à $0.01°$ près.

6.2.3 Perspectives

Si la caméra est embarquée sur un véhicule, la transposition directe de cette méthode fonctionne mal. En effet, les vibrations dues aux suspensions induisent du bruit sur les points suivis. Il faudrait ajouter une étape de filtrage pour que ce bruit ait moins de répercussions sur la détection du foyer d'expansion. De plus, nos véhicules peuvent avoir un offset (inconnu) sur l'actionneur de la direction de l'essieu avant. Ce biais induit une faible courbure à la trajectoire du véhicule qui est commandé pour avancer en ligne droite. Il nous faut donc mettre en œuvre une méthode qui n'est plus spécifique à un mouvement donné. Il serait envisageable de généraliser cette approche basée sur le flot optique grâce à l'équation générale du mouvement (valable pour un mouvement quelconque). Toutefois, dans la section suivante, nous proposons une seconde approche qui utilise les résultats de la Section 5.3 pour déterminer l'orientation de la caméra par rapport au repère cinématique du véhicule.

FIGURE 6.4 – Détermination partielle de l'orientation de la caméra par rapport au véhicule se déplaçant en ligne droite, à partir du foyer d'expansion (pour trois séquences d'images).

6.3 Modèle cinématique et étalonnage “essieu-caméra”

Dans cette section, nous cherchons à déterminer l'orientation d'une caméra dans le repère cinématique du véhicule. Contrairement à la section précédente, nous n'imposons plus que le véhicule se déplace en ligne droite. Nous étudierons comment intégrer dans la méthode les informations issues d'autres capteurs comme les odomètres.

6.3.1 Principe

Le but principal est de déterminer l'orientation d'une caméra dans le repère cinématique du véhicule. Nous verrons qu'il est également possible d'estimer partiellement la position de la caméra par rapport à ce repère. L'idée de l'étalonnage proposé consiste à confronter la trajectoire de la caméra (estimée au préalable) avec sa trajectoire obtenue de proche en proche via le modèle cinématique d'évolution du véhicule.

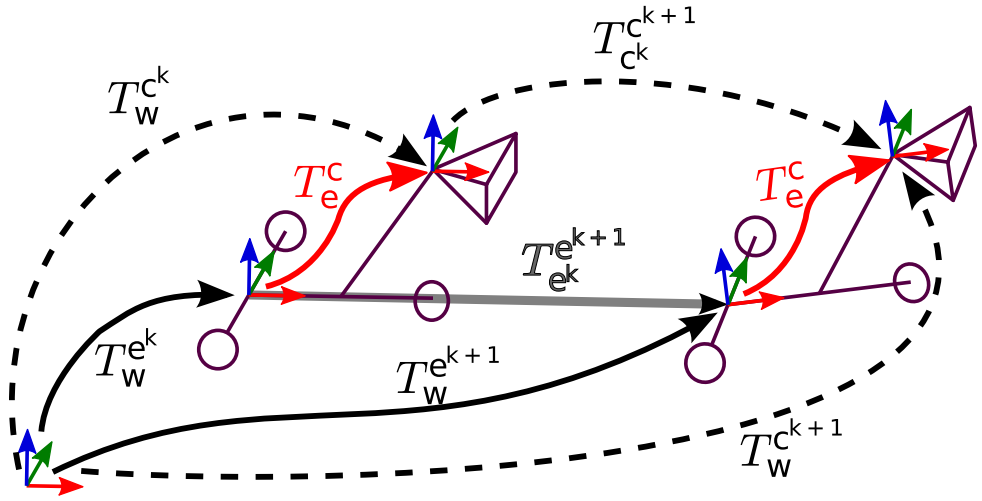


FIGURE 6.5 – Schéma illustrant un véhicule doté d'une caméra se déplaçant entre les instants k et $k+1$. Les poses de l'essieu et de la caméra sont définies dans un repère monde.

On considère les $K+1$ poses de la caméra (et donc les K mouvements) enregistrées aux instants $\{\tau_k\}_{k\in[\![0..K]\!]}$. La Figure 6.5 schématise le déplacement d'un véhicule et de sa caméra (dessinés en violet) entre deux instants k et $k+1$. Le repère associé au véhicule est situé sur le centre de l'essieu arrière. La convention pour le repère du véhicule est : $\mathbf{x}$ vers l'avant, $\mathbf{y}$ vers la gauche et $\mathbf{z}$ vers le haut. Afin de ne pas surcharger les équations, cette convention est également utilisée pour les caméras dans cette section. La Figure représente plusieurs transformations homogènes entre différentes entités comme : le repère monde (noté w), le repère du véhicule à l'instant k (noté e^k) et le repère de la caméra à l'instant k noté c^k.

Les données fournies comme entrées de l'algorithme d'étalonnage sont :

– les poses $\hat{T}_w^{c^k}$ de la caméra (exprimées dans un repère monde) obtenues par une reconstruction euclidienne (dans notre cas, avec un algorithme de type SFM avec un système multi-caméra étalonné, *cf* Section 5.3),
– une pose initiale T_e^c de la caméra par rapport au repère véhicule. La translation $\mathbf{t}_e^c$ entre l'essieu et la caméra doit être mesurée (quand elle n'est pas optimisée). La rotation R_e^c peut être approximative (puisqu'elle sera toujours estimée),
– d'éventuelles mesures odométriques v_o^k sur l'essieu arrière,
– d'éventuelles mesures δ_{fo}^k de l'angle de braquage.

Les paramètres à déterminer avec l'étalonnage sont :
– la pose T_e^c entre l'essieu arrière et la caméra,
– les K vitesses instantanées v^k du véhicule (pour $k \in [\![0..K-1]\!]$) ,
– les K rotations $R_{e^k}^{e^{k+1}}$ de l'essieu arrière entre les instants k et $k+1$.
– la première pose $T_w^{e^0}$ de l'essieu arrière dans le repère monde [1].
– des coefficients sur le gain et le biais des capteurs odométrique et d'angle au volant.

Cinématique : Soit $T_{e^k}^{e^{k+1}}$ la transformation homogène de la pose de l'essieu arrière entre les instants k et $k+1$. $T_{e^k}^{e^{k+1}}$ modélise donc le mouvement du véhicule (donné par son modèle cinématique) et est définie par l'équation 6.5.

$$T_{e^k}^{e^{k+1}} \stackrel{def}{=} \begin{pmatrix} & v^k \tau_k^{k+1} \\ R_{e^k}^{e^{k+1}} & 0 \\ & 0 \\ 0_{1\times 3} & 1 \end{pmatrix} \tag{6.5}$$

Où le temps écoulé entre la pose k et $k+1$ est donné par $\tau_k^{k+1} \stackrel{def}{=} \tau_{k+1} - \tau_k$.

Pour un mouvement général (3D), on décompose la rotation $R_{e^k}^{e^{k+1}}$ selon successivement les axes $\mathbf{z}$, $\mathbf{y}$ et $\mathbf{x}$ du repère e^k (pour les angles de lacet φ_k^{k+1}, de tangage ψ_k^{k+1} et de roulis ϕ_k^{k+1}), comme le montre l'équation 6.6.

$$R_{e^k}^{e^{k+1}} = R_x(\phi_k^{k+1}) R_y(\psi_k^{k+1}) R_z(\varphi_k^{k+1}) \tag{6.6}$$

Hypothèses : Or, on suppose par la suite que le véhicule évolue sur un sol plat. Pour modéliser le mouvement plan du véhicule, seuls les angles de lacet φ_k^{k+1} seront estimés (les autres angles sont supposés nuls).

Sous l'hypothèse de roulement sans glissement et en négligeant la dynamique, on peut représenter un véhicule à 4 roues (2 roues arrière non-orientables et 2 roues avant directrices) par le modèle d'évolution de type tricycle. Ce modèle est cinématiquement équivalent au modèle unicycle et bicyclette (également appelé modèle d'Ackermann) lorsque la vitesse est non-nulle. La Figure 6.6 illustre la cinématique du véhicule à l'instant k. Soit L la distance entre les essieux

1. Optimiser cette pose permet de répartir les erreurs sur toutes les poses de caméra.

avant et arrière (appelée empattement). On écrit les équations de la cinématique pour le centre de l'essieu arrière, en considérant que le mouvement du véhicule est une rotation instantanée autour de l'axe de rotation $\mathbf{n}_k$. En continu, la dérivée $\dot{\varphi}$ de l'orientation du véhicule s'exprime :

$$\dot{\varphi} = \frac{v}{L}\tan(\delta) \tag{6.7}$$

En discret et avec une approximation au premier ordre ($\dot{\varphi}_k \simeq \frac{\varphi_k^{k+1}}{\tau_k^{k+1}} = \frac{\varphi_{k+1}-\varphi_k}{\tau_k^{k+1}}$), on obtient :

$$\varphi_{k+1} = \varphi_k + \frac{v^k \tau_k^{k+1}}{L}\tan(\delta_k) \tag{6.8}$$

Si la vitesse est non-nulle, l'équation 6.8 est équivalente à l'équation 6.9.

$$\delta_k = \tan^{-1}\left(\frac{L}{v^k \tau_k^{k+1}}(\varphi_{k+1} - \varphi_k)\right) \tag{6.9}$$

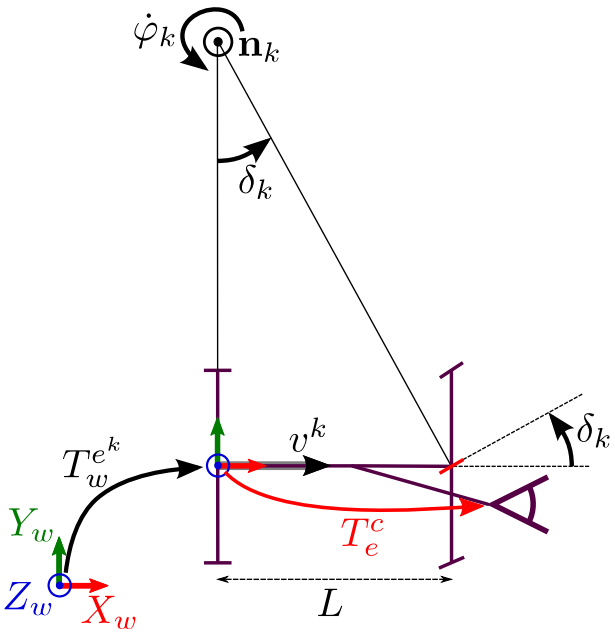


FIGURE 6.6 – Schéma illustrant le modèle cinématique du véhicule. Il s'agit du modèle tricycle. La vitesse longitudinale du véhicule est v^k, l'angle de braquage virtuel des roues avant est δ_k.

Etalonnage uniquement à partir des données vision : Notre approche consiste à minimiser l'erreur entre la trajectoire de la caméra reconstruite par vision (dont les poses sont notées $\hat{T}_c^k$),

et la trajectoire de la caméra estimée par le modèle cinématique. La fonction de coût f_{vision} que l'on minimise est définie par :

$$f_{\text{vision}} \overset{def}{=} \sum_{k'=0}^{K} \left\| \boldsymbol{\epsilon} \left(T_w^{e^0} \left(\prod_{k=0}^{k'-1} T_{e^k}^{e^{k+1}} \right) T_e^c \ , \ \hat{T}_w^{c^{k'}} \right) \right\|^2 \tag{6.10}$$

$\boldsymbol{\epsilon}(T_1, T_2)$ renvoie la concaténation de l'erreur en translation et en rotation entre deux transformations euclidiennes. Plus précisément, $\boldsymbol{\epsilon}(T_1, T_2)$ concatène l'erreur en translation $\boldsymbol{\epsilon}(\mathbf{t_1}, \mathbf{t_2}) \overset{def}{=} \mathbf{t_1} - \mathbf{t_2} \in \mathbb{R}^3$ et l'erreur en rotation $\boldsymbol{\epsilon}(R_1, R_2) \in \mathbb{R}^3$ (le vecteur des angles d'Euler de $R_1^\top R_2$).

Remarque : On ne peut pas prendre pour $\boldsymbol{\epsilon}(R_1, R_2)$ la distance $d(R_1, R_2)$ (définie par l'équation 4.11) qui ne renvoie qu'un angle positif, car cette valeur positive ne contraindrait pas le sens de l'optimisation. De plus, comme la fonction choisie renvoie 3 angles, on obtient plus de contraintes.

Intégration facultative de l'odométrie de l'essieu arrière et/ou des mesures de l'angle de braquage Afin d'améliorer l'estimation de la trajectoire et de l'étalonnage essieu-caméra, on peut rajouter des informations issues d'autres capteurs, comme des odomètres. Comme nous le verrons, cela permettra d'estimer également la position $\mathbf{t}_e^c$ de la caméra par rapport au repère cinématique du véhicule. Considérons le cas du VipaLab, qui est équipé d'un odomètre moteur (selon la formule 6.11), et d'un capteur mesurant l'angle de braquage des roues avant (selon la formule 6.12). On suppose que ces mesures (odométriques et angle de braquage) sont disponibles aux instants $\tau = \tau_k$ (*c.-à-d.* aux mêmes dates que les poses de la caméra).

$$v^k = \alpha_r v_o^k \tag{6.11}$$

$$\delta_f^k = \alpha_f \delta_{fo}^k + \beta_f \tag{6.12}$$

v^k correspond à la vitesse de l'essieu arrière et v_o^k à sa mesure odométrique. Le coefficient non-nul α_r (sans unité) est le rapport du rayon réel de la roue sur le rayon estimé. δ_f^k est l'angle de braquage avant (de la roue du modèle tricycle) et δ_{fo}^k sa mesure odométrique. Ces derniers sont reliés par une loi affine. α_f est un réel non-nul sans unité. Le coefficient β_f correspond à l'offset de commande sur l'angle de braquage. La commande du véhicule utilise la mesure odométrique afin d'asservir l'angle de braquage. En négligeant la phase transitoire de l'asservissement, on peut considérer que les paramètres α_f et β_f liés à la mesure sont également les paramètres nécessaires à la commande.

Soit f_{odo} la fonction de coût liée aux erreurs de mesures odomériques. Elle concatène les résidus $\frac{v^k}{\alpha_r} - v_o^k$ lorsque l'odométrie arrière est disponible. Soit f_{braq} la fonction de coût liée aux erreurs de mesures d'angle de braquage. Elle concatène les résidus les résidus $\frac{\delta_f^k - \beta_f}{\alpha_f} - \delta_{fo}^k$ lorsque les mesures d'angle de braquage sont disponibles.

Si une connaissance *a priori* sur l'écart-type des résidus (de f_{vision}, f_{odo} et f_{braq}) est disponible, il suffit de diviser les résidus par leurs écarts-types respectifs afin que l'estimateur proposé soit maximum de vraisemblance (en supposant que les bruits sur les mesures soient gaussiens).

Enoncé de l'étalonnage proposé : L'algorithme 10 récapitule l'étalonnage "essieu-caméra" pouvant intégrer des mesures odométriques et/ou d'angle de braquage.

Algorithme 10: Étalonnage essieu-caméra

Entrées : $\hat{T}_w^{c^k}$, pose initiale T_e^c, τ_k, facultativement : v_o^k et δ_{fo}^k
Sorties : T_e^c, $R_{e^k}^{e^{k+1}}$, v^k, facultativement : α_r, α_f et β_f

$1^{ère}$ *étape : Initialisation :*

Pour chaque *instant* $k \in [\![0..K]\!]$ **:**
Calculer la pose de l'essieu.

En déduire les K vitesses v^k et les $R_{e^k}^{e^{k+1}}$

$2^{ème}$ *étape : Optimisation non-linéaire :*

$$\operatorname{argmin} \left(f_{\text{vision}} + \underbrace{f_{odo} + f_{braq}}_{\text{facultatifs}}\right) \tag{6.13}$$

L'initialisation de l'algorithme 10 dépend des capteurs présents. Si on ne dispose que des poses de la caméra, et d'une valeur approximative de T_e^c, on déduit une initialisation des poses de l'essieu avec $T_w^{e^k} = \hat{T}_w^{c^k}(T_e^c)^{-1}$. Si l'odométrie de l'essieu arrière et les mesures de l'angle de braquage sont disponibles (avec une valeur approximative du rayon des roues et des coefficients α_f et β_f), on peut également initialiser de proche en proche les poses $T_w^{e^k}$.

L'optimisation non-linéaire est effectuée avec l'algorithme de Levenberg-Marquardt. La fonction de coût totale que l'on cherche à minimiser provient d'un bruit sur les poses caméras obtenues par vision, et de l'approximation de la dynamique du véhicule par un modèle cinématique (ex. vibrations dues aux suspensions). Si d'autres capteurs (odomètre, capteur de l'angle de braquage) sont utilisés, nous cherchons à minimiser leurs bruits de détection.

Interprétation On peut interpréter cette minimisation comme une optimisation sous la contrainte de non-holonomie imposée par la cinématique du véhicule. Cette contrainte est respectée par construction de la fonction de coût (*cf* équation 6.10). A titre indicatif, en absence de bruit sur les données, les équations 6.14 doivent être vérifiées :

$$\forall k \in [\![0..K-1]\!],\ T_{e^k}^{e^{k+1}} T_e^c = T_e^c T_{c^k}^{c^{k+1}} \tag{6.14}$$

Ces équations, obtenues par un simple changement de base, rappellent les équations 5.3 vues pour l'initialisation linéaire de l'étalonnage multi-caméra (Section 5.2.2). Les différences apparaissent sur : les matrices $T_{e^k}^{e^{k+1}}$ qui sont à estimer à l'aide de la connaissance du modèle cinématique (et éventuellement de l'odométrie et des mesures de l'angle de braquage), et sur la matrice T_e^c dont la translation $\mathbf{t}_e^c$ est supposée totalement ou partiellement connue.

Quels mouvements faut-il utiliser pour étalonner ? Tout comme l'étude des cas singuliers pour l'étalonnage camera/caméra, une translation pure d'axe $\mathbf{t}$ est un mouvement singulier : on

ne peut observer ni la translation de T_e^c , ni l'angle de rotation autour de l'axe $\mathbf{t}$. De même, pour des mouvements plans, on ne peut pas observer la hauteur de la caméra par rapport au véhicule. Afin de parcourir l'ensemble des états admissibles par la commande du véhicule, on utilise pour l'étalonnage une trajectoire où le véhicule s'est déplacé en ayant braqué totalement à gauche et à droite.

6.3.2 Résultats

Nous présentons ci-dessous des résultats de l'étalonnage "essieu-caméra" avec des données synthétiques et réelles. Les données synthétiques ont été générées soit manuellement, soit avec un simulateur fournissant directement des données (images) réalistes par rapport aux expériences réalisées sur le véhicule VipaLab.

6.3.2.1 Résultats avec des données synthétiques

Données synthétiques générées manuellement Pour tester l'étalonnage avec une vérité terrain, nous avons généré une trajectoire de synthèse de 31 poses. Le véhicule et sa caméra suivent les trajectoires illustrées par la Figure 6.7. La caméra est située 2 m vers l'avant du véhicule ($\mathbf{t}_e^c = (2m\ 0\ 2m)^\top$). Qualitativement, sa trajectoire amplifie celle de l'essieu, par une sorte de bras de levier.

La trajectoire de l'essieu n'est pas fournie à l'algorithme d'étalonnage. Un bruit est appliqué aux poses $\hat{T}_w^{c^k}$ de la caméra. Pour cela un bruit additif gaussien est appliqué sur chaque composante des translations $\mathbf{t}_w^{c^k}$ (avec un écart-type de $1\ cm$). Chaque matrice de rotation $\hat{R}_w^{c^k}$ est multipliée par une matrice de rotation créée avec trois angles d'Euler générés aléatoirement avec un bruit gaussien d'écart-type $0.01°$. Lorsque les données odométriques sont utilisées, elles sont préalablement bruitées (avec un écart-type de $1.5\ cm/s$ pour l'odométrie de l'essieu arrière, et de $0.1°$ pour l'angle de braquage δ_{fo}).

Une fois bruitées, ces données sont prises en compte pour étalonner soit uniquement à partir de la vision (*c.-à-d.* les poses $\hat{T}_w^{c^k}$), soit avec la vision, l'odométrie de l'essieu arrière et/ou des mesures de l'angle de braquage. Le tableau 6.1 récapitule les valeurs théoriques, initiales et finales des grandeurs utilisées. Nous avons renouvelé cette expérience en estimant également la position de la caméra par rapport à l'essieu. Seule la hauteur de la caméra est supposée connue : les deux premières composantes de $\mathbf{t}_e^c$ sont maintenant optimisées (elles sont initialisées avec une erreur de $20cm$). De même, le tableau 6.2 récapitule les résultats des étalonnages (en optimisant cette fois la position latérale et longitudinale de la caméra). Les optimisations non-linéaires (pour les deux tableaux) ont convergé en entre 4 et 6 itérations (sauf celle du tableau 6.2, sans mesures odométrique ni d'angle au volant, qui a convergé en 13 itérations).

Le tableau 6.1 montre que les données vision suffisent pour déterminer l'orientation de la caméra par rapport au repère cinématique du véhicule (pour une trajectoire plane), à moins de $0.2°$ près (relativement aux bruits réalistes des données d'entrée). Si des mesures odométriques

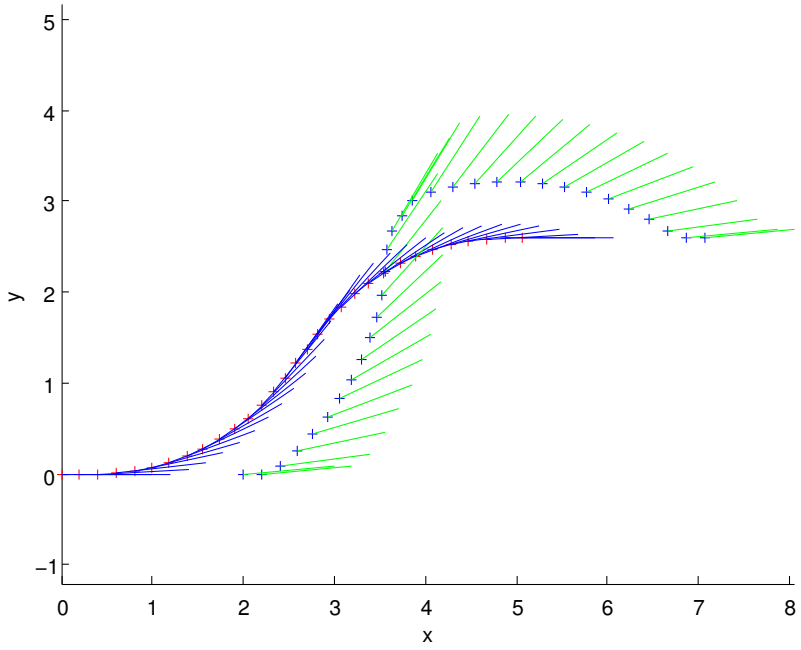


FIGURE 6.7 – Vue de dessus de la trajectoire synthétique de l'essieu du véhicule (dessinée avec des croix rouges et des caps bleus) et de la trajectoire suivis par la caméra (dessinée avec des croix bleues et des axes optiques verts). Le mouvement est plan, mais la pose de la caméra par rapport au véhicule est bien une transformation euclidienne de l'espace ($\mathbf{t}_e^c = (2m\ 0\ 2m)^\top$ et $R_e^c = \rho((5°\ 5°\ 5°)^\top)$).

ou d'angle de braquage sont disponibles, alors les inconnues introduites (les coefficients α_r, α_f et β_f) sont également estimés précisément par l'algorithme.

Le tableau 6.2 montre que si aucune information odométrique n'est disponible, alors les données visions seules ne permettent pas de contraindre la position latérale de la caméra. Au contraire, si on prend en compte des données odométriques (de l'essieu arrière et/ou de l'angle de braquage) alors la position (latérale et longitudinale) de la caméra est également estimable (ici, à moins d'un centimètre près).

Données générées par un simulateur Afin d'avoir une vérité terrain réaliste, nous avons utilisé une séquence d'images de synthèse générées avec le simulateur développé par [Mal11] et [Del11]. De la même manière que les VIPA ou VipaLab, un véhicule synthétique (*cf* Figure 6.8) est équipée de deux caméras (aux paramètres intrinsèques et extrinsèques connus). Il suit une séquence sur un sol plat, pendant laquelle on a successivement braqué totalement à gauche et à droite. Les images avant et arrière sont enregistrées à 15 fps. L'algorithme de sélection des poses clés et de SfM multi-caméra à paramètres extrinsèques constants (*cf* Section 5.3) est appliqué. Les poses estimées de la caméra avant sont utilisées comme entrée de notre algorithme d'étalonnage "essieu-camera". La position de la caméra par rapport à l'essieu du véhicule (juste

Variables	R_e^c	α_r	α_f	β_f
Valeurs théoriques	$\xi_e^c = (5^\circ\ 5^\circ\ 5^\circ)^\top$	1	1.3	3°
Valeurs initiales	$\xi_e^c = (0^\circ\ 0^\circ\ 0^\circ)^\top$	0.9	1	0°
Données prises en compte	Valeurs finales (ou erreur commise)			
Vision	$d(R_e^c, \hat{R}_e^c) = 0.19^\circ$	n/a	n/a	n/a
Vision + odométrie arrière	$d(R_e^c, \hat{R}_e^c) = 0.11^\circ$	0.9987	n/a	n/a
Vision + angle de braquage	$d(R_e^c, \hat{R}_e^c) = 0.19^\circ$	n/a	1.3082	3.07°
Vision + odométrie arrière + angle de braquage	$d(R_e^c, \hat{R}_e^c) = 0.09^\circ$	0.9993	1.3023	3.03°

TABLE 6.1 – Résultats de l'étalonnage "essieu-caméra" sur une trajectoire de synthèse. Les données prises en compte sont soit uniquement la vision (*c.-à-d.* les poses $\hat{T}_w^{c^k}$), soit la vision avec les mesures odométriques et/ou d'angle de braquage. L'orientation R_e^c de la caméra est toujours estimée, ainsi que les coefficients odométriques dès que possible.

la translation) est supposée connue. L'expérience est répétée pour trois orientations R_e^c différentes de la caméra avant (et donc trois séquences différentes). Pour la première, la caméra est parfaitement dirigée vers l'avant ($R_e^c = I_3$). Pour la seconde, la caméra avant est légèrement orientée vers droite (avec 5° de lacet). Enfin, pour la dernière 5° de lacet, 5° de roulis puis 5° de tangage sont appliqués. Pour ces 3 configurations, la rotation initiale R_e^c est initialisée avec des angles nuls et l'optimisation non-linéaire a convergé en 4 itérations. Le tableau 6.3 récapitule les résultats. La rotation R_e^c est retrouvée avec une précision inférieure à 0.35°.

FIGURE 6.8 – Expériences à l'aide du simulateur.

Variables	$\mathbf{t}_e^c$	R_e^c	α_r	α_f	β_f
Valeurs théoriques	$\mathbf{t}_e^c = \begin{pmatrix} 2m \\ 0 \\ 2m \end{pmatrix}$	$\xi_e^c = (5° \; 5° \; 5°)^\top$	1	1.3	3°
Valeurs initiales	$\mathbf{t}_e^c = \begin{pmatrix} 2.2m \\ 0.2m \\ 2m \end{pmatrix}$	$\xi_e^c = (0° \; 0° \; 0°)^\top$	0.9	1	0°
Données prises en compte	Valeurs finales (ou erreur commise)				
Vision	$\mathbf{t}_e^c = \begin{pmatrix} 2.003m \\ 1.455m \\ 2m \end{pmatrix}$	$d(R_e^c, \hat{R}_e^c) = 0.86°$	n/a	n/a	n/a
Vision + odométrie arrière	$\mathbf{t}_e^c = \begin{pmatrix} 2.006m \\ 0.003m \\ 2m \end{pmatrix}$	$d(R_e^c, \hat{R}_e^c) = 0.12°$	0.9987	n/a	n/a
Vision + angle de braquage	$\mathbf{t}_e^c = \begin{pmatrix} 2.006m \\ 0.003m \\ 2m \end{pmatrix}$	$d(R_e^c, \hat{R}_e^c) = 0.19°$	n/a	1.3080	3.05°
Vision + odométrie (arrière et angle de braquage)	$\mathbf{t}_e^c = \begin{pmatrix} 2.007m \\ 0.003m \\ 2m \end{pmatrix}$	$d(R_e^c, \hat{R}_e^c) = 0.10°$	0.9993	1.3021	3.02°

TABLE 6.2 – Résultats de l'étalonnage "essieu-caméra" sur une trajectoire de synthèse. Les données prises en compte sont soit uniquement la vision (*c.-à-d.* les poses $\hat{T}_w^{c^k}$), soit la vision avec les mesures odométriques et/ou d'angle de braquage. L'orientation R_e^c de la caméra et les deux premières composantes de $\mathbf{t}_e^c$ sont toujours estimées, ainsi que les coefficients odométriques dès que possible.

Configuration	n°1	n°2	n°3
Erreur angulaire $d(R_e^c, \hat{R}_e^c)$	0.22°	0.08°	0.34°
Erreur maximale en translation sur les poses des caméras $\boldsymbol{\epsilon}(\mathbf{t}_w^{c^k}, \mathbf{t}_w^{c^k})$ (cm)	2.4	1.5	1.5
Erreur maximale en rotation sur les poses caméra $\boldsymbol{\epsilon}(R_e^c, \hat{R}_e^c)$	0.05°	0.08°	0.11°

TABLE 6.3 – Résultats sur la précision de la rotation $\hat{R}_e^c$ estimée par l'étalonnage sur trois configurations. Les séquences d'images sont obtenues avec le simulateur.

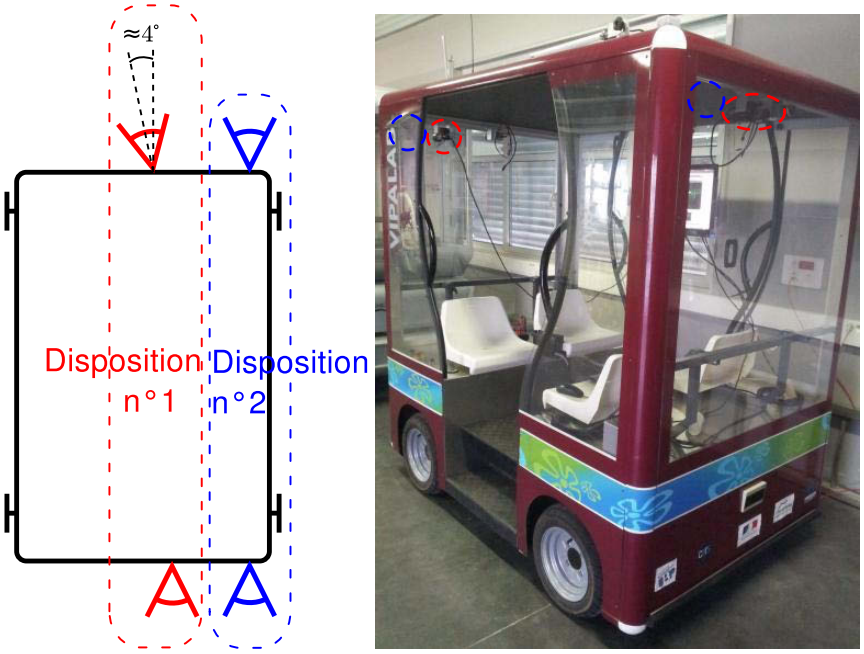
6.3.2.2 Résultats avec des données réelles

Nous allons présenter des étalonnages du VipaLab réalisés sur la plateforme PAVIN. La procédure ci-après démontre expérimentalement la pertinence de la méthode d'étalonnage pour des applications de navigation autonome.

Protocole : L'absence de vérité terrain nous impose de mettre en place un protocole expérimental particulier :

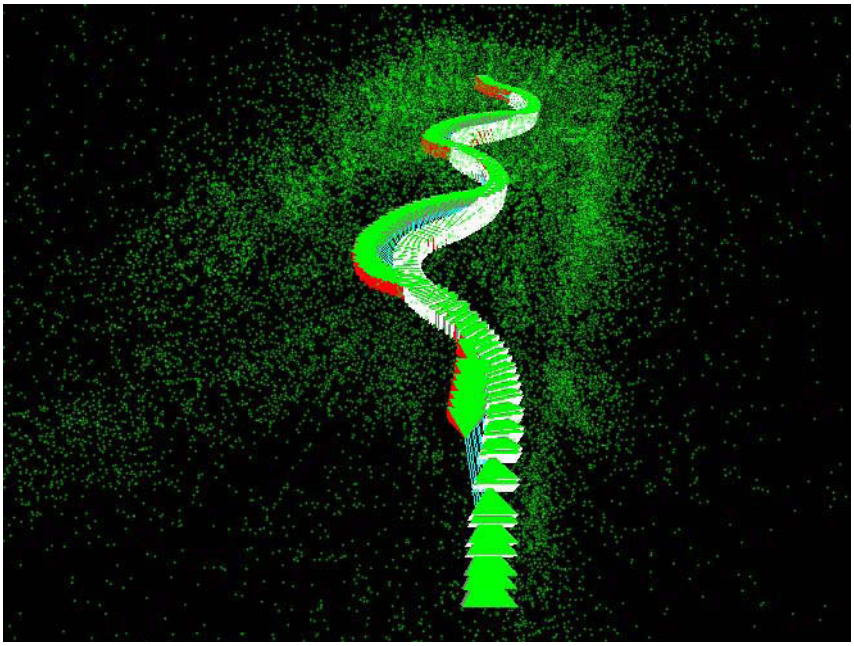
- La paire de caméras à champs disjoints est placée dans la disposition n°1 (vers le milieu, *cf* Figure 6.9).
  1. Le véhicule est conduit manuellement sur une trajectoire d'apprentissage (présentant un demi-tour). Cette trajectoire sera dénommée *trajectoire de référence* 1_{ref}.
  2. La trajectoire de référence est apprise en reconstruisant l'environnement 3D et en calibrant extrinsèquement les caméras à l'aide de l'algorithme présenté dans la section 5.3.
  3. Le véhicule avance en zigzag (sur un sol plat) en braquant au maximum à gauche puis à droite.
  4. Cette trajectoire est reconstruite (*cf* Figure 6.10) avec les paramètres extrinsèques fixes déterminés au point n° 2. Puis, on effectue l'étalonnage essieu-caméra.
  5. Avec les paramètres obtenus avec ces 2 étalonnages, on navigue automatiquement en rejouant la trajectoire de référence. Un second rejeu est réalisé dans des conditions totalement identiques. Ces rejeux sont dénommés 1 et $1'$.
  6. Un troisième rejeu (1_b) est fait avec une orientation R_e^c volontairement erronée. L'erreur est de $-2°$ d'angle de lacet (*c.-à-d.* on pose $R_e^c = \hat{R}_e^c R_z(-2°)$, où $\hat{R}_e^c$ est la rotation estimée par l'étalonnage).
- La paire de caméras à champs disjoint est placée dans la disposition n°2 (vers le côté droit du véhicule, *cf* Figure 6.9).
  1. On étalonne extrinsèquement les caméras (à l'aide d'une séquence similaire à la trajectoire de référence).
  2. Un second étalonnage essieu-caméra est obtenu sur une deuxième trajectoire en zigzag.
  3. On rejoue la trajectoire de référence 1_{ref} (enregistrée avec la disposition n°1). Ce rejeu est noté 2.

Pour tous ces enregistrements suivants, une seule personne était à la même place dans le véhicule (qu'il soit conduit manuellement ou automatiquement). L'inclinaison du véhicule était donc sensiblement identique sur les différentes trajectoires. Pour tous les rejeux en navigation automatique, les consignes de commande (comme la vitesse maximale du véhicule) étaient les



(a) (b)

FIGURE 6.9 – (a) Vue de dessus du véhicule, avec les deux dispositions pour la paire de caméras à champs non-recouvrant. Pour la disposition n°1, la caméra avant est légèrement orientée vers la gauche (avec angle de lacet d'environ $4°$). (b) VipaLab avec les caméras Marlin™ dans la disposition n°1 (rouge), les caméras sont ensuite déplacées dans la position n°2 (bleu).



(a)

(b)

(c)

FIGURE 6.10 – (a) Vue 3D de la reconstruction de la trajectoire en zigzag servant à l'étalonnage essieu-caméra (pour la disposition n°1). Exemples d'images avant (b) et arrière (c) issues de la séquence.

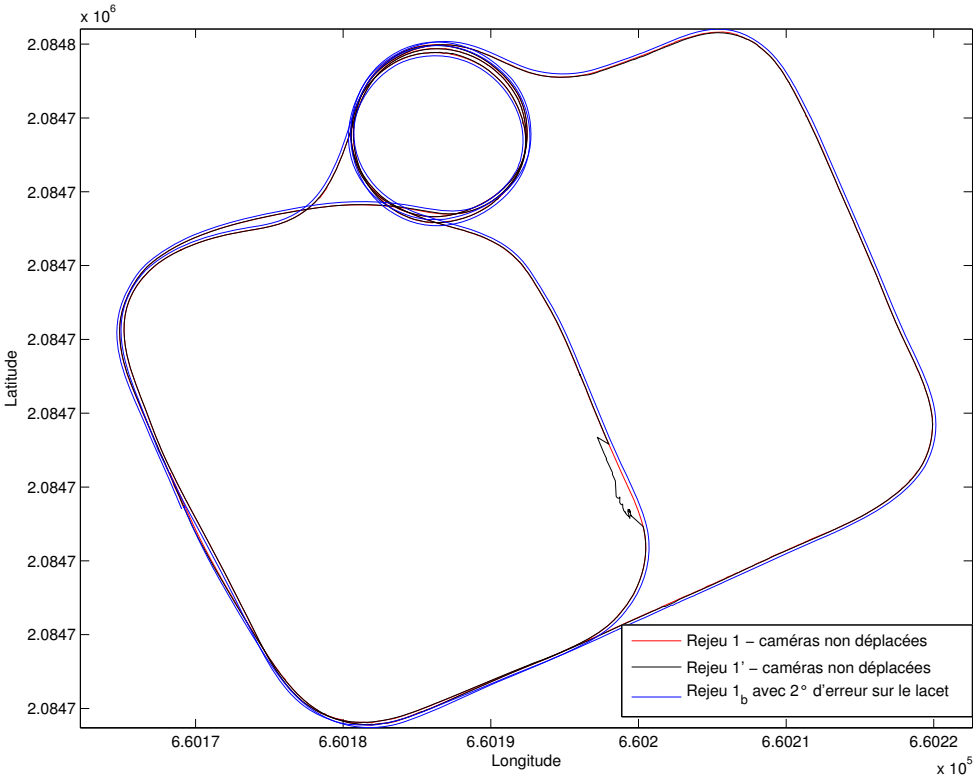
même et seules les informations issues des caméras sont utilisées (on ne fusionne pas les données odométriques ni les données GPS).

Pour les étalonnages essieu-caméra, l'algorithme utilise les poses de la caméra avant. Pour chaque disposition, la translation $\mathbf{t}_e^c$ entre le milieu de l'essieu arrière et la caméra avant a été mesurée à l'aide d'un mètre. Ces translations étant connues, nous ne les avons pas estimées lors des étalonnages essieu-caméra. Les deux étalonnages ont convergé en 4 itérations (en initialisant les angles d'euler ξ_e^c à $(0^\circ\ 0^\circ\ 0^\circ)^\top$). L'expérience va montrer si l'orientation R_e^c de la caméra a été déterminée avec suffisamment de précision.

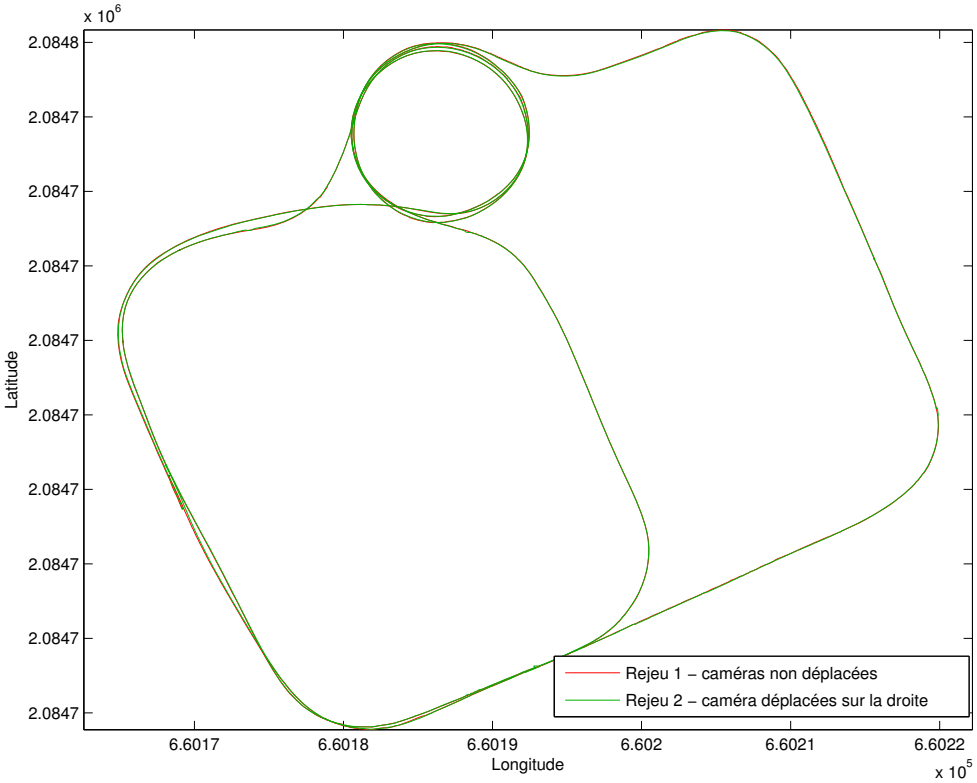
Résultats : Le signal DGPS a été enregistré pour la trajectoire de référence et ses différents rejeux. Pour comparer deux traces DGPS tr_1 et tr_2, on étudie leur écart latéral. Pour cela, on exprime tout d'abord leurs coordonnées en Lambert II étendu. Pour chaque point (X_1, Y_1) de tr_1, on trouve le point (X_2, Y_2) de tr_2 le plus proche dans un intervalle donné. On calcule ensuite l'écart latéral (distance signée) entre les 2 points en approximant localement la normale à la courbe tr_2.

- Les comparaisons de la trajectoire de référence 1_{ref} et les rejeux 1 ou 2 permettent de quantifier les erreurs dues à la commande et la localisation. Cette erreur est donnée à titre indicatif, car l'objectif de cet étalonnage n'est pas de diminuer l'erreur due à la commande. L'erreur de commande est vraisemblablement due à l'approximation d'une trajectoire globalement plane. Les futures versions de la commande seront améliorées en faisant une approximation d'une trajectoire localement plane.
- Les rejeux 1 et $1'$ servent à quantifier la précision de la navigation autonome pour des paramètres donnés. On s'affranchit donc des erreurs de commande, et l'écart latéral s'interprète comme une erreur de localisation (à laquelle s'ajoute du bruit de mesure dû aux vibrations et au DGPS). L'écart latéral vaut en moyenne $1.09\ cm$ (avec un écart-type de $8.8\ cm$). La Figure 6.11a illustre en rouge et en noir les traces GPS de 1 et $1'$ (qui sont quasiment superposées). L'irrégularité sur la trace GPS du rejeux $1'$ est simplement due à une baisse locale de la réception du signal DGPS.
- Les rejeux 1 (ou $1'$) et 1_b servent à apprécier quantitativement l'influence de l'étalonnage essieu-caméra sur la trajectoire rejouée. La Figure 6.11a illustre le décalage entre les traces GPS de 1 (en rouge) et de 1_b (en bleu). L'écart latéral vaut en moyenne $-19.87\ cm$, avec un écart-type de $2.3\ cm$.
- L'interprétation des rejeux 1 et 2 est la plus importante dans cette expérience. Ils permettent de vérifier si l'étalonnage permet au véhicule de repasser au même endroit avec deux dispositions différentes de la paire de caméras. La Figure 6.11b illustre le faible décalage entre les traces GPS de 1 (en rouge) et de 2 (en vert). L'écart latéral vaut en moyenne $1.12\ cm$ avec un écart-type de $2.1\ cm$.

Un zoom sur de la partie supérieure droite des trajectoires 1, $1'$, 1_b et 2 est illustré par la Figure 6.12.



(a) Rejeux 1 (en rouge) et $1'$ (en noir) dans les mêmes conditions (traces quasiment identiques). Rejeu 1_b (en bleu) avec une orientation volontairement biaisée (le véhicule roulait en moyenne environ $20\ cm$ trop à droite par rapport au rejeu 1).



(b) Rejeu 1 (en rouge) sans avoir déplacé les caméras. Rejeu 2 (en vert) après avoir déplacé et ré-étalonné les caméras. L'écart latéral moyen entre les deux traces vaut environ $1\ cm$.

FIGURE 6.11 – Traces GPS de différents rejeux de la trajectoire de référence 1_{ref}.

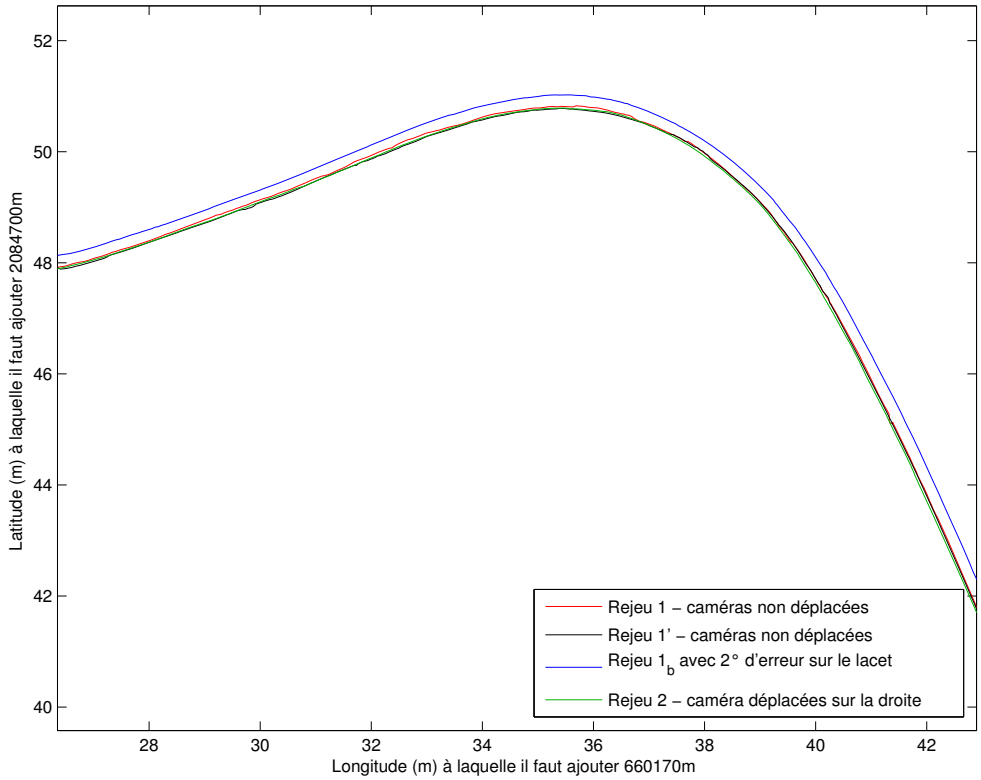


FIGURE 6.12 – Zoom sur les traces GPS de différents rejeux de la trajectoire de référence 1_{ref}. Rejeux 1 (en rouge) et $1'$ (en noir) dans les mêmes conditions (traces quasiment identiques). Rejeu 1_b (en bleu) avec une orientation volontairement biaisée (le véhicule roulait en moyenne environ 20 cm trop à droite par rapport au rejeu 1). Rejeu 2 (en vert) après avoir déplacé et ré-étalonné les caméras. L'écart latéral moyen entre les traces 1 et 2 vaut environ 1 cm.

Interprétation des résultats : On remarque qu'une estimation erronée de l'orientation de la caméra engendre un biais sur l'écart latéral moyen : $2°$ d'erreur (sur l'angle de lacet) engendre un décalage d'environ $20cm$. En effet, en indiquant que la caméra est plus orientée sur la droite par rapport à la réalité, si le véhicule est situé sur la trajectoire à suivre, il estimera que son cap est trop tourné vers la gauche (par rapport à la référence). La commande tente ensuite de minimiser l'erreur de cap (l'angle entre le cap estimé du véhicule et le cap à suivre) en faisant tourner le véhicule vers la droite. Comme la commande cherche également à minimiser l'écart latéral, on conclut qu'une caméra estimée trop sur tournée vers la droite par rapport à la réalité engendre effectivement un rejeu décalé sur la droite. L'écart latéral entre les rejeux 1 et 2 est du même ordre de grandeur que l'écart latéral entre les rejeux 1 et $1'$. Or, il est fortement improbable qu'un même biais soit appliqué aux estimations des matrices de rotation R_e^c pour les dispositions 1 et 2. Les estimations de la position du milieu de l'essieu arrière et le cap du véhicule ont donc été sensiblement identiques entre les rejeux 1 et 2 pour que la commande fasse naviguer le véhicule au même endroit. Nous pouvons donc conclure que les étalonnages se sont correctement déroulés avec une précision suffisante pour la navigation autonome du véhicule.

6.3.3 Conclusion et perspectives

Nous avons validé, via des simulations et des expériences réelles, l'étalonnage déterminant l'orientation (définie par trois angles) de la caméra dans le repère cinématique du véhicule uniquement à partir des données vision (pouvant être bruitées). Les simulations ont montré, que si des données odométriques sont disponibles, il est également possible d'estimer la position latérale et longitudinale de la caméra. Ainsi, il est possible que plusieurs véhicules ayant un emplacement de caméra légèrement différent puissent, une fois étalonnés, emprunter le même parcours lors d'un rejeu de la trajectoire de référence.

Il serait pertinent de détailler une initialisation linéaire (similaire à l'étalonnage bras-œil, ou à notre initialisation caméra/caméra) pour estimer T_e^c. D'autre part, la fonction de coût f_{vision} prend uniquement en compte les poses de la caméra avant. Mais elle pourrait considérer toutes les caméras disponibles : si on dispose d'au moins 2 caméras étalonnées (intrinsèquement et extrinsèquement), une fonction de coût uniquement métrique pourrait être suffisante. Cependant, la méthode d'étalonnage deviendrait alors dépendante des paramètres extrinsèques (de l'étalonnage extrinsèque multi-caméra) : une erreur sur les paramètres extrinsèques pourrait biaiser l'étalonnage essieu-caméra.

Dans la méthode proposée, on suppose que les mesures sont les poses bruitées de la caméra. En effet, les erreurs de détection des points d'intérêt se propagent aux poses des caméras. Il serait toutefois plus rigoureux de se baser sur les observations issues des images, comme par exemple avec un algorithme SfM avec un modèle cinématique qui minimise les erreurs de reprojection. Ainsi, l'optimisation des poses des caméras serait remplacée par l'optimisation de l'évolution du modèle cinématique. En intégrant la notion de temps d'acquisition dans cette approche, la synchronisation des caméras ne serait peut-être plus nécessaire. Les approches basées

EKF seraient en théorie bien adaptées comme l'explique [Fox02], où les étapes de reconstruction et d'étalonnage sont découplées.

Les expérimentations ont été effectuées en supposant que le mouvement est plan. Pour tendre vers un mouvement 3D, il suffirait d'optimiser également les angles de tangage ψ_k et de roulis ϕ_k. Cela revient à supposer que le mouvement est localement plan à chaque instant k. Des études supplémentaires sont nécessaires pour déterminer les singularités de cette approche. Toutefois, en guise de résultats préliminaires, nous avons testé cette variante 3D de l'étalonnage sur la trajectoire simulée (*cf* Section 6.3.2.1). On constate que si la translation $\mathbf{t}_e^c$ est connue et constante, alors l'odométrie de l'essieu arrière et/ou les mesures de l'angle de braquage sont nécessaires pour que l'algorithme converge (vers des précisions similaires au tableau 6.1). Pour cette variante 3D, l'odométrie et les mesures de l'angle de braquage ne semblent pas suffire à contraindre le problème si les deux premières composantes de $\mathbf{t}_e^c$ sont à estimer. Au lieu de tendre vers un mouvement 3D, une autre possibilité serait de considérer des sous-ensembles de la trajectoire 3D où les mouvements sont coplanaires (tout comme dans [RLPK10]).

D'autres expériences avec des données réelles sont nécessaires avec des données vision et odométriques pour valider l'étalonnage déterminant à la fois l'orientation de la caméra et sa position latérale et longitudinale. Il serait également possible d'intégrer des données issues un GPS (bas-coût et/ou différentiel) dans l'algorithme afin de déterminer sa position (latérale et longitudinale) dans le repère du véhicule (tout en géoréférençant la trajectoire vision).

Chapitre 7

Conclusion et perspectives

7.1 Conclusion

L'objectif de cette thèse était de développer et de mettre en œuvre des méthodes souples permettant d'étalonner un ensemble de caméras dont les champs de vue sont totalement disjoints. Ces caméras sont embarquées sur deux types de véhicule : le VIPA et le VipaLab, qui ont été créés dans le cadre d'un projet pluridisciplinaire ambitieux (le projet VIPA). Les travaux de ce manuscrit s'inscrivent dans une démarche de recherche avec l'exploration de différentes pistes (miroir, ajustement de faisceaux multi-caméra), en abordant plusieurs domaines : vision par ordinateur, programmation, robotique mobile, optique, traitement du signal et de l'image.

De part les aspects théoriques développés et les expériences concrètes mises en œuvre, ces travaux s'intègrent de manière complémentaire aux travaux réalisés par mes collègues (membres de l'équipe VIPA ou du service technique du LASMEA). Les étalonnages proposés (intrinsèques, extrinsèques ou par rapport au repère du véhicule) ont contribué à l'utilisation d'un système multi-caméra embarqué sur ces véhicules.

Principales contributions Nous avons proposé plusieurs étalonnages extrinsèques d'un ensemble de caméras à champs disjoints. L'un utilise un miroir plan mobile et une scène de géométrie inconnue (qui est reconstruite grâce à ses reflets sur le miroir). De plus, l'influence de la réfraction induite par la lame de verre du miroir est étudiée, et une méthode d'estimation de pose d'un objet réfracté est proposée.

Deux autres approches étalonnent extrinsèquement un système multi-caméra à partir de ses mouvements. Nous avons étudié quels sont les mouvements singuliers (pour lesquels l'étalonnage est partiel) et quelle procédure mettre en œuvre pour tout de même réaliser un étalonnage complet (dans le cas des mouvements plans). La première approche utilise des cibles et atteint des précisions inférieures au millimètre pour les translations extrinsèques et inférieures à $0.05°$ pour les rotations extrinsèques. Dans la deuxième approche, nous avons montré que l'étalonnage extrinsèque peut être obtenu simultanément à la reconstruction 3D (par exemple lors de

la phase d'apprentissage), en utilisant des points d'intérêt. Nous avons détaillé un algorithme multi-caméra détectant des fermetures de boucle (*c.-à-d.* on détecte si le système réemprunte un chemin déjà emprunté) et un ajustement de faisceaux multi-caméra (MCBA, qui optimise à la fois les points 3D, les poses du système multi-caméra et les paramètres extrinsèques). Nous avons montré que le MCBA (seul ou après avoir détecté des fermetures de boucle) permet d'améliorer les reconstructions qui sont obtenues avec des paramètres extrinsèques constants et approximatifs. L'algorithme 8 synthétise de manière détaillée l'implémentation creuse de l'ajustement de faisceaux multi-caméra (MCBA). Il peut se placer comme une référence (au même titre que l'algorithme [HZ04, A.6.7] pour le monoculaire).

Ce manuscrit présente en outre différentes contributions telles que les cibles, un étalonnage intrinsèque (gérant plusieurs modèles de caméra et détectant automatiquement la mire d'étalonnage), la correction automatique d'images distordues, et un étalonnage déterminant la pose de la caméra par rapport au repère véhicule (en se basant sur son modèle cinématique).

7.2 Perspectives

Plusieurs améliorations ponctuelles et des travaux futurs se dégagent de ce mémoire. Parmi ces améliorations ponctuelles, il y a l'accélération de la détection sous-pixellique des cibles, estimer les incertitudes associées à chaque paramètres lors de l'étalonnage intrinsèque, ou encore gérer les mouvements 3D pour l'étalonnage "essieu/caméra".

Concernant des perspectives plus globales, plusieurs améliorations portent sur l'algorithme de SfM multi-caméra. Il serait pertinent qu'au lieu de faire une reconstruction approximative globale (avec des paramètres extrinsèques constants), on optimise plutôt dès que possible certains ou tous les paramètres extrinsèques. En d'autres termes, dès que le mouvement le permet (*c.-à-d.* dès que l'on est pas dans un cas singulier), on estime les paramètres extrinsèques observables. Il serait pertinent de proposer une variante du MCBA qui estime les incertitudes associées à chaque paramètre. La propagation des incertitudes dans la reconstruction permettrait de pré-sélectionner les poses voisines de la fermeture de boucle. Concernant la classification des points 2D en inliers ou outliers, il serait pertinent que cette sélection intervienne indépendamment dans chaque caméra (avec N reconstructions monoculaires) puis que la reconstruction multi-caméra ne les remette pas en cause. Cette approche pourrait ainsi tendre vers une reconstruction incrémentale et temps réel. De plus, comme le MCBA est massivement parallélisable, il serait envisageable de l'implémenter sur GPU.

Des expérimentations seraient à réaliser avec d'autres systèmes multi-caméra, comme par exemple le capteur Ladybug®. Il serait également pertinent de fusionner des informations d'autres capteurs en se basant sur la reconstruction multi-caméra. En effet, partant de l'ensemble des poses du système multi-caméra, il est possible de localiser les poses d'autres capteurs (LIDAR, radar...) et d'incorporer leurs mesures dans la reconstruction multi-caméra.

D'autre part, pour des applications de type convoi de véhicules, le but est de maintenir une interdistance souhaitée entre les véhicules. Le véhicule de tête reconstruit alors la scène dans

laquelle devront naviguer les véhicules suivants. Or, l'utilisation d'une seule caméra engendre le problème de dérive de facteur d'échelle, et donc, de dérive de l'interdistance. Une reconstruction multi-caméra permettrait de résoudre cette problématique.

Comment détecter que l'étalonnage n'est plus à jour ? Pour tirer parti d'un système multi-caméra, il faut pouvoir utiliser chaque caméra indépendamment (*c.-à-d.* une seule caméra fonctionnelle doit être suffisante pour localiser le système) mais aussi simultanément. Or, si l'une des caméras a été déplacée de son support, les paramètres extrinsèques ne sont plus à jour et peuvent pénaliser l'ensemble des caméras (qui perçoivent alors moins d'inliers). Il faut donc pouvoir détecter en ligne que les paramètres extrinsèques ne sont plus à jour. En localisation, un moyen simple pour vérifier la qualité de l'étalonnage est de masquer artificiellement $N-1$ caméras (toutes sauf une). Pour une caméra donnée, si son nombre d'inliers est notablement plus élevé en mono-caméra qu'en multi-caméra, alors l'étalonnage n'est pas assez précis. Au contraire, si le nombre d'inliers n'évolue pas, alors l'étalonnage est suffisamment précis. Ici, pour déterminer les points inliers, on peut calculer la pose en mono ou multi-caméra. Comme ce calcul inclut une méthode non déterministe (le RANSAC), le nombre d'inliers peut fluctuer légèrement.

Peut-on intégrer les étalonnages intrinsèques à une reconstruction multi-caméra sans perte de qualité ? Ceci permettrait d'obtenir de bonnes reconstructions euclidiennes tout en garantissant plus de souplesse de mise en œuvre (affranchissement de l'étape préalable d'étalonnage intrinsèque). Il faudrait alors adapter l'étude des cas singuliers. On peut se référer aux travaux [Stu97] et [PVG00] pour le cas monoculaire. Il serait intéressant de ne pas se limiter au modèle sténopé, pour pouvoir utiliser d'autres modèles intrinsèques comme le modèle unifié ou les caméra génériques.

Annexe A

A.1 Changement de base et matrice de passage

Pour passer d'un repère à un autre, différentes conventions peuvent être choisies. Ceci se vérifie tout particulièrement en vision par ordinateur, où on peut appliquer d'abord la rotation, puis la translation, ou inversement. Autre source d'ambigüité : selon les articles, les rotations représentent soit le déplacement du repère, soit le déplacement des vecteurs.

Il est donc de bonne augure d'arrêter un choix sur une convention. Celle qui est présentée ci-après a été choisie pour l'ensemble du manuscrit.

Soit $x \in \mathbb{R}^n$ un vecteur unicolonne de dimension n. Dans les cas rencontrés dans ce manuscrit, $n = 2$ ou $n = 3$. Soit $X = \begin{pmatrix} x \\ w \end{pmatrix} \in \mathbb{P}^n$, le même vecteur exprimé en coordonnées homogènes.

Soit $\mathcal{B}_1$, $\mathcal{B}_2$ et $\mathcal{B}_3$ trois bases de $\mathbb{R}^n$. X_1, X_2 et X_3 représentent le vecteur x respectivement dans la base $\mathcal{B}_1$, $\mathcal{B}_2$ et $\mathcal{B}_3$ en coordonnées homogènes.

Définition 1. *On appelle* matrice de passage homogène *de la base $\mathcal{B}_1$ à la base $\mathcal{B}_2$, la matrice T_1^2 de taille $(n+1) \times (n+1)$ vérifiant :*

$$X_1 = T_1^2 X_2 \tag{A.1}$$

On utilise également le terme *transformation homogène* pour désigner la matrice T_1^2. Il faut bien noter que la formule A.1 permet d'obtenir le vecteur exprimé dans $\mathcal{B}_1$ à partir de celui exprimé dans $\mathcal{B}_2$ (et non pas l'inverse !).

Interprétation (pour $n = 3$) : $T_1^2 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$, $T_1^2 \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$, $T_1^2 \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$, sont les 3 vecteurs de la base $\mathcal{B}_2$ exprimés dans la base $\mathcal{B}_1$ en coordonnées homogènes.

Composition de changements de base Soient T_1^3 la matrice de passage homogène de la base $\mathcal{B}_1$ à la $\mathcal{B}_3$ et T_2^3 la matrice de passage homogène entre $\mathcal{B}_2$ et $\mathcal{B}_3$. On rappelle la formule qui

compose deux changements de bases successifs :

$$T_1^3 = T_1^2 T_2^3 \tag{A.2}$$

En effet,

$$\begin{cases} X_1 = T_1^2 X_2 \\ X_1 = T_1^3 X_3 \\ X_2 = T_2^3 X_3 \end{cases} \Rightarrow \begin{cases} X_1 = T_1^2 T_2^3 X_3 \\ X_1 = T_1^3 X_3 \end{cases} \tag{A.3}$$

A.2 Quelques transformations géométriques

A.2.1 Transformation euclidienne

Soit R la matrice de rotation entre les bases $\mathcal{B}_1$ et $\mathcal{B}_2$. Soit $\mathbf{t}$ le vecteur de translation entre les bases $\mathcal{B}_1$ et $\mathcal{B}_2$. L'équation A.4 représente une transformation euclidienne entre les deux bases. Les transformations euclidiennes sont des isométries qui préservent l'orientation.

$$\begin{pmatrix} x_1 \\ 1 \end{pmatrix} = \begin{pmatrix} R & \mathbf{t} \\ 0_{1\times n} & 1 \end{pmatrix} \begin{pmatrix} x_2 \\ 1 \end{pmatrix} \tag{A.4}$$

En dimension 3, toute transformation euclidienne peut s'interpréter comme un vissage dans une base donnée.

A.2.2 Similitude directe

Soit $s \in \mathbb{R}^{*+}$. L'équation A.5 représente une similitude directe entre les deux bases. Cette similitude directe s'interprète comme la composition d'une homothétie de rapport s avec la transformation euclidienne $(R, \mathbf{t})$.

$$\begin{pmatrix} x_1 \\ 1 \end{pmatrix} = \begin{pmatrix} sR & \mathbf{t} \\ 0_{1\times n} & 1 \end{pmatrix} \begin{pmatrix} x_2 \\ 1 \end{pmatrix} \tag{A.5}$$

A.2.3 Affinité

Soit $A \in \mathrm{GL}_n(\mathbb{R})$ une matrice inversible. L'équation A.6 représente une transformation affine entre les deux bases.

$$\begin{pmatrix} x_1 \\ 1 \end{pmatrix} = \begin{pmatrix} A & \mathbf{t} \\ 0_{1\times n} & 1 \end{pmatrix} \begin{pmatrix} x_2 \\ 1 \end{pmatrix} \tag{A.6}$$

A.2.4 Transformation projective

Considérons le cas où $n = 2$ (utilisé dans le chapitre 2.2.1.3). Soit $\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in \mathbb{R}^2$ et $v \in \mathbb{R}$. L'équation A.7 représente une transformation projective (*c.-à-d.* une homographie) entre les deux bases.

$$\begin{pmatrix} x_1 \\ w_1 \end{pmatrix} = \begin{pmatrix} A & \mathbf{t} \\ \mathbf{v}^\top & v \end{pmatrix} \begin{pmatrix} x_2 \\ w_2 \end{pmatrix} \tag{A.7}$$

On pose $H = \begin{pmatrix} A & \mathbf{t} \\ \mathbf{v}^\top & v \end{pmatrix}$, qui fait intervenir 9 paramètres indépendants. Or, comme l'homographie est utilisée à un facteur multiplicatif près (à cause de la division par la coordonnée homogène), on obtient bien ses 8 degrés de libertés. Pour les imposer, il est possible de rajouter une contrainte (en général, nous imposons que $h_{33} = 1 \Leftrightarrow v = 1$).

A.2.5 De la transformation euclidienne à la transformation projective

H est une homographie pouvant se décomposer en un produit de plusieurs matrices, qui représentent plusieurs transformations élémentaires successivement appliquées. Il est possible de choisir la décomposition présentée par l'équation A.8 pour le cas où $n = 2$, illustrée par la Figure A.1 :

$$\begin{aligned} H = H_e\, H_a\, H_p &= \begin{pmatrix} R & \mathbf{t} \\ 0_{1\times 2} & 1 \end{pmatrix} \begin{pmatrix} U & 0_{2\times 1} \\ 0_{1\times 2} & 1 \end{pmatrix} \begin{pmatrix} I_n & 0_{2\times 1} \\ \mathbf{v}^\top & v \end{pmatrix} \\ &= \begin{pmatrix} RU + t\mathbf{v}^\top & \mathbf{t} \\ \mathbf{v}^\top & v \end{pmatrix} \end{aligned} \tag{A.8}$$

, où $U = \begin{pmatrix} \lambda_1 & r \\ 0 & \lambda_2 \end{pmatrix}$. λ_1 et λ_2 sont les rapports d'agrandissement selon les deux axes. r est le facteur de non-orthogonalité des axes (appelé "shear"). Si les axes sont orthogonaux, r est nul.

Autre décomposition : Comme expliqué dans [HZ04, p43], l'inverse d'une homographie $H^{-1} = H_p^{-1} H_a^{-1} H_e^{-1}$ est aussi une homographie. De plus, H_p^{-1}, H_a^{-1}, et H_e^{-1} reste respectivement des transformations projectives, affine et euclidienne. Une homographie H peut donc également se décomposer sous la forme présentée par l'équation A.9.

$$\begin{aligned} H = H'_p\, H'_a\, H'_e &= \begin{pmatrix} I_n & 0_{n\times 1} \\ \mathbf{v'}^\top & v' \end{pmatrix} \begin{pmatrix} U' & 0_{n\times 1} \\ 0_{1\times n} & 1 \end{pmatrix} \begin{pmatrix} R' & \mathbf{t}' \\ 0_{1\times n} & 1 \end{pmatrix} \\ &= \begin{pmatrix} U'R' & U'\mathbf{t}' \\ \mathbf{v'}^\top U'R' & \mathbf{v'}^\top U'\mathbf{t}' + v' \end{pmatrix} \end{aligned} \tag{A.9}$$

Les valeurs de $\mathbf{v}'$, v', $U' = \begin{pmatrix} \lambda'_1 & r' \\ 0 & \lambda'_2 \end{pmatrix}$, R' et $\mathbf{t}'$ diffèrent de celles de la décomposition de l'équation A.8. Là encore, on peut imposer que $h_{33} = 1 \Leftrightarrow v' = 1 - \mathbf{v'}^\top U'\mathbf{t}'$.

A.3 Valeurs et vecteurs propres

A.3.1 Matrice de rotation

Soit R, une matrice de rotation représentant la rotation d'angle θ autour de l'axe dirigé par le vecteur unitaire n.

Propriété 9. *n est un vecteur propre de R associé à la valeur propre* 1.

Démonstration : L'axe de rotation est invariant par R :

$$Rn = 1\ n \tag{A.10}$$

□

Propriété 10. *Les valeurs propres de R sont* 1, $e^{\mathtt{i}\theta}$ *et* $e^{-\mathtt{i}\theta}$.

Démonstration : D'après la propriété 9, on sait déjà que 1 est valeur propre.
La matrice R est semblable à la matrice :

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix} \tag{A.11}$$

Ainsi, si θ n'est pas un multiple de π (*c.-à-d.* si $\sin(\theta) \neq 0$), R n'est pas diagonalisable dans $\mathbb{R}$.
Soient a et b, deux nombres complexes tels que R soit semblable à la matrice diagonale :

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \end{pmatrix} \tag{A.12}$$

En utilisant les propriétés sur la trace et le déterminant de R, on obtient :

$$\begin{cases} \det(R) = ab \\ \mathrm{tr}(R) = 1 + a + b \end{cases} \Leftrightarrow \begin{cases} 1 = ab \\ 1 + 2\cos(\theta) = 1 + a + b \end{cases} \Leftrightarrow \begin{cases} ab = 1 \\ a + b = e^{\mathtt{i}\theta} + e^{-\mathtt{i}\theta} \end{cases} \tag{A.13}$$

On en déduit les 2 dernières valeurs propres conjuguées de R : $e^{\mathtt{i}\theta}$ et $e^{-\mathtt{i}\theta}$. □

On note $n \in \mathbb{R}^3$, $u \in \mathbb{C}^3$ et $v \in \mathbb{C}^3$ les 3 vecteurs propres unitaires de R respectivement associés aux valeurs propres 1, $e^{\mathtt{i}\theta}$ et $e^{-\mathtt{i}\theta}$.

Propriété 11. *Les vecteurs propres u et v sont conjugués.*

Démonstration : Par définition, on a :

$$\begin{cases} Ru = e^{\mathtt{i}\theta}u \\ Rv = e^{-\mathtt{i}\theta}v \end{cases} \Leftrightarrow \begin{cases} Ru^* = e^{-\mathtt{i}\theta}u^* \\ Rv = e^{-\mathtt{i}\theta}v \end{cases} \Rightarrow v = u^*, \text{ car } u \text{ et } v \text{ sont unitaires.} \tag{A.14}$$

□

A.3.2 Produit de Kronecker

Définition 2. *Soient deux matrices $A \in \mathbb{R}^{m\times n}$ et $B \in \mathbb{R}^{p\times q}$. Le produit de Kronecker $A \otimes B \in \mathbb{R}^{mp\times nq}$ est défini par blocs :*

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix} \tag{A.15}$$

Propriété 12. Produit mixte *Soient A, B, C, D quatre matrices telles que les produits matriciels AC et BD existent, on a :*

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD) \tag{A.16}$$

Propriété 13. *Soient deux matrices $A \in \mathbb{R}^{n\times n}$ et $B \in \mathbb{R}^{m\times m}$. Soit $(\lambda_i)_{i\in[\![1..n]\!]}$ les n valeurs propres de A associées aux vecteurs propres $(\mathbf{a}_i)_{i\in[\![1..n]\!]}$. Soit $(\mu_j)_{j\in[\![1..m]\!]}$ les m valeurs propres de B associées aux vecteurs propres $(\mathbf{b}_j)_{j\in[\![1..m]\!]}$.*

Les np valeurs propres de $A \otimes B$ sont $(\lambda_i\mu_j)_{(i,j)\in[\![1..n]\!]\times[\![1..m]\!]}$ associées aux vecteurs propres $(\mathbf{a}_i \otimes \mathbf{b}_j)_{(i,j)\in[\![1..n]\!]\times[\![1..m]\!]}$

Démonstration : Soient $i \in [\![1..n]\!]$ et $j \in [\![1..m]\!]$. Par définition, on a :

$$\begin{cases} A\mathbf{a}_i = \lambda_i\mathbf{a}_i \\ B\mathbf{b}_j = \mu_j\mathbf{b}_j \end{cases} \Rightarrow (A\mathbf{a}_i) \otimes (B\mathbf{b}_j) = (\lambda_i\mathbf{a}_i) \otimes (\mu_j\mathbf{b}_j) \tag{A.17}$$

$$(A.16)\ \&\ (A.17) \Rightarrow (A \otimes B)(\mathbf{a}_i\mathbf{b}_j) = \lambda_i\mu_j(\mathbf{a}_i \otimes \mathbf{b}_j) \tag{A.18}$$

□

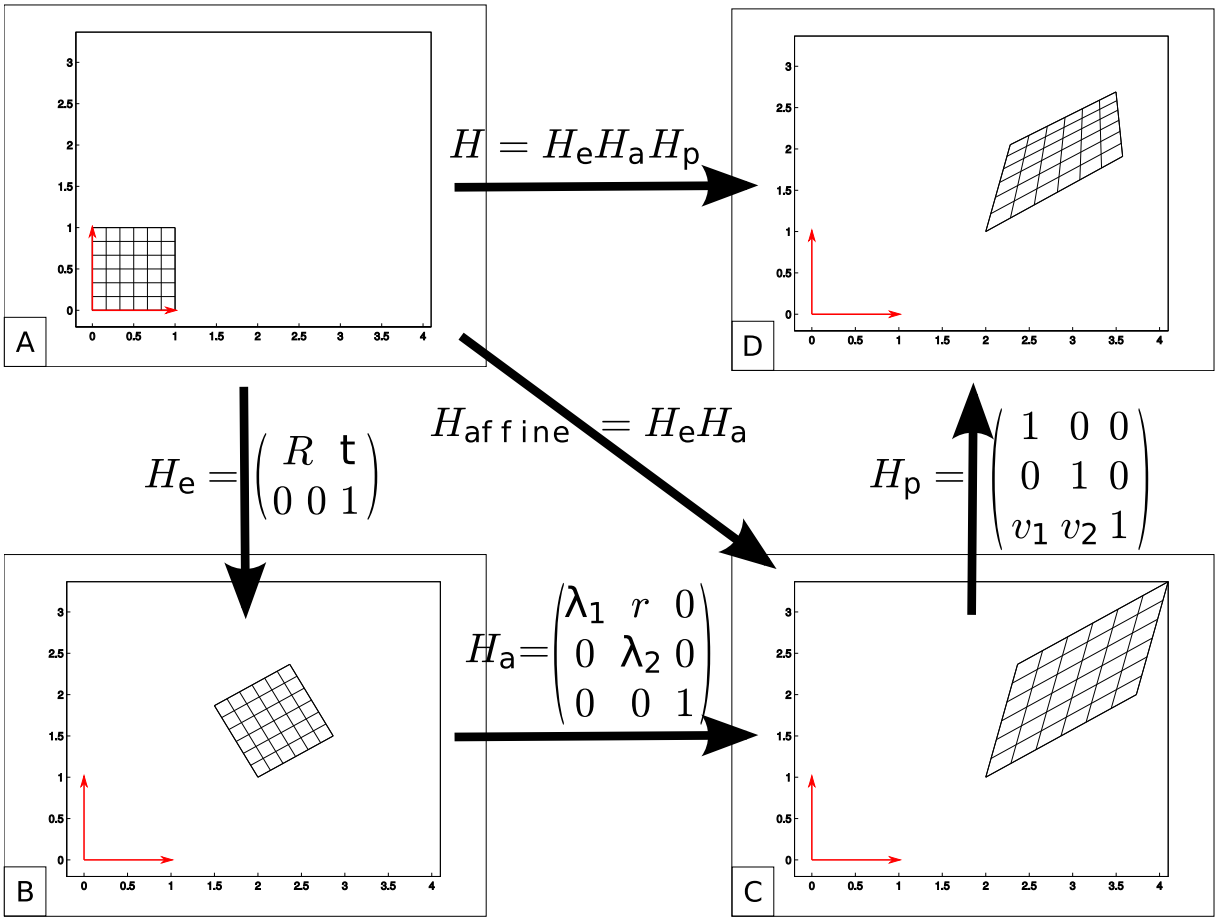


FIGURE A.1 – Illustration de quelques transformations géométriques du plan. L'homographie H_p s'obtient par la composition de la transformation euclidienne H_e, de la transformation H_a (qui applique un rapport λ_1 selon le premier axe, λ_2 selon le deuxième axe et modifie l'angle entre les 2 axes), et H_p (qui applique la non-linéarité liée à la transformation projective).

Annexe B

Dans cette annexe, nous détaillons l'ajustement de faisceaux multi-caméra (MCBA, *cf* Section 5.4).

B.1 Changement de repère monde/caméras

On souhaite exprimer un point du monde dans le repère de la caméra l'ayant observé (afin de le projeter dans celle-ci). La Figure 5.8 illustre l'ensemble du système.

$\overline{P}_i = (X_i\ Y_i\ Z_i\ w_i)^\top$ est un point 3D de la scène $S = \{\overline{P}_i\}_{i\in[\![1..M]\!]}$, exprimé dans le repère monde. Rappelons que K_j est la matrice intrinsèque de C_j. On rappelle également que π est la fonction, définie par l'équation 2.7, permettant de passer des coordonnées homogènes aux coordonnées cartésiennes pour un point 2D.

Monde $\rightarrow$ caméra maître D'après la formule de changement de base (*cf* équation A.1), les coordonnées homogènes de $\overline{P}_i$ exprimées dans le repère de C_1^k ($k^{ème}$ pose de la caméra maître) sont ${T_1^k}^{-1}\overline{P}_i$.

La projection perspective d'un point 3D du monde vu par la $k^{ème}$ pose de la caméra maître C_1 s'écrit donc : $\pi(K_j[{R_1^k}^\top | -{R_1^k}^\top \mathbf{t}_1^k]\overline{P}_i)$.

Monde $\rightarrow$ caméra esclave De même, d'après la formule de changement de base (*cf* équation A.1) et l'équation A.2, les coordonnées homogènes de $\overline{P}_i$ exprimées dans le repère de C_j^k ($k^{ème}$ pose de la caméra C_j) sont $\left(T_1^k \Delta T_j\right)^{-1}\overline{P}_i$, c'est à dire ${\Delta T_j}^{-1}{T_1^k}^{-1}\overline{P}_i$.

Projection perspective d'un point 3D vu par une caméra esclave La projection perspective d'un point 3D du monde vu par la $k^{ème}$ pose de la caméra esclave C_j s'écrit donc : $\pi(K_j{\Delta R_j}^\top[{R_1^k}^\top | -{R_1^k}^\top \mathbf{t}_1^k - \Delta\mathbf{t}_j]\overline{P}_i)$.

B.2 Fonction de coût

Maintenant que nous avons exprimé analytiquement la projection des points de la scène dans chaque caméra, nous pouvons détailler la fonction de coût de l'ajustement de faisceaux multi-

caméra (MCBA), définie par l'équation 5.37. $p_{ij}^k = (u_{ij}^k \; v_{ij}^k)^\top$ correspond aux coordonnées du point $\overline{P}_i$ observé par la caméra C_j à l'instant k.

Le vecteur $\boldsymbol{\epsilon}_{ij}^k$ de résidu de l'observation du point $\overline{P}_i$ par la caméra C_j à l'instant k est donc :

$$\boldsymbol{\epsilon}_{ij}^k = \pi(K_j \Delta {R_j}^\top [{R_1^k}^\top | - {R_1^k}^\top \mathbf{t}_1^k - \Delta \mathbf{t}_j] \overline{P}_i) - p_{ij}^k \tag{B.19}$$

En définissant $\Delta \mathbf{t}_1 = 0_{3\times 1}$ et $\Delta R_1 = I_3$, l'équation (B.19) est valable aussi bien les résidus des mesures de la caméra maître et des caméras esclaves, qui sont respectivement $\boldsymbol{\epsilon}_{i1}^k = \boldsymbol{\epsilon}_{i1}^k(\overline{P}_i, R_1^k, \mathbf{t}_1^k)$ et $\boldsymbol{\epsilon}_{ij}^k = \boldsymbol{\epsilon}_{ij}^k(\overline{P}_i, \mathbf{t}_1^k, R_1^k, \Delta \mathbf{t}_j, \Delta R_j)$. Si un point $\overline{P}_i$ n'est pas visible par la caméra C_j à l'instant k, alors le résidu $\boldsymbol{\epsilon}_{ij}^k$ est nul (dans le cas de l'implémentation creuse, ce résidu n'est même pas déclaré).

B.3 Jacobienne analytique

B.3.1 Propositions préalables

Composition d'applications linéaires et dérivée partielle Si $\psi \in \mathbb{R}^n$ est un vecteur colonne et $g : \mathbb{R}^4 \times \mathbb{R}^n \to \mathbb{R}^3$ est une application linéaire telle que $g(\overline{P}_i, \psi) = \overline{p}$, alors la dérivée partielle de la composition d'applications linaires permet d'écrire :

$$\frac{\partial \pi \circ g(\overline{P}_i, \psi)}{\partial \psi} = \frac{\partial \pi}{\partial \overline{p}} \frac{\partial g(\overline{P}_i, \psi)}{\partial \psi} \tag{B.20}$$

où

$$\frac{\partial \pi}{\partial \overline{p}} = \begin{pmatrix} 1/Z_p & 0 & -X_p/Z_p^2 \\ 0 & 1/Z_p & -Y_p/Z_p^2 \end{pmatrix} \tag{B.21}$$

Levenberg-Marquardt et matrices de rotation Pour chaque itération de l'algorithme de Levenberg-Marquardt, une rotation locale $\rho(\xi)$ est composée avec l'estimation courante de la rotation r pour obtenir la mise à jour de la rotation R (*cf* équation B.22).

$$R = R(\xi) = \rho(\xi) r \tag{B.22}$$

Où $\xi = (\alpha \; \beta \; \gamma)^\top$ sont les angles d'Euler locaux, qui permettent d'éviter des blocages de cardan, tel que :

$$\rho(\xi) = \begin{pmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{pmatrix} \tag{B.23}$$

A l'aide d'une approximation au premier ordre :

$$\rho(\xi) \simeq I_3 + [\xi]_\times = \begin{pmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{pmatrix} \tag{B.24}$$

Alors,

$$\frac{\partial \rho(\xi)}{\partial \alpha} \simeq \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}_\times, \frac{\partial \rho(\xi)}{\partial \beta} \simeq \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}_\times, \frac{\partial \rho(\xi)}{\partial \gamma} \simeq \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}_\times \tag{B.25}$$

$$\Rightarrow \frac{\partial(\rho(\xi)\overline{p})}{\partial \xi} = [\overline{p}]_\times \tag{B.26}$$

$$\text{Finalement : } \frac{\partial(R(\xi)^\top \overline{p})}{\partial \xi} \simeq -r^\top [\overline{p}]_\times \tag{B.27}$$

B.3.2 Blocs analytiques du Jacobien

Dérivée partielle par rapport aux points 3D

$$\begin{aligned} \frac{\partial \boldsymbol{\epsilon}_{ij}^k}{\partial \overline{P}_i} =& \frac{\partial \pi}{\partial \overline{p}} \frac{\partial (K_j \Delta {R_j}^\top [{R_1^k}^\top | - {R_1^k}^\top \mathbf{t}_1^k - \Delta \mathbf{t}_j] \overline{P}_i)}{\partial \overline{P}_i} \\ =& \frac{\partial \pi}{\partial \overline{p}} K_j \Delta {R_j}^\top [{R_1^k}^\top | - {R_1^k}^\top \mathbf{t}_1^k - \Delta \mathbf{t}_j] \end{aligned} \tag{B.28}$$

$\frac{\partial \boldsymbol{\epsilon}_{ij}^k}{\partial \overline{P}_i}$ est une matrice :

- 2×4 si on optimise les 4 coordonnées de $\overline{P}_i$,
- 2×3 si on optimise les 3 coordonnées (X, Y, Z) de $\overline{P}_i$. On ne conserve que les 3 premières colonnes de la formule B.28.

Dérivée partielle par rapport à la translation de la caméra maître

$$\frac{\partial \boldsymbol{\epsilon}_{ij}^k}{\partial \mathbf{t}_1^k} = -w_i \frac{\partial \pi}{\partial \overline{p}} K_j \Delta {R_j}^\top R_1^k \tag{B.29}$$

Dérivée partielle par rapport à la rotation de la caméra maître En utilisant les mêmes notations : $R_1^k = R_1^k(\xi_1^k) = \rho(\xi_1^k) r_1^k$ et $(B.27)$:

$$\frac{\partial \boldsymbol{\epsilon}_{ij}^k}{\partial \xi_1^k} = -\frac{\partial \pi}{\partial \overline{p}} K_j \Delta {R_j}^\top {r_1^k}^\top \left[[I_3| - \mathbf{t}_1^k] \overline{P}_i \right]_\times \tag{B.30}$$

Dérivée partielle par rapport aux translations extrinsèques $\forall j \in [\![2..N]\!]$,

$$\frac{\partial \boldsymbol{\epsilon}_{ij}^{k}}{\partial \Delta \mathbf{t}_j} = -w_i \frac{\partial \pi}{\partial \overline{p}} K_j \Delta R_j{}^{\top} \tag{B.31}$$

Dérivée partielle par rapport aux rotations extrinsèques En utilisant les mêmes notations : $\Delta R_j = \Delta R_j(\Delta\xi_j) = \rho(\Delta\xi_j)\Delta r_j$ et (B.27) :

$$\frac{\partial \boldsymbol{\epsilon}_{ij}^{k}}{\partial \Delta \xi_j} = \frac{\partial \pi}{\partial \overline{p}} \Delta r_j{}^{\top} \left[[R_1^{k\top} | - R_1^{k\top} \mathbf{t}_1^k - \Delta \mathbf{t}_j] \overline{P}_i \right]_{\times} \tag{B.32}$$

B.4 MCBA : Détail de l'expression des blocs

Les blocs matriciels de l'équation normale 5.38 et du Système Caméra-Pose 5.54 permettent d'obtenir une implémentation creuse de l'algorithme 8. La description de cet algorithme est suffisante pour l'implémenter et pour calculer les blocs de manière incrémentale.

Cependant, pour un détail complet de l'algorithme, nous fournissons les expressions complètes des blocs de l'équation normale. Puis, nous détaillons les étapes intermédiaires pour les calculs des blocs diagonaux et non-diagonaux du Système Caméras/Poses.

B.4.1 De l'équation normale

Afin de détailler l'implémentation de la structure secondaire des équations normales, il est nécessaire de définir deux index supplémentaires.

- Soit $\mathcal{J}_i^k$ l'index des caméras qui ont vu le point $\overline{P}_i$ pour la $k^{ème}$ pose.
- Soit $\mathcal{K}_{ij}$ l'index des poses où le point $\overline{P}_i$ est vu par la $j^{ème}$ caméra.

$\mathcal{J}_i^k$ est un sous-ensemble de $[\![1..N]\!]$. $\mathcal{K}_{ij}$ est un sous-ensemble de $[\![0..K]\!]$.

$$\forall k \in [\![0..K]\!],\ U^k = \sum_{i \in \mathcal{I}^k} A_i^{k\top} A_i^k = \sum_{i \in \mathcal{I}^k} \sum_{j \in \mathcal{J}_i^k} A_{ij}^{k\top} A_{ij}^k \tag{B.33}$$

$$\forall i \in [\![1..M]\!],\ V_i = \sum_{k \in \mathcal{K}_i} B_i^{k\top} B_i^k = \sum_{k \in \mathcal{K}_i} \sum_{j \in \mathcal{J}_i^k} B_{ij}^{k\top} B_{ij}^k \tag{B.34}$$

$$\forall j \in [\![2..N]\!],\ Q_j = \sum_{k=0}^{K} \sum_{i \in \mathcal{I}_j^k} C_{ij}^{k\top} C_{ij}^k = \sum_{i \in \mathcal{I}_j} \sum_{k \in \mathcal{K}_{ij}} C_{ij}^{k\top} C_{ij}^k \tag{B.35}$$

$$\left.\begin{array}{l} \forall k \in [\![0..K]\!],\ \forall i \in \mathcal{I}^k, \\ \text{ou } \forall i \in [\![1..M]\!],\ \forall k \in \mathcal{K}_i, \end{array}\right\} W_i^k = A_i^{k\top} B_i^k = \sum_{j \in \mathcal{J}_i^k} A_{ij}^{k\top} B_{ij}^k \tag{B.36}$$

$$\forall j \in [\![2..N]\!],\ \forall k \in [\![0..K]\!],\ E_j^k = \sum_{i \in \mathcal{I}_j \bigcap \mathcal{I}^k} {C_{ij}^k}^\top A_{ij}^k \tag{B.37}$$

$$\left.\begin{array}{l} \forall j \in [\![2..N]\!],\ \forall i \in \mathcal{I}_j, \\ \text{ou } \forall i \in [\![1..M]\!],\ \forall j \in \mathcal{J}_i \bigcap [\![2..N]\!], \end{array}\right\} F_{ij} = \sum_{k \in \mathcal{K}_{ij}}^{K} {C_{ij}^k}^\top B_{ij}^k \tag{B.38}$$

$$\forall j \in [\![2..N]\!],\ g(\Delta T_j) = \sum_{i \in \mathcal{I}_j} \sum_{k \in \mathcal{K}_{ij}} {C_{ij}^k}^\top \boldsymbol{\epsilon}_{ij}^k \tag{B.39}$$

$$\forall k \in [\![0..K]\!],\ g(T_1^k) = \sum_{i \in \mathcal{I}^k} \sum_{j \in \mathcal{J}_i^k}^{N} {A_{ij}^k}^\top \boldsymbol{\epsilon}_{ij}^k \tag{B.40}$$

$$\forall i \in [\![1..M]\!],\ g(\overline{P}_i) = \sum_{k \in \mathcal{K}_i} \sum_{j \in \mathcal{J}_i^k} {B_{ij}^k}^\top \boldsymbol{\epsilon}_{ij}^k \tag{B.41}$$

B.4.2 Du Système Caméra-Pose (CPS)

$\forall j \in [\![2..N]\!]$,

$$\begin{aligned} \bar{\mathbf{A}}_{jj} &= Q_j - (FV^{-1}F^\top)_{jj} \\ &= Q_j - \sum_{i \in \mathcal{I}_j} \underbrace{F_{ij}{V_i}^{-1}}_{\stackrel{def}{=} Y_{ij}} {F_{ij}}^\top \\ &= Q_j - \sum_{i \in \mathcal{I}_j} Y_{ij} {F_{ij}}^\top \end{aligned} \tag{B.42}$$

$\forall j_1 \in [\![2..N]\!], \forall j_2 \in [\![2..N]\!],\ j_1 < j_2$,

$$\begin{aligned} \bar{\mathbf{A}}_{j_1 j_2} &= -(FV^{-1}F^\top)_{j_1 j_2} \\ &= - \sum_{i \in \mathcal{I}_{j_1} \bigcap \mathcal{I}_{j_2}} \underbrace{F_{ij_1}{V_i}^{-1}}_{\stackrel{def}{=} Y_{ij_1}} {F_{ij_2}}^\top \\ &= - \sum_{i \in \mathcal{I}_{j_1} \bigcap \mathcal{I}_{j_2}} Y_{ij_1} {F_{ij_2}}^\top \end{aligned} \tag{B.43}$$

$\forall j \in [\![2..N]\!],\ \forall k \in [\![0..K]\!]$,

$$\begin{aligned} \bar{\mathbf{B}}_j^k &= E_j^k - (FV^{-1}W^\top)_j^k \\ &= E_j^k - \sum_{i \in \mathcal{I}_j \bigcap \mathcal{I}^k} F_{ij}{V_i}^{-1}{W_i^k}^\top \\ &= E_j^k - \sum_{i \in \mathcal{I}_j \bigcap \mathcal{I}^k} Y_{ij}{W_i^k}^\top \end{aligned} \tag{B.44}$$

$\forall k \in [\![0..K]\!]$,

$$\begin{aligned}\bar{\mathbf{D}}^{kk} &= U^k - (WV^{-1}W^\top)^{kk} \\ &= U^k - \sum_{i\in\mathcal{I}^k} \underbrace{W_i^k V_i^{-1}}_{\stackrel{def}{=} Y_i^k} {W_i^k}^\top \\ &= U^k - \sum_{i\in\mathcal{I}^k} Y_i^k {W_i^k}^\top\end{aligned} \tag{B.45}$$

$\forall k_1 \in [\![0..K]\!],\ \forall k_2 \in [\![0..K]\!],\ k_1 < k_2$,

$$\begin{aligned}\bar{\mathbf{D}}^{k_1k_2} &= -(WV^{-1}W^\top)^{k_1k_2} \\ &= -\sum_{i\in\mathcal{I}^{k_1}\bigcap\mathcal{I}^{k_2}} \underbrace{W_i^{k_1} V_i^{-1}}_{\stackrel{def}{=} Y_i^{k_1}} {W_i^{k_2}}^\top \\ &= -\sum_{i\in\mathcal{I}^{k_1}\bigcap\mathcal{I}^{k_2}} Y_i^{k_1} {W_i^{k_2}}^\top\end{aligned} \tag{B.46}$$

$$\begin{aligned}\forall j \in [\![2..N]\!],\ (g'_\Delta)_j &= g(\Delta T_j) - (FV^{-1}g_P)_j \\ &= g(\Delta T_j) - \sum_{i\in\mathcal{I}_j} F_{ij} V_i^{-1} g(\overline{P}_i) \\ &= g(\Delta T_j) - \sum_{i\in\mathcal{I}_j} Y_{ij} g(\overline{P}_i)\end{aligned} \tag{B.47}$$

$$\begin{aligned}\forall k \in [\![0..K]\!],\ (g'_T)^k &= g(T_1^k) - (WV^{-1}g_P)^k \\ &= g(T_1^k) - \sum_{\in\mathcal{I}^k} W_i^k V_i^{-1} g(\overline{P}_i) \\ &= g(T_1^k) - \sum_{\in\mathcal{I}^k} Y_i^k g(\overline{P}_i)\end{aligned} \tag{B.48}$$

Publications dans le cadre de cette thèse

Congrès internationaux :

[1] Pierre Lébraly, Clément Deymier, Omar Ait-Aider, Eric Royer, and Michel Dhome. Flexible Extrinsic Calibration of Non-Overlapping Cameras Using a Planar Mirror : Application to Vision-Based Robotics. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS 2010.

[2] Pierre Lébraly, Eric Royer, Omar Ait-Aider, Deymier Clément, and Michel Dhome. Fast Calibration of Embedded Non-Overlapping Cameras. In *International Conference on Robotics and Automation*, ICRA 2011.

[3] Pierre Lébraly, Eric Royer, Omar Ait-Aider, and Michel Dhome. Calibration of non-overlapping cameras - application to vision-based robotics. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 10.1–10.12. BMVA Press, 2010. doi :10.5244/C.24.10.

Congrès nationaux :

[4] Clément Deymier, Pierre Lébraly, and Thierry Chateau. Auto-étalonnage de caméras fisheyes en environnement structuré. In *RFIA 2012, 18e congrès francophone AFRIF-AFIA Reconnaissance des Formes et Intelligence Artificielle*. (accepté).

[5] Pierre Lébraly, Omar Ait-Aider, Michel Dhome, and Eric Royer. Calibrage extrinsèque multi-caméras à champs non-recouvrants à l'aide d'un miroir plan. In *XXIIe colloque GRETSI (traitement du signal et des images), Dijon (FRA), 8-11 septembre 2009*. GRETSI, Groupe d'Etudes du Traitement du Signal et des Images, 2009.

[6] Pierre Lébraly, Omar Ait-Aider, Eric Royer, and Michel Dhome. Calibrage extrinsèque multi-caméras à champs non-recouvrants à l'aide d'un miroir plan - Application pour un robot mobile. In *RFIA 2010, 17e congrès francophone AFRIF-AFIA Reconnaissance des Formes et Intelligence Artificielle*.

[7] Pierre Lébraly, Omar Ait-Aider, Eric Royer, and Michel Dhome. Comment calibrer extrinsèquement des caméras à champs non-recouvrants ? Application pour un robot mobile. In *Congrès des jeunes chercheurs en vision par ordinateur*, ORASIS 2011.

Brevet :

[8] Procédé d'étalonnage d'un système de vision par ordinateur embarqué sur un mobile, 2011. Brevet déposé.

Bibliographie

[ACV04] F. Abad, E. Camahort, and R. Vivó. Camera calibration using two concentric circles. *Image Analysis and Recognition*, pages 688–696, 2004.

[AHE01] N. Andreff, R. Horaud, and B. Espiau. Robot hand-eye calibration using structure-from-motion. *The International Journal of Robotics Research*, 20(3) :228, 2001.

[AmS11] A. Agha-mohammadi and D. Song. Robust recognition of planar mirrored walls using a single view. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1186–1191. IEEE, 2011.

[AP09] R. Angst and M. Pollefeys. Static multi-camera factorization using rigid motion. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1203–1210. IEEE, 2009.

[ASB02] X. Armangué, J. Salvi, and J. Batlle. A comparative review of camera calibrating methods with accuracy evaluation. *Pattern Recognition*, 35(7) :1617–1635, 2002.

[ATC] N. Anjum, M. Taj, and A. Cavallaro. Relative position estimation of non-overlapping cameras. In *IEEE International Conference on Acoustics, Speech and Signal Processing, 2007. ICASSP 2007*, volume 2.

[BA03] J.P. Barreto and H. Araujo. Paracatadioptric camera calibration using lines. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1359–1365. IEEE, 2003.

[BA06] J.P. Barreto and H. Araujo. Fitting conics to paracatadioptric projections of lines. *Computer Vision and Image Understanding*, 101(3) :151–165, 2006.

[Baj10] F. Bajramovic. *Self-Calibration of Multi-Camera Systems*. Logos Verlag Berlin, 2010.

[BALM] A. Benzerrouk, L. Adouane, L. Lequievre, and P. Martinet. Navigation of multi-robot formation in unstructured environment using dynamical virtual structures. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5589–5594. IEEE.

[Bar03] J.P. Barreto. *General central projection systems, modeling, calibration and visual servoing*. Ph.D., University of Coimbra, 2003.

[Bar06] J.P. Barreto. A unifying geometric representation for central projection systems. *Computer Vision and Image Understanding*, 103(3) :208–217, 2006.

[BB] DH Ballard and CM Brown. Computer vision. 1982.

[BCLF04] E. Bayro-Corrochano and C. López-Franco. Omnidirectional vision : Unified model using conformal geometry. *Computer Vision-ECCV 2004*, pages 536–548, 2004.

[BD08a] F. Bajramovic and J. Denzler. Global uncertainty-based selection of relative poses for multi camera calibration. In *Proceedings of the British Machine Vision Conference (BMVC)*, volume 2, pages 745–754, 2008.

[BD08b] Ferid Bajramovic and Joachim Denzler. An Efficient Shortest Triangle Paths Algorithm for Uncertainty-based Multi Camera Calibration. In *The 8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras - OMNIVIS*, Marseille, France, 2008. Rahul Swaminathan and Vincenzo Caglioti and Antonis Argyros.

[BHS11a] A. Borkar, M. Hayes, and M. Smith. A new multi-camera approach for lane departure warning. *Advances Concepts for Intelligent Vision Systems*, pages 58–69, 2011.

[BHS11b] A. Borkar, M. Hayes, and M.T. Smith. A non overlapping camera network : Calibration and application towards lane departure warning. In *International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV 2011)*, July 2011.

[Bon06] Thomas Bonfort. *Reconstruction de Surfaces Réfléchissantes à partir d'Images*. These, Institut National Polytechnique de Grenoble - INPG, February 2006.

[Bou02] J.Y. Bouguet. Complete camera calibration toolbox for matlab. 2002. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[Bra95] Pascal Brand. *Reconstruction tridimensionnelle à partir d'une caméra en mouvement : de l'influence de la precision*. These, Université Claude Bernard - Lyon I, October 1995. theses/1995/Brand.Pascal theses/1995/Brand.Pascal.

[Bra00] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[Bro71] D.C. Brown. Close-range camera calibration. *Photogrammetric engineering*, 37(8) :855–866, 1971.

[CAD11a] G. Carrera, A. Angeli, and A.J. Davison. Lightweight slam and navigation with a multi-camera rig. *European Conference on Mobile Robots*, 2011.

[CAD11b] G. Carrera, A. Angeli, and A.J. Davison. Slam-based automatic extrinsic calibration of a multi-camera rig. In *International Conference on Robotics and Automation*, ICRA 2011.

[Can86] J. Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6) :679–698, 1986.

[CCPB09] B. Cannelle, D. Craciun, N. Paparoditis, and D. Boldo. Bundle adjustement and pose estimation of images of a multiframe panoramic camera. In *In 9th Conference on Optical 3-D*, July 2009.

[CE11] G. Caron and D. Eynard. Multiple camera types simultaneous stereo calibration. In *IEEE Int. Conf. on Robotics and Automation, ICRA'11*, Shanghai, China, May 2011.

[CF05] D. Claus and A.W. Fitzgibbon. A plumbline constraint for the rational function lens distortion model. In *Proceedings of the British Machine Vision Conference*, pages 99–108, 2005.

[CGC+11] Lilian Calvet, Pierre Gurdjos, Vincent Charvillat, Simone Gasparini, and Peter Sturm. Suivi de caméra à partir de marqueurs plans composés de cercles concentriques : paradigme et algorithmes. In *ORASIS - Congrès des jeunes chercheurs en vision par ordinateur*, Praz-sur-Arly, France, 2011. INRIA Grenoble Rhône-Alpes.

[CGN10] S. Choudhary, S. Gupta, and PJ Narayanan. Practical time bundle adjustment for 3d reconstruction on the gpu. *Computer Vision on GPUs–ECCV 2010 Workshops*, 2010.

[CGPV03] R. Cucchiara, C. Grana, A. Prati, and R. Vezzani. A hough transform-based method for radial lens distortion correction. In *Image Analysis and Processing, 2003. Proceedings. 12th International Conference on*, pages 182–187. IEEE, 2003.

[CI01] Y. Caspi and M. Irani. Alignment of non-overlapping sequences. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 76–83. IEEE, 2001.

[CKF+08] B. Clipp, J.H. Kim, J.M. Frahm, M. Pollefeys, and R. Hartley. Robust 6dof motion estimation for non-overlapping, multi-camera systems. In *Applications of Computer Vision, 2008. WACV 2008. IEEE Workshop on*, pages 1–8. IEEE, 2008.

[Cli10] B.S. Clipp. *Multi-camera simultaneous localization and mapping*. PhD thesis, University of North Carolina, 2010.

[CMEM07] J. Courbon, Y. Mezouar, L. Eck, and P. Martinet. A generic fisheye camera model for robotic applications. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1683–1688. IEEE, 2007.

[Cou09] Jonathan Courbon. *Navigation de robots mobiles par mémoire sensorielle*. PhD thesis, Université Blaise Pascal, Clermont-Ferrand, France, December 2009.

[CPRR06] Á. Catalá-Prat, J. Rataj, and R. Reulke. Self-calibration system for the orientation of a vehicle camera. In *Proc. of the ISPRS Com. V Symposium :" Image Engineering and Vision Metrology*, pages 68–73, 2006.

[CS09] V. Chari and P. Sturm. Multi-view geometry of the refractive plane. In *Proceedings of British Machine Vision Conference 2009 (BMVC2009)*, pages 1–11. Citeseer, 2009.

[CW] J. Chen and K.H. Wong. Calibration of an articulated camera system. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

[CWM+11] Z. Chen, KKY Wong, Y. Matsushita, X. Zhu, and M. Liu. Self-calibrating depth from refraction. In *International Conference on Computer Vision*, 2011.

[CWYO07] M.M.Y. Chang, K.H. Wong, Y.K. Yu, and S.H. Or. A robust head pose tracking system based on multiple cameras. Tech. rep. university of hong kong, 2007.

[CXF06] Xiaochun Cao, Jiangjian Xiao, and Hassan Foroosh. Camera motion quantification and alignment. In *Proceedings of the 18th International Conference on Pattern Recognition - Volume 02*, ICPR '06, pages 13–16, Washington, DC, USA, 2006. IEEE Computer Society.

[DD95] Daniel F. Dementhon and Larry S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15 :123–141, 1995.

[Del11] Pierre Delmas. *Génération active des déplacements d'un véhicule agricole dans son environnement.* PhD thesis, Université Blaise Pascal - Clermont II, 2011.

[DES10] G. Dubbelman, I. Esteban, and K. Schutte. Efficient trajectory bending with applications to loop closure. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1–7, Taipei, Taiwan, Oct. 18-22 2010.

[DF01] Frédéric Devernay and Olivier Faugeras. Straight lines have to be straight : automatic calibration and removal of distortion from scenes of structured enviroments. *Machine Vision and Applications*, 13(1) :14–24, 2001. Erratum available at http ://devernay.free.fr/publis/distcalib-erratum.html.

[DHLH] Y. Dai, M. He, H. Li, and R. Hartley. Factorization-based structure-and-motion computation for generalized camera model. In *Signal Processing, Communications and Computing (ICSPCC), 2011 IEEE International Conference on*, pages 1–6. IEEE.

[Dho09] M. Dhome. *Visual Perception Through Video Imagery*. Wiley-ISTE, 2009.

[DTL+10] Y. Dai, J. Trumpf, H. Li, N. Barnes, and R. Hartley. Rotation Averaging with Application to Camera-Rig Calibration. *Computer Vision–ACCV 2009*, pages 335–346, 2010.

[EL09] A. Eudes and M. Lhuillier. Error propagations for local bundle adjustment. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2411–2418. IEEE, 2009.

[EWK07a] S. Esquivel, F. Woelk, and R. Koch. Calibration of a Multi-camera Rig from Non-overlapping Views. In *Pattern Recognition : 29th DAGM Symposium, Heidelberg, Germany, September 12-14, 2007, Proceedings*, page 82. Springer-Verlag New York Inc, 2007.

[FKK04] J.M. Frahm, K. Koser, and R. Koch. Pose Estimation for Multi-camera Systems. In *Pattern recognition : 26th DAGM Symposium, Tübingen, Germany, August 30-1 September 2004 : proceedings*, page 286. Springer-Verlag New York Inc, 2004.

[FL05] I. Fassi and G. Legnani. Hand to sensor calibration : A geometrical interpretation of the matrix equation AX= XB. *Journal of Robotic Systems*, 22(9) :497, 2005.

[FMW03] A. R.J. François, G. G. Medioni, and R. Waupotitsch. Mirror symmetry 2-view stereo geometry. *Image and Vision Computing*, 21(2) :137–143, February 2003.

[Fox02] E.M. Foxlin. Generalized architecture for simultaneous localization, auto-calibration, and map-building. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, pages 527–533. IEEE, 2002.

[FP01] H. Farid and AC Popescu. Blind removal of lens distortion. *Journal of the Optical Society of America. A, Optics, image science, and vision*, 18(9) :2072, 2001.

[GD00] C. Geyer and K. Daniilidis. A unifying theory for central panoramic systems and practical implications. *Computer Vision—ECCV 2000*, pages 445–461, 2000.

[GdVLL10] P. Gupta, N. da Vitoria Lobo, and JJ Laviola. Markerless tracking using polar correlation of camera optical flow. In *Virtual Reality Conference (VR), 2010 IEEE*, pages 223–226. IEEE, 2010.

[GN98] J. Gluckman and S.K. Nayar. A real-time catadioptric stereo system using planar mirrors. In *Proceedings of the 1998 DARPA Image Understanding Workshop*, pages 309–313. Citeseer, 1998.

[GN01a] J. Gluckman and S.K. Nayar. Catadioptric stereo using planar mirrors. *International Journal of Computer Vision*, 44(1) :65–79, 2001.

[GN01b] Michael D. Grossberg and Shree K. Nayar. A general imaging model and a method for finding its parameters. In *In Proc. International Conference on Computer Vision*, pages 108–115, 2001.

[Gra01] Claus Gramkow. On averaging rotations. *Journal of Mathematical Imaging and Vision*, 15(1-2) :7–16, 2001.

[GSW06] P. Gurdjos, P. Sturm, and Y. Wu. Euclidean structure from n$\geqslant$ 2 parallel circles : theory and algorithms. *Computer Vision–ECCV 2006*, pages 238–252, 2006.

[HC09] C. Hu and L.F. Cheong. Linear quasi-parallax sfm using laterally-placed eyes. *International journal of computer vision*, 84(1) :21–39, 2009.

[Hey00] A. Heyden. Tensorial properties of multilinear constraints. *Mathematical Methods in the Applied Sciences*, 23 :169–202, 2000.

[HGJD09] C. Hughes, M. Glavin, E. Jones, and P. Denny. Wide-angle camera technology for automotive applications : a review. *Intelligent Transport Systems, IET*, 3(1) :19–31, 2009.

[HJ] R.A. Horn and C.R. Johnson. Matrix analysis. 1985. *Cambridge, Cambridge*.

[HK05] R.I. Hartley and S.B. Kang. Parameter-free radial distortion correction with centre of distortion estimation. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1834–1841. IEEE, 2005.

[HMR08a] Joel A. Hesch, Anastasios I. Mourikis, and Stergios I. Roumeliotis. Determining the camera to robot-body transformation from planar mirror reflections. In *IROS08*, volume 1, pages 3865–3871, Nice, France, Sept 22 – Sept 26 2008.

[HMR08b] Joel A. Hesch, Anastasios I. Mourikis, and Stergios I. Roumeliotis. Mirror-based extrinsic camera calibration. In *In The Eighth International Workshop on the Algorithmic Foundations of Robotics (WAFR'08)*, Mexico, Dec 22 – Dec 26 2008.

[HMR10] J. Hesch, A. Mourikis, and S. Roumeliotis. Extrinsic camera calibration using multiple reflections. *Computer Vision–ECCV 2010*, pages 311–325, 2010.

[HS88] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

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

[IKL+10] I. Ihrke, K.N. Kutulakos, H. Lensch, M. Magnor, and W. Heidrich. Transparent and specular object reconstruction. In *Computer Graphics Forum*. Wiley Online Library, 2010.

[ISY03] S. Ikeda, T. Sato, and N. Yokoya. A calibration method for an omnidirectional multi-camera system. In *Proc. SPIE*, volume 5006, pages 499–507, 2003.

[JNS+10] Y. Jeong, D. Nister, D. Steedly, R. Szeliski, and I.S. Kweon. Pushing the envelope of modern methods for bundle adjustment. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1474–1481. IEEE, 2010.

[JQ05] G. Jiang and L. Quan. Detection of concentric circles for camera calibration. In *Computer Vision. ICCV. Tenth IEEE International Conference on*, volume 1, pages 333–340. IEEE, 2005.

[JRAS03] O. Javed, Z. Rasheed, O. Alatas, and M. Shah. Knight/spl trade : a real time surveillance system for multiple and non-overlapping cameras. In *icme*, pages 649–652. IEEE, 2003.

[JSO10] W. Jiang, M. Shimizu, and M. Okutomi. Single-camera multi-baseline stereo using fish-eye lens and mirrors. *Computer Vision–ACCV 2009*, pages 347–358, 2010.

[Kae08] M. Kaess. *Incremental Smoothing and Mapping*. Ph.D., Georgia Institute of Technology, Dec 2008.

[KC06] J.H. Kim and M.J. Chung. Absolute motion and structure from stereo image sequences without stereo correspondence and analysis of degenerate cases. *Pattern Recognition*, 39(9) :1649–1661, 2006.

[KCL05] F. Kahn, M. Chapman, and J. Li. Camera calibration for a robust omni-directional photogrammetry system. In *Proceedings of the 5th International Symposium on Mobile Mapping Technology, ser. MMT07*, pages 1–8, 2005.

[KCPL08] J.H. Kim, M.J. Chung, C.J. Park, and I.H. Lee. Sequential motion and scene reconstruction from image sequences captured by a multi-camera system. In *Proceedings of the British Machine Conference, pages*, pages 116–1, 2008.

[KD06] M. Kaess and F. Dellaert. Visual slam with a multi-camera rig. *Georgia Institute of Technology, Tech. Rep. GIT-GVU-06-06*, 2006.

[KD10] M. Kaess and F. Dellaert. Probabilistic structure matching for visual SLAM with a multi-camera rig. *Computer Vision and Image Understanding, CVIU*, 114 :286–296, Feb 2010.

[KGB11] R. Kummerle, G. Grisetti, and W. Burgard. Simultaneous calibration, localization, and mapping. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 3716–3721. IEEE, 2011.

[KGK05] J.S. Kim, P. Gurdjos, and I.S. Kweon. Geometric and algebraic constraints of projected concentric circles and their applications to camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(4) :637–642, 2005.

[KHFP07] J.H. Kim, R. Hartley, J.M. Frahm, and M. Pollefeys. Visual Odometry for Non-Overlapping Views Using Second-Order Cone Programming. *Lecture Notes in Computer Science*, 4844 :353, 2007.

[KHK08] Jun-Sik Kim, Myung Hwangbo, and T. Kanade. Motion estimation using multiple non-overlapping cameras for small unmanned aerial vehicles. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3076 –3081, 19-23 2008.

[KHK10] J.S. Kim, M. Hwangbo, and T. Kanade. Spherical approximation for multiple cameras in motion estimation : Its applicability and advantages. *Computer Vision and Image Understanding*, 2010.

[KIFP] R.K. Kumar, A. Ilie, J.-M. Frahm, and M. Pollefeys. Simple calibration of non-overlapping cameras with a mirror. *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1 –7.

[Kim08] J.H. Kim. *Camera Motion Estimation for Multi-Camera Systems*. PhD thesis, The Australian National University, 2008.

[Kin93] BA King. Optimisation of bundle adjustments for stereo photography. *International Archives Of Photogrammetry And Remote Sensing*, 29 :168–168, 1993.

[KJCTC10] J.H. Kim, M. Jin Chung, and B. Tae Choi. Recursive estimation of motion and a scene model with a two-camera system of divergent view. *Pattern Recognition*, 43(6) :2265–2280, 2010.

[KJR+11] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. isam2 : Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3281–3288. IEEE, 2011.

[KKK02] J.S. Kim, H.W. Kim, and I.S. Kweon. A camera calibration method using concentric circles for vision applications. *ACCV2002, Melbourne, Australia*, 2002.

[KRD08] M. Kaess, A. Ranganathan, and F. Dellaert. isam : Incremental smoothing and mapping. *Robotics, IEEE Transactions on*, 24(6) :1365–1378, 2008.

[KT99] F. Kahl and B. Triggs. Critical motions in euclidean structure from motion. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 1999.

[KTS11] Christian Kurz, Thorsten Thormählen, and Hans-Peter Seidel. Bundle adjustment for stereoscopic 3d. In André Gagalowicz and Wilfried Philips, editors, *Computer Vision/Computer Graphics Collaboration Techniques*, volume 6930 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2011.

[LA09] M.I.A. Lourakis and A.A. Argyros. Sba : A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software (TOMS)*, 36(1) :2, 2009.

[LB10] J. Lim and N. Barnes. Estimation of the epipole using optical flow at antipodal points. *Computer Vision and Image Understanding*, 114(2) :245–253, 2010.

[LBL10] J. Lim, N. Barnes, and H. Li. Estimating relative camera motion from the antipodal-epipolar constraint. *IEEE transactions on pattern analysis and machine intelligence*, pages 1907–1914, 2010.

[LBR+10] P. Lothe, S. Bourgeois, E. Royer, M. Dhome, and S. Naudet-Collette. Real-time vehicle global localisation with a single camera in dense urban areas : Exploitation of coarse 3d city models. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 863–870. IEEE, 2010.

[LD07] J. Lobo and J. Dias. Relative pose calibration between visual and inertial sensors. *The International Journal of Robotics Research*, 26(6) :561, 2007.

[LdInMH02] Diego López de Ipiña, Paulo R. S. Mendonça, and Andy Hopper. Trip : A low-cost vision-based location system for ubiquitous computing. *Personal Ubiquitous Comput.*, 6(3) :206–219, 2002.

[LF93] Q.T. Luong and O. Faugeras. Self-calibration of a stereo rig from unknown camera motions and point correspondences. *Rapports de recherche-INRIA*, 1993.

[LHK08] Hongdong Li, Richard I. Hartley, and Jae-Hak Kim. A linear approach to motion estimation using generalized camera models. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2008*, 2008.

[lig] http://www.automobiles-ligier.com.

[Lim10] J.J.K. Lim. *Egomotion Estimation with Large Field-of-View Vision*. PhD thesis, 2010.

[LNCS10] J.L. Lerma, S. Navarro, M. Cabrelles, and A.E. Seguí. Camera calibration with baseline distance constraints. *The Photogrammetric Record*, 25(130) :140–158, 2010.

[Lou10] Manolis I.A. Lourakis. Sparse non-linear least squares optimization for geometric vision. In *European Conference on Computer Vision*, volume 2, pages 43–56, 2010.

[LRFK07] B. Lamprecht, S. Rass, S. Fuchs, and K. Kyamakya. Extrinsic camera calibration for an on-board two-camera system without overlapping field of view. In *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pages 265 –270, sept. 2007.

[LRL00] J.M. Lavest, G. Rives, and J.T. Lapresté. Underwater camera calibration. In *Proceedings of the 6th European Conference on Computer Vision-Part II*, pages 654–668. Springer-Verlag, 2000.

[LRL03] JM Lavest, G. Rives, and JT Lapreste. Dry camera calibration for underwater applications. *Machine Vision and Applications*, 13(5) :245–253, 2003.

[LRWW99] J.C. Lagarias, J.A. Reeds, M.H. Wright, and P.E. Wright. Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM Journal on Optimization*, 9(1) :112–147, 1999.

[LVD98] Jean-Marc Lavest, Marc Viala, and Michel Dhome. Do we really need an accurate calibration pattern to achieve a reliable camera calibration ? In *Proceedings of the 5th European Conference on Computer Vision-Volume I-Volume I*, pages 158–174, London, UK, 1998. Springer-Verlag.

[LVD99] Jean M. Lavest, Marc Viala, and Michel Dhome. Quelle précision pour une mire d'étalonnage. *GRETSI Traitement du Signal*, 16(3) :241–254, 1999.

[LZWS11] Z. Liu, G. Zhang, Z. Wei, and J. Sun. A global calibration method for multiple vision sensors based on multiple targets. *Measurement Science and Technology*, 22 :125102, 2011.

[MAHW11] D. Muhle, S. Abraham, C. Heipke, and M. Wiggenhagen. Estimating the mutual orientation in a multi-camera system with a non overlapping field of view. *Photogrammetric Image Analysis*, pages 13–24, 2011.

[Mal11] Florent Malartre. *Perception intelligente pour la navigation rapide de robots mobiles en environnement naturel*. PhD thesis, Université Blaise Pascal - Clermont II, 2011.

[Mar09] A. Martinelli. Local decomposition and observability properties for automatic calibration in mobile robotics. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 4182–4188. IEEE, 2009.

[MCMar] Y. Mezouar, J. Courbon, and P. Martinet. Evaluation of the unified model on the sphere for fisheye cameras in robotic applications. *Advanced Robotics*, To appear.

[MGP93] HG Maas, A. Gruen, and D. Papantoniou. Particle tracking velocimetry in three-dimensional flows. *Experiments in Fluids*, 15(2) :133–146, 1993.

[MH03] J. Mitchelson and A. Hilton. Wand-based multiple camera studio calibration. *Center Vision, Speech and Signal Process*, 2003.

[MLD+09] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Generic and real-time structure from motion using local bundle adjustment. *Image and Vision Computing*, 27(8) :1178–1193, 2009.

[MM05] E. Malis and E. Marchand. Méthodes robustes d'estimation pour la vision robotique. In *Journées nationales de la recherche en robotique, JNRR'05*, Guidel, France, France, 2005.

[Moa02] M. Moakher. Means and averaging in the group of rotations. *SIAM Journal on Matrix Analysis and Applications*, 24(1) :1–16, 2002.

[Mou07] Etienne Mouragnon. *Reconstruction 3D et localisation simultannée de caméras mobiles : une approche temps-réel par ajustement de faisceaux local*. PhD thesis, Ecole Doctorale Sciences Pour l'Ingénieur de Clermont-Ferrand, 2007.

[MSA05] C. Mei and I. Sophia-Antipolis. Camera calibration toolbox for matlab. *Aug*, 4 :5, 2005.

[MSC+10] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. Rslam : A system for large-scale mapping in constant-time using stereo. *International Journal of Computer Vision*, 2010.

[MSMP09] G.L. Mariottini, S. Scheggi, F. Morbidi, and D. Prattichizzo. Planar catadioptric stereo : single and multi-view geometry for calibration and localization. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 1510–1515. IEEE, 2009.

[MWS06] A. Martinelli, J. Weingarten, and R. Siegwart. Theoretical results on on-line sensor self-calibration. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 43–48. IEEE, 2006.

[NHA10] P. Nyman, A. Heyden, and K. Aström. Multi-camera platform calibration using multi-linear constraints. In *2010 International Conference on Pattern Recognition*, pages 53–56. IEEE, 2010.

[NJ04] Martins N. and Dias J. Camera calibration using reflections in planar mirrors and object reconstruction using volume carving method. *Imaging Science Journal, The*, 52 :117–130(14), 2004.

[NN98] S.A. Nene and S.K. Nayar. Stereo with mirrors. In *Computer Vision, 1998. Sixth International Conference on*, pages 1087–1094. IEEE, 1998.

[NPD08] A. Negre, C. Pradalier, and M. Dunbabin. Robust vision-based underwater target identification and homing using self-similar landmarks. In *Field and Service Robotics : Results of the 6th International Conference*, page 51. Springer Verlag, 2008.

[NVMG06] S. Nedevschi, C. Vancea, T. Marita, and T. Graf. On-line calibration method for stereovision systems used in vehicle applications. In *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*, pages 957–962. IEEE, 2006.

[NW99] J. Nocedal and S.J. Wright. *Numerical optimization*. Springer verlag, 1999.

[oPSTH80] American Society of Photogrammetry, C.C. Slama, C. Theurer, and S.W. Henriksen. *Manual of photogrammetry*. American Society of Photogrammetry, 1980.

[Pag] F. Pagel. Calibration of non-overlapping cameras in vehicles. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pages 1178–1183. IEEE.

[Pen98] Xavier Pennec. Computing the Mean of Geometric Features Application to the Mean Rotation. Technical Report RR-3371, INRIA, 03 1998.

[PFF+10] M. Pollefeys, J.M. Frahm, F. Fraundorfer, C. Zach, C. Wu, B. Clipp, and D. Gallup. Challenges in wide-area structure-from-motion. *IPSJ Transactions on Computer Vision and Applications*, 2(0) :105–120, 2010.

[PFTV92] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C : The Art of Scientific Computing, Second Edition*. Cambridge University Press, 2 edition, October 1992.

[Ple03] R. Pless. Using many cameras as one. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2003. Proceedings*, pages 587–593, 2003.

[PST99] F. Pedersini, A. Sarti, and S. Tubaro. Accurate and simple geometric calibration of multi-camera systems. *Signal Processing*, 77(3) :309–334, 1999.

[PVG00] M. Pollefeys and L. Van Gool. Some geometric insight in self-calibration and critical motion sequences. Technical report nr. kul/esat/psi/0001, k.u.leuven, 2000.

[PW] F. Pagel and D. Willersinn. Motion-based online calibration for non-overlapping camera views. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 843–848. IEEE.

[RBN10] R. Rodrigues, J. Barreto, and U. Nunes. Camera pose estimation using images of planar mirror reflections. *Computer Vision–ECCV 2010*, pages 382–395, 2010.

[RDD04] A. Rahimi, B. Dunagan, and T. Darrell. Simultaneous calibration and tracking with a network of non-overlapping sensors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004*, volume 1, pages I–187 – I–194 Vol.1, 27 2004.

[RHK+11] J.Y. Rau, A.F. Habib, A.P. Kersting, K.W. Chiang, K.I. Bang, Y.H. Tseng, and Y.H. Li. Direct sensor orientation of a land-based mobile mapping system. *Sensors*, 11(7) :7243–7261, 2011.

[RL11] E. Rosten and R. Loveland. Camera distortion self-calibration using the plumb-line constraint and minimal hough entropy. *Machine Vision and Applications*, 22(1) :77–85, 2011.

[RLDL07] E. Royer, M. Lhuillier, M. Dhome, and J.M. Lavest. Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision*, 74(3) :237–260, 2007.

[RLPK10] T. Ruland, H. Loose, T. Pajdla, and L. Kruger. Hand-eye autocalibration of camera positions on vehicles. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 367–372. IEEE, 2010.

[RLS06] S. Ramalingam, S.K. Lodha, and P. Sturm. A generic structure-from-motion framework. *Computer Vision and Image Understanding*, 103(3) :218–228, 2006.

[RPK] T. Ruland, T. Pajdla, and L. Kruger. Global optimization of extended hand-eye calibration. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 740–745. IEEE.

[RW] ME Ragab and KH Wong. Multiple nonoverlapping camera pose estimation. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 3253–3256. IEEE.

[RW07] ME Ragab and KH Wong. Extended kalman filter based pose estimation using multiple cameras. Technical report, Internal Report, CUHK, http ://www. cse. cuhk. hk/khwong/papers. html, 2007.

[RWCC] ME Ragab, KH Wong, JZ Chen, and M. Chang. Ekf pose estimation : How many filters and cameras to use ? In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 245–248. IEEE.

[RWCC07] ME Ragab, KH Wong, JZ Chen, and MMY Chang. EKF Based Pose Estimation using Two Back-to-Back Stereo Pairs. In *IEEE International Conference on Image Processing, 2007. ICIP 2007*, volume 6, 2007.

[SÅ04] H. Stewenius and K. Åström. Structure and motion problems for multiple rigidly moving cameras. *European Conference on Computer Vision, ECCV 2004*, pages 252–263, 2004.

[SB06] P. Sturm and T. Bonfort. How to compute the pose of an object without a direct view ? *Computer Vision–ACCV 2006*, pages 21–31, 2006.

[SBSV10] J. Santos, A. Bernardino, and J. Santos-Victor. Sensor-based self-calibration of the icub's head. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5666–5672. IEEE, 2010.

[SI03] A. Sugimoto and T. Ikeda. Diverging viewing-lines in binocular vision : A method for estimating ego motion by mounted active cameras. In *Proc. of Australia-Japan Advanced Workshop on Computer Vision*, pages 126–133. Citeseer, 2003.

[SIY04] T. Sato, S. Ikeda, and N. Yokoya. Extrinsic camera parameter recovery from multiple image sequences captured by an omni-directional multi-camera system. *Computer Vision-ECCV 2004*, pages 326–340, 2004.

[SMP05] Tomas Svoboda, Daniel Martinec, and Tomas Pajdla. A convenient multi-camera self-calibration for virtual environments. *PRESENCE : Teleoperators and Virtual Environments*, 14(4) :407–422, August 2005.

[SMRN09] G. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Robotics Science and Systems Conference*. Citeseer, 2009.

[SMRN10] G. Sibley, C. Mei, I. Reid, and P. Newman. Vast-scale outdoor navigation using adaptive relative bundle adjustment. *The International Journal of Robotics Research*, 29(8) :958, 2010.

[SOÅ05] Henrik Stewénius, Magnus Oskarsson, and Kalle Åström. Reconstruction from planar motion image sequences with applications for autonomous vehicles. In *Scandinavian Conf. on Image Analysis*, pages 609–618, 2005.

[Soi10] P. Soille. *Morphological Image Analysis : Principles and Applications*. Springer, 2010.

[SR03] Peter Sturm and Srikumar Ramalingam. A Generic Calibration Concept : Theory and Algorithms. Technical Report RR-5058, INRIA, December 2003.

[SRL05] P. Sturm, S. Ramalingam, and SK Lodha. On calibration, structure-from-motion and multi-view geometry for panoramic camera models. In *Panoramic Photogrammetry Workshop, Berlin, Germany*. Citeseer, 2005.

[SRT+11] Peter Sturm, Srikumar Ramalingam, Jean-Philippe Tardif, Simone Gasparini, and Joao Barreto. Camera Models and Fundamental Concepts Used in Geometric Computer Vision. *Foundations and Trends in Computer Graphics -New York-Association for Computing Machinery- and Vision*, 6(1-2) :1–183, 2011.

[SS97] W.H. Steeb and T.K. Shi. *Matrix calculus and Kronecker product with applications and C++ programs*. World Scientific Publishing Co., Inc., 1997.

[Ste03] H. Stewenius. Simplified vehicle calibration using multilinear constraints. In *Proceedings of the 13th Scandinavian conference on Image analysis*, SCIA'03, pages 669–676, Berlin, Heidelberg, 2003. Springer-Verlag.

[Ste05] H Stewénius. *Gröbner Basis Methods for Minimal Problems in Computer Vision*. PhD thesis, Centre for Mathematical Sciences LTH, Lund University, Sweden, 2005.

[Stu97] P. Sturm. Critical motion sequences for monocular self-calibration and uncalibrated euclidean reconstruction. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1100–1105. IEEE, 1997.

[Stu99] P. Sturm. Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length. In *British Machine Vision Conference, Nottingham, England*, pages 63–72. Citeseer, 1999.

[Tar94] Jean-Philippe Tarel. Calibration de camera fondée sur les ellipses. Rapport de recherche RR-2200, INRIA, January 1994. Projet SYNTIM.

[TL89] R.Y. Tsai and R.K. Lenz. A new technique for fully autonomous and efficient 3d robotics hand/eye calibration. *Robotics and Automation, IEEE Transactions on*, 5(3) :345–358, 1989.

[TMHF00] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment-a modern synthesis. *Vision algorithms : theory and practice*, pages 153–177, 2000.

[TSR06] J.P. Tardif, P. Sturm, and S. Roy. Self-calibration of a general radially symmetric distortion model. *Computer Vision–ECCV 2006*, pages 186–199, 2006.

[TSS08] T. Treibitz, Y.Y. Schechner, and H. Singh. Flat refractive geometry. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[VA09] R. Valkenburg and N. Alwesh. Calibration of the relative poses of multiple cameras. In *Image and Vision Computing New Zealand, 2009. IVCNZ'09. 24th International Conference*, pages 185–190. IEEE, 2009.

[WACS11] C. Wu, S. Agarwal, B. Curless, and S.M. Seitz. Multicore bundle adjustment. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3057–3064. IEEE, 2011.

[Wel86] W.T. Welford. *Aberrations of optical systems*. The Adam Hilger series on optics and optoelectronics. A. Hilger, 1986.

[WLS11] Q. Wang, Y. Liu, and Y. Shen. An accurate extrinsic camera self-calibration method in non-overlapping camera sensor networks. In *Instrumentation and Measurement Technology Conference (I2MTC), 2011 IEEE*, pages 1–6. IEEE, 2011.

[WZHW04] Y. Wu, H. Zhu, Z. Hu, and F. Wu. Camera calibration from the quasi-affine invariance of two parallel circles. *Computer Vision-ECCV 2004*, pages 190–202, 2004.

[YHZ06] X. Ying, Z. Hu, and H. Zha. Fisheye lenses calibration using straight-line spherical perspective projection constraint. *Computer Vision–ACCV 2006*, pages 61–70, 2006.

[YZ07] X. Ying and H. Zha. An efficient method for the detection of projected concentric circles. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 6, pages VI–560. IEEE, 2007.

[Zha97] Z. Zhang. Parameter estimation techniques : A tutorial with application to conic fitting. *Image and vision Computing*, 15(1) :59–76, 1997.

[Zha00] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on PAMI*, 22(11) :1330–1334, 2000.

Glossaire

Caméra Désigne le capteur physique réalisant des acquisitions d'images. Contrairement au jargon de nombreux articles de vision monoculaire, le terme "caméra" ne désigne pas la pose de celle-ci.

ComSee Acronyme de « Computers that See ».

Coordonnées homogènes Coordonnées dans un espace projectif.

Coordonnées cartésiennes Coordonnées dans un espace euclidien (parfois appelées *inhomogeneous coordinate* en anglais).

EKF Filtre de Kalman étendu (*Extended Kalman Filter* en anglais). Algorithme estimant l'état d'un système avec des modèles d'observation et d'évolution différentiables.

GRAVIR Acronyme de « Groupe d'Automatique, VIsion et Robotique ».

Inlier Entité n'étant pas aberrante.

LASMEA Acronyme de « LAboratoire des Sciences et Matériaux pour l'Electronique et d'Automatique ». Au 1^{er} janvier 2012, les laboratoires LASMEA, LaMI et LGCB se sont regroupés pour former l'*Institut Pascal* (UMR 6602 Université Blaise Pascal/CNRS/IFMA).

LIDAR Acronyme de « Light Detection and Ranging » désignant un capteur de télédétection par laser.

LM Acronyme de l'algorithme d'optimisation non-linéaire de Levenberg-Marquardt.

MCBA Ajustement de faisceaux multi-caméra (pour *Multi-Camera Bundle Adjustment*, défini en Section 5.4, p89).

Outlier Entité aberrante (ex : un point 2D outlier peut être une mauvaise détection).

Paramètres intrinsèques Paramètres modélisant le processus de formation de l'image interne à une caméra.

Paramètres extrinsèques Paramètres modélisant la transformation rigide entre la caméra maître et une autre (caméra esclave). On conserve cette définition, car ces paramètres sont externes à une caméra. Cependant, et bien que ce ne soit pas le cas dans ce manuscrit, la notion de paramètres extrinsèques peut être associée à la *pose* d'une caméra (ou d'un système multi-caméra).

Pose La pose d'un objet est la rotation et la translation entre un repère de référence et le repère attaché à l'objet. Une pose peut s'interpréter comme une transformation euclidienne (*cf* Annexe A.2.1).

VIPA Acronyme de « Véhicule Individuel Public Autonome ».

Vissage d'axe $\mathbf{a}$ rotation d'axe $\mathbf{a}$ suivi d'une translation selon le même axe.

Scène Ensemble d'amers visuels observables par une caméra.

SFM (ou SfM) Structure From Motion. Catégorie d'algorithmes reconstruisant la scène observée à partir du mouvement.

SVD Décomposition en valeurs singulières (*Singular Value Decomposition* en anglais).

Étalonnage de caméras à champs disjoints et reconstruction 3D - Application à un robot mobile

Ces travaux s'inscrivent dans le cadre du projet VIPA « Véhicule Individuel Public Autonome », au cours duquel le LASMEA et ses partenaires ont mis au point des véhicules capables de naviguer automatiquement, sans aucune infrastructure extérieure dédiée, dans des zones urbaines (parkings, zones piétonnes, aéroports). Chaque véhicule est doté de deux caméras, l'une à l'avant, et l'autre à l'arrière. Avant son déploiement, il doit tout d'abord être étalonné et conduit manuellement afin de reconstruire la carte d'amers visuels dans laquelle il naviguera ensuite automatiquement.

Les travaux de cette thèse ont pour but de développer et de mettre en œuvre des méthodes souples permettant d'étalonner cet ensemble de caméras dont les champs de vue sont totalement disjoints. Après une étape préalable d'étalonnage intrinsèque et un état de l'art sur les systèmes multi-caméra, nous développons et mettons en œuvre différentes méthodes d'étalonnage extrinsèque (déterminant les poses relatives des caméras à champs de vue disjoints).

La première méthode présentée utilise un miroir plan pour créer un champ de vision commun aux différentes caméras. La seconde approche consiste à manœuvrer le véhicule pendant que chaque caméra observe une scène statique composée de cibles (dont la détection est sous-pixellique). Dans la troisième approche, nous montrons que l'étalonnage extrinsèque peut être obtenu simultanément à la reconstruction 3D (par exemple lors de la phase d'apprentissage), en utilisant des points d'intérêt comme amers visuels. Pour cela un algorithme d'ajustement de faisceaux multi-caméra a été développé avec une implémentation creuse. Enfin, nous terminons par un étalonnage déterminant l'orientation du système multi-caméra par rapport au véhicule.

Mots-clés : Étalonnage, reconstruction 3D, ajustement de faisceaux, caméras à champs de vue non-recouvrants, miroir plan, VIPA.

Non-overlapping camera calibration and 3D reconstruction : Application to Vision-Based Robotics

My research was involved in the VIPA « Automatic Electric Vehicle for Passenger Transportation » project. During which, the LASMEA and its partnerships have developed vehicles able to navigate autonomously, without any outside dedicated infrastructure in an urban environment (parking lots, pedestrian areas, airports). Two cameras are rigidly embedded on a vehicle : one at the front, another at the back. Before being available for autonomous navigation tasks, the vehicle has to be calibrated and driven manually in order to build a visual 3D map (calibration and learning steps). Then, the vehicle will use this map to localize itself and drive autonomously.

The goals of this thesis are to develop and apply user friendly methods, which calibrate this set of non-overlapping cameras. After a first step of intrinsic calibration and a state of the art on multi-camera rigs, we develop and test several methods to extrinsically calibrate non-overlapping cameras (*i.e.* estimate the camera relative poses).

The first method uses a planar mirror to create an overlap between views of the different cameras. The second procedure consists in manoeuvring the vehicle while each camera observes a static scene (composed of a set of targets, which are detected accurately). In a third procedure, we solve the 3D reconstruction and the extrinsic calibration problems simultaneously (the learning step can be used for that purpose) relying on visual features such as interest points. To achieve this goal a multi-camera bundle adjustment is proposed and implemented with a sparse data structures. Lastly, we present a calibration of the orientation of a multi-camera rig relative to the vehicle.

Keywords : Calibration, 3D reconstruction, bundle adjustment, non-overlapping cameras, planar mirror, VIPA.