Contribution à la localisation pour le véhicule autonome

Rémi Boutteau

Normandie Université

Habilitation à Diriger des Recherches

Ecole doctorale MIIS – ED 590

Spécialité : Automatique, Signal, Productique, Robotique

Contribution à la localisation pour le véhicule autonome

Présentée et soutenue par
Rémi BOUTTEAU

HDR soutenue publiquement le 7 Décembre 2018 devant le jury composé de		
M. Peter STURM	Directeur de Recherche INRIA Grenoble Rhône-Alpes	Président du jury
Mme Samia BOUCHAFA-BRUNEAU	Professeur des Universités Université d'Evry Val d'Essonne / Université Paris-Saclay, IBISC	Rapporteur
M. Philippe MARTINET	Directeur de Recherche INRIA Sophia-Antipolis Méditerranée	Rapporteur
M. Stéphane VIOLLET	Directeur de Recherche, CNRS, Institut des Sciences du Mouvement	Rapporteur
M. Cédric DEMONCEAUX	Professeur des Universités Université de Bourgogne Franche-Comté, LE2I	Examinateur
M. Xavier SAVATIER	Enseignant-Chercheur HDR ESIGELEC, IRSEEM	Examinateur
M. Pascal VASSEUR	Professeur des Universités Université de Rouen Normandie, LITIS	Examinateur – Garant

Institut de Recherche en Systèmes Electroniques Embarqués (IRSEEM) – EA 4353

UNIVERSITÉ DE ROUEN NORMANDIE

MIIS

irseem

ESIGELEC

Liste des abréviations, sigles et acronymes

ADAS	Advanced Driver-Assistance Systems
AF	Ajustement de faisceaux
AGV	Automatic Guided Vehicle
ANR	Agence Nationale de la Recherche
APP	Apprentissage Par Problèmes
BoW	Bag of Words
CISE	Campus Intégration Systèmes Embarqués
CM	Cours Magistraux
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
ddl	degré de liberté
fps	frames per second
GPS	Global Positioning System
GPU	Graphics Processing Unit
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
ISE-VA	Ingénierie des Systèmes Embarqués - Véhicule Autonome
ISYMED	Ingénierie des Systèmes Médicaux
ITE	Institut pour la Transition Énergétique
LIDAR	LIght Detection And Ranging
MCTGE	MéCaTronique et Génie Électrique
ML	Multi Layer
NFI	Nouvelle France Industrielle
PIA	Programme d'Investissements d'Avenir
PnL	Perspective-n-Line
PnP	Perspective-n-Point
RANSAC	RANdom SAmple Consensus
RMSE	Root Mean Square Error
ROV	Remotely Operated Vehicle
RTK	Real Time Kinematic
SfM	Structure from Motion
SL	Single Layer

SLAM	Simultaneous Localization And Mapping
SoC	System-on-Chip
SVD	Singular Value Decomposition
TD	Travaux Dirigés
TP	Travaux Pratiques
UAV	Unmanned Aerial Vehicle
UWB	Ultra Wide Band
VO	Visual Odometry

Table des matières

Liste des abréviations, sigles et acronymes **3**

Introduction **9**

1 Synthèse de mes activités scientifiques, administratives et pédagogiques **11**
1.1 Mon parcours . . . 13
1.1.1 Fonctions actuelles . . . 13
1.1.2 Activités professionnelles . . . 13
1.1.3 Formation . . . 14
1.2 Descriptions de mes activités d'enseignement et de mes responsabilités pédagoqiques 14
1.2.1 Synthèse des enseignements dispensés . . . 14
1.2.2 La dominante ISE-VA . . . 15
1.2.3 Implication dans la pédagogie active . . . 16
1.3 Encadrements . . . 17
1.3.1 Co-encadrement de thèses (soutenues) . . . 17
1.3.2 Co-encadrement de thèses (en cours) . . . 18
1.3.3 Encadrement de stages de Master 2 . . . 18
1.3.4 Encadrement de stages de fin d'études . . . 18
1.3.5 Autres stages . . . 19
1.3.6 Ingénieurs de recherche et post-doctorants . . . 19
1.4 Responsabilités au sein de l'Institut de Recherche en Systèmes Électroniques Embarqués . . . 19
1.4.1 Contribution à la mise en place du Campus Intégration des Systèmes Embarqués . . . 19
1.4.2 Montage, coordination et participation à des projets . . . 20
1.4.3 Valorisation de la recherche par des contrats . . . 21
1.5 Rayonnement scientifique . . . 21
1.5.1 Participation à des comités de relecture . . . 21
1.5.2 Membre du comité de domaine de recherche 2 de l'institut VEDECOM . 22
1.5.3 Participation à des GDRs . . . 22
1.5.4 Expertise de projets . . . 23
1.5.5 Formation professionnelle . . . 23
1.5.6 Collaborations . . . 23
1.6 Liste de mes publications . . . 24
1.6.1 Articles dans des revues internationales . . . 24

1.6.2 Communications avec actes dans un congrès international 25
1.6.3 Communications avec actes dans un congrès national 26
1.6.4 Chapitres de livre 27
1.6.5 Mémoires 28

2 Localisation basée vision du véhicule 29
2.1 Introduction et contexte 30
2.2 Structure-from-Motion à partir d'un réseau de caméras asynchrone 31
2.2.1 Etat de l'art 32
2.2.2 La méthode des triangles 34
2.2.3 Optimisation par ajustement de faisceaux 38
2.2.4 Résultats expérimentaux 41
2.3 Localisation visuelle multimodale visible/infrarouge 48
2.3.1 État de l'art 48
2.3.2 Méthodologie 51
2.3.3 Application de PHROG au problème de *place recognition* 58
2.4 Architectures embarquées pour la localisation basée vision 65
2.4.1 Estimation du mouvement relatif basé CUDA à partir de correspondances 2D-2D 66
2.4.2 Implémentation d'un algorithme de *place recognition* sur GPU 70
2.5 Conclusion 73

3 Solutions Minimales pour l'odométrie visuelle et l'estimation de pose 75
3.1 Introduction et contexte 76
3.2 Odométrie Visuelle connaissant une direction commune 77
3.2.1 Méthodes d'estimation du mouvement 78
3.2.2 Résultats expérimentaux 86
3.3 Estimation de l'attitude et de l'altitude d'un drone 94
3.3.1 Formulation du problème 95
3.3.2 Une solution géométrique pour l'estimation de l'altitude et de l'attitude . 97
3.3.3 Estimations robustes 100
3.3.4 Calibrage du système 107
3.3.5 Résultats expérimentaux 109
3.4 Conclusion 115

4 Localisation 6 ddl basée lidar 117
4.1 Introduction et contexte 118
4.2 État de l'art de la localisation en milieu industriel 119
4.2.1 SLAM et localisation 119
4.2.2 Technologies en milieu industriel 120
4.2.3 Travaux analogues basés lidar 120
4.2.4 Contraintes de l'environnement par rapport aux solutions existantes . . . 121
4.3 Méthodologie 121
4.3.1 Notions sur la théorie de la localisation 121
4.3.2 Méthode de filtrage retenue 122
4.3.3 Champ de vraisemblance 3D 124

4.3.4 Stockage du champ de vraisemblance . . . 126
4.4 Résultats expérimentaux . . . 127
4.4.1 Performance du stockage du champ de vraisemblance par l'octree hybride 129
4.4.2 Caractéristiques de la fonction de vraisemblance . . . 132
4.4.3 Etude de la convergence de l'algorithme . . . 134
4.4.4 Localisation en simulation . . . 136
4.4.5 Evaluation des performances sur robots réels . . . 137
4.5 Conclusion . . . 139

5 Bilan et Perspectives **141**
5.1 Bilan . . . 142
5.2 Perspectives de Recherche . . . 142
5.2.1 Approches non-conventionnelles pour la localisation . . . 142
5.2.2 Solutions minimales pour l'estimation de pose . . . 143
5.2.3 Localisation Lidar . . . 144
5.2.4 Localisation multi-modale . . . 144
5.2.5 Utilisation d'informations sémantiques pour la localisation . . . 145
5.3 Moyens mis en œuvre pour développer le projet . . . 145
5.3.1 Utilisation des moyens issus des projets existants . . . 145
5.3.2 Renforcement des contrats industriels . . . 145
5.3.3 Montage de nouveaux projets . . . 145
5.3.4 Renforcement des collaborations . . . 146

Bibliographie **147**

Introduction

Ces dernières années, nous avons pu assister à l'avènement du véhicule autonome. Chaque jour, des constructeurs, équipementiers automobiles, géants du web ou laboratoires proposent de nouveaux prototypes ou de nouvelles fonctionnalités pour essayer de ne pas se faire distancer dans cette course à la voiture autonome. Le véhicule totalement autonome désigne un véhicule entièrement piloté par une intelligence artificielle. Celle-ci doit être capable de diriger le véhicule, en percevant et en comprenant son environnement pour adapter sa conduite en fonction des évènements (présence d'un accident ou de travaux par exemple), et des autres usagers de la route (voitures, cyclistes, piétons, etc). Cette révolution des transports est annoncée dès 2020, mais elle présente encore de nombreux enjeux et défis technologiques. L'intelligence artificielle promet d'écarter les erreurs humaines, à l'origine de plus de 94% des accidents de la route [1]. Elle n'est cependant pas encore synonyme du risque zéro, comme en témoignent les quelques accidents survenus avec des véhicules autonomes.

Les véhicules sont de plus en plus intelligents (reconnaissance de panneaux, détection de piétons, assistance à la conduite, etc), mais, pour se déplacer en totale autonomie, ils doivent être capables de se localiser précisément dans leur environnement. Cette localisation peut être absolue (localisation du véhicule en coordonnées GPS ou en coordonnées métriques par rapport à une carte connue) ou relative (localisation du véhicule par rapport à sa voie de circulation, position d'un drone par rapport au plan du sol, etc) en fonction des besoins. La précision de la localisation nécessaire au véhicule autonome n'est pas compatible avec celle des assistants GPS que nous connaissons et dont la précision avoisine quelques mètres. Il est possible d'améliorer cette précision en utilisant des systèmes plus évolués, tel qu'un GPS-RTK associé à une centrale inertielle, mais le coût de cet ensemble ne permet pas, pour le moment, de l'envisager pour le grand-public. Le GPS - et les systèmes similaires (GLONASS, Galileo, Beidou, *etc*) - est de plus sujet à des pertes de signal, ce qui n'est pas envisageable dans le contexte de la conduite automatisée.

Mes travaux de recherche sont concentrés sur cette problématique de la localisation pour le véhicule autonome. Ces travaux concernent le véhicule au sens large : voitures autonomes évidemment, mais aussi robots terrestres et drones. Pour se localiser, les véhicules autonomes s'appuient sur une batterie de capteurs : GPS, centrales inertielles, capteurs odométriques, caméras et lidars. Mes travaux reposent donc sur l'utilisation de toutes ces technologies, mais en particulier sur la vision par ordinateur et l'utilisation des lidars.

Ce mémoire, rédigé en vue de l'obtention de l'Habilitation à Diriger des Recherches, présente une synthèse de mes activités d'enseignement et de recherche menées à l'ESIGELEC et au sein de son Institut de Recherche en Systèmes Electroniques EMbarqués (IRSEEM, EA 4353), depuis l'obtention de mon doctorat de l'Université de Rouen en 2010. Il est organisé en 5 chapitres.

Le **Chapitre 1** résume l'ensemble de mon parcours en tant qu'enseignant au sein de l'ESIGELEC, et chercheur au sein du pôle Instrumentation, Informatique et Systèmes à l'IRSEEM. Dans ce chapitre, je présente mon parcours, mes activités d'enseignement et mes responsabilités pédagogiques, mes encadrements, mes responsabilités au sein de l'IRSEEM, ainsi qu'une liste complète de mes publications.

Les chapitres suivants décrivent mes contributions au travers de mes activités de recherche et de mes encadrements.

Le **Chapitre 2** est consacré à la localisation basée vision du véhicule. Il est organisé en trois parties. Dans un premier temps, nous proposons une méthode de *Structure-from-Motion* (SfM) à partir d'un réseau de caméras asynchrones. Nous montrons dans ces travaux qu'il est possible d'estimer les déplacements et de reconstruire l'environnement à l'échelle malgré l'utilisation de caméras non-synchronisées en posant quelques hypothèses sur le déplacement du véhicule. Ces travaux ont été menés dans le cadre de la thèse de Rawia Mhiri. Nous présentons dans un second temps nos travaux sur la localisation visuelle multimodale visible/infrarouge. L'objectif est d'être capable de se re-localiser grâce à une mémoire visuelle apprise au préalable, malgré les changements de caractéristiques du capteur de vision (champ de vue, résolution, bande spectrale) entre la construction de la mémoire visuelle et la phase de re-localisation. Ces travaux ont été menés dans le cadre de la thèse de Fabien Bonardi. Enfin, nous présentons dans ce chapitre nos travaux sur la parallélisation des algorithmes de localisation basée vision sur des architectures GPU. Nous proposons dans ces travaux des stratégies permettant d'améliorer le temps de calcul bien que les algorithmes de localisation soient intrinsèquement séquentiels. Ces travaux ont été réalisés dans le cadre de la thèse de Safa Ouerghi.

Le **Chapitre 3** est dédié à l'étude de solutions minimales pour l'odométrie visuelle et l'estimation de poses. Nous expliquons dans ce chapitre l'intérêt des solutions minimales, notamment lorsqu'elles sont intégrées dans des algorithmes d'estimation robuste de type RANSAC (*RANdom SAmple Consensus*). Nous présentons ensuite deux travaux sur cette thématique. Le premier concerne l'estimation du mouvement d'une caméra basée sur une homographie et en ayant la connaissance du vecteur de gravité. Ces travaux ont été réalisés dans le cadre d'une collaboration internationale (ETH Zürich, LITIS, IRSEEM, LE2I et TU Graz). Le second travail concerne l'estimation de l'attitude et de l'altitude d'un drone basé sur la projection d'un laser circulaire au sol. La projection de ce laser au sol est une conique, ce qui est un outil mathématique très intéressant en géométrie projective. Nous proposons différentes solutions pour l'estimation de la pose du drone, ainsi qu'une méthode de calibrage du système. Ces travaux ont été menés dans le cadre d'une collaboration nationale entre quatre laboratoires (INRIA, LITIS, LE2I, IRSEEM).

Le **Chapitre 4** présente nos contributions dans le domaine de la localisation lidar à 6 degrés de liberté (ddl). Nous avons proposé un algorithme de localisation robuste et temps-réel fonctionnant dans des environnements complexes. Ces travaux ont été menés dans le cadre du challenge international de robotique ARGOS et de la thèse de Pierre Merriaux. Ces travaux ont ensuite été étendus au véhicule autonome par un contrat de recherche et de développement pour le compte de Transdev dans le cadre du projet *Rouen Normandy Autonomous Lab*.

Le **Chapitre 5** conclut sur mes activités de recherche et ouvre des perspectives sur les axes de travail que je souhaite poursuivre.

Dans un souci de synthèse et de cohérence, il est important de noter que certains de mes travaux ne sont pas présentés dans ce manuscrit.

Chapitre 1

Synthèse de mes activités scientifiques, administratives et pédagogiques

Sommaire

1.1 Mon parcours . . . 13
1.1.1 Fonctions actuelles . . . 13
1.1.2 Activités professionnelles . . . 13
1.1.3 Formation . . . 14
1.2 Descriptions de mes activités d'enseignement et de mes responsabilités pédagoqiques . . . 14
1.2.1 Synthèse des enseignements dispensés . . . 14
1.2.2 La dominante ISE-VA . . . 15
1.2.3 Implication dans la pédagogie active . . . 16
1.3 Encadrements . . . 17
1.3.1 Co-encadrement de thèses (soutenues) . . . 17
1.3.2 Co-encadrement de thèses (en cours) . . . 18
1.3.3 Encadrement de stages de Master 2 . . . 18
1.3.4 Encadrement de stages de fin d'études . . . 18
1.3.5 Autres stages . . . 19
1.3.6 Ingénieurs de recherche et post-doctorants . . . 19
1.4 Responsabilités au sein de l'Institut de Recherche en Systèmes Électroniques Embarqués . . . 19
1.4.1 Contribution à la mise en place du Campus Intégration des Systèmes Embarqués . . . 19
1.4.2 Montage, coordination et participation à des projets . . . 20
1.4.3 Valorisation de la recherche par des contrats . . . 21
1.5 Rayonnement scientifique . . . 21
1.5.1 Participation à des comités de relecture . . . 21
1.5.2 Membre du comité de domaine de recherche 2 de l'institut VEDECOM 22
1.5.3 Participation à des GDRs . . . 22
1.5.4 Expertise de projets . . . 23

1.5.5 Formation professionnelle 23
1.5.6 Collaborations 23
1.6 Liste de mes publications 24
1.6.1 Articles dans des revues internationales 24
1.6.2 Communications avec actes dans un congrès international 25
1.6.3 Communications avec actes dans un congrès national 26
1.6.4 Chapitres de livre 27
1.6.5 Mémoires 28

Ce premier chapitre présente mes activités scientifiques, administratives et pédagogiques. Dans un premier temps, je résume mon parcours scolaire et professionnel. Je présente ensuite l'ensemble de mes activités depuis l'obtention de ma thèse en 2010.

1.1 Mon parcours

1.1.1 Fonctions actuelles

Je suis actuellement **Enseignant-Chercheur** à l'ESIGELEC (Ecole d'ingénieurs-es généralistes) et à l'IRSEEM (Institut de Recherche en Systèmes Electroniques EMbarqués, institut de recherche de l'ESIGELEC, EA 4353).

J'exerce mes activités pédagogiques au sein du département "Systèmes Embarqués et Instrumentation (SEI)" de l'ESIGELEC dans lequel je suis **Responsable de la dominante "Ingénierie des Systèmes Embarqués - Véhicule Autonome (ISE-VA)"**. Mes activités de recherche s'inscrivent dans le pôle de Recherche "Instrumentation, Informatique et Systèmes (IIS)" de l'IRSEEM.

1.1.2 Activités professionnelles

Depuis 2015	**Responsable de la dominante** "Ingénierie des Systèmes Embarqués - Véhicule Autonome (ISE-VA)".
Depuis 2009	**Enseignant-Chercheur** à l'ESIGELEC : enseignement, encadrement de thèses, coordination de projets, réalisation de contrats industriels.
2007-2009	**Vacataire** à l'ESIGELEC.
2006-2009	**Ingénieur de Recherche** dans le cadre d'une convention CIFRE - IPSIS (Ingénierie pour Signaux et Systèmes) - Groupe IT-Link.
Mars-Août 2006	**Projet de fin d'études / Stage de Master Recherche** – École des Mines de Douai (équipe Systèmes et Processus Industriels). - Travail dans le cadre des projets RaViOLi (Radar et Vision Orientables, Lidar) et AutoRIS (Automatique pour la Route Intelligente et Sûre). - Sujet : Étude de méthodes d'estimation du mouvement dans des séquences d'images prises par une caméra embarquée sur un véhicule dans le but de développer un système de compensation mécanique des mouvements.
Mai-Sept 2005	**Stage ingénieur adjoint** – IFTH – Villeneuve d'Ascq. - Mise au point de méthodes de mesures basées sur la vision 3D pour les tests et essais dans les secteurs textile et automobile. - Automatisation de mesures basées sur la vision 3D : développement d'une carte de commande 3 axes pour tourner l'objet à numériser, développement logiciel pour automatiser les traitements, etc.

- Développement d'un logiciel de commande d'une machine de numérisation 3D par imagerie confocale chromatique.

1.1.3 Formation

2010 **Doctorat** de l'Université de Rouen : "Reconstruction tridimensionnelle de l'environnement d'un robot mobile, à partir d'informations de vision omnidirectionnelle, pour la préparation d'interventions".

2006 **Master Recherche** en Automatique, Génie Informatique et Image (AG2I) - Option Signal, Image et Décision (SID) - Université des Sciences et Technologies de Lille.

2006 **Diplôme d'Ingénieur** de l'Ecole des Mines de Douai - Option Productique (Gestion Industrielle, Programmation, Microcontrôleurs, Robotique, Vision, Asservissement, Réseaux, DAO, CAO, etc.).

2002 **Classe Préparatoire aux Grandes Écoles** PTSI (Physique, Technologie et Sciences de l'Ingénieur) – Lycée Baggio de Lille

2001 **Baccalauréat S** (Mention Bien) – Spécialités Mathématiques et Sciences de l'Ingénieur – Lycée Camille Claudel de Fourmies

1.2 Descriptions de mes activités d'enseignement et de mes responsabilités pédagoqiques

1.2.1 Synthèse des enseignements dispensés

J'enseigne à l'ESIGELEC depuis 2007 : de 2007 à 2009 en tant que vacataire pendant ma thèse, puis en tant que permanent depuis 2009. Je présente dans cette section une synthèse des enseignements que j'ai effectués à l'ESIGELEC.

J'interviens principalement dans les enseignements du Département Systèmes Embarqués et Instrumentation (SEI) à hauteur de 250 heures équivalent TD par an. Les matières que j'enseigne portent principalement sur les architectures matérielles, sur la programmation en langage C pour les systèmes embarqués, sur la robotique mobile, ainsi que sur les algorithmes de traitement d'images et de vision par ordinateur. La figure 1.1 synthétise la répartition de ces enseignements en les regroupant en trois grandes familles : Vision et traitement d'images, Robotique et Systèmes Embarqués.

La figure 1.2 montre l'évolution de ma charge pédagogique en heure équivalent TD par an, ainsi que la répartition en CM/TD/TP. Les intitulés des matières que j'enseigne, ainsi que le type d'enseignement et le public concerné sont donnés ci-dessous. Les matières pour lesquelles je suis (ou j'ai été) responsable du module d'enseignement sont indiquées en caractères gras :

- **Vision par ordinateur**, Ingénieur 3ème année en dominante ISE-VA, (CM, TD, TP),
- **Fondamentaux en Vision et Traitement d'images**, Ingénieur 2ème année en dominante ISYMED, (CM, TD, TP),
- **Robotique Mobile et Perception**, Ingénieur 3ème année en tronc commun, (CM, TP),

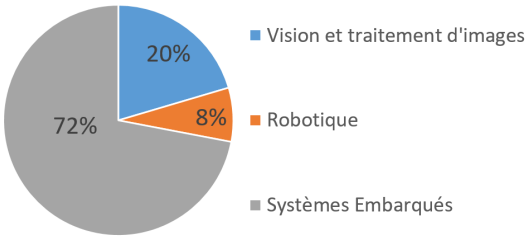


FIGURE 1.1 – Répartition des enseignements par matière.

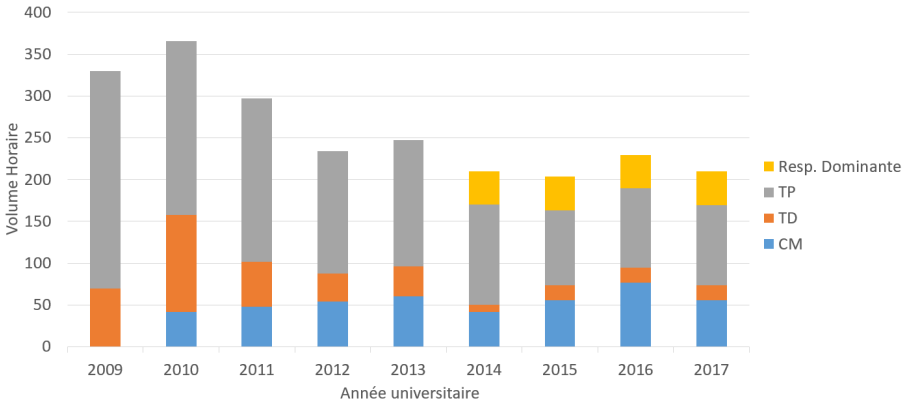


FIGURE 1.2 – Evolution de ma charge pédagogique en heures équivalent TD par an.

— **Introduction to C programming**, Master International SEE, (CM, TP),
— **Interfaçage des Systèmes Numériques**, Ingénieur 2ème année en tronc commun, (CM, TP).
— Systèmes à microprocesseurs, Ingénieur 2ème année en tronc commun, (TD, TP, APP),
— Instrumentation et Systèmes, Ingénieur 2ème année en tronc commun, (CM, TP),
— Architecture des ordinateurs, Ingénieur 1ème année en tronc commun, (TD),
— Microprocessors, Master International SEE, (CM, TP),
— Application à microprocesseurs, Ingénieur 3ème année en dominante ISE, (CM, TP),
— Découverte des Systèmes Embarqués et Instrumentation, Ingénieur 1ème année en tronc commun, (TP),
— Systèmes Embarqués, Ingénieur 3ème année en dominante ISE, (CM, TP),
— Pilotage d'actionneurs, Ingénieur 2ème année en dominante ISE et MCTGE, (CM, TP).

A noter également que depuis 2009, j'interviens dans les masters internationaux SEE (Systèmes Électroniques Embarqués) et CEI (Control and Embedded Instrumentation), où les cours se déroulent totalement en anglais.

1.2.2 La dominante ISE-VA

Afin de renforcer l'adéquation entre les travaux de recherche menés à l'IRSEEM, et l'enseignement dispensé à l'ESIGELEC, j'ai été chargé en 2015 par la direction des Etudes de structurer une nouvelle dominante liée au Véhicule Autonome. Je suis donc responsable, depuis le 1er mars 2015, de la dominante Ingénierie des Systèmes Embarqués – Véhicule Autonome

(ISE-VA). Comme il s'agit d'une nouvelle dominante proposée à l'ESIGELEC, j'ai eu l'opportunité de définir entièrement le contenu pédagogique.

Mes principales missions pour cette responsabilité sont donc :

- de faire évoluer le programme de la dominante de façon cohérente, équilibrée et attractive, en rapport avec les recommandations du comité de pilotage et avec les exigences pédagogiques (enseignement bilingue, référentiel de compétences, etc),
- de rechercher et de contacter les intervenants (industriels et académiques), en France et à l'étranger,
- de planifier les enseignements, de répondre aux demandes du service de la scolarité et des relations internationales,
- de faire un suivi des enseignements dispensés, d'être attentif aux attentes des étudiants dans un souci d'amélioration continue,
- d'organiser un comité de pilotage régulièrement, de faire de la veille sur l'état du marché de l'emploi et les évolutions technologiques,
- de proposer des axes d'amélioration en cohérence avec les enseignements du département SEI et avec ses orientations,
- de communiquer sur la dominante auprès des étudiants.

1.2.3 Implication dans la pédagogie active

La pédagogie active a été introduite au début des années 2000 à l'ESIGELEC et j'ai pu y apporter ma contribution à plusieurs titres.

Les projets ingénieurs sont une des déclinaisons de la pédagogie active à l'ESIGELEC. Constitués en équipe des six, les étudiants de deuxième et troisième années doivent mener un projet complet, de l'étude à la réalisation d'une maquette, et ce pour le compte d'un commanditaire, en général un industriel. Je suis intervenu dans ce dispositif :

- en tant que binôme d'encadrement avec un rôle de contrôle et d'accompagnement sur le plan du management de projet (j'ai suivi en moyenne deux à trois équipes chaque année) ;
- en tant qu'instructeur chargé du suivi technique de projets (deux à trois équipes par an) ;
- en tant que commanditaire d'études et de réalisations dans le cadre des activités de recherche du pôle instrumentation, informatique et systèmes de l'IRSEEM.

J'interviens également dans des dispositifs similaires au sein des Masters Internationaux SEE et CEI. Chaque année, je propose en moyenne deux sujets et assure le suivi des équipes travaillant sur ces sujets.

A partir de 2012, l'ESIGELEC a mis en place une nouvelle forme de pédagogie active : l'Apprentissage Par Problèmes (APP), où l'étudiant est au cœur du dispositif pédagogique. L'objectif de cette réforme était de mettre les étudiants en situation professionnelle en développant leur autonomie, leur esprit d'initiative, et leur capacité de travail en équipe. Les activités sont organisées en séances de travail en groupe tutorées et en activités individuelles, le tout couplé à des séances de cours de restructuration dont l'objectif est de consolider les connaissances acquises et de clarifier les points difficiles. Deux départements de l'Ecole ont été pilotes de cette pédagogie innovante : le département TIC (Technologies de l'Information et de la Communication) et le département SEI (Systèmes Embarqués et Instrumentation) auquel je suis rattaché. Dans ce contexte, nous avons donc décidé d'enseigner le module de "Systèmes à Microprocesseurs", module de tronc commun de deuxième année, en utilisant ce type de pédagogie. J'ai

particulièrement été impliqué dans cette réforme : j'ai été force de proposition sur le choix d'une nouvelle plateforme de développement, j'ai défini les problèmes que les étudiants devaient traiter, et j'ai rédigé une partie des livrets pédagogiques pour ce module.

1.3 Encadrements

Depuis ma prise de fonction en 2009, j'ai encadré 8 thèses de doctorat (dont 4 soutenues), 5 stages de Master 2 Recherche ou projets de fin d'études, 5 post-doctorants et ingénieurs de recherche, et de nombreux autres stages. Ces encadrements sont détaillés ci-dessous.

1.3.1 Co-encadrement de thèses (soutenues)

Nom du doctorant (Date de soutenance)	**Titre de la thèse**	**Directeur(s) de thèse**	**Ecole doctorale**	**Taux encadrement**
[Th1] Rawia Mhiri (Décembre 2015)	Approches 2D-2D pour le SfM à partir d'un réseau de caméras asynchrones	P. Vasseur et A. Bensrhair	SPMII (ED 351)	30 %
[Th2] Pierre Merriaux (Octobre 2016)	Contribution à la localisation robuste embarquée pour la navigation autonome	P. Vasseur et X. Savatier	SPMII (ED 351)	30 %
[Th3] Enjie Ghorbel (Octobre 2017)	Fast and accurate human action recognition using RGB-D cameras	X. Savatier et S. Lecoeuche	MIIS (ED 590)	30 %

J'ai également été impliqué dans l'encadrement de la thèse de Fabien Bonardi [Th4] dans le cadre du projet Savemore. Cette thèse, dirigée par Pascal Vasseur, a été soutenue en novembre 2017 et porte sur la localisation visuelle multimodale visible/infrarouge pour la navigation autonome. Comme évoqué dans l'introduction, dans un souci de cohérence, les travaux effectués dans le cadre de la thèse de Enjie Ghorbel ne sont pas présentés dans ce mémoire.

1.3.2 Co-encadrement de thèses (en cours)

Nom du doctorant (Année de soutenance prévue)	Titre de la thèse	Directeur(s) de thèse	Ecole doctorale	Taux encadrement
[Th5] Safa Ouerghi (2018)	Embedded Architecture for Vision-based Mobile Localization	F. Tlili et X. Savatier	EDTIC (Tunisie)	30 %
[Th6] Louis Le Crosnier (2019)	Perception de scènes multimodales et coopérative drone-véhicule terrestre	P. Vasseur et X. Savatier	MIIS (ED 590)	30 %
[Th7] Oumaima Ait-Boutargante (2020)	Contribution à la vision plénoptique pour l'odométrie visuelle d'un robot mobile	X. Savatier	MIIS (ED 590)	30 %
[Th8] Mickaël Delamare (2021)	Localisation précise en intérieur et capture simultanée du geste d'une personne par fusion d'une localisation Ultra Wide Band (UWB) et d'une mesure de mouvement par capteurs inertiels et vision	P. Descamps	MIIS (ED 590)	30 %

1.3.3 Encadrement de stages de Master 2

Xingyi ZHOU (Stage M2 IGIS-STIM 2016) : Développement d'algorithmes de localisation par vision dans des environnements complexes.

Harold DESERT-LEGENDRE (Stage M2 IGIS-STIM 2012) : Étude et évaluation d'une bibliothèque de SLAM monoculaire.

Mahmoud HAKIM (Stage M2 IGIS-STIM 2012) : Planification de trajectoires de navigation pour une flotte de robots mobiles.

1.3.4 Encadrement de stages de fin d'études

Zhengfeng DING (2018), Projet de fin d'Etudes de Polytech Grenoble : Développement d'algorithmes de SLAM Fisheye.

Imen BEN SALAH (2015), Projet de fin d'Etudes de l'Ecole Nationale d'Ingénieurs de Sfax : Localisation absolue d'un véhicule en utilisant des données visuelles.

Dejie SUN (2013), Projet de fin d'Etudes ESIGELEC : Filtrage particulaire par vision pour la localisation de robots mobiles.

1.3.5 Autres stages

Benoit FAURE (Stage Technicien CESI, 2017) : Développement d'algorithmes de Vision par Ordinateur pour la reconstruction 3D et l'odométrie visuelle.

Cyril ROUYER (Stage Initiation à la recherche CESI, 2017) : Veille technologique sur les systèmes SONAR pour la bathymétrie.

Tony Jean Jacques KAFANDO (Stage technicien ESIGELEC, 2015) : Développement d'une carte de synchronisation de caméras.

René Emmanuel DATONDJI (Stage technicien ESIGELEC, 2013) : Développement d'une librairie de fonctions pour le traitement d'images par la vision.

1.3.6 Ingénieurs de recherche et post-doctorants

Lei QIN (Post-Doctorant, 2016-2017) : Application de la vision omnidirectionnelle à la localisation d'un ROV.

Pailin CHANUC (Ingénieur de Recherche, 2016-2017) : Ingénieur de recherche travaillant sur le projet VIATIC².

Yassine NASRI (Ingénieur de Recherche, 2015-2016) : Ingénieur de recherche travaillant sur le projet VIATIC².

Fengchun DONG (Post-Doctorant, 2013-2014) : Optimisation de la navigation d'une flotte de robots mobiles pour l'interception de cibles.

Assia BELBACHIR (Post-Doctorant, 2012-2013) : Implémentation d'algorithmes pour la navigation autonome.

1.4 Responsabilités au sein de l'Institut de Recherche en Systèmes Électroniques Embarqués

1.4.1 Contribution à la mise en place du Campus Intégration des Systèmes Embarqués

L'IRSEEM a connu une forte croissance avec la création du Campus Intégration des Systèmes Embarqués (CISE) en 2012. Il s'agit d'une extension de 4700 m^2 pour les activités de recherches avec 4 plate-formes : navigation autonome, compatibilité électromagnétique, essais moteurs hybrides, et nacelles (dispositif d'accueil des réacteurs d'avion). Dans ce contexte, j'ai participé à la définition des équipements du Laboratoire de Navigation Autonome, à la rédaction des appels d'offres, à l'analyse des réponses aux appels d'offres puis à la mise en place des équipements.

Les principaux équipements sont un système de capture du mouvement, un laser 3D permettant de numériser un environnement en 3D avec une portée de 300m, un système de trajectographie embarquable sur un véhicule (Centrale Inertielle, GPS-RTK, Velodyne, etc), des robots mobiles d'intérieur et d'extérieur, un atelier mécanique de prototypage, etc.

1.4.2 Montage, coordination et participation à des projets

Durant ces années, j'ai participé, monté et coordonné de nombreux projets internationaux ou nationaux. Je détaille ci-dessous les différents projets dans lesquels j'ai été impliqué, ainsi que le rôle que j'ai eu au sein de ces projets.

Nom du projet	**Dispositif de financement (Budget IR-SEEM)**	**Partenaires**	**Rôle dans le projet**	**Dates Début/Fin**
LOCADYN : LOCAlisation et sécurisation DYNamique des flux de personnes, d'objets et d'information sur les sites industriels	Région/RIN Recherche/Pôle Sciences du Numérique (194 k€)	IRSEEM, CESI	**Chef de projet IRSEEM**	2017-2019
COPTER : COoPération drone véhicule TERrestre	FEDER + Région/GRR EEM (403 k€)	IRSEEM, CEREMA, LITIS	Participation scientifique	2016-2020
VIATIC^2 : VIabilité et AuTonomie des systèmes en environnement Incertain et Contraint 2	ANR Astrid Maturation (291 k€)	IRSEEM, MBDA, VIMADES	**Coordinateur de l'ANR**	2014-2018
ARGOS : Autonomous Robot for Gas and Oil Sites	Challenge Total/ANR (500k€)	IRSEEM, Sominex	**Responsable de la partie vision**	2014-2017
NAVALIS	ADEME (522k€)	IRSEEM, MSI Chantiers Allais	Participation scientifique	2012-2015
VIATIC : VIabilité et AuTonomie des systèmes en environnement Incertain et Contraint	ANR Astrid (193 k€)	IRSEEM, EHESS, MBDA	**Chef de Projet IRSEEM**	2011-2014
SAVEMORE : Smart Autonomous VEhicle for urban MObility using Renewable Energy	Interreg IV + GRR EEM (253 k€)	IRSEEM, LITIS, Université du Kent	Participation scientifique	2013-2015

PLUTON : PLateforme Unifiée pour le Test des systèmes de visiON)	FEDER + Région/GRR EEM (138 k€)	IRSEEM, LITIS	Participation scientifique	2010-2014
MIRIADE : MIlitarisation de la RestItution d'Attitude Déduite des Etoiles	DGA Rapid (102k€)	IRSEEM, STARNAV, SOMINEX	**Chef de projet IRSEEM**	2009-2011

1.4.3 Valorisation de la recherche par des contrats

L'IRSEEM est un laboratoire ayant de fortes interactions avec le tissu économique local et national. Dans ce cadre, je suis régulièrement amené à répondre à des contrats directs avec des industriels. Le tableau ci-dessous résume mes activités contractuelles avec des industriels.

Industriel	**Descriptif du projet**	**Rôle dans le projet**	**Année (Durée)**
PSA	Développement d'un outil de référence pour les capteurs ADAS	Définition des algorithmes et encadrement d'un ingénieur de recherche	2016 (5 mois)
AREVA	Développement d'un système de localisation d'un ROV à partir de vision fisheye	Responsable technique du projet, définition et développement des algorithmes, encadrement d'un ingénieur de recherche, rédaction des documents	2016 (10 mois)
TRANSDEV	Développement d'un système de localisation basé Lidar	Expertise technique	2017 (3 ans)

1.5 Rayonnement scientifique

1.5.1 Participation à des comités de relecture

Comités de relecture dans des revues

J'ai effectué des relectures d'articles pour les revues suivantes :

Robotica	
IJARS	International Journal of Advanced Robotic Systems
JMIV	Journal of Mathematical Imaging and Vision
IET Computer Vision	
IEEE RA-L	IEEE Robotics and Automation Letters
PLOS ONE	

Comités de relecture dans des conférences

J'ai été relecteur pour les congrès suivants :

ITSC	IEEE Conference on Intelligent Transportation Systems	2014, 2015, 2018
IROS	IEEE/RSJ International Conference on Intelligent Robots and Systems	2014, 2015, 2016, 2017
ICRA	IEEE International Conference on Robotics and Automation	2011, 2016, 2017, 2018
ICIP	IEEE International Conference on Image Processing	2014, 2015
ICARCV	International Conference on Control, Automation, Robotics and Vision	2014
IV	IEEE Intelligent Vehicles Symposium	2015, 2017, 2018
CASE	IEEE International Conference on Automation Science and Engineering	2017

1.5.2 Membre du comité de domaine de recherche 2 de l'institut VEDECOM

Je suis représentant de l'ESIGELEC au comité de domaine 2 (Délégation de Conduite et Connectivité) de l'institut VEDECOM depuis janvier 2017.

Créé en février 2014, VEDECOM est un Institut pour la Transition Energétique (ITE) mis en place dans le cadre du Programme d'Investissements d'Avenir (PIA) du Gouvernement Français, dédié à la mobilité individuelle, décarbonnée et durable. Il appartient à ce titre au "Plan Véhicule Autonome" de la Nouvelle France Industrielle (NFI). L'ESIGELEC est membre fondateur de cet institut.

VEDECOM est une fondation partenariale de près de 40 membres regroupés sur une collaboration inédite entre industriels de la filière automobile, aéronautique, opérateurs d'infrastructures et de services de l'écosystème de la mobilité, établissements de recherche académiques et collectivités locales d'Ile-de-France.

VEDECOM répond aux enjeux du véhicule autonome et des mobilités de demain. VEDECOM a pour ambition de devenir l'institut de référence de l'automobile autonome, connectée et de ses usages et, à terme, de créer de nouveaux standards visant à faire évoluer le cadre réglementaire et normatif aux niveaux national et européen.

Les missions du Comité de Domaine 2 auquel je participe sont :
— décliner la feuille de route VEDECOM au niveau des activités du domaine 2,
— partager des priorités sur les livrables par les membres du CD 2 (industriels et académique),
— veiller à la cohérence feuille de la feuille de route du CD2 de VEDECOM avec les axes de recherches du plan NFI et des travaux de la PFA (Filière Automobile et mobilités),
— émettre des avis consultatifs à destination du bureau VEDECOM sur l'engagement, le suivi et l'arbitrage des projets.

1.5.3 Participation à des GDRs

Je participe régulièrement aux journées du GDR ISIS et du GDR Robotique.

1.5.4 Expertise de projets

J'ai été sollicité pour expertiser des projets de recherche :
— expertise d'un projet ANR à l'appel à projets générique 2017 (ANR-AAPG2017),
— expertise d'un projet à l'appel à projets 2017 de la région Normandie au Pôle Stratégique de Recherche et de Formation Sciences du Numérique (PSRF-SN 2017).

1.5.5 Formation professionnelle

J'interviens en tant que formateur chez Renault dans le cadre d'une formation intitulée "Les enjeux techniques du véhicule autonome". J'anime au sein de cette formation une session sur la vision par ordinateur pour le véhicule autonome.

1.5.6 Collaborations

Au niveau international, je collabore ou ai collaboré avec :

— Ecole Supérieure des Télécommunications de Tunis (SUP'COM), Tunisie, avec Fethi Tlili. Cette collaboration s'est inscrite dans le cadre du co-encadrement de la thèse de Safa Ouerghi [Th5]. Je collabore également avec Adel Ghazel, par des échanges pour l'enseignement, et des travaux dans le cadre de projets de recherche.
— Graz University of Technology, Autriche, avec Friedrich Fraundorfer. Nous avons collaboré sur certains travaux [2] et dans le cadre de la thèse de Louis Le Crosnier [Th6].
— ETH Zurich, Suisse, avec Marc Pollefeys et Olivier Saurer. Nous avons collaboré dans le cadre des travaux publiés dans [2].

Au niveau national, je collabore ou ai collaboré avec :

— LITIS (Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes), avec Pascal Vasseur. Nous avons collaboré sur de nombreux travaux de recherche et dans le cadre de l'encadrement de plusieurs thèses. J'ai également collaboré avec Stéphane Mousset, Abdelaziz Bensrhair et Samia Ainouz, dans le cadre de l'encadrement des thèses de Rawia Mhiri [Th1], Fabien Bonardi [Th4] et Louis Le Crosnier [Th6].
— INRIA Grenoble Rhône-Alpes, avec Peter Sturm, dans le cadre des travaux de recherche sur l'estimation d'attitude et d'altitude d'un drone [3].
— Le2i, avec Cédric Demonceaux, dans le cadre des travaux de recherche sur l'estimation d'attitude et d'altitude d'un drone [3] et sur l'odométrie visuelle [2].
— CEREMA (Centre d'Etudes et d'expertise sur les Risques, l'Environnement, la Mobilité et l'Aménagement), avec Yohan Dupuis, dans le cadre de l'encadrement de la thèse de Pierre Merriaux [Th2] et de Fabien Bonardi [Th4], ainsi que dans le cadre du projet COPTER.
— Institut Mines-Télécom Lille-Douai, avec Jacques Boonaert et Stéphane Lecoeuche, dans le cadre de l'encadrement de la thèse d'Enjie Ghorbel [Th3].
— CESI, avec David Baudry, Anne Louis, Fabrice Duval, Sahbi Baccar, M'hammed Sahnoun et Vincent Havard, dans le cadre d'échanges scientifiques et de projets de recherche (LOCADYN).
— ENSTA ParisTech, Unité de Mathématiques Appliquées, avec Anya Désilles dans le cadre des projets VIATIC et VIATIC^2.

— LASTRE (Laboratoire d'Applications des Systèmes Thychastiques Régulés), avec Jean-Pierre Aubin et Patrick Saint-Pierre dans le cadre des projets VIATIC et VIATIC².
— IFSTTAR (Institut Français des Sciences et Technologies des Transports, de l'Aménagement et des Réseaux), avec Jean-Marc Blosseville, dans le cadre de travaux [4] et des formations professionnelles chez Renault.

1.6 Liste de mes publications

1.6.1 Articles dans des revues internationales

[R10] Merriaux, P., Rossi, R., **Boutteau, R.**, Vauchey, V., Qin, L., Chanuc, P., Rigaud, F., Roger, F., Decoux, B., Savatier, X., "VIKINGS autonomous inspection robot for the ARGOS challenge", IEEE Robotics and Automation Magazine (IEEE-RAM), Special Issue on Robots for in-situ Interventions, 2018 (JCR IF 3.573).

[R09] Ghorbel, E., Boonaert, J., **Boutteau, R.**, Lecoeuche, S., Savatier, X., "An extension of kernel learning methods using a modified Log-Euclidean distance for fast and accurate skeleton-based Human Action Recognition", Computer Vision and Image Understanding (CVIU), 2018 (JCR IF 2.391).

[R08] Ghorbel, E., **Boutteau, R.**, Boonaert, J., Savatier, X., Lecoeuche, S., "Kinematic Spline Curves : A temporal invariant descriptor for fast action recognition", Image and Vision Computing (IVC), vol. 77, pp 60-71, 2018 (JCR IF 2.671).

[R07] Ouerghi, S., **Boutteau, R.**, Tlili, F., Savatier, X., "Visual Odometry and Place Recognition Fusion for Vehicle Position Tracking in Urban Environments", Sensors, 2018, 18, 939 (JCR IF 2.677).

[R06] **Boutteau, R.**, Sturm, P., Vasseur, P., Demonceaux, C., "Circular Laser/Camera-based attitude and altitude estimation : minimal and robust solutions", Journal of Mathematical Imaging and Vision (JMIV), vol. 60, Issue 3, pp 382-400, 2018, (JCR IF 1.927).

[R05] Merriaux, P., Dupuis, Y., **Boutteau, R.**, Vasseur, P., Savatier, X., "A Study of Vicon System Positioning Performance", Sensors, 17, 1591, 2017, (JCR IF 2.677).

[R04] Merriaux, P., Dupuis, Y., **Boutteau, R.**, Vasseur, P., Savatier, X., "Robust Robot Localization in a Complex Oil and Gas Industrial Environment", Journal of Field Robotics (JFR), 2017, (JCR IF 4.882).

[R03] Bonardi, F. , Ainouz, S., **Boutteau, R.**, Dupuis, Y., Savatier, X., Vasseur, P., "PHROG : A multimodal Feature for Place Recognition", Sensors, 17, 1167, 2017, (JCR IF 2.677).

[R02] Saurer, O., Vasseur, P., **Boutteau, R.**, Demonceaux, C., Pollefeys, M., Fraundorfer, F., "Homography Based Egomotion Estimation with a Common Direction", IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), February 2017, vol. 39, Issue 2, pp 327-341, 2017, (JCR IF 9.455).

[R01] **Boutteau, R.**, Savatier, X., Ertaud, J.Y., Mazari, B., "An Omnidirectional Stereoscopic System for Mobile Robot Navigation", Sensors and Transducers Journal, Special Issue on Robotic and Sensors Environments, vol. 5, pp 3-17, 2009.

1.6.2 Communications avec actes dans un congrès international

[C17] Bonardi, F. , Ainouz, S., **Boutteau, R.**, Dupuis, Y., Savatier, X., Vasseur, P., "A novel Global Image Description approach for Long Term Vehicle Localization", European Signal Processing Conference (EUSIPCO), Kos Island, Greece, 2017.

[C16] Ouerghi, S., **Boutteau, R.**, Tlili, F., Savatier, X., "CUDA-based SeqSLAM for Real Time Place Recognition", International Conference on Computer Graphics, Visualization and Computer Vision (WSCG), Plzen, Czech Republic, 2017.

[C15] Ouerghi, S., **Boutteau, R.**, Savatier, X., Tlili, F., "CUDA Accelerated Visual Egomotion Estimation for Robotic Navigation", International Conference on Computer Vision Theory and Applications (VISAPP), Porto, Portugal, 2017.

[C14] Ghorbel, E., **Boutteau, R.**, Boonaert, J., Savatier, X., Lecoeuche, S., "A Fast and Accurate Motion Descriptor for Human Action Recognition Applications", International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 2016.

[C13] Ouerghi, S., **Boutteau, R.**, Merriaux, P., Ragot, N., Savatier, X., Vasseur, P., "Absolute Localization Using Visual Data for Autonomous Vehicles", International Conference on Computer Vision Theory and Applications (VISAPP), Rome, Italy, 2016.

[C12] Ghorbel, E., **Boutteau, R.**, Boonaert, J., Savatier, X., Lecoeuche, S., "3D real time human action recognition using a spline interpolation approach", International Conference on Image Processing Theory, Tools and Applications (IPTA), Orléans, France, 2015.

[C11] Mhiri, R., Vasseur, P., Mousset, S., **Boutteau, R.**, Bensrhair, A., " Accurate Scale estimation based on unsynchronized camera network", IEEE International Conference on Image Processing (ICIP), Québec City, Canada, 2015.

[C10] Merriaux, P., **Boutteau, R.**, Vasseur, P., Savatier, X., "IMU LIDAR based positioning of a gangway for maintenance operations on wind farms", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, Illinois, USA, 2014.

[C09] Dupuis, Y., Merriaux, P., Subirats, P., **Boutteau, R.**, Savatier, X., Vasseur, P., "GPS-based Preliminary Map Estimation for Autonomous Vehicle Mission Preparation", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)., Chicago, Illinois, USA, 2014.

[C08] Mhiri, R., Vasseur, P., Mousset, S., **Boutteau, R.**, Bensrhair, A., "Visual odometry with unsynchronized multi-cameras setup for intelligent vehicle application", IEEE Intelligent Vehicles Symposium (IV), Dearborn, Michigan, USA, 2014.

[C07] Le Ménec, S., Lecointre, V., **Boutteau, R.**, Savatier, X., Jaulin, L., "Robotic Demonstration of Collision Avoidance Based on Differential Games", 16th International Symposium on Dynamic Games and Applications, Amsterdam, 2014.

[C06] **Boutteau, R.**, Savatier, X., Ertaud, J.Y., "A dynamic programming algorithm applied to omnidirectional vision for dense 3D reconstruction", IAPR Asian Conference on Pattern Recognition (ACPR), Naha, Okinawa, Japan, 2013.

[C05] **Boutteau, R.**, Savatier, X., Bonardi, F., Ertaud, J.Y., "Road- line detection and 3D reconstruction using fisheye cameras", IEEE International Conference on Intelligent Transportation Systems (ITSC), The Hague, The Netherlands, 2013.

[C04] Belbachir, A., **Boutteau, R.**, Merriaux, P., Blosseville, J.M., Savatier, X., "From Autonomous Robotics Toward Autonomous Cars", IEEE Intelligent Vehicles Symposium (IV), Gold Coast, Australia, 2013.

[C03] Savatier, X., **Boutteau, R.**, Merriaux, P., Le Ménec, S., "Demonstration of the Salvo Enhanced No Escape Zone Concept using ground mobile robots", Fifteenth International Symposium on Dynamic Games and Applications (SDG), Bysice, Czech Republic, 2012.

[C02] Iraqui, A., Dupuis, Y., **Boutteau, R.**, Ertaud, J. Y., Savatier, X., "Fusion of Omnidirectional and PTZ cameras for face detection and tracking", International Conference on Emerging Security Technologies, Canterbury, United Kingdom, 2010.

[C01] **Boutteau, R.**, Savatier, X., Ertaud, J.-Y., Mazari, B., "An omnidirectional stereoscopic system for mobile robot navigation", IEEE International Workshop on RObotic and Sensors Environments (ROSE), Ottawa, Canada, 2008.

1.6.3 Communications avec actes dans un congrès national

[N09] Merriaux, P., **Boutteau, R.**, Rossi, R., Coru, G., Vauchey, V., Savatier, X., "Synchronisation et calibrage entre un Lidar 3D et une centrale inertielle pour la localisation précise d'un véhicule autonome", Journées scientifiques d'URSI-France (URSI), Meudon, France, 2018.

[N08] Bonardi, F. , Ainouz, S., **Boutteau, R.**, Dupuis, Y., Savatier, X., Vasseur, P., "Localisation visuelle multimodale à long terme", Colloque GRETSI (GRETSI), Juan-Les-Pins, France, 2017.

[N07] Merriaux, P., Dupuis, Y., **Boutteau, R.**, Vasseur, P., Savatier, X., "Correction de nuages de points lidar embarqué sur véhicule pour la reconstruction d'environnement 3D vaste", Reconnaissance de Formes et Intelligence Artificielle (RFIA), Clermont Ferrand, France, 2016.

[N06] Ghorbel, E., **Boutteau, R.**, Boonaert J., Savatier, X., Lecoeuche, S., "Vers une reconnaissance en ligne d'actions à partir de caméras RGB-D", Reconnaissance de Formes et Intelligence Artificielle (RFIA), Clermont Ferrand, France, 2016.

[N05] Merriaux, P., Dupuis, Y., **Boutteau, R.**, Vasseur, P., Savatier, X., "Localisation robuste en milieu industriel complexe", Colloque GRETSI (GRETSI), Lyon, France, 2015.

[N04] Mhiri, R., Vasseur, P., Mousset, S., **Boutteau, R.**, Bensrhair, A., "Estimation du mouvement et de la structure à l'échelle absolue à partir d'un réseau multi-caméras non synchronisées", Journées francophones des jeunes chercheurs en vision par ordinateur (ORASIS), Amiens, France, 2015.

[N03] Datondji, R., Ragot, N., Nasri, Y., Khemmar, R., **Boutteau, R.**, "Odométrie visuelle par vision omnidirectionnelle pour la navigation autonome d'une chaise roulante motorisée", Journées francophones des jeunes chercheurs en vision par ordinateur (ORASIS), Amiens, France, 2015.

[N02] Mhiri, R., Vasseur, P., Mousset, S., **Boutteau, R.**, Bensrhair, A., "Estimation à l'échelle du mouvement d'un réseau multi-caméras non synchronisées", Reconnaissance de Formes et Intelligence Artificielle (RFIA), Rouen, France, 2014.

[N01] Merriaux, P., **Boutteau, R.**, Vasseur, P., Savatier, X., "Algorithme de positionnement d'une passerelle à mouvements compensés à partir de mesures inertielles et lidar pour les opérations de maintenance des parcs éoliens offshore", Reconnaissance de Formes et Intelligence Artificielle (RFIA), Rouen, France, 2014.

1.6.4 Chapitres de livre

[L02] Ouerghi, S., **Boutteau, R.**, Savatier, X., Tlili, F. "CUDA-accelerated Feature-based Egomotion Estimation", Communications in Computer and Information Science, VISIGRAPP 2017 - Extended and revised papers, Springer, 2018.

[L01] **Boutteau, R.**, Savatier, X., Ertaud, J. Y., Mazari, B., "Chapter 1 : A 3D Omnidirectional Sensor For Mobile Robot Applications", Mobile Robots Navigation, In-Tech Book, ISBN : 978-953-307-076-6, 2010.

1.6.5 Mémoires

[M03] **Boutteau**, R., "Reconstruction tridimensionnelle de l'environnement d'un robot mobile, à partir d'informations de vision omnidirectionnelle, pour la préparation d'interventions", Thèse de Doctorat de l'Université de Rouen, 2010.

[M02] **Boutteau, R.**, "Étude des méthodes d'estimation du mouvement dans des séquences d'images prises par une caméra embarquée sur un véhicule", Mémoire de Master 2 Recherche, Université des Sciences et Technologies de Lille (USTL), 2006.

[M01] **Boutteau, R.**, "Étude des méthodes d'estimation du mouvement dans des séquences d'images prises par une caméra embarquée sur un véhicule", Mémoire de projet de fin d'études de l'Ecole des Mines de Douai, 2006

Chapitre 2

Localisation basée vision du véhicule

Sommaire

2.1 Introduction et contexte 30
2.2 Structure-from-Motion à partir d'un réseau de caméras asynchrone 31
2.2.1 Etat de l'art 32
2.2.2 La méthode des triangles 34
2.2.3 Optimisation par ajustement de faisceaux 38
2.2.4 Résultats expérimentaux 41
2.3 Localisation visuelle multimodale visible/infrarouge 48
2.3.1 État de l'art 48
2.3.2 Méthodologie 51
2.3.3 Application de PHROG au problème de *place recognition* 58
2.4 Architectures embarquées pour la localisation basée vision 65
2.4.1 Estimation du mouvement relatif basé CUDA à partir de correspondances 2D-2D 66
2.4.2 Implémentation d'un algorithme de *place recognition* sur GPU 70
2.5 Conclusion 73

2.1 Introduction et contexte

Ces dernières années, les systèmes d'aide à la conduite (ADAS), initialement réservés aux véhicules de luxe sont devenus disponibles sur les modèles destinés au grand-public. Ces véhicules sont maintenant équipés de série avec de nombreuses fonctions d'aide à la conduite telles que l'alerte de franchissement de ligne [5], la surveillance des angles morts [6], la reconnaissance de panneaux [7] [8], le parking automatique ou la conduite autonome dans les embouteillages [9]. Dans ce contexte de développement des ADAS, la caméra est devenue l'un des capteurs les plus couramment utilisés car en plus de son faible coût par rapport à d'autres capteurs tels que les lidars, la caméra est un capteur polyvalent qui permet de développer de nouvelles fonctions sur une architecture déjà existante. Dans le futur, les ADAS offriront de plus en plus d'assistances automatisées et le développement des véhicules totalement autonomes impliqueront une perception complète et robuste de l'environnement autour du véhicule, tout en maintenant des coûts abordables pour le secteur automobile.

Parmi les différentes fonctions nécessaires au véhicule partiellement ou totalement automatisé, la localisation tient une place fondamentale. Elle est généralement basée sur des capteurs performants mais onéreux (GPS-RTK, IMU, lidars, etc) et qui ne sont pas infaillibles (pertes de signal, performances moindres dans des conditions climatiques sévères, etc). De nombreux travaux ont donc été menés sur la localisation basée vision pour le véhicule autonome, soit pour remplacer les autres capteurs, soit pour venir en complément par fusion de données.

Nous pouvons distinguer plusieurs types de localisation. Le premier niveau de localisation est l'estimation du mouvement propre du véhicule, on parle alors d'*ego-motion*. Ce problème fondamental du véhicule autonome est généralement réalisé avec des encodeurs qui mesurent directement la rotation des roues et permettent de retrouver la trajectoire du véhicule par intégration dans le temps. Ces méthodes sont imprécises lorsque la roue glisse, notamment sur des terrains accidentés ou mouillés. L'odométrie visuelle ou l'odométrie basée sur le lidar, qui estiment le mouvement à partir d'images ou de rayons lasers, sont devenus très populaires car ils sont moins sensibles à ces conditions et ils peuvent corriger les erreurs lorsque des lieux précédemment visités sont reconnus, par une méthode appelée fermeture de boucles [10] [11]. L'odométrie visuelle peut être basée sur une caméra monoculaire [12] ou un système stéréoscopique qui a l'avantage de pouvoir estimer directement l'échelle à partir de la *baseline* connue entre les deux caméras [13].

Lorsqu'une carte de l'environnement est nécessaire, ce qui est généralement le cas pour les tâches de planification de trajectoires et de navigation d'un véhicule autonome, l'odométrie visuelle n'est plus suffisante. Si la carte n'est pas connue ou incomplète, le véhicule doit se localiser tout en construisant cette carte. Cette problématique est connue dans la communauté robotique sous le terme SLAM : *Simultaneous Localization and Mapping*. Contrairement aux robots évoluant dans des espaces limités, le véhicule autonome doit être capable de gérer des environnements très vastes et en temps-réel, tout en tenant compte des changements de l'environnement, ce qui est à l'heure actuelle encore un challenge. Lorsque les données utilisées pour cette tâche sont uniquement visuelles, on parle également de *Structure-from-Motion* (SfM). L'un des systèmes les plus aboutis de SLAM visuel est ORB-SLAM [14], qui combine des méthodes de détection et de fermeture de boucles, ainsi qu'une optimisation de graphes de poses dans un seul système.

Contrairement aux méthodes de localisation décrites précédemment qui utilisent et construisent généralement des cartes métriques, le dernier niveau de localisation a pour but de

retrouver le lieu le plus probable à partir d'une base d'images précédemment acquises [15] ; il s'agit donc de reconnaissance de lieu ou *place recognition*. Ces méthodes utilisent des bases d'images associées à une localisation connue et transforment le problème de localisation en un problème de recherche d'image par le contenu (*image retrieval*). Elles se basent sur des mesures de similarité pour retrouver dans la base d'images (la mémoire visuelle) l'image la plus semblable à l'image acquise (la requête) et ainsi résoudre le problème de localisation. Ces méthodes ont l'avantage de pouvoir résoudre également le problème du *robot kidnapping* [16]. Cependant, il reste de nombreux challenges à relever pour faire face aux changements d'apparence, aux lieux qui se ressemblent, et aux changements de point de vue. La taille de l'environnement est également une problématique, puisque plus la base d'images est importante, plus la localisation est difficile.

Ce chapitre est consacré aux travaux que j'ai menés sur la localisation basée vision pour le véhicule autonome et aux différents niveaux de localisation décrits précédemment : odométrie visuelle, SfM et reconnaissance visuelle de lieu.

Dans un premier temps, nous proposons une méthode de *Structure-from-Motion* à partir d'un réseau de caméras asynchrones. Nous montrons dans ces travaux qu'il est possible d'estimer les déplacements et de reconstruire l'environnement à l'échelle malgré l'utilisation de caméras non-synchronisées en posant quelques hypothèses sur le déplacement du véhicule. Ces travaux ont été menés dans le cadre de la thèse de Rawia Mhiri [Th1] [1] et des projets PLUTON (Plateforme Unifiée pour le Test des systèmes de visiON) et DrAACaR (*Driver Assistance by Asynchronous Camera Ring*).

Nous présentons dans un second temps nos travaux sur la localisation visuelle multimodale visible/infrarouge. L'objectif est d'être capable de se re-localiser grâce à une mémoire visuelle apprise au préalable, malgré les changements de caractéristiques du capteur de vision (champ de vue, résolution, bande spectrale) entre la construction de la mémoire visuelle et la phase de re-localisation. La problématique de ces travaux est donc la reconnaissance visuelle de lieu par des capteurs aux caractéristiques très différentes. Ces travaux ont été menés dans le cadre de la thèse de Fabien Bonardi [Th4] et du projet SAVEMORE (*Smart Autonomous VEhicle for urban MObility using Renewable Energy*).

Enfin, nous présentons dans ce chapitre nos travaux sur la parallélisation des algorithmes de localisation basée vision sur des architectures GPU. Nous proposons dans ces travaux des stratégies permettant d'améliorer le temps de calcul bien que les algorithmes de localisation soient intrinsèquement séquentiels. Ces travaux ont été réalisés dans le cadre de la thèse de Safa Ouerghi [Th5], en collaboration avec l'Ecole Supérieure des Télécommunications de Tunis (SUP'COM).

2.2 Structure-from-Motion à partir d'un réseau de caméras asynchrone

La surveillance de l'environnement d'un véhicule peut être réalisée avec un nombre très réduit de caméras. Une observation omnidirectionnelle de la scène est en effet possible avec l'utilisation d'un capteur catadioptrique, c'est-à-dire en combinant une caméra avec un miroir convexe [17]. De tels systèmes ont largement été étudiés dans la communauté robotique [18], [19], et des essais ont également été réalisés dans le domaine du véhicule autonome [20]. Cependant, l'utilisation

1. Les références données ici se rapportent à la section 1.3

de ces capteurs pour surveiller l'environnement d'un véhicule n'est pas optimale puisque ces capteurs doivent être placés sur le toit du véhicule pour avoir une vue à 360 degrés autour de celui-ci, et le toit peut occulter une partie de l'image correspondant à l'environnement proche du capteur. Pour atténuer ce phénomène, ils doivent être suffisamment surélevés, ce qui rend leur utilisation dans le domaine automobile peu crédible, aussi bien pour des raisons pratiques qu'esthétiques.

Une autre possibilité permettant d'obtenir une vision omnidirectionnelle est d'utiliser des réseaux de caméras. Il existe dans la littérature et sur le marché des systèmes multi-caméras, le plus connu étant sans aucun doute la Ladybug® [2]. Ces systèmes, très compacts, présentent plusieurs inconvénients. En effet, à l'instar des capteurs catadioptriques, ils doivent être placés sur le toit et à une certaine hauteur pour avoir une vue omnidirectionnelle. De plus, ils ne disposent que d'une faible *baseline*, quand il ne s'agit pas d'un point de vue quasi-unique, ce qui ne permet pas d'obtenir des points 3D par triangulation. Enfin, ces systèmes nécessitent une synchronisation précise pour obtenir des images de la scène prises exactement au même instant.

Les constructeurs automobiles intègrent de plus en plus de caméras dans leurs véhicules, notamment pour développer des systèmes d'aide à la conduite comme le système Nissan Around View® Monitor [3]. Ces systèmes sont généralement composés de 4 caméras, avec des champs de recouvrement assez faibles, et avec des caméras bas-coûts ne disposant pas de système de synchronisation. Nous nous sommes intéressés à ce type de configuration pour proposer un système de *Structure-from-Motion* utilisant un réseau de caméras non-synchronisées. Les avantages d'un tel réseau sont les suivants :

- il s'agit d'un système peu onéreux, qui ne nécessite pas de câblage supplémentaire ni de carte de synchronisation,
- il est possible d'exploiter des caméras différentes, et dans ce cas le système ne dépend plus de la caméra la plus lente,
- les images sont acquises en continu, ce qui limite dans une certaine mesure les problèmes de bande passante et permet d'avoir une localisation à une fréquence plus élevée, c'est-à-dire à chaque fois qu'une nouvelle image est acquise,
- il est possible de rajouter ou de supprimer une des caméras du système sans nuire à son bon fonctionnement.

Nos contributions ont été une nouvelle méthode d'estimation du mouvement à l'échelle, que nous avons appelée méthode des triangles. Cette méthode se base sur l'utilisation d'un triplet d'images provenant de deux caméras pour estimer les poses relatives entre les images avec des algorithmes de SfM classiques. Les facteurs d'échelle absolue sont ensuite estimés en intégrant une pose virtuelle d'une des caméras dans le triangle formé par le triplet d'images. Une méthode d'optimisation a ensuite été proposée afin d'améliorer la précision de l'estimation initiale. Cette optimisation est basée sur un ajustement de faisceaux portant sur les facteurs d'échelle et sur la structure 3D.

2.2.1 Etat de l'art

La reconstruction 3D à l'échelle de la scène à partir d'un réseau de caméras embarquées nécessite la capacité d'en établir la géométrie épipolaire et peut être effectuée selon plusieurs approches en fonction des connaissances *a priori* sur ce système. Si on considère le réseau

2. https ://www.ptgrey.com/ladybug5-30-mp-usb-30-spherical-digital-video-camera-black
3. http ://www.nissan-global.com/EN/TECHNOLOGY/MAGAZINE/around_view_monitor.html

calibré, à la fois intrinsèquement et extrinsèquement, ainsi que synchronisé, et avec des zones de recouvrement, alors les techniques classiques proposées dans [21] peuvent être appliquées à chaque capture d'un ensemble d'images. Il est également possible dans ce cas d'intégrer des images consécutives du réseau en mouvement dans un tenseur multifocal pour une estimation plus précise [22].

Pour les systèmes spécifiquement omnidirectionnels, [23] propose une approche basée SLAM. Dans ces travaux, les cinq caméras autour du véhicule sont synchronisées et les points d'intérêt doivent être visibles dans le champ de vue de deux caméras simultanément pour pouvoir obtenir le facteur d'échelle. Dans [24] et [25], un système de visualisation de l'environnement autour d'un véhicule est également présenté. Ces travaux concernent plus particulièrement l'optimisation au niveau matériel avec l'utilisation de SoC (System-on-Chip). Les caméras de ce système sont synchronisées par un signal externe, et le calibrage extrinsèque est rudimentaire puisque le véhicule (heureusement miniature) est placé au centre d'une mire contenant quatre carrés de positions et de dimensions connues.

Dans le cas spécifique du véhicule, [26] présente un réseau de caméras embarquées associé à une méthode d'étalonnage pour un usage dédié à l'analyse de risques dans un ADAS. Récemment, une méthode dédiée au système Around View® Monitor a également été proposée dans [27] pour effectuer la reconstruction de l'environnement. Cependant, la synchronisation des caméras étant de la plus haute importance, celle-ci a été effectuée par un signal externe.

Si au moins l'une des trois conditions initiales (calibrage, synchronisation, recouvrement) est relâchée, alors des méthodes dédiées sont nécessaires. Ainsi, dans le cas où le système est calibré et synchronisé mais sans recouvrement entre les caméras, il est possible de définir une contrainte sur le mouvement rigide des deux caméras pour assembler deux vues séparées. Un exemple de ce type pour un système stéréoscopique fish-eye synchrone est présenté dans [28]. Pour s'affranchir de la contrainte de chevauchement des champs de vision, un algorithme d'odométrie visuelle est appliqué séparément sur chaque caméra à l'échelle près. Une solution linéaire est définie afin de fusionner les facteurs d'échelle provenant des deux estimations en imposant la transformation rigide connue entre les deux capteurs pour retrouver enfin l'échelle métrique absolue. Plus récemment, les mêmes problèmes ont été abordés par [29] qui propose une procédure d'initialisation de l'échelle avec des résultats intéressants. L'ensemble de ces méthodes peut plus globalement s'apparenter à la notion de caméra généralisée [30][31][32].

Dans le cas où le calibrage extrinsèque est absent également, les méthodes effectuent généralement à la fois la reconstruction de la scène et l'estimation des paramètres de calibrage. Ainsi, un système multi-caméras embarqué sur un véhicule a été développé dans le cadre du projet V-Charge (*Autonomous Valet Parking and Charging for e-Mobility*) [33]. Le calibrage extrinsèque est présenté dans [34] et [35] où l'approche est basée sur l'utilisation de données d'odométrie du véhicule en complément des données visuelles. Un algorithme de VO suivi d'un ajustement de faisceaux est utilisé pour chaque caméra. Les données de VO ainsi obtenues ont alors des échelles différentes pour chaque caméra. Les points d'intérêt utilisés pour la VO sont alors triangulés grâce à une première approximation de la transformation caméra-odométrie et des données d'odométrie fournies par le véhicule. Un second ajustement de faisceaux est alors appliqué en gardant fixes les poses des caméras et en optimisant les points 3D et la transformation odométrie-caméra. La précision n'est alors pas encore suffisante car les erreurs de reprojection restent élevées. Pour pallier ce problème, les points d'intérêt sont appariés dans les images de plusieurs caméras et des fermetures de boucles sont également réalisées lorsqu'elles sont détectées. Un dernier ajustement de faisceaux est alors mis en œuvre afin d'optimiser l'ensemble

des paramètres. L'inconvénient de cette méthode est qu'elle nécessite l'utilisation de données supplémentaires en plus des données visuelles.

Dans [36], le même problème est abordé en intégrant en plus les décalages temporels dus à l'effet de *rolling shutter*. Cependant, dans cette approche, les caméras sont considérées comme toutes identiques intrinsèquement, uniformément réparties et placées suffisamment proches pour approximer un point de vue unique. L'approche repose essentiellement sur un ajustement de faisceaux intégrant l'ensemble de tous les paramètres du réseau et de la scène reconstruite à partir de la méthode proposée dans [37]. Ils estiment le décalage temporel en utilisant les premières images d'une séquence vidéo. Cela leur permet de synchroniser les caméras à une image près. Un ajustement de faisceaux est alors réalisé puisque les images sont cette fois supposées synchronisées. Pour obtenir également le décalage dû au *rolling shutter*, un second ajustement de faisceaux est mis en œuvre en ajoutant ce paramètre additionnel. Notre méthode diffère de celle-ci pour plusieurs raisons : nous ne posons aucune hypothèse sur les caméras et elles peuvent fonctionner avec des fréquences différentes (et même non-constantes). De plus, dans cet article, l'ajustement de faisceaux est réalisé *a posteriori* sur la totalité de la séquence vidéo, alors que nous faisons un ajustement de faisceaux en temps-réel. Enfin, le système étudié dans ces travaux est supposé avoir un point de vue unique, ce qui signifie se ramener à une configuration monoculaire.

Dans le cas de caméras non-synchronisées, l'accent a surtout été mis sur l'estimation du décalage temporel entre les différents flux de données vidéo [38][39][40]. En effet, dans un flux vidéo avec des objets en mouvement, les objets se déplacent de la même façon dans toutes les vues et peuvent être utilisés pour calculer le décalage temporel. Pour une séquence entière d'une scène dynamique observée par des caméras non-synchronisées, la mise en correspondance de primitives et la synchronisation peuvent être post-traitées. Cependant, ces méthodes ne sont pas applicables au cas du véhicule puisque les traitements sont effectués hors-ligne.

Dans [41], l'approche consiste à simuler l'image manquante d'un système asynchrone pour calculer la structure 3D en utilisant l'odométrie du robot. Les points 2D de l'image manquante sont obtenus par interpolation des points 2D détectés dans les images acquises. L'approche repose sur la linéarité du déplacement entre deux images consécutives d'une même caméra.

Dans tous les travaux cités précédemment, dès lors qu'il est nécessaire d'avoir une reconstruction à l'échelle absolue, il faut soit ajouter des capteurs supplémentaires, soit avoir des caméras synchronisées et des points visibles par aux moins deux caméras.

2.2.2 La méthode des triangles

Cette section présente la méthode que nous avons proposée pour retrouver le mouvement d'un système de caméras asynchrones. Cette méthode repose sur trois hypothèses principales :

— Les caméras doivent être montées sur un système rigide et avoir des champs de vue commun deux à deux. Comme les caméras sont fixées rigidement, les transformations entre les différentes caméras sont connues par calibrage. Le champ de vue commun entre deux caméras consécutives permet de mettre en correspondance et de trianguler des points.

— Les caméras doivent être calibrées hors ligne pour connaitre à la fois leurs paramètres intrinsèques et extrinsèques. Les paramètres extrinsèques vont être utilisés pour retrouver les facteurs d'échelle absolue et les paramètres intrinsèques vont permettre l'estimation de la pose et la triangulation.

— Le mouvement entre deux vues consécutives est supposé linéaire et par conséquent approximé par un segment. Avec cette approximation, nous considérons que les vecteurs de translation entre trois positions de la même caméra sont colinéaires. Cette hypothèse correspond à la réalité, car les fréquences d'acquisition des caméras sont suffisamment élevées pour que l'intervalle de temps entre trois images soit faible. Par exemple, pour un véhicule équipé d'une caméra à 30 images par secondes et roulant à 50km/h, la distance parcourue entre les trois images est d'environ 1.30m, soit 45 centimètres entre chaque image.

Nous séparons la méthode en deux parties : l'estimation de la pose relative dans un premier temps, puis l'estimation du facteur d'échelle absolue. L'estimation des poses relatives des caméras est effectuée par des algorithmes d'odométrie visuelle classiques [42]. Les facteurs d'échelle absolue sont ensuite estimés en utilisant le calibrage extrinsèque et l'hypothèse de linéarité du mouvement. Pour simplifier l'explication de notre méthode, et sans perte de généralité, nous considérons un système à deux caméras qui vont acquérir des images à trois instants différents. La première et la deuxième caméra, C_i et C_j, vont acquérir les images I_i et I_j respectivement. L'instant d'acquisition est noté en indice du nom de la caméra. Par exemple, la position de la caméra C_i à l'instant t_0 est notée C_{i0}. La transformation Euclidienne de la caméra C_{i0} vers la caméra C_{j1} est donnée par $\mathbf{T}_{i0}^{j1}$. De la même manière, la matrice de rotation et le vecteur unitaire de translation de la caméra C_{i0} vers la caméra C_{j1} sont notés $\mathbf{R}_{i0}^{j1}$ et $\mathbf{t}_{i0}^{j1}$ respectivement.

Estimation de pose

La transformation euclidienne entre les deux poses relatives d'une même caméra peut être décrite par un vecteur unitaire de translation $\mathbf{t}$, par une matrice de rotation $\mathbf{R}$ et un facteur d'échelle λ. La transformation entre deux positions peut donc être exprimée par :

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \lambda\mathbf{t} \\ 0^\intercal & 1 \end{bmatrix}. \tag{2.1}$$

Comme notre système est calibré, nous pouvons estimer la matrice de rotation $\mathbf{R}$ et le vecteur de translation $\mathbf{t}$ entre les poses de deux caméras par la mise en correspondance de points d'intérêt et en exploitant la géométrie épipolaire entre ces deux vues [21].

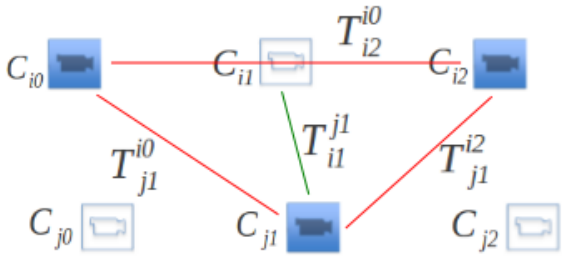


FIGURE 2.1 – La méthode des triangles pour deux caméras non synchronisées : les lignes rouges représentent les transformations obtenues par SfM et la ligne verte représente la transformation rigide obtenue par calibrage extrinsèque.

La méthode des triangles repose sur l'estimation de trois matrices essentielles entre les images acquises à trois instants différents : deux images capturées par la même caméra (C_i par exemple)

et une image par l'autre caméra (C_j dans ce cas). Ce triangle est illustré sur la figure 2.1 qui représente toutes les transformations possibles entre les trois images formant ce triangle. Les transformations $\mathbf{T}_{i2}^{i0}$, $\mathbf{T}_{j1}^{i0}$ et $\mathbf{T}_{j1}^{i2}$ entre les trois images sont obtenues par l'algorithme des 5 points [42]. La transformation rigide $\mathbf{T}_{i1}^{j1}$ est obtenue par le calibrage extrinsèque effectué hors ligne.

La première étape de notre algorithme est donc l'extraction et la mise en correspondance des points d'intérêt détectés dans les trois images formant ce triangle. Nous avons pour cela utilisé le détecteur FAST [43] et le descripteur BRIEF [44]. Ces étapes sont appliquées sur les images I_{i0}, I_{i2} and I_{j1}.

La caméra C_i passe par une position intermédiaire où elle ne fait pas l'acquisition d'une image à cause de la non-synchronisation. Il s'agit de la pose C_{i1} qui peut être estimée en utilisant la transformation $\mathbf{T}_{i1}^{j1}$. Pour résumer, nous utilisons quatre transformations : trois transformations calculées par SfM ($\mathbf{T}_{i2}^{i0}$, $\mathbf{T}_{j1}^{i0}$ et $\mathbf{T}_{j1}^{i2}$) et une transformation rigide ($\mathbf{T}_{i1}^{j1}$) issue du calibrage extrinsèque.

Estimation des facteurs d'échelle

Jusqu'à maintenant, les facteurs d'échelle absolue sont inconnus entre les poses des caméras. Avec les hypothèses que nous avons posées, il est possible d'exprimer les poses par quatre transformations principales, comme l'illustre la figure 2.2. Les trois images (I_{i0}, I_{i2} et I_{j1}) forment le triangle principal entre les poses de C_{i0}, C_{i2} et C_{j1}. La pose virtuelle de la caméra C_{i1} peut être considérée comme une position intermédiaire dans ce triangle. Cette position génère deux sous-triangles : le premier est formé par C_{i0}, C_{i1} et C_{j1}, et le second par C_{i1}, C_{i2} et C_{j1}.

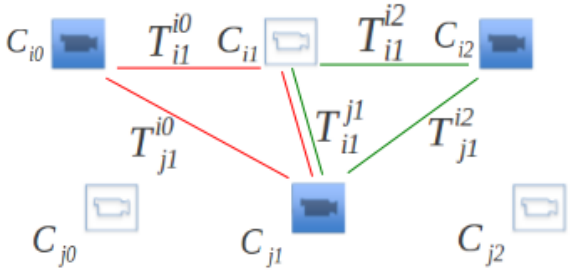


FIGURE 2.2 – Le premier sous-triangle entre les caméras (C_{i0}, C_{i1} et C_{j1}) et le second sous-triangle entre les caméras (C_{i1}, C_{i2} et C_{j1}).

Comme illustré sur la figure 2.2, dans le premier sous-triangle, la transformation $\mathbf{T}_{i1}^{i0}$ de la caméra C_{i1} vers la caméra C_{i0} est égale à la transformation $\mathbf{T}_{j1}^{i0}$ de C_{j1} vers C_{i0} multipliée par la transformation $\mathbf{T}_{i1}^{j1}$:

$$\mathbf{T}_{i1}^{i0} = \mathbf{T}_{j1}^{i0}\mathbf{T}_{i1}^{j1}. \tag{2.2}$$

Les transformations euclidiennes sont exprimées en coordonnées homogènes. La transformation rigide $\mathbf{T}_{i1}^{j1}$ obtenue par le calibrage extrinsèque est une transformation à l'échelle, c'est-à-dire que le facteur d'échelle de cette transformation est connu. En ré-injectant l'équation (2.1) dans l'équation (2.2), et en introduisant les deux facteurs d'échelle inconnus λ_1 and α, nous obtenons :

$$\begin{bmatrix} \mathbf{R}_{i1}^{i0} & \lambda_1 \mathbf{t}_{i1}^{i0} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{j1}^{i0} & \alpha \mathbf{t}_{j1}^{i0} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{i1}^{j1} & \mathbf{t}_{i1}^{j1} \\ 0 & 1 \end{bmatrix}, \tag{2.3}$$

où λ_1 est le facteur d'échelle associé à la transformation $\mathbf{T}_{i1}^{i0}$ et α est le facteur d'échelle associé à la transformation $\mathbf{T}_{j1}^{i0}$.

A partir de (2.3), nous pouvons obtenir les deux équations suivantes :

$$\mathbf{R}_{i1}^{i0} = \mathbf{R}_{j1}^{i0} \mathbf{R}_{i1}^{j1}, \tag{2.4}$$

et

$$\lambda_1 \mathbf{t}_{i1}^{i0} - \alpha \mathbf{t}_{j1}^{i0} = \mathbf{R}_{j1}^{i0} \mathbf{t}_{i1}^{j1}. \tag{2.5}$$

De plus, l'équation (2.5) peut être formulée :

$$\begin{bmatrix} \mathbf{t}_{i1}^{i0} & -\mathbf{t}_{j1}^{i0} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \alpha \end{bmatrix} = \mathbf{R}_{j1}^{i0} \mathbf{t}_{i1}^{j1}. \tag{2.6}$$

Comme illustré par la figure 2.2, dans le second sous-triangle, la transformation $\mathbf{T}_{i1}^{i2}$ de C_{i1} vers C_{i2}, est égale à la transformation $\mathbf{T}_{j1}^{i2}$ de la caméra C_{j1} vers la caméra C_{i2} multipliée par la transformation rigide $\mathbf{T}_{i1}^{j1}$ de la caméra C_{i1} vers la caméra C_{j1} :

$$\mathbf{T}_{i1}^{i2} = \mathbf{T}_{j1}^{i2} \mathbf{T}_{i1}^{j1}. \tag{2.7}$$

De la même manière, l'équation (2.7) est développée pour obtenir :

$$\begin{bmatrix} \mathbf{R}_{i1}^{i2} & \lambda_2 \mathbf{t}_{i1}^{i2} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{j1}^{i2} & \beta \mathbf{t}_{j1}^{i2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{i1}^{j1} & \mathbf{t}_{i1}^{j1} \\ 0 & 1 \end{bmatrix}. \tag{2.8}$$

Dans cette équation, λ_2 représente le facteur d'échelle associé à la transformation $\mathbf{T}_{i1}^{i2}$. β représente le facteur d'échelle associé à la transformation $\mathbf{T}_{i2}^{j1}$.

Si nous séparons dans l'équation (2.8) les termes de rotation et de translation, nous obtenons alors :

$$\mathbf{R}_{i1}^{i2} = \mathbf{R}_{j1}^{i2} \mathbf{R}_{i1}^{j1} \tag{2.9}$$

et

$$\lambda_2 \mathbf{t}_{i1}^{i2} - \beta t_{j1}^{i2} = \mathbf{R}_{j1}^{i2} \mathbf{t}_{i1}^{j1}. \tag{2.10}$$

Dans le triangle principal formé par le triplet C_{i0}, C_{i2} et C_{j1}, les transformations Euclidiennes entre les poses peuvent être exprimées comme dans l'équation (2.11). La transformation $\mathbf{T}_{i2}^{i0}$ de C_{i2} vers C_{i0} est égale à la transformation $\mathbf{T}_{j1}^{i0}$ de C_{j1} vers C_{i0} multipliée par la transformation $\mathbf{T}_{i2}^{j1}$ de C_{i2} vers C_{j1} :

$$\mathbf{T}_{i2}^{i0} = \mathbf{T}_{j1}^{i0} \mathbf{T}_{i2}^{j1}. \tag{2.11}$$

Comme pour les sous-triangles, et en utilisant l'hypothèse de linéarité, l'équation (2.11) devient :

$$\begin{bmatrix} \mathbf{R}_{i2}^{i0} & (\lambda_1+\lambda_2)\mathbf{t}_{i2}^{i0} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{j1}^{i0} & \alpha\mathbf{t}_{j1}^{i0} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{i2}^{j1} & \beta\mathbf{t}_{i2}^{j1} \\ 0 & 1 \end{bmatrix}. \tag{2.12}$$

Après le développement de l'équation (2.12), nous séparons les termes de rotation et de translation. La partie concernant la translation mène à l'équation (2.13) :

$$\begin{bmatrix} \mathbf{t}_{i2}^{i0} & \mathbf{t}_{i2}^{i0} & -\mathbf{t}_{j1}^{i0} & -\mathbf{R}_{j1}^{i0}\mathbf{t}_{i2}^{j1} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \alpha \\ \beta \end{bmatrix} = \mathbf{0}. \tag{2.13}$$

Par conséquent, les équations basées sur la translation pour les trois triangles peuvent être exprimées par le système d'équations suivant :

$$\begin{cases} \lambda_1\mathbf{t}_{i_1}^{i^0} - \alpha\mathbf{t}_{j_1}^{i^0} = \mathbf{R}_{j_1}^{i^0}\mathbf{t}_{i_1}^{j^1} \\ \lambda_2\mathbf{t}_{i_1}^{i^2} - \beta\mathbf{t}_{j_1}^{i^2} = \mathbf{R}_{j_1}^{i^2}\mathbf{t}_{i_1}^{j^1} \\ \lambda_1\mathbf{t}_{i_2}^{i^0} + \lambda_2\mathbf{t}_{i_2}^{i^0} - \beta\mathbf{R}_{j_1}^{i^0}\mathbf{t}_{i_2}^{j^1} - \alpha\mathbf{t}_{j_1}^{i^0} = \mathbf{0}. \end{cases} \tag{2.14}$$

Pour résoudre ces équations et obtenir les facteurs d'échelle, il est possible d'écrire ce système comme suit :

$$\begin{bmatrix} \mathbf{t}_{i1}^{i0} & 0 & -\mathbf{t}_{j1}^{i0} & 0 \\ 0 & \mathbf{t}_{i1}^{i2} & 0 & -\mathbf{t}_{j1}^{i2} \\ \mathbf{t}_{i2}^{i0} & \mathbf{t}_{i2}^{i0} & -\mathbf{t}_{j1}^{i0} & -\mathbf{R}_{j1}^{i0}\mathbf{t}_{i2}^{j1} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{j1}^{i0}\mathbf{t}_{i1}^{j1} \\ \mathbf{R}_{j1}^{i2}\mathbf{t}_{i1}^{j1} \\ \mathbf{0} \end{bmatrix}. \tag{2.15}$$

Les facteurs d'échelle absolue peuvent être estimés par la résolution de l'équation (2.15) au sens des moindres carrés puisque cette équation peut être exprimée comme un système linéaire de la forme :

$$\mathbf{AX} = \mathbf{B}, \tag{2.16}$$

où $\mathbf{X}$ est le vecteur composé des facteurs d'échelle λ_1, λ_2, α et β. Il est par conséquent possible d'estimer toutes les poses relatives, y compris les facteurs d'échelle absolue avec notre méthode.

2.2.3 Optimisation par ajustement de faisceaux

Les estimations initiales des facteurs d'échelle ne sont pas suffisamment précises à cause des hypothèses sur lesquelles repose la méthode des triangles, en particulier pour les trajectoires ayant un fort rayon de courbure et/ou des vitesses élevées. Pour cette raison, l'estimation initiale ne peut pas être utilisée directement dans des applications de navigation, de détection d'obstacles, etc.

Dans cette section, nous proposons d'appliquer un ajustement de faisceaux local, c'est-à-dire un ajustement de faisceaux limité à un certain nombre de vues. Dans ces travaux, les caméras sont supposées calibrées et nous considérons par conséquent qu'il n'est pas nécessaire d'estimer les paramètres intrinsèques. Les paramètres à optimiser sont par conséquent les poses des caméras et les coordonnées des points 3D.

L'ajustement de faisceaux consiste à optimiser simultanément les coordonnées 3D des points de la scène et les poses des caméras. Le critère habituellement minimisé dans un algorithme d'ajustement de faisceaux est l'erreur de reprojection, c'est-à-dire l'erreur mesurée entre les points observés dans l'image et la projection estimée de leurs correspondants 3D (qui dépendent des paramètres à estimer). Dans le cas d'une caméra de type sténopé, la projection $\mathbf{x}$ d'un point 3D $\mathbf{X}$ dont les coordonnées sont exprimées en coordonnées homogènes dans un repère monde sont données par :

$$\mathbf{x} \sim [\mathbf{K}\ \mathbf{0}] \underbrace{\begin{bmatrix} \mathbf{R} & s\,\mathbf{t} \\ \mathbf{0}^\intercal & 1 \end{bmatrix}}_{\mathbf{T}_{world}^{cam}} \mathbf{X} = \mathbf{P}\mathbf{X}, \tag{2.17}$$

où $\mathbf{K}$ est la matrice des paramètres intrinsèques et $\mathbf{T}_{world}^{cam}$ est la matrice de transformation du repère monde vers le repère caméra. La matrice résultante $\mathbf{P}$ est la matrice de projection de la caméra.

Dans le cas général, l'ajustement de faisceaux est capable d'optimiser les paramètres intrinsèques (la matrice $\mathbf{K}$), les paramètres extrinsèques (la rotation et la translation de la caméra par rapport au repère monde) et les points 3D. Le problème doit par conséquent être résolu en minimisant l'erreur de reprojection d'une fonction non-linéaire, en utilisant par exemple l'algorithme de Levenberg-Marquardt [45].

Formulation du problème

Soient $\widehat{\mathbf{X}}_i$ les coordonnées estimées du i$^{\text{ème}}$ point et n le nombre de points. Soient $\widehat{\mathbf{P}}^j$ la matrice de projection de la j$^{\text{ème}}$ caméra et m le nombre de caméras. $\mathbf{x}_i^j$ est le point image extrait correspondant au i$^{\text{ème}}$ point dans l'image de la j$^{\text{ème}}$ caméra. La fonction coût à minimiser peut par conséquent être exprimée par :

$$\min_{\widehat{\mathbf{X}}_i,\, \widehat{\mathbf{P}}^j} \sum_{j=1}^{m} \sum_{i=1}^{n} d(\widehat{\mathbf{P}}^j\, \widehat{\mathbf{X}}_i,\ \mathbf{x}_i^j)^2 \tag{2.18}$$

où $d()$ est la distance Euclidienne. La minimisation de la fonction coût est effectuée par l'algorithme de Levenberg-Marquardt. Cet algorithme consiste essentiellement à calculer la matrice Jacobienne de cette fonction et à optimiser itérativement le système d'équations. L'étape clé de cet algorithme est la résolution de l'équation normale augmentée :

$$(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})\Delta = -\mathbf{J}^T\mathbf{e}, \tag{2.19}$$

où $\mathbf{J}$ est la matrice Jacobienne de la fonction de projection, λ est un scalaire qui varie d'itération en itération, Δ est le vecteur de correction des paramètres, et $\mathbf{I}$ est la matrice identité.

Positionnement par rapport à l'état de l'art

L'ajustement de faisceaux a atteint un certain niveau de maturité dans la littérature [46] [47]. L'ajustement de faisceaux a été largement étudié pour l'odométrie visuelle et pour les applications de *Structure-from-Motion* mais il n'existe pas, à notre connaissance, de travaux qui ont été effectués sur les systèmes asynchrones.

Dans la méthode proposée par Engels [46], un ajustement de faisceau fenêtré a été introduit pour optimiser localement les poses des caméras. Cette méthode a été exploitée dans le cadre de la reconstruction d'un objet disposé sur une table tournante et en optimisant tous les paramètres. Sur une échelle plus grande, Mouragnon [48] a aussi présenté un ajustement de faisceaux local adapté aux applications temps-réel pour de longues séquences obtenues avec des caméras calibrées.

Dans le cas où les caméras sont calibrées, les paramètres intrinsèques sont connus et ne nécessitent par conséquent pas d'être optimisés. La différence entre notre méthode et un algorithme d'ajustement de faisceaux classique appliqué à des caméras calibrées réside dans le nombre de paramètres à estimer. D'une part, un algorithme d'ajustement de faisceaux classique a pour but d'optimiser six paramètres par caméras (3 paramètres pour la rotation en utilisant la paramétrisation des angles d'Euler, et 3 paramètres pour la translation), ainsi que 3 paramètres par point. D'autre part, notre algorithme a pour but d'optimiser seulement les facteurs d'échelle et les points 3D, et par conséquent un paramètre par caméra et 3 paramètres par point.

Dans les travaux de Fraundorfer [49], un ajustement de faisceaux sous contrainte a été appliqué pour l'estimation de l'odométrie visuelle d'une seule caméra embarquée dans un véhicule. La principale différence entre cette méthode et les approches conventionnelles est la séparation de l'estimation relative du mouvement, présentée initialement dans [50], et l'estimation du facteur d'échelle. En effet, les auteurs mettent l'accent sur l'estimation cohérente de l'échelle en optimisant uniquement les distances entre les caméras voisines (les facteurs d'échelle relative). Les rotations et les directions des translations estimées initialement par le 1-point RANSAC sont considérées comme étant fixes, et les points 3D sont calculés à chaque itération du processus d'optimisation. C'est un ajustement de faisceaux global qui optimise toutes les échelles de la trajectoire. Les principales différences entre cette méthode et la nôtre sont :

- la configuration des caméras : la méthode de Fraundorfer [49] est appliquée à un système monoculaire alors que notre méthode est appliquée à un réseau de caméras asynchrones,
- il propose un algorithme basé sur l'hypothèse d'un mouvement circulaire alors que le nôtre est basé sur l'hypothèse d'un mouvement rectiligne,
- nous optimisons les facteurs d'échelle et les points 3D là où la méthode proposée par Fraundorfer optimise seulement les facteurs d'échelle [49]. Ils calculent les points 3D à chaque itération avec le nouveau facteur d'échelle.

Optimisation des facteurs d'échelle et de la structure 3D avec notre ajustement de faisceaux local

Nous supposons $\mathbf{R}$ et $\mathbf{t}$ connus et nous optimisons uniquement le facteur d'échelle s et les coordonnées du point 3D $\mathbf{X}$. La projection décrite par l'équation (2.17) peut par conséquent être écrite comme une fonction F dépendant de s et de $\mathbf{X}$:

$$F(\mathbf{X}, s) = \mathbf{P}\mathbf{X}. \tag{2.20}$$

Dans notre méthode, la matrice Jacobienne $\mathbf{J}$ est calculée par dérivation de la fonction de projection F par rapport au facteur d'échelle s et au point 3D $\mathbf{X}$ uniquement. Soient $\mathbf{J_X}$ la matrice Jacobienne (matrice 2×3) de F par rapport au point $\mathbf{X}$, et $\mathbf{J_S}$ la matrice Jacobienne (matrice 2×1) de F par rapport au facteur d'échelle s, alors :

$$\mathbf{J_X} = \left[\frac{\partial F}{\partial \mathbf{X}}\right]_{2\times 3} \text{ et } \mathbf{J_S} = \left[\frac{\partial F}{\partial s}\right]_{2\times 1}. \tag{2.21}$$

Pour chaque point 3D et chaque pose de caméra, les matrices $\mathbf{J_X}$ et $\mathbf{J_S}$ sont calculées pour la fenêtre glissante considérée. La matrice Jacobienne résultante $\mathbf{J}$ a une structure éparse comme l'illustre la figure 2.3. Si nous considérons m caméras et n points 3D, la Jacobienne est une matrice de dimension $(2\cdot n\cdot m)\times(m+3\cdot n)$. Une fois la matrice Jacobienne calculée, l'algorithme de Levenberg-Marquardt est appliqué.

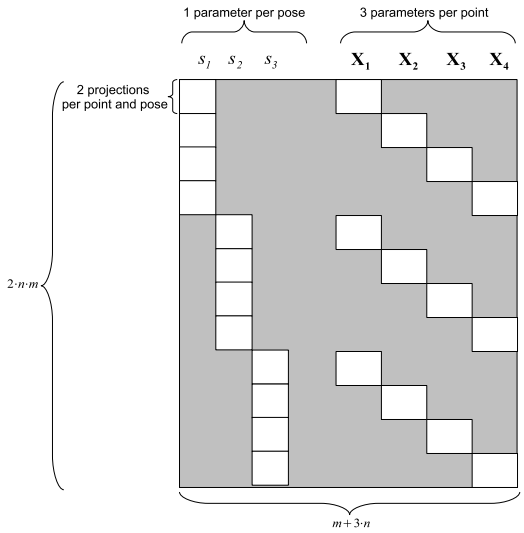


FIGURE 2.3 – Structure de la matrice Jacobienne pour 3 poses et 4 points. Les éléments nuls sont représentés en gris.

2.2.4 Résultats expérimentaux

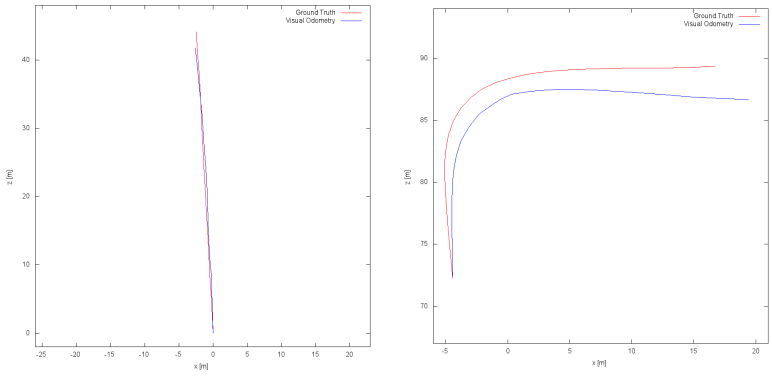
Évaluation de la méthode des triangles

Pour évaluer la méthode des triangles, nous avons vérifié la validité des hypothèses sur lesquelles repose cette méthode sur des séquences réelles de la base de données KITTI [51] [52]. La base de données KITTI est une base publique fournissant les données de capteurs embarqués sur un véhicule : caméras, lidar, GPS-RTK - Centrale Inertielle.

Nous avons évalué notre méthode en utilisant les images de deux caméras embarquées et en comparant nos résultats à la vérité terrain obtenue par un système GPS-Centrale inertielle. Nous avons utilisé les séquences issues de la paire stéréoscopique en ne prenant qu'une image à chaque instant pour simuler la désynchronisation. En d'autres termes, nous prenons les images paires de la caméra de gauche avec les images impaires de la caméra de droite pour obtenir des images non-synchronisées. Les poses relatives et les facteurs d'échelle sont calculés pour chaque ensemble de trois images formant un triangle principal.

Pour chaque série de trois images (trois instants), les points d'intérêts sont extraits en utilisant le détecteur FAST et le descripteur BRIEF. L'algorithme des 5-points permet ensuite d'estimer la matrice essentielle tout en rejetant les mauvais appariements grâce à une approche

RANSAC pour chaque paire d'images. Les rotations et les translations relatives sont obtenues par décomposition de ces matrices essentielles. Les points d'intérêts sont alors triangulés pour retrouver leurs coordonnées 3D. Le système d'équations (2.15) est résolu pour calculer les facteurs d'échelle absolue λ_1, λ_2, α et β.



(a) Ligne droite (b) Virage

FIGURE 2.4 – Estimation de la trajectoire par la méthode des triangles (en bleu) et vérité terrain (en rouge) dans deux cas spécifiques : (a) Ligne droite, (b) Virage.

Pour les trajectoires rectilignes, nous obtenons une estimation très proche de la vérité terrain comme illustré dans la figure 2.4.a. La figure 2.4.b correspond à la trajectoire estimée dans un virage. Même dans un virage (une rotation d'à peu près 90 degrés), la méthode des triangles donne de bons résultats. En effet, le véhicule ne roule généralement pas très rapidement dans les virages. De plus, les caméras de nos jours ont des fréquences d'acquisition suffisamment élevées. En tenant compte de ces deux points, l'approximation du déplacement entre deux images d'une même caméra par un segment de droite reste valide. La trajectoire dans un virage peut donc être interprétée comme une succession de trajectoires rectilignes entre deux images d'une caméra.

La figure 2.5 montre l'évaluation moyenne des dix séquences en termes d'erreurs de translation et de rotation. Nous avons obtenu des erreurs de rotations entre 0.041 et 0.015 degrés par mètre pour des séquences entre 100 à 800 mètres (figure 2.5.a) et entre 0.14 et 0.02 degrés par mètres pour des vitesses de 5 à 90 kilomètres par heure. Pour la translation, les erreurs moyennes sont entre 7 et 9% pour des séquences de 100 et 800 mètres et entre 5 et 22% pour des vitesses entre 5 et 90 km/h.

Nous remarquons que les erreurs de rotations sont plus élevées pour les séquences courtes et pour les vitesses lentes. Les erreurs de translations croissent d'à peu près 2% pour les séquences les plus longues. Cependant, les autres méthodes qui effectuent des optimisations, tel que l'ajustement de faisceaux, obtiennent des allures décroissantes pour les erreurs de translation en fonction des longueurs des séquences. Les 2% d'erreurs en plus sont certainement dues à l'accumulation des erreurs. Nous expliquons ces erreurs par le calcul des facteurs d'échelle. De faibles erreurs dues à l'approximation de la trajectoire linéaire s'accumulent pour les séquences les plus longues.

La figure 2.6 illustre les résultats obtenus par la méthode des triangles sur quelques séquences de la base de données KITTI. Ces séquences sont très différentes, que ce soit au niveau des trajectoires réalisées par le véhicule, ou au niveau du contenu des scènes (végétations, autres véhicules en mouvement, bâtiments, etc).

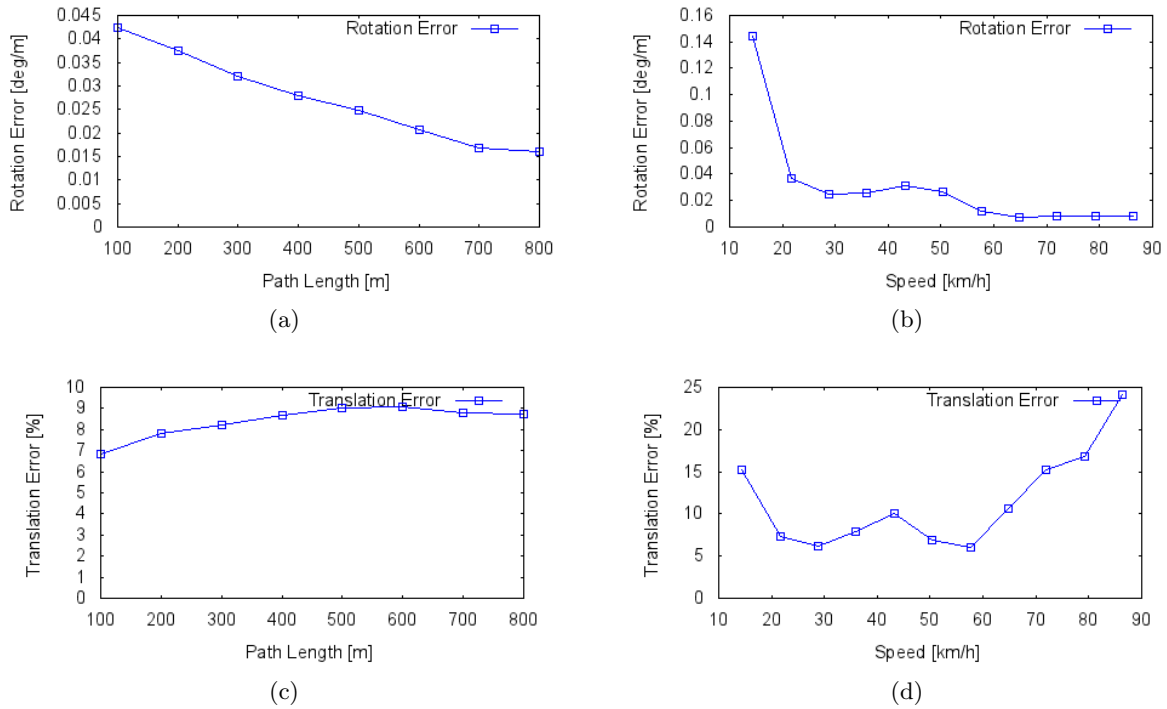


FIGURE 2.5 – Erreurs moyennes sur les dix séquences de la base de données KITTI fournissant le vérité terrain.

Évaluation de l'optimisation par ajustement de faisceaux

Intérêt et évaluation d'un ajustement de faisceaux optimal

Dans cette partie, nous présentons les résultats de la méthode d'optimisation des facteurs d'échelle décrite précédemment. L'algorithme est appliqué sur une séquence d'images réelles de la base de données KITTI. Nous simulons l'aspect asynchrone de la même manière que pour les expérimentations décrites précédemment. Nous avons choisi de tester l'ajustement de faisceaux (AF) local sur deux triangles consécutifs (cinq images).

Les points 3D et les poses des caméras à l'échelle sont initialement estimés par la méthode des triangles. Pour appliquer l'AF, nous exprimons les poses et les points 3D dans le repère de la première caméra de la fenêtre glissante. Ensuite, nous appliquons l'algorithme de Levenberg-Marquardt.

Les résultats obtenus sont présentés pour quelques images de la séquence 0 de la base de données KITTI. Dans un premier temps, nous validons la méthode sur une estimation parfaite : les points 3D triangulés à l'aide de la vérité terrain issue des données du GPS différentiel (synchronisé avec les images). Nous appliquons ensuite l'AF sur ces données. Nous avons appelé ce test "Ajustement de Faisceaux Optimal" puisqu'il sert de référence à laquelle nous allons ensuite comparer l'AF appliqué sur la méthode des triangles.

Pour évaluer l'AF optimal, nous calculons le rapport des facteurs d'échelle avant et après l'AF comme indiqué dans les équations (2.22) et (2.23). Les facteurs d'échelle sont les normes des vecteurs de translation. Les rapports obtenus sont très proches de 1, ce qui signifie que le processus de traitement et que les données sont très proches de la réalité.

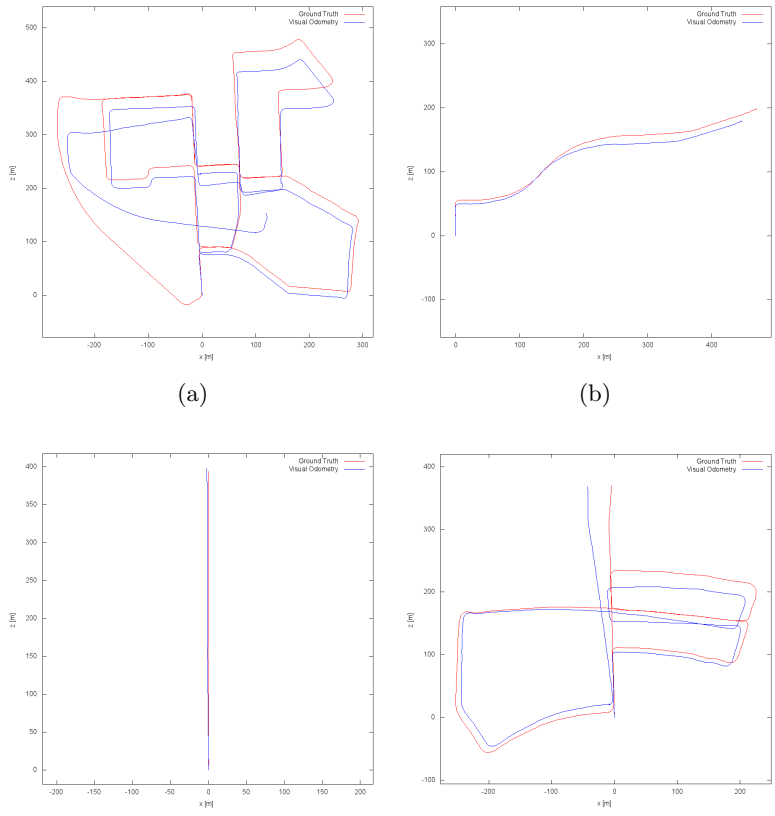


FIGURE 2.6 – Trajectoires estimées par la méthodes des triangles (en bleu) sur quelques séquences de la base de données KITTI. La vérité terrain issue du GPS est représentée en rouge.

$$\text{Rapport avant} = \frac{\text{facteur d'échelle évalué avant AF}}{\text{facteur d'échelle de la vérité terrain}} \tag{2.22}$$

$$\text{Rapport après } = \frac{\text{facteur d'échelle évalué après AF}}{\text{facteur d'échelle après l'AF optimal}} \tag{2.23}$$

Pour évaluer la performance de l'approche, nous ajoutons un bruit gaussien ($\sigma = 0.01$) à la vérité terrain (avant l'AF). Le bruit est directement ajouté aux valeurs des facteurs d'échelle (avant l'AF) car nous cherchons à optimiser ces paramètres. Les points 3D sont ensuite calculés à partir des poses bruitées avant que l'ensemble ne soit optimisé. Les facteurs d'échelle évalués avant l'AF sont donc différents de ceux de la vérité terrain.

Les résultats obtenus sont très satisfaisants puisque la trajectoire obtenue après l'AF est presque identique à celle de départ (la trajectoire obtenue par le GPS). Les résultats des ratios avant et après l'AF sont présentés dans la table 2.1. L'échelle 1 est le facteur d'échelle de la pose de la deuxième caméra de la fenêtre dans le repère de la première caméra de la fenêtre, l'échelle 2 est le facteur d'échelle de la pose de la troisième caméra dans le repère de la première. Il en est de même pour les échelles 3 et 4. Les rapports calculés sont proches de 1 ; nous jugeons donc que notre algorithme donne des résultats très précis.

TABLE 2.1 – Rapports avant et après l'AF pour la vérité terrain bruitée par un bruit gaussien ($\sigma = 0.01$).

	échelle 1	échelle 2	échelle 3	échelle 4
avant AF	1.0014	0.9993	0.9996	0.9994
après AF	1.0003	0.999803	01.0001	1.003

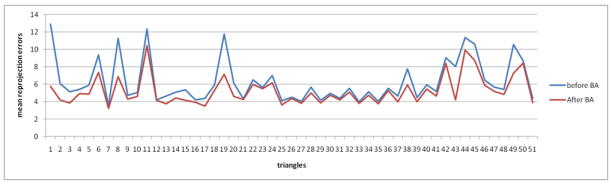


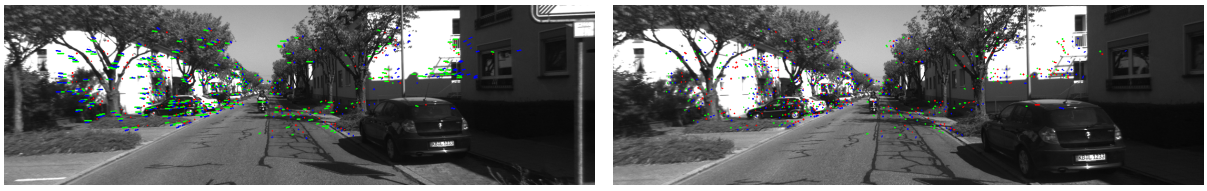
FIGURE 2.7 – Erreurs de reprojection moyennes pour les 52 triangles avant et après AF appliqué aux données issues de la méthode des triangles.

Pour résumer, les erreurs obtenues sont très faibles et dues à plusieurs raisons : l'imprécision des détecteurs de points d'intérêt, les erreurs de mise en correspondance, les erreurs de calcul de la triangulation et de la reprojection des points 3D dans les images. En bruitant les facteurs d'échelle et les points 3D calculés, l'AF permet de minimiser les erreurs de reprojection pour améliorer les facteurs d'échelle et les points 3D. Lorsque les paramètres du mouvement sont parfaits (la vérité terrain), quelques erreurs subsistent encore. Ces erreurs sont probablement dues à des imprécisions dans le calibrage, la synchronisation des capteurs, ou même des imprécisions du système GPS/INS. C'est pour toutes ces raisons que nous comparons nos résultats obtenus par la méthode des triangles à un ajustement de faisceaux optimal.

Evaluation quantitative de l'AF appliqué à la méthode des triangles

Dans cette partie, nous présentons les résultats de l'ajustement de faisceaux appliqué aux données issues de la méthode des triangles.

La figure 2.7 montre la distribution des erreurs de reprojection moyennes avant et après l'AF pour une séquence de 200 images. Les erreurs de reprojection ont considérablement baissé après l'AF. La figure 2.8 illustre les erreurs de reprojection des points 3D dans cinq caméras d'une fenêtre glissante. Les erreurs de reprojection sont représentées dans une image de la fenêtre par des couleurs différentes pour chaque caméra.



(a) (b)

FIGURE 2.8 – Exemples d'erreurs de reprojection des points 3D avant (a) et après (b) l'ajustement de faisceaux. Chaque couleur représente les erreurs dans l'une des caméras de la fenêtre glissante (5 images).

Nous calculons les rapports de la même façon (équations (2.22) et (2.23)) avec les facteurs d'échelle estimés par la méthode des triangles avant et après l'AF. Les résultats sont résumés dans la table 2.2. Les trajectoires obtenues sont représentées dans la figure 2.9. Cette figure montre que la trajectoire optimisée par la méthode proposée avec AF est plus proche de la vérité terrain que la trajectoire initialement estimée par la méthode des triangles.

TABLE 2.2 – Rapports avant et après un AF appliqué sur l'estimation initiale par la méthode des triangles.

	échelle 1	échelle 2	échelle 3	échelle 4
Avant AF	0.9516	0.944478	0.9555	0.9500
Après AF	1.037	1.002	1.0286	1.0609

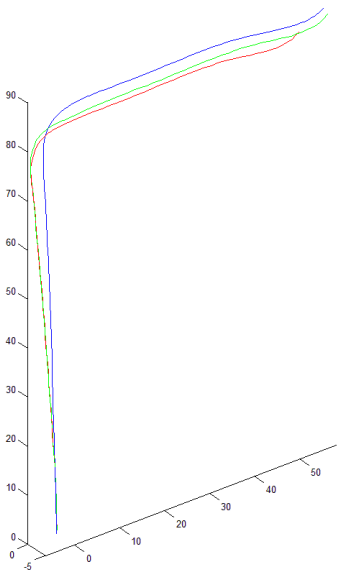


FIGURE 2.9 – Trajectoires obtenues à partir de 200 images : estimation initiale par la méthode des triangles en rouge, vérité terrain en bleu, méthode des triangles après AF en vert.

Évaluation qualitative de notre méthode sur un système plus complexe

Pour démontrer que notre méthode peut être généralisée à un système comportant plus de caméras, nous avons développé un système composé de 5 caméras rigidement liées et montées sur le toit d'un véhicule.

Les caméras utilisées pour mener ces expérimentations sont des caméras Basler Ace 1600 (GigE, CCD 1/1.8", 1624x1234). Ces caméras sont équipées de lentilles de focale 6mm, ce qui donne un champ de vue de 60°. L'ensemble du système couvre un champ de vue de 180° à l'avant du véhicule puisque chaque paire de caméra a une zone de recouvrement d'une demi-image. La

FIGURE 2.10 – Exemples d'images obtenues par le réseau de caméras non-synchronisé.

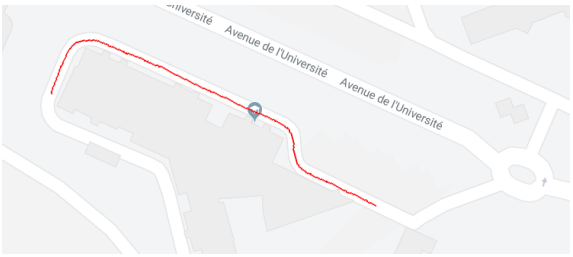


FIGURE 2.11 – La trajectoire estimée lors de l'expérimentation sur des données réelles.

plateforme expérimentale a été embarquée sur un véhicule et un jeu de données a été collecté pendant un parcours d'une distance d'environ 350 mètres autour de notre campus.

La figure 2.10 montre quelques images capturées pendant l'expérimentation. Pour chaque nouvelle image, nous commençons par la vérification des transformations possibles. Les primitives sont extraites par FAST et décrites par BRIEF et les transformations relatives ont été estimées en utilisant l'algorithme des 5 points. Une transformation relative est valide quand le nombre d'appariements obtenus est plus grand qu'un seuil fixé à 50. Les triangles possibles sont alors vérifiés à chaque nouvelle pose.

La méthode des triangles suivie d'un ajustement de faisceaux est appliquée dans chaque triangle pour calculer les facteurs d'échelle absolue λ_1, λ_2, α et β et obtenir les poses des caméras et la structure 3D à l'échelle. La trajectoire reconstruite sur cet itinéraire et représentée sur la figure 2.11 montre qualitativement des résultats intéressants puisqu'elle correspond bien à la route, aussi bien dans les portions en ligne droite que dans les virages, et avec une échelle correcte.

Discussion sur les expérimentations

Nous avons présenté des résultats quantitatifs et qualitatifs pour notre méthode. Il est légitime de se demander pourquoi ces résultats n'ont pas été comparés aux méthodes de l'état de l'art. Comme nous l'avons mentionné dans la section 2.2.1, il n'y a, à notre connaissance, aucun travail qui a été mené sur les systèmes asynchrones. Notre méthode ne peut donc pas être comparée aux autres pour plusieurs raisons.

En premier lieu, il n'est pas possible de comparer notre algorithme avec les méthodes monoculaires puisque dans ce cas, le facteur d'échelle ne peut pas être estimé en utilisant les images uniquement. Il s'agit là de l'intérêt principal de notre méthode : estimer les facteurs d'échelle sans synchronisation entre les caméras. Nous pourrions l'obtenir de la vérité terrain, mais dans ce cas il est évident que notre méthode serait moins précise puisque nous comparerions notre méthode à cette même vérité terrain.

Il ne serait pas plus juste de comparer notre algorithme aux systèmes stéréoscopiques synchronisés puisque dans notre méthode, nous avons une difficulté supplémentaire introduite par la non-synchronisation. Pour résoudre ce problème, nous avons posé des hypothèses et nos résultats seraient logiquement inférieurs à ces méthodes.

Enfin, nous sommes conscients que les résultats obtenus ne sont pas aussi bons que ceux des meilleures méthodes de l'état de l'art, mais cela est dû au fait que nous nous sommes posés un problème différent. En effet, toutes les méthodes de la catégorie "odométrie" classées sur le site de la base de données KITTI sont soit des méthodes basées lidar, soit des méthodes utilisant de la stéréovision synchronisée. Nous proposons dans ces travaux d'étudier ce qu'il est possible de faire quand les caméras ne sont pas synchronisées, ce qui ajoute des difficultés supplémentaires et par conséquent les résultats sont inévitablement moins bons que ceux des méthodes décrites précédemment.

2.3 Localisation visuelle multimodale visible/infrarouge

La reconnaissance à long terme des lieux dans les environnements extérieurs reste un défi en raison des changements d'apparence dans l'environnement. Le problème devient encore plus difficile lorsque la correspondance entre deux scènes doit être faite avec des informations provenant de différentes sources visuelles, en particulier avec des gammes spectrales différentes. Par exemple, une caméra infrarouge est utile pour la vision nocturne en combinaison avec une caméra visible. Dans ces travaux, nous insistons sur notre travail de test des extracteurs de points caractéristiques habituels sous deux contraintes : répétabilité à travers les gammes spectrales et apparence à long terme. Nous développons une nouvelle méthode d'extraction de caractéristiques dédiée à l'amélioration de la répétabilité à travers les gammes spectrales. Nous procédons à une évaluation de la robustesse des caractéristiques sur des jeux de données à long terme provenant de différentes sources d'imagerie (optique, taille des capteurs et gammes spectrales) avec une approche "*Bag-of-Words*". Les tests que nous effectuons démontrent que notre méthode apporte une amélioration significative sur le problème de recherche d'image par le contenu dans un contexte de reconnaissance visuelle de lieu, en particulier quand il est nécessaire d'associer des images de différentes gammes spectrales telles que l'infrarouge et le visible.

2.3.1 État de l'art

Les approches classiques de localisation, basées uniquement sur des données visuelles, utilisent généralement les mêmes outils que les approches de recherche d'images par le contenu : dans un premier temps, des caractéristiques sont extraites des images, puis un algorithme identifie les données les plus discriminantes (principalement invariantes aux changements tels que l'illumination), et enfin les données discriminantes sont comparées sur la base d'une métrique choisie (par exemple la distance L_1). Les caractéristiques utilisées pour la localisation visuelle peuvent être classées en deux catégories principales : les descripteurs d'image globaux et les caractéristiques locales [15].

Images et représentations intermédiaires

Plusieurs méthodes émergentes dépendent directement des données brutes de la caméra. En effet, les capteurs visuels sont particulièrement sensibles aux changements d'apparence à haute

dynamique dans les environnements extérieurs (changements dus à l'illumination de la scène, à la présence d'ombres, etc). Une stratégie pertinente consiste à effectuer une étape de prétraitement sur les images brutes appelée transformation invariante à l'illumination avant de calculer une description globale [53]. D'autres auteurs choisissent également des descriptions globales des images, appelées signatures d'images, et comparent directement ces représentations compactes [54] [55]. Ces méthodes donnent des résultats intéressants mais exigent que l'image requête et les images stockées en mémoire partagent un point de vue très proche.

Les caractéristiques locales, ou caractéristiques ponctuelles, sont largement utilisées dans les applications de vision par ordinateur. Certains comparatifs sur les caractéristiques locales pour des applications générales ont déjà été produits, par exemple par [56] ou plus récemment par [57] qui met l'accent sur l'évaluation des caractéristiques binaires. Par rapport aux descripteurs d'image globaux, les caractéristiques ponctuelles présentent certains avantages. Elles peuvent en effet être utilisées ultérieurement pour une estimation de la pose relative entre deux caméras ou pour retrouver la structure 3D de la scène par triangulation. Cependant, la taille des données extraites des images dépend du nombre des points caractéristiques détectés alors que les signatures d'image globales ont une taille fixe.

Une récente étude classe les techniques de "*Bag of Words*" (BoW) comme une troisième catégorie [58]. Ces sacs de mots visuels sont l'une des approches les plus communes en robotique. L'approche BoW est le résultat d'une analogie avec les mots en recherche de textes par le contenu. Chaque image est représentée de la même façon qu'un document contenant des mots (les caractéristiques). L'ensemble des mots visuels, ou statistiques obtenues à partir de cet ensemble, est donc codé en tant que représentation intermédiaire de l'image, ce qui est parfois appelé caractéristiques *mid-level* [59]. Une requête sur un ensemble de données peut être exécutée avec un ou plusieurs mots et un classement est établi en fonction de leur pouvoir discriminant dans les documents. Ces techniques sont globalement efficaces, mais leur complexité peut être un fardeau pour un système embarqué ou pour des exigences de calcul en temps-réel.

Depuis quelques années, l'évolution des capacités matérielles, notamment les technologies GPU, a incité à la reprise des travaux sur les réseaux neuronaux. La communauté de vision par ordinateur montre un intérêt croissant pour les caractéristiques apprises grâce aux améliorations apportées aux réseaux de neurones profonds et à leurs variantes convolutives. Les premières approches d'apprentissage profond (*Deep Learning*) sur les images ont tendance à utiliser les images complètes en entrée et à laisser le système apprendre par lui-même un modèle empirique. Les approches d'apprentissage profond ont donné des résultats pertinents sur des problèmes classiques de vision par ordinateur. Par exemple, les approches présentées dans [60] [61] [62] surpassent les méthodes mentionnées précédemment. D'autres auteurs substituent uniquement une partie de leur processus de vision par ordinateur : par exemple, dans [63], l'auteur remplace uniquement le calcul des descripteurs et le calcul de la métrique par un réseau de neurones profond.

Néanmoins, l'apprentissage profond souffre d'inconvénients non négligeables : les réseaux neuronaux convolutionnels sont extrêmement gourmands et nécessitent d'énormes jeux de données pour réussir le processus d'apprentissage, ainsi que des ressources informatiques massives. De plus, ces approches peuvent être vues comme des "boîtes noires" et la théorie sous-jacente est un domaine de recherche toujours très actif.

Caractéristiques et multimodalité

Une autre source de problèmes peut surgir en raison de la diversité des capteurs qui conduit à des images différentes d'une même scène. Différents capteurs visuels ont été utilisés pour le SLAM visuel (V-SLAM) dans la littérature. Par exemple, monoSLAM décrit dans [64] propose l'utilisation d'une seule caméra. Dans [65], les auteurs ont développé une méthode similaire avec des capteurs infrarouges. Ce changement de capteur soulève un autre problème connu par la communauté sous le nom de multimodalité. En plus de la nécessité d'une invariance aux changements d'apparence décrits ci-dessus, le processus d'appariement doit être étendu pour faire face à des données provenant de capteurs ayant des gammes spectrales différentes. Le traitement d'images multimodales est bien documenté dans le domaine de la recherche pour l'imagerie médicale et aérienne, mais il est encore récent dans le domaine de la robotique.

Dédiée à l'asservissement visuel, la méthode décrite dans [66] est un bon exemple qui met l'accent sur l'information mutuelle sur des images entières pour la mise en correspondance d'images et le suivi. Dans [67], les auteurs utilisent des images multimodales au cœur d'un processus de SLAM. [68] propose également une approche multimodale impliquant des systèmes d'imagerie visible et thermique et revendique ses performances en plein jour comme de nuit. Néanmoins, ces deux méthodes utilisent simultanément deux plages spectrales, c'est-à-dire qu'ils utilisent à tout moment les mêmes informations provenant d'un ensemble spécifique de capteurs. Dans ces travaux, nous supposons que le jeu de capteurs peut changer avec le temps ou entre différents agents et, par conséquent, que le processus de mise en correspondance d'images doit être tolérant à ces changements.

Certaines autres méthodes traitent également de la multimodalité mais au lieu d'adapter des algorithmes issus de l'imagerie médicale, ils utilisent des descripteurs standards tels que SIFT ou SURF habituellement utilisés en vision par ordinateur : Ricaurte, par exemple, évalue l'efficacité des points caractéristiques à la fois sur les spectres visible et infrarouge [69]. D'autres auteurs apportent des modifications supplémentaires aux descripteurs habituels et affirment que ces nouveaux descripteurs sont plus robustes pour la mise en correspondance multimodale [70] [71].

Supervision de la reconnaisance visuelle de lieu

Le processus de recherche d'images est généralement inclus dans un algorithme de décision global dans le contexte d'un système SLAM : si le sous-système de vision suggère des associations d'images erronées, celles-ci peuvent être rectifiées par une position estimée dans une carte et par les acquisitions suivantes du capteur. [15] donne des détails sur les cartes et les processus de décision généralement utilisés. Les travaux présentés ici ne mettent l'accent que sur le processus de mise en correspondance d'images sans supervision globale.

La publication la plus récente et probablement la plus proche de nos travaux est [63] : dans cet article, les auteurs prennent la décision de substituer la description des points caractéristiques et les étapes d'évaluation métrique par différents réseaux neuronaux convolutionnels (CNN). Ils montrent que l'un des CNN proposés donne des résultats légèrement meilleurs que les méthodes traditionnelles face à la multimodalité. Cependant, ils gèrent la détection de points caractéristiques avec différents algorithmes, SIFT [72] et FAST [43], et les divers paramètres sont réglés manuellement. Ils concluent qu'il s'agit toujours d'un problème ouvert. De plus, leur CNN a été entraîné et évalué sur des jeux de données dont les paires d'images ont été rectifiées (même résolution et même point de vue). Dans nos travaux, le point de vue de

la même scène peut varier. De plus, nous évaluons notre approche sur des jeux de données avec des changements d'apparence majeurs afin d'estimer sa robustesse face aux changements à long terme. Nous justifions l'utilisation du détecteur de caractéristiques de Harris et proposons un nouveau descripteur de caractéristiques qui améliore les résultats expérimentaux de la reconnaissance visuelle de lieu et à son extension à la multimodalité. Par rapport à la plupart des articles qui conduisent des tests sur deux modalités (et parfois trois), nous proposons une comparaison entre le visible, le proche infrarouge (NIR, *near infrared*), l'infrarouge à courte longueur d'onde (SWIR, *short-wavelength infrared*) et l'infrarouge à grande longueur d'onde (LWIR, *long-wavelength infrared*).

2.3.2 Méthodologie

Une des méthodes les plus courantes en robotique permettant d'estimer la pose d'une caméra est le calcul de la matrice fondamentale [21]. À l'aide d'une approche de type RANSAC, ce calcul permet d'éliminer les faux appariements. La figure 2.12 est un exemple d'appariements restant après extraction de points SIFT et calcul de la matrice fondamentale entre une image visible et une image infrarouge. La différence d'apparence entre les deux modalités spectrales est telle que les descriptions de mêmes points de l'espace sont peu semblables, si bien que l'algorithme ne converge pas vers une solution satisfaisante. Cette constatation nous a amenés à nous tourner plutôt vers des approches développées dans le domaine de la recherche d'image par le contenu (*content-based image retrieval*).

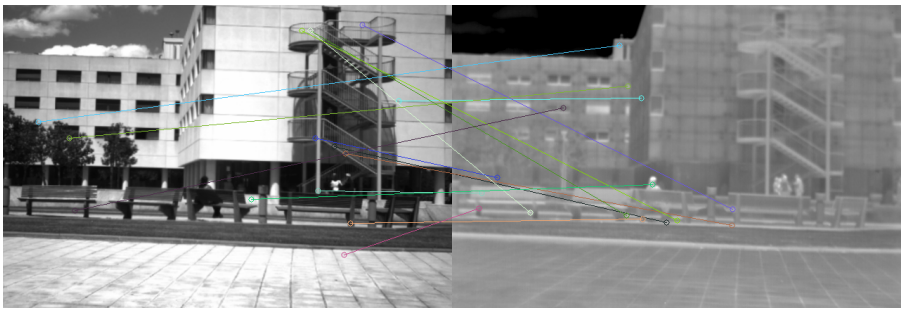


FIGURE 2.12 – Calcul de la matrice fondamentale à l'aide de points SIFT extraits. Les nombreux faux appariements ne permettent pas à l'algorithme de converger vers un résultat cohérent.

Nous détaillons notre travail dans les parties suivantes. Nous mettons tout d'abord en évidence le processus d'extraction de caractéristiques et ses sous-étapes. Ensuite, nous nous concentrons sur l'étude de deux approches principales pour détecter les points d'intérêt dans les images. Nous montrons que les détecteurs de coins sont plus adaptés aux jeux de données multimodaux que des méthodes telles que *Differences-of-Gaussians*. Nous en tirons des conclusions pertinentes et proposons une nouvelle approche pour décrire le voisinage des points d'intérêts que nous appelons PHROG (*Plural Histograms of Restricted Oriented Gradients*). Enfin, dans la section 2.3.3, nous montrons comment appliquer PHROG au problème de *place recognition*. La vue d'ensemble de notre méthode est présentée sur la figure 2.13.

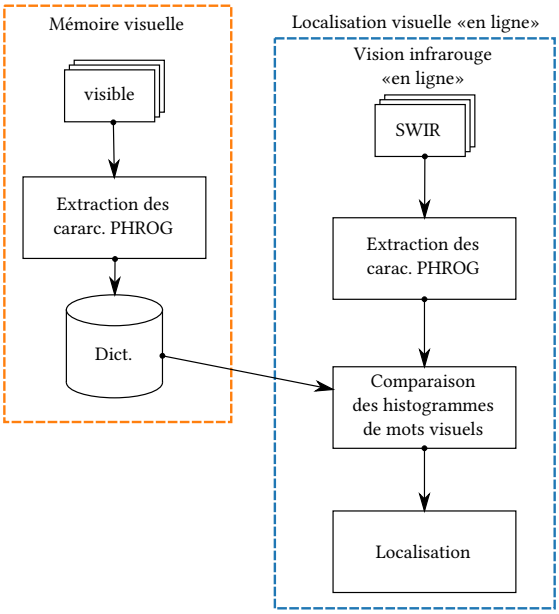


FIGURE 2.13 – Vue globale de la méthode proposée : les caractéristiques PHROG sont extraites de chaque image d'une séquence que l'on considère comme la mémoire. Un dictionnaire est déterminé par les centres des clusters calculés. Chaque image issue d'une nouvelle séquence que l'on nomme "en ligne" est comparée par la suite successivement avec chaque image de la mémoire en fonction de leurs histogrammes de mots visuels.

Extraction des caractéristiques : détection et description

Une méthode usuelle pour extraire des informations pertinentes à partir d'images consiste à choisir un type d'entité particulier ; les entités peuvent être des points, des régions, des arêtes ou des lignes droites par exemple. Les détecteurs de points d'intérêt sont les plus courants dans les applications de SLAM ou de SfM, principalement parce qu'ils permettent un calcul plus poussé des relations géométriques entre plusieurs images ou une reconstruction 3D éparse de l'environnement. De nombreuses méthodes différentes d'extracteurs de caractéristiques ont été proposées dans la littérature. Le processus est généralement divisé en deux parties : la détection et la description du point caractéristique. Le lecteur peut se référer à deux recensements des détecteurs [73] [74], une étude sur les descripteurs locaux [56] et une étude comparative sur les descripteurs binaires [57] pour obtenir des détails pratiques sur les différents algorithmes impliqués. Les descripteurs binaires ont été préférés depuis quelques années pour leur vitesse de calcul, mais font face au problème de "l'inversion du gradient" posé par la multi-modalité [70].

Tests préliminaires sur la répétabilité des détecteurs de points d'intérêt

Historiquement, les premiers détecteurs de caractéristiques locales étaient des détecteurs de coins : ils utilisaient le voisinage proche de chaque pixel pour distinguer les coins des pixels non pertinents. Les références [75] [76] calculent la carte de saillance d'une image grâce au calcul de matrices d'auto-corrélation et d'un critère sur leurs valeurs propres. Plus tard, des études ont été menées sur des détecteurs avec la capacité d'estimer une caractéristique et son "échelle" appropriée pour la description afin d'améliorer la reconnaissance des caractéristiques même si le

point de vue a changé. Il en résulte des méthodes telles que SIFT [72] ou SURF [77] qui utilisent respectivement les "Différences de Gaussiennes" ou une formule approximative pour trouver une caractéristique et son échelle. Les caractéristiques sont alors plus proches de "*patches* d'intérêt" que de coins. Ces ensembles de pixels connectés sont généralement appelés *blobs*.

Comme l'auteur le remarque dans [63], les objets apparaissent différemment dans les images infrarouges et les images couleurs. Les objets perdent globalement leurs textures : par exemple, une sérigraphie avec de nombreuses couleurs et formes semble homogène dans les images infrarouges (figure 2.14). Un autre exemple remarquable est celui de la végétation qui apparaît beaucoup plus sombre dans les images visibles que dans le spectre infrarouge (un paysage semble "enneigé" comme dans la figure 2.15).

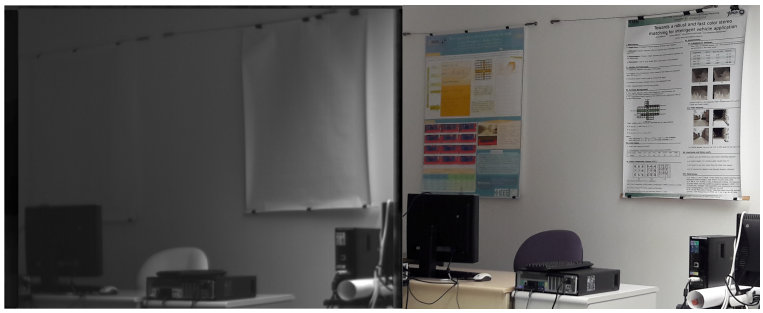


FIGURE 2.14 – Exemple de sérigraphie observée dans le spectre visible et dans le spectre infrarouge : alors que les formes imprimées sont perceptibles dans le visible, le poster semble vierge dans le spectre infrarouge.

FIGURE 2.15 – Une paire infrarouge-visible extraite de notre jeu de données.

Sachant ce qui se passe globalement avec les textures dans différentes plages spectrales, nous avons décidé de valider l'étape de détection des caractéristiques elle-même afin de choisir la meilleure approche de détection, à savoir la détection de coins ou la détection de *patches* d'intérêt. Nous avons pris des paires d'images provenant du jeu de données LWIR-visible introduit dans [78]. Chaque paire a été ajustée afin de rendre les points de vue et les résolutions des deux images (infrarouges et visibles) identiques. Nous avons ensuite exécuté des algorithmes de détection pour chaque modalité et vérifié si les caractéristiques se retrouvent dans les deux modalités. Nous avons concentré nos tests sur le détecteur de coin Harris ainsi que le détecteur "Différences de Gaussiennes" implémentés dans la bibliothèque OpenCV. Nous avons considéré la répétabilité comme critère d'évaluation, qui est donnée, pour chaque paire, comme un rapport entre le nombre de points d'intérêt détectés à la même position dans les deux images,

sur le nombre total de points d'intérêt renvoyés par l'algorithme dans les deux images. Les paramètres de chaque algorithme ont été réglés pour obtenir les meilleurs résultats selon la méthode suivante : nous avons calculé la répétabilité sur une séquence complète avec plusieurs valeurs pour chaque paramètre de détecteur. Les figures 2.16 et 2.17 montrent les résultats obtenus avec le détecteur Harris lors de l'ajustement des paramètres de *niveau de qualité* et de *distance minimale* entre deux points détectés contigus. Le réglage du ratio de qualité peut apporter des améliorations significatives à la répétabilité de certaines paires d'images. Globalement, les résultats sont meilleurs lorsqu'un niveau de qualité faible (égal à $0,0001$) est choisi. La distance minimale entre deux pixels sélectionnés comme caractéristiques est un critère moins déterminant sur la répétabilité et nous avons choisi de garder une distance minimale de 2 pixels afin de détecter moins de points tout en préservant la répétabilité.

Nous définissons une tolérance de deux pixels dans les positions renvoyées par chaque algorithme. La figure 2.18 donne les résultats de répétabilité en fonction de chaque paire du jeu de données visibles-LWIR [78]. Pour chaque paire d'images, la répétabilité est bien meilleure avec une approche de détection de type coin. Nous supposons que ces résultats sont dus au fait que les formes de l'objet sont essentiellement les mêmes dans les deux modalités, alors que les textures ont tendance à différer. La conception des caractéristiques de type coin est plus corrélée aux formes de l'objet que les détecteurs de type *patches*. Par conséquent, nous avons choisi d'utiliser la méthode Harris comme procédure de détection.

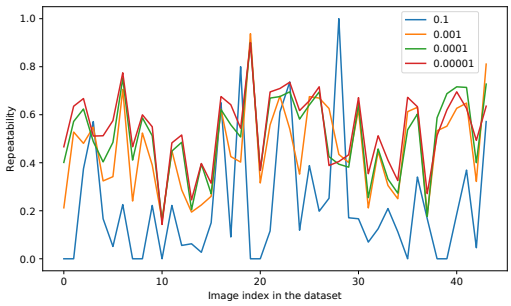


FIGURE 2.16 – Ajustement des paramètres du détecteur : répétabilité des points détectés en fonction des images de la séquence et du paramètre de qualité.

Proposition d'un descripteur ponctuel : PHROG

Motif de description multi-échelle

Compte tenu de la nature du détecteur de Harris, nous n'avons aucune information préalable concernant l'échelle du *patch* de l'image à décrire. Par conséquent, nous nous sommes résolus à établir plusieurs descriptions à différents niveaux d'échelle. Nous avons appelé cette approche PHROG : *Plural Histograms of Restricted Oriented Gradients*. De manière analogue aux caractéristiques invariantes à l'échelle comme SIFT, nous calculons une "Pyramide de Gaussiennes", le premier niveau d'échelle étant l'image originale. Pour composer les niveaux suivants, nous convoluons l'image source avec un noyau Gaussien de 5×5 pixels pour lisser et enlever ses composantes à hautes fréquences. Nous sélectionnons ensuite l'image résultante en prenant un pixel sur deux selon les deux axes x et y. Nous faisons de nouvelles itérations comme celle-ci en fonction du nombre de niveaux d'échelle désiré. Au cours de nos expérimentations, comme

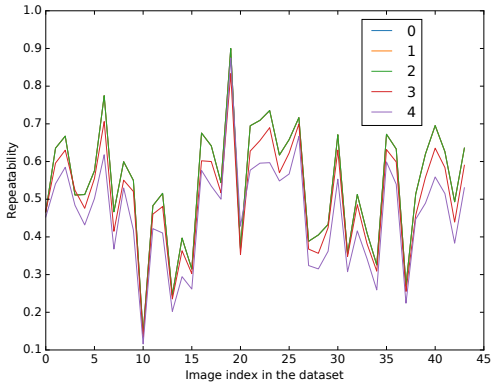


FIGURE 2.17 – Ajustement des paramètres du détecteur : répétabilité des points détectés en fonction des images de la séquence et de la distance minimale (en pixels) entre deux points détectés.

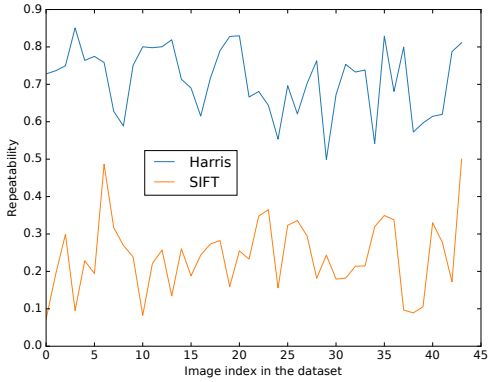


FIGURE 2.18 – Répétabilité des détecteurs Harris et SIFT pour chaque paire d'images du jeu de données visible-infrarouge lointain (LWIR).

nous avons obtenu des images à partir de capteurs ayant des résolutions différentes et avec des points de vue variés, nous avons évalué l'efficacité de PHROG avec plusieurs niveaux d'échelle de description différents. Nous utilisons les courbes *Precision-Recall* et leur AUC (*Area Under the Curve*, aire sous la courbe) comme une évaluation de la performance de PHROG avec différents paramètres d'échelle. Nous donnons dans la figure 2.19 un exemple des résultats obtenus sur le jeu de données VPrice (qui est présenté dans Section 2.3.3). Nous constatons que 5 niveaux de description sont appropriés car le calcul des niveaux supplémentaires n'entraîne aucun avantage significatif sur les résultats correspondants et constitue un bon compromis pour le coût mémoire de notre proposition.

Histogramme des gradients orientés réduits

Sur chaque niveau, nous composons un descripteur inspiré par le motif utilisé dans SIFT (voir figure 2.20) : nous considérons un voisinage de 4×4 zones de 4×4 pixels. Les zones centrales se chevauchent de sorte que le pixel correspondant au coin détecté (la position centrale de la caractéristique) est inclus dans 4 zones et chaque pixel ayant la même abscisse ou la même ordonnée appartient à 2 zones. Contrairement au descripteur SIFT, nous n'effectuons aucune pondération supplémentaire sur la valeur d'intensité des pixels car, compte tenu de la zone de

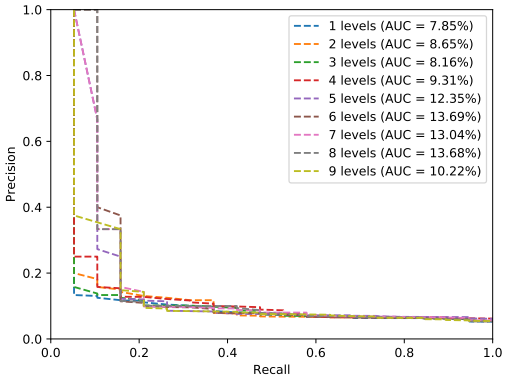


FIGURE 2.19 – Courbes Precision-Recall et leur AUC (aire sous la courbe) en fonction du nombre de niveaux de description utilisés dans PHROG, appliqué sur le jeu de données VPRiCE.

chevauchement, les informations des pixels centraux sont déjà considérées deux fois par rapport aux autres pixels (et même quatre fois pour le pixel central).

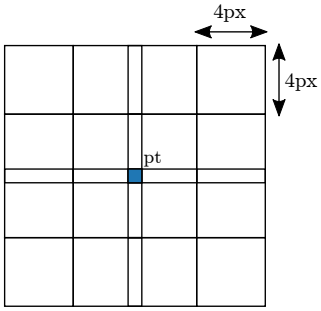


FIGURE 2.20 – Motif utilisé pour la phase de description. Celui-ci définit les zones à extraire autour du point d'intérêt.

Pour chaque zone donnée, nous calculons un histogramme de gradients orientés de taille N. Le problème des "gradients inversés" dû à l'imagerie multispectrale est expliqué dans [70] : un matériau particulier a des propriétés de réflectance différentes selon les plages spectrales, de sorte que deux matériaux différents peuvent avoir des réponses variables lorsqu'une scène est observée avec plusieurs capteurs. En particulier, une zone de contraste élevé dans une image au niveau des bords d'un objet composé d'un matériau donné devant un autre objet avec un matériau différent peut apparaître comme son propre négatif avec un autre capteur : les pixels blancs dans la première image apparaissent comme noirs dans la seconde et *vice-versa*. Si nous calculons un descripteur de type HOG sur les deux images, les gradients auront la même orientation et à peu près la même norme mais une direction opposée.

Afin de gérer les inversions de gradient, nous avons restreint un descripteur HOG traditionnel à une demi-taille dont les directions de gradient opposées sont additionnées (équations (2.24) et (2.25)). Ce concept est illustré par la figure 2.21 : de cette façon, nous conservons dans le descripteur les informations d'orientation de gradient sans information de direction.

$$h_i = \sum_k \alpha_{\theta_k} r_k \tag{2.24}$$

$$\alpha_{\theta_k} = \begin{cases} 1 & \text{si } \theta_k \in [\frac{i}{N}\pi, \frac{i+1}{N}\pi] \cup [\frac{i}{N}\pi + \pi, \frac{i+1}{N}\pi + \pi] \\ 0 & \text{sinon} \end{cases} \tag{2.25}$$

avec N, le nombre choisi d'intervalles dans l'histogramme, h_i le $i^{\text{ème}}$ intervalle, θ_k l'orientation du gradient au pixel k et r_k l'amplitude du gradient au pixel k. Le coefficient α_{θ_k} est égal à 1 lorsque l'orientation du gradient au pixel k est inclue dans l'intervalle $[\frac{i}{N}\pi, \frac{i+1}{N}\pi]$ ou dans sa direction opposée (en $[\frac{i}{N}\pi + \pi, \frac{i+1}{N}\pi + \pi]$).

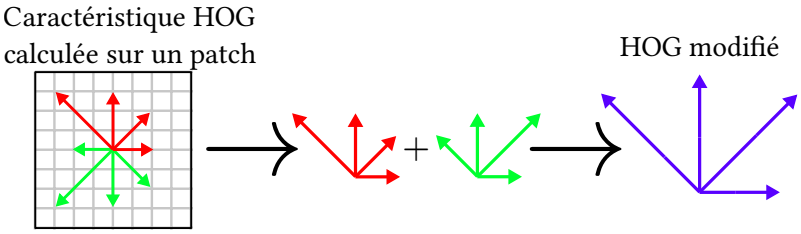


FIGURE 2.21 – Représentation schématique de la modification d'une description HOG de manière à la rendre invariante aux inversions du sens des gradients : les gradients de même direction mais de sens opposés sont additionnés. La taille des vecteurs de description est ainsi deux fois plus petite que celle des descriptions originales.

Application du noyau de Hellinger sur les descripteurs

Il a été prouvé dans plusieurs études que l'utilisation de la distance Euclidienne n'est pas la meilleure pratique pour comparer les caractéristiques qui portent des informations sous forme d'histogrammes. Pour ces cas particuliers, χ^2 ou la métrique de Hellinger sont de meilleurs choix. Cette considération faite, les auteurs de [79] proposent quelques modifications sur le descripteur SIFT et l'appellent RootSIFT. L'idée derrière cette évolution est que la comparaison des descripteurs RootSIFT avec l'aide de la distance Euclidienne est la même que l'application du noyau Hellinger sur les descripteurs SIFT originaux. Nous appliquons le même processus sur nos histogrammes de gradients : soient $\mathbf{h}_1$ et $\mathbf{h}_2$ deux vecteurs unitaires selon la norme euclidienne ($\|\mathbf{h}_i\|_2 = 1$), leur distance Euclidienne est donnée par les équations (2.26) et (2.27) :

$$d_{Eucl}(\mathbf{h}_1, \mathbf{h}_2) = \|\mathbf{h}_1 - \mathbf{h}_2\|_2 = \sqrt{\|\mathbf{h}_1\|_2^2 + \|\mathbf{h}_2\|_2^2 - 2{\mathbf{h}_1}^T\mathbf{h}_2}, \tag{2.26}$$

$$d_{Eucl}(\mathbf{h}_1, \mathbf{h}_2) = \sqrt{2 - 2K_{Eucl}(\mathbf{h}_1, \mathbf{h}_2)}, \tag{2.27}$$

où $K_{Eucl}(\mathbf{h}_1, \mathbf{h}_2) = {\mathbf{h}_1}^T\mathbf{h}_2$, est le noyau Euclidien (ou similitude). Nous souhaitons remplacer cette similitude par le noyau de Hellinger donné dans l'équation (2.28) :

$$K_{Hell}(\mathbf{h}_1, \mathbf{h}_2) = \sum_{j=1}^{N} \sqrt{h_{1j}h_{2j}}, \tag{2.28}$$

pour $\mathbf{h}_1$ et $\mathbf{h}_2$ deux histogrammes normalisés selon $L1$ ($\sum_{j=1}^{N} h_{ij} = 1$ et $h_{ij} \geq 0$). Un moyen simple de calculer la similitude de Hellinger sur les descripteurs est de normaliser les vecteurs d'histogrammes et de passer à la racine carrée chaque élément des histogrammes. Ainsi, $K_{Eucl}(\sqrt{\mathbf{h}_1}, \sqrt{\mathbf{h}_2}) = \sqrt{\mathbf{h}_1}^T\sqrt{\mathbf{h}_2} = K_{Hell}(\mathbf{h}_1, \mathbf{h}_2)$ et l'utilisation de la distance Euclidienne sur ces descripteurs modifiés équivaut à utiliser la similitude de Hellinger sur les descripteurs initiaux.

Requêtes par "Sac-de-mots"

Une stratégie courante utilisée pour la recherche d'images par le contenu est de calculer un "sac-de-mots" (*Bag-of-Words, BoW*) comme expliqué dans [80]. Les données utilisées sont divisées en deux ensembles : le premier compose la mémoire et le second est appelé séquence *live* ou "en ligne" car cette information est généralement acquise progressivement pendant le processus de localisation. La recherche *via* une approche *BoW* tire profit d'une étape de prétraitement appliquée sur les images de la partie mémoire de l'ensemble des données. Un schéma global du processus *Bag-of-Words* est présenté dans la figure 2.22. Toutes les caractéristiques locales sont d'abord extraites de toutes les images de la mémoire. Un algorithme *K-means* sépare alors l'espace entier des descripteurs selon K clusters (de 1000 à 8000 clusters selon les différents cas de test). L'ensemble des descripteurs moyens de chaque cluster est décrit comme le *vocabulaire*. Tous les descripteurs sont ensuite quantifiés par rapport à ce vocabulaire. Chaque *mot* du vocabulaire est ensuite pondéré par un score *TF-IDF* (*Term Frequency-Inverse Document Frequency*). Le score TF-IDF est le produit de deux termes :

— $\mathrm{tf}_{i,j}$ (*Term Frequency* ou la "fréquence") est défini comme :

$$\mathrm{tf}_{i,j} \triangleq \frac{n_{i,j}}{\sum_{k=1}^{|I|} n_{k,j}}, \tag{2.29}$$

où $n_{i,j}$ est le nombre d'occurrences du mot d'index i dans le dictionnaire, dans l'image d'index j de la séquence d'images composant la mémoire, et $|I|$ est le nombre total d'images dans le *corpus*.

— idf_i (*Inverse Document Frequency* ou "fréquence inverse de document") est défini par :

$$\mathrm{idf}_i \triangleq \log \frac{|I|}{|\{i_j : w_i \in i_j\}|}, \tag{2.30}$$

où $|I|$ est le nombre total d'images dans le corpus et $|\{i_j : w_i \in i_j\}|$ est le nombre d'images dans lesquelles apparaissent le mot d'index w_i.

Ainsi, les scores $\text{tf-idf}_{i,j} = \mathrm{tf}_{i,j} \times \mathrm{idf}_i$ permettent de spécifier les mots qui sont les plus pertinents à la fois dans une image donnée de la mémoire et dans toute la base de données. Pendant le traitement de la séquence "en-ligne", nous extrayons des caractéristiques de chaque image et les quantifions par rapport au dictionnaire calculé avec la séquence en mémoire. Les mots les plus pertinents trouvés dans l'image courante décrivent l'image la plus proche présente en mémoire.

2.3.3 Application de PHROG au problème de *place recognition*

Comme nous l'avons mentionné précédemment, plusieurs travaux et développements ont été réalisés pour faire face au problème de la reconnaissance visuelle de lieu à long terme. Ces travaux se concentrent sur deux approches principales : la première concerne la description de l'image elle-même et propose des études pour améliorer l'appariement d'images une à une. L'autre approche considère une séquence et non pas une image seule. Ainsi, la cohérence temporelle permet d'envisager un filtrage afin d'éliminer de potentiels appariements erronés. Les travaux présentés dans cette partie ne se concentrent que sur la première approche. Notre objectif ici est d'améliorer l'appariement de deux images de la même scène lorsque des capteurs avec différentes plages spectrales sont utilisés. Afin d'évaluer notre méthode par rapport aux méthodes bien

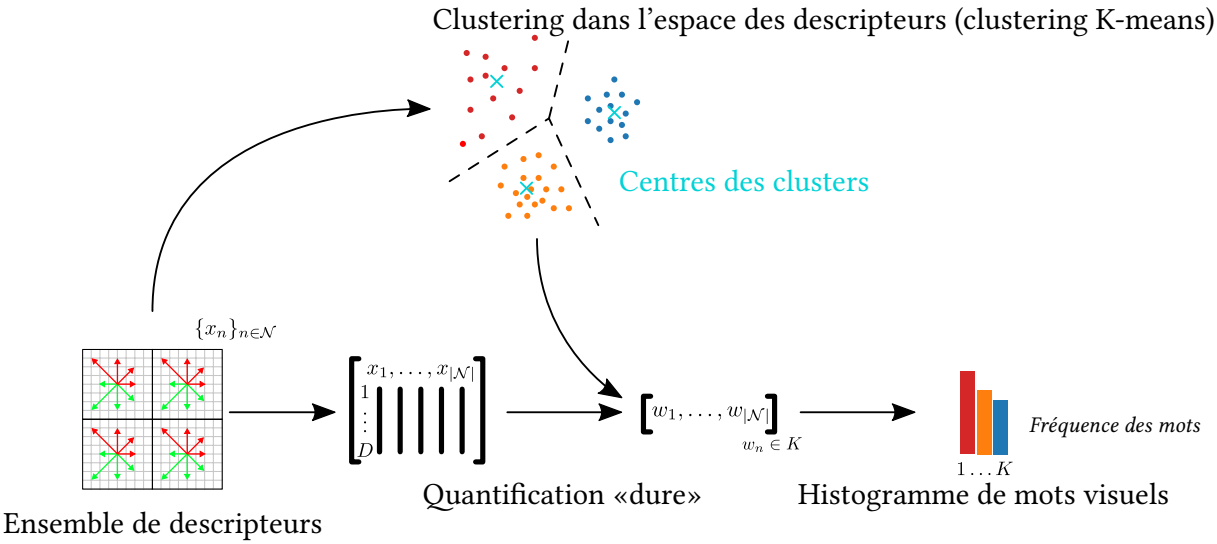


FIGURE 2.22 – Représentation graphique d'une approche *Bag-of-Words* : un ensemble d'apprentissage est d'abord utilisé afin d'agréger les caractéristiques extraites dans l'espace de description. Les caractéristiques sont ensuite quantifiées selon le *cluster* le plus proche. La représentation résultante des images est un "histogramme de mots visuels".

connues de la littérature, nous utilisons plusieurs jeux de données. Certains proviennent de travaux récents disponibles dans la communauté [81], [82], [63]. Nous avons également réalisé notre propre jeu de données pour évaluer les différentes techniques face à des contraintes et des difficultés accrues. Pour chaque cas de test, nous prenons une seule modalité à partir d'un jeu de données (visible ou infrarouge) et construisons un dictionnaire hors ligne. Nous désignons ce sous-ensemble comme "mémoire". Après quelques expérimentations, nous avons déterminé que les dictionnaires composés de 1000 mots sont un bon compromis entre efficacité et vitesse de calcul. Nous désignons l'autre sous-ensemble comme séquence "en ligne" (ou *live*). Il est composé des images de l'autre modalité composant le jeu de données. En d'autres termes, pour chaque cas de test, le dictionnaire est construit grâce à une seule modalité et l'étape de requête est toujours effectuée avec une autre modalité. Nous essayons de faire correspondre chaque image composant le sous-ensemble dit *live* en trouvant l'image la plus proche dans la mémoire en fonction des histogrammes de mots. Si l'image de la mémoire retournée par l'algorithme et l'image *live* proviennent de la même paire, nous considérons le test comme un "vrai positif", sinon comme un "faux positif". Nous expérimentons cette méthode sur les cas de test présentés dans les parties suivantes. Nous donnons pour chacun le ratio entre le nombre de vrais positifs et le nombre total d'images *live* du jeu de données. Nous montrons également avec la figure 2.23 un exemple où notre algorithme proposé échoue en raison de l'*aliasing* dominant entre les deux scènes (forme du chemin de fer, bâtiment sur le côté droit, arrière-plan avec des montagnes, *etc*).

Configuration matérielle des expérimentations

Nous avons mené les expérimentations sur un ordinateur de bureau exécutant *Ubuntu 16.04 LTS* avec un processeur *Intel Core i7* et 8 Go de RAM. Nous avons limité à 10000 le nombre de points caractéristiques détectés dans chaque image. Nous avons donné des détails sur le motif et

FIGURE 2.23 – Un exemple d'appariement erroné lorsque l'algorithme proposé échoue. On remarque que la confusion (*aliasing*) entre la requête (image de gauche) et l'image de la mémoire retournée par l'algorithme est forte.

la méthode de description par PHROG précédemment. Un vecteur de description PHROG est donc 2 fois plus petit (64 scalaires) que celui de SIFT. Avec de tels paramètres, le calcul d'un dictionnaire de 1000 mots visuels dure entre une heure et deux heures selon le jeu de données considéré.

Expérimentations sur des images visibles et infrarouge proche

Le premier jeu de données provient de l'EPFL (*École Polytechnique Fédérale de Lausanne* en Suisse) et est présenté dans [81]. Il est composé de plusieurs sous-groupes triés : *country*, *field*, *forest*, *indoor*, *mountain*, *oldbuilding*, *street*, *urban* et *water*. Chaque sous-ensemble est composé d'environ 50 paires d'images. Chaque paire comprend une image visible et une image correspondante dans l'infrarouge proche (NIR). Les images de chaque paire ont été corrigées par les auteurs afin que les points de vue et les résolutions soient strictement identiques. Un exemple de paire est donné dans la figure 2.24. Nous choisissons de nous concentrer sur les sous-ensembles *urban*, *street* et *country* qui sont les plus proches des cas d'utilisation de la robotique et de la navigation. Le sous-ensemble *country* a effectivement été utilisé dans [63] comme l'ensemble d'apprentissage pour l'ensemble des expérimentations. Nous choisissons alternativement les images NIR et l'ensemble visible comme mémoire et comme séquence *live*.

Les résultats sont donnés dans la table 2.3 et les figures 2.25 et 2.27 pour chaque sous-ensemble et associations détecteur-descripteur, et nous montrons un exemple de matrice de confusion obtenue sur l'ensemble de données *urban* avec PHROG (figure 2.28). Nous pouvons voir facilement que la diagonale de la matrice de confusion présente les distances les plus faibles entre les images : les jeux de données ont été synchronisés afin de calculer les performances de manière simple. Cela nous permet de considérer que les images associées à la diagonale sont les bons appariements et que les autres sont des correspondances fausses. Cette configuration des matrices de confusion rend possible le calcul des courbes *Precision-Recall* (*PR*) et leur AUC.

Comme on peut le voir, les images NIR ne sont pas si différentes des images visibles. Il semble que la recherche d'images par le contenu avec des fonctions traditionnelles donne de très bons taux de bons appariements (voir les jeux de données *urban* et *street*). Néanmoins, les résultats sur le jeu de données données *country* sont plus disparates : une explication possible est que la végétation est beaucoup plus présente dans ce jeu de données et, par conséquent, le problème d'inversion du sens des gradients apparaît plus fréquemment qu'avec des bâtiments et matières inorganiques. Les courbes PR montrent que PHROG est le meilleur sur les jeux de données *urban*

FIGURE 2.24 – Une paire visible-infrarouge proche issue du jeu de l'EPFL.

	Jeux de données					
	Urban		**Street**		**Country**	
SIFT-SIFT	96%	94%	78%	66%	40%	34%
SIFT-GISIFT	98%	94%	70%	64%	34%	32%
FAST-SIFT	100%	100%	96%	96%	75%	76%
Harris-SIFT	100%	98%	88%	92%	73%	61%
Harris-GISIFT	100%	100%	90%	90%	69%	67%
Harris-PHROG	100%	100%	96%	94%	73%	80%

TABLE 2.3 – Taux de bons appariements sur le jeu visible-infrarouge proche.

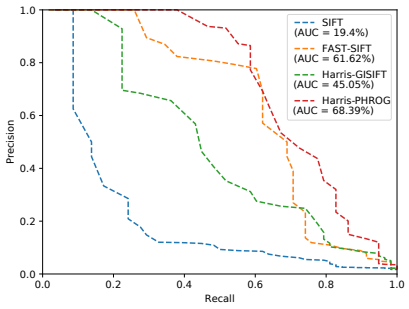


FIGURE 2.25 – Courbes Precision-Recall et leur AUC respective obtenues pour chaque méthode sur le jeu de données *urban*.

et *street* : son AUC est plus élevée d'au moins 9% que les autres méthodes évaluées. Sur le jeu de données *country*, PHROG n'obtient pas la meilleure AUC, mais sa précision reste la meilleure lorsque le rappel est faible. Ce résultat est significatif car cela veut dire que si le processus de recherche renvoie un seul résultat, PHROG donne la meilleure réponse.

Expérimentations sur des images visible et infrarouge lointain

Cette partie considère des images infrarouges provenant d'une gamme spectrale beaucoup plus éloignée du spectre visible que la précédente. Nous avons utilisé le jeu de données infrarouge-visible introduit dans [63] et [78]. Cette gamme spectrale est appréciée pour sa réponse thermique et son utilisation possible en tant que système de vision nocturne. Ce jeu de données englobe des scènes en extérieur du campus de Barcelone. Les images ont également été corrigées par leurs

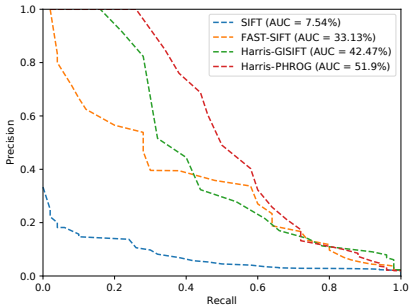


FIGURE 2.26 – Courbes Precision-Recall et leur AUC respective obtenues pour chaque méthode sur le jeu de données *street*.

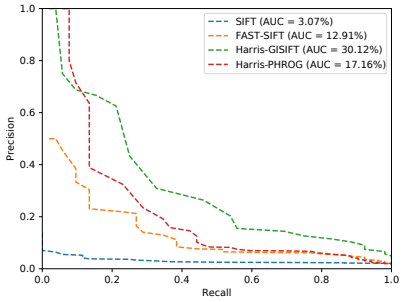


FIGURE 2.27 – Courbes Precision-Recall et leur AUC respective obtenues pour chaque méthode sur le jeu de données *country*.

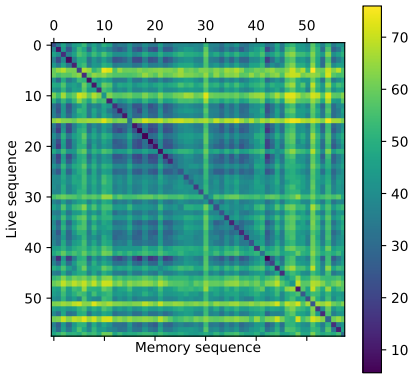


FIGURE 2.28 – Matrice de confusion entre la séquence mémoire et la séquence *live* obtenue avec PHROG. Les valeurs dans la matrice correspondent aux distances calculées pour chaque paire d'images possible.

auteurs afin que les résolutions et les points de vue soient identiques. Un exemple est donné sur la figure 2.29.

Nous expérimentons deux situations en changeant la modalité utilisée comme mémoire. Les résultats sont résumés dans la table 2.4 et la figure 2.30 présente les courbes PR. Nous pouvons constater que le ratio d'appariements corrects est légèrement meilleur lorsque le sous-ensemble LWIR est utilisé comme source pour la composition du dictionnaire. Nous supposons que les

images moins texturées améliorent le calcul du dictionnaire de mots visuels. En outre, notre descripteur PHROG démontre sa valeur sur ce jeu de données. PHROG donne des résultats nettement meilleurs que les autres méthodes avec exactement les mêmes paramètres que sur le jeu de données EPFL. Cependant, nous pouvons noter que les AUC sont très basses (inférieures à 5%), même si la précision de PHROG est bonne lorsque le rappel est faible. Cela signifie que les distances entre les images pour les correspondances vraies et fausses sont très proches l'une de l'autre mais assez discriminantes lorsqu'il est nécessaire de ne récupérer qu'une seule image. Nous remarquons ainsi que plus les plages spectrales sont éloignées, plus il est difficile d'apparier les images avec des méthodes courantes de l'état de l'art.

FIGURE 2.29 – Une paire visible-infrarouge lointain issue du jeu de l'université de Barcelone.

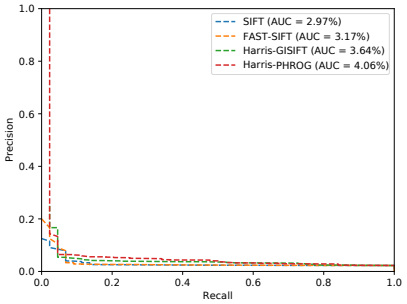


FIGURE 2.30 – Courbes Precision-Recall et leur AUC respective obtenues pour chaque méthode sur le jeu de données de l'université de Barcelone.

	Modalité utilisée en mémoire	
Méthodes	**LWIR**	**Visible**
SIFT-SIFT	9%	9%
SIFT-GISIFT	11%	9%
FAST-SIFT	22%	9%
Harris-SIFT	18%	20%
Harris-GISIFT	52%	38%
Harris-PHROG	61%	56%

TABLE 2.4 – Taux de bons appariements sur le jeu de l'université de Barcelone.

Expérimentations sur le jeu de données VPRiCE

Dans cette partie, nous utilisons des images sélectionnées parmi le jeu de données VPRiCE [4]. Un exemple d'images est donné dans la figure 2.31. Il a été conçu pour évaluer la reconnaissance de lieu à long terme. Les deux séquences de ce jeu de données ont été prises en utilisant des capteurs sensibles dans le spectre visible, mais à deux moments différents, de sorte que les changements saisonniers sont prépondérants. Par rapport aux deux jeux de données précédents, les images ne sont pas rectifiées de sorte que deux images de la même paire ont un point de vue différent. Nous appliquons le même protocole d'évaluation sans approche séquentielle sur ce jeu de données et obtenons les résultats présentés dans la table 2.5 et la figure 2.32.

FIGURE 2.31 – Une paire d'images issue du jeu de données VPRiCE.

Méthodes	**Efficacité**
SIFT-SIFT	36%
SIFT-GISIFT	42%
FAST-SIFT	68%
Harris-SIFT	52%
Harris-GISIFT	47%
Harris-PHROG	73%

TABLE 2.5 – Taux de bons appariements sur le jeu de données VPRiCE.

Même si ce jeu de données est très difficile à appréhender, la caractéristique PHROG présente un appariement faux une fois sur quatre et surpasse les autres méthodes que nous avons testées. PHROG présente la meilleure AUC et la meilleure précision lorsque le rappel est faible.

Expérimentations sur le jeu de données visible-SWIR

Le dernier cas de test est réalisé sur notre propre jeu de données. Il a été réalisé avec une caméra visible et un capteur SWIR. Ce jeu de données est beaucoup plus difficile car la résolution et les points de vue des images ne sont pas identiques. De plus, un sous-ensemble a été acquis plusieurs mois plus tard, de sorte que l'aspect de la végétation est très différent (avec et sans feuilles). La figure 2.33 est tirée de ce jeu de données. L'ensemble SWIR, avec la plus petite résolution, a été utilisé comme mémoire dans un premier temps, puis les deux modalités ont été échangées. Les résultats sont présentés dans la table 2.6 et la figure 2.34. Notre méthode montre

4. https ://roboticvision.atlassian.net/wiki/pages/viewpage.action ?pageId=14188617

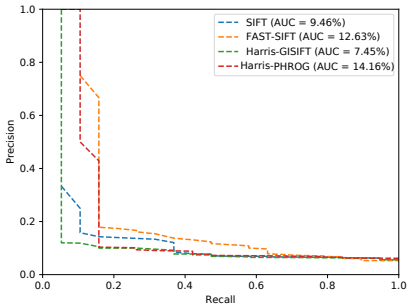


FIGURE 2.32 – Courbes Precision-Recall et leur AUC respective obtenues pour chaque méthode sur le jeu de données VPRiCE.

encore de bons résultats par rapport aux descripteurs habituels. Une remarque notable doit être faite concernant le choix de la mémoire : les résultats sont meilleurs lorsque le SWIR compose la mémoire plutôt que l'inverse. Nous supposons que la faible résolution et les images bruitées de la caméra SWIR conduisent à des descripteurs moins discriminants et à des représentations plus générales qu'avec l'ensemble visible. Néanmoins, ces résultats sont médiocres en termes absolus, les AUC sont très faibles et aucune méthode n'obtient de résultats significatifs si on considère uniquement les courbes Précision-Rappel. Nous avons effectué les tests avec cette caméra SWIR afin d'éprouver notre méthode dans une situation très contrainte. De toute évidence, cette caméra n'est pas le meilleur choix à faire lors de la conception d'un système embarqué, en raison de sa faible résolution, du bruit généré sur les images et de son faible intérêt pour les situations de faible luminosité.

	Modalité utilisée en mémoire	
Méthodes	**SWIR**	**Visible**
SIFT-SIFT	15%	5%
SIFT-GISIFT	20%	10%
FAST-SIFT	15%	20%
Harris-SIFT	15%	10%
Harris-GISIFT	25%	15%
Harris-PHROG	35%	15%

TABLE 2.6 – Taux de bons appariements sur notre jeu de données (multimodal à long terme).

2.4 Architectures embarquées pour la localisation basée vision

L'objectif général de ces travaux est de concevoir une architecture embarquée pour la localisation du véhicule autonome basée sur la vision. Pour atteindre cet objectif, nous avons dans un premier temps étudié les facteurs permettant d'obtenir une accélération rapide de deux algorithmes classiques d'estimation de l'odométrie visuelle. Nous avons choisi une architecture ayant un faible coût et largement disponible sur la plupart des ordinateurs : le GPU-CUDA [5] (*Graphics Processing Unit - Compute Unified Device Architecture*). Ce choix a été motivé par

5. https ://www.nvidia.fr/object/gpu-programming-fr.html

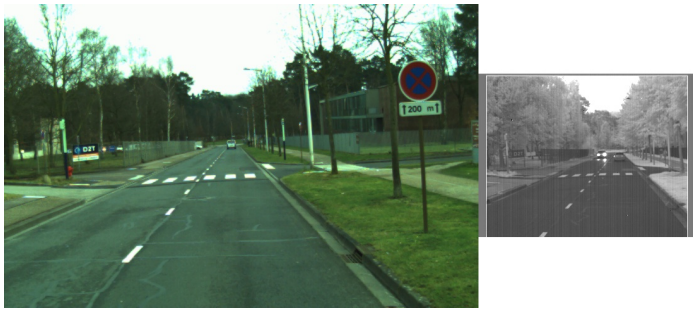


FIGURE 2.33 – Une paire visible-SWIR issue de notre jeu de données.

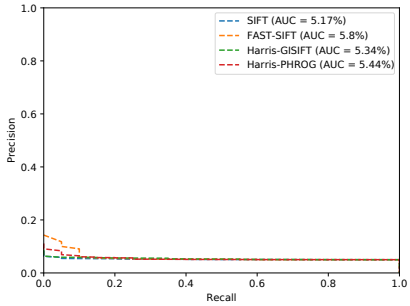


FIGURE 2.34 – Courbes Precision-Recall et leur AUC respective obtenues pour chaque méthode sur notre jeu de données.

le fait que les GPU modernes surpassent généralement les processeurs haut de gamme sur des tâches adaptées à leur parallélisme. Les algorithmes d'odométrie visuelle sont des problèmes non-graphiques [6] pour lequel il n'est pas toujours possible de dérouler les opérations en parallèle. Nos travaux ont été de repenser ces implémentations séquentielles pour qu'elles soient exécutées sur GPU avec le meilleur gain de temps possible. De la même façon, nous avons également étudié la problématique de la reconnaissance de lieu pour porter un algorithme de *place recognition* fonctionnant dans des conditions diverses (de jour ou de nuit, par temps ensoleillé ou pluvieux) sur une architecture GPU-CUDA.

2.4.1 Estimation du mouvement relatif basé CUDA à partir de correspondances 2D-2D

La complexité de la mise en œuvre des algorithmes existants sur GPU dépend fortement du flux de traitement de l'algorithme. En effet, certains algorithmes (tels que les traitements d'images "classiques") peuvent être classés comme *parallèlement embarrassants* lorsque peu d'efforts sont requis pour diviser le processus en tâches parallèles, et sont souvent facilement portés sur GPU [83]. En revanche, d'autres algorithmes sont *intrinsèquement séquentiels*, ce qui implique des difficultés d'ordonnancement qui interdisent la parallélisation d'une part, et augmentent considérablement l'effort requis pour implémenter une solution CUDA efficace d'autre part. Par conséquent, le développement d'une implémentation parallèle efficace dépend forte-

6. Dans le sens où il ne s'agit pas de traitement d'images effectués sur tous les pixels mais plutôt d'algorithmes de Vision par Ordinateur, d'estimation robuste, etc.

ment de l'algorithme d'entrée.

Les algorithmes d'estimation du mouvement relatif s'exécutent généralement dans un RANSAC pour traiter le problème des valeurs aberrantes. Notre approche de parallélisation est basée sur l'exécution des itérations RANSAC requises en parallèle sur CUDA pour atteindre une certaine accélération. Ce niveau de parallélisme suggère la prise en compte des itérations RANSAC comme un lot de calculs parallèles, chacun traitant un petit sous-ensemble de données.

Nous avons porté plusieurs variantes d'algorithmes d'estimation de la matrice essentielle sur GPU : l'algorithme des 8-points [84], des 7-points [21] et celui des 5-points [85]. Notre principale contribution porte sur ce dernier algorithme, notamment sur la variante proposée par Stewenius [85], car il s'agit d'un algorithme complexe à cause de l'utilisation d'une base de Gröbner. Le portage d'un tel algorithme sur GPU n'est pas chose aisée, car en plus de sa complexité, certains calculs d'algèbre linéaire ne sont présents dans CUDA : calcul de valeurs singulières, calcul de matrices échelonnées en ligne, extraction de racines d'un polynôme, etc. Nous avons donc dû trouver des stratégies d'implémentation et des variantes d'algorithmes pour arriver à nos fins. Les détails de mise en œuvre de l'estimation de la pose relative à partir de 5, 7 et 8 correspondances en utilisant le modèle de programmation CUDA peuvent être trouvés dans [86] et [87]. Dans cette section, nous ne présentons que les résultats qui ont été obtenus.

Résultats expérimentaux

Dans cette partie, nous évaluons la vitesse d'éxécution de nos algorithmes basés sur un RANSAC et implémentés sur CUDA, et nous les comparons aux versions CPU fournies dans la librairie OpenGV [88].

Pour générer des données synthétiques, nous avons utilisé le *benchmark* automatique pour l'estimation de poses inclus dans cette bibliothèque. Nous avons utilisé des poses relatives créées aléatoirement. Cet outil fournit des observations à partir de deux points de vue, ainsi que les valeurs de la vérité terrain, c'est-à-dire les paramètres de la transformation. Nous avons utilisé un ensemble de 1000 points mis en correspondance, ce qui correspond au nombre moyen de correspondances obtenues à partir d'images réelles.

Nous avons mesuré les temps de calcul moyens à la fois sur CPU et sur GPU. Pour calculer ces temps moyens, chaque estimation est répétée 20 fois. Nous ne présentons que les résultats pour les calculs en simple précision (32 bits) car la perte de précision de ce type n'affecte pas l'estimation de la localisation dans un scénario réel. De plus, les algorithmes d'odométrie visuelle sont généralement couplés à des processus d'optimisation pour réduire la dérive causée par les accumulations d'erreur. Le système sur lequel est exécuté le code est équipé d'un CPU Intel Core i7 à 3.5GHz. Le GPU est une carte NVIDIA GeForce GTX 850M fonctionnant à 876 MHz, avec 4096MB de mémoire GDDR5. L'évaluation a été exécutée avec CUDA version 7.5 intégré à VisualStudio 2012.

A la première exécution de l'estimation, des allocations mémoires doivent être réalisées. Cela n'est nécessaire qu'une fois et prend environ 6ms. Pour évaluer notre implémentation, dix taux d'*outliers* sont évalués, de $\epsilon = 0.05$ à $\epsilon = 0.5$ par pas de $\epsilon = 0.05$. La figure 2.35 représente le nombre d'itérations requises (cf équation (3.1)) pour le calcul de la matrice essentielle à partir de 5, 7 et 8 correspondances et pour les dix taux d'*outliers*.

La figure 2.36, montre les résultats de performance pour l'estimation de la pose relative de la caméra à partir d'un jeu de 5 mises en correspondance. Premièrement, la figure 2.36.a compare le temps de calcul moyen des implémentations CPU et GPU. Nous montrons un temps de calcul

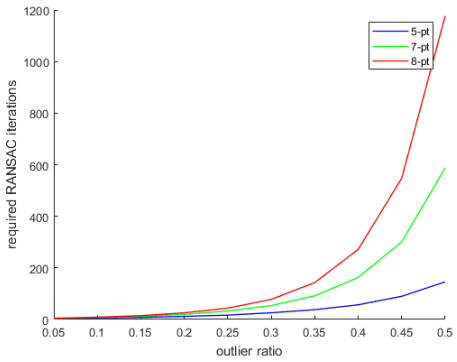
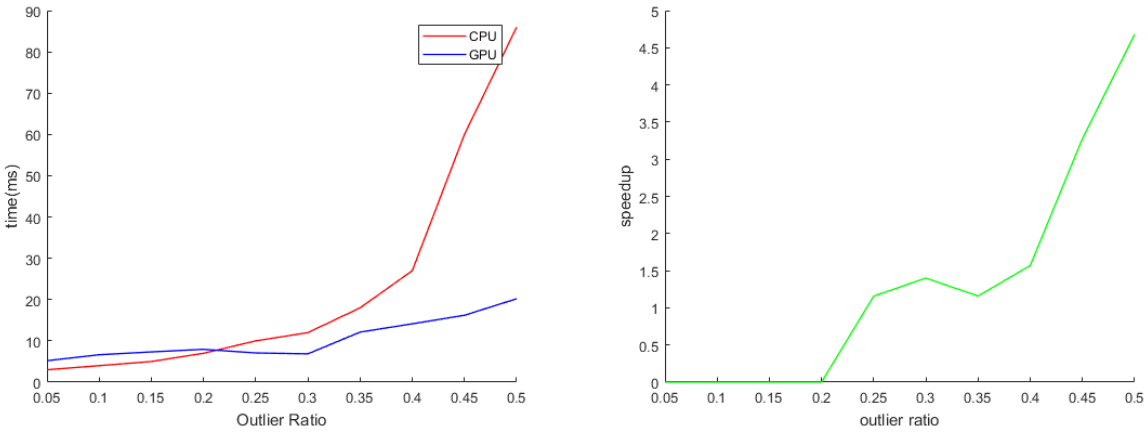


FIGURE 2.35 – Nombre d'itérations RANSAC en fonction du taux d'*outliers*.

bien supérieur en CPU, atteignant 86ms pour un ratio d'*outliers* de $\epsilon = 0.5$ contre 18ms pour le GPU. Avec un taux d'outliers de $\epsilon = 0.5$, ce qui est assez courant pour le calcul de matrices essentielles à partir de mises en correspondance automatiques de points, nous obtenons une accélération d'un facteur supérieur à 4 par rapport à l'implémentation CPU.

En outre, il est intéressant de visualiser l'intersection des courbes entre les versions CPU et GPU, c'est-à-dire de trouver le pourcentage d'*outliers* pour lequel le rapport de vitesse est égal à un. La figure 2.36 montre qu'il n'y a pas d'accélération pour les taux d'*outliers* inférieurs à $\epsilon \leq 0.2$.



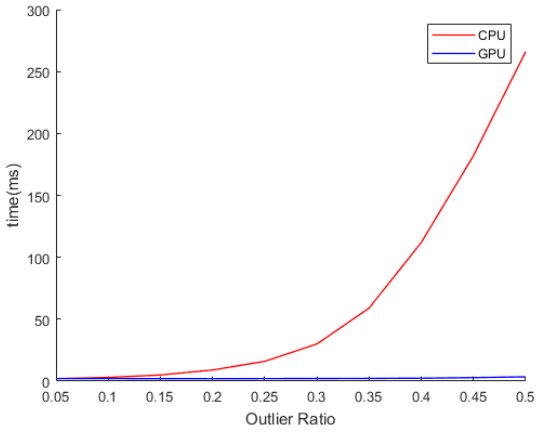
(a) Temps de calcul.

(b) Accélération.

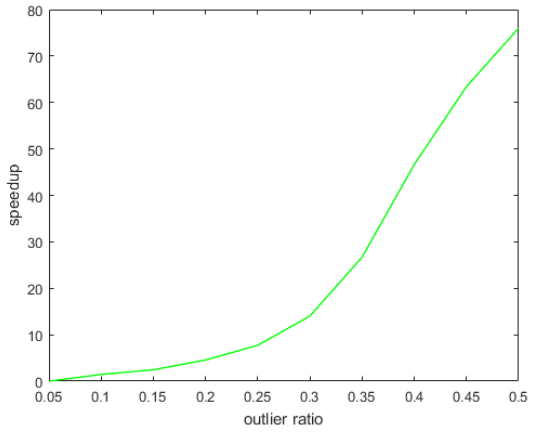
FIGURE 2.36 – Performance de l'estimation de la matrice essentielle avec un RANSAC basé sur 5 mises en correspondance.

Dans la figure 2.37, nous montrons les résultats de l'estimation de la pose relative de la caméra à partir d'ensembles de sept correspondances. Dans la figure 2.37.a, nous comparons le temps de calcul moyen des implémentations CPU et GPU. Nous obtenons un temps de calcul moyen pour le processeur pouvant atteindre 266ms pour un ratio d'*outliers* $\epsilon = 0,5$ contre 3,5ms pour le GPU, ce qui représente une accélération très importante d'un facteur 76 comme illustré

dans la figure 2.37.



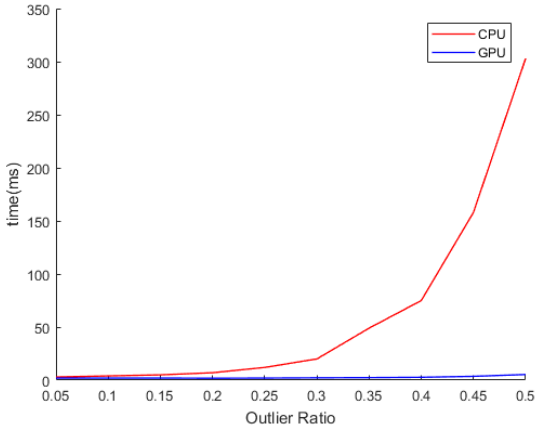
(a) Temps de calcul.



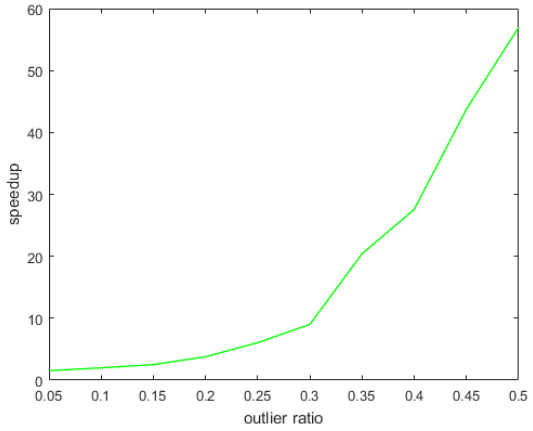
(b) Accélération.

FIGURE 2.37 – Performance de l'estimation de la matrice essentielle avec un RANSAC basé sur 7 mises en correspondance.

La figure 2.38 montre les résultats de l'estimation de la pose relative de la caméra à partir de la mise en correspondance de 8 points. Comme le montre la figure 2.38.a, une accélération de près de 57× est obtenue pour un ratio d'*outliers* de $\epsilon = 0,5$.



(a) Temps de calcul.

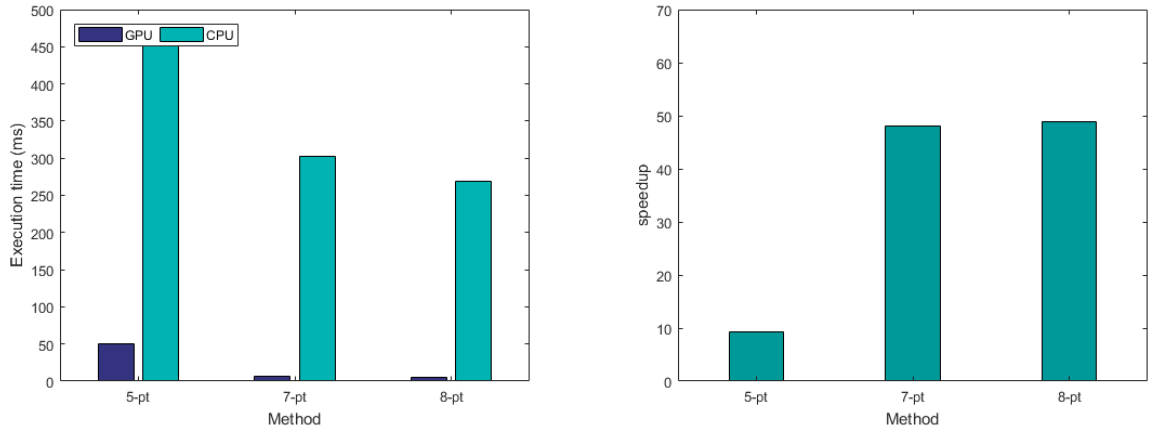


(b) Accélération.

FIGURE 2.38 – Performance de l'estimation de la matrice essentielle avec un RANSAC basé sur 8 mises en correspondance.

Dans une deuxième série d'expérimentations, un nombre fixe d'itérations RANSAC, égal à 1024, est utilisé pour évaluer les performances des trois méthodes d'estimation de l'odométrie visuelle. En effet, un nombre élevé d'itérations, préalablement déterminé, est parfois nécessaire pour estimer la covariance des estimations, comme par exemple dans [89]. Dans la figure 2.39, nous évaluons le temps en ms et l'accélération de nos implémentations basées sur CUDA de l'estimation de la matricielle essentielle à partir de 5, 7 et 8 correspondances. En raison de

la complexité des implémentations, nous ne lançons que 256 *threads* parallèles dans le cas de l'algorithme 5-pt et les noyaux sont donc lancés 4 fois en série par le processeur pour effectuer les 1024 itérations. Dans le cas des algorithmes 7-pt et 8-pt, 512 *threads* sont émis en parallèle et les noyaux sont lancés en série 2 fois. La figure 2.39 montre un temps de calcul important, atteignant 464ms, 303ms et 269ms pour les algorithmes 5-pt, 7-pt et 8-pt respectivement contre 50.34ms, 6.31ms et 5.5ms pour les implémentations basées sur CUDA. Cela permet d'atteindre des accélérations d'un facteur 9 pour l'algorithme 5-pt et presque 48 pour les deux autres algorithmes.



(a) Temps de calcul.

(b) Accélération.

FIGURE 2.39 – Performance de l'estimation de la matrice essentielle avec un RANSAC basé sur 5, 7 et 8 mises en correspondance.

2.4.2 Implémentation d'un algorithme de *place recognition* sur GPU

Après la problématique de l'odométrie visuelle, nous avons également proposé une implémentation parallèle sous CUDA d'un algorithme de reconnaissance de lieu : SeqSLAM [54]. SeqSLAM est une approche bien connue pour la localisation et la reconnaissance visuelle de lieu fonctionnant dans des conditions très variables : changements saisonniers, cycles jour/nuit, etc.

Notre méthode de parallélisation est basée sur l'allocation des trois grandes étapes de cette méthode à trois noyaux GPU, en optimisant les paramètres de chacun d'entre-eux pour obtenir les meilleures performances possibles. Ces trois étapes sont le calcul de la matrice des différences, l'amélioration de son contraste, et le calcul du chemin optimal dans celle-ci.

Les détails de notre implémentation GPU sont donnés dans [90]. Nous présentons dans cette partie les résultats obtenus.

Résultats expérimentaux

Pour les expériences décrites ci-après, nous avons extrait les images de la vidéo originale du jeu de données Nordland [7]. Nous les avons rééchantillonnées à 64×32 et les avons converties en niveaux de gris.

7. `http://nrkbeta.no/2013/01/15/nordlandsbanen-minute-by-minute-season-by-season/`

De plus, toutes les données résident dans la mémoire du GPU au début de chaque test. Il n'y a donc pas de transfert de données vers le processeur pendant les tests afin d'éviter les interactions avec d'autres facteurs de l'étude. La performance des expérimentations pour la version parallèle de SeqSLAM est mesurée en temps t en millisecondes (ms). Le système sur lequel notre implémentation a été évaluée est équipé d'un processeur Intel Core i7 fonctionnant à 3,5 GHz. Le périphérique CUDA est une carte graphique NVIDIA GeForce GTX 850M fonctionnant à 876 MHz avec 4096 Mo de mémoire GDDR5. L'évaluation a été réalisée avec la version CUDA 7.5 intégrée à VisualStudio 2012.

Lors de la première exécution de SeqSLAM, des allocations de mémoire doivent être effectuées. Ceci n'est requis qu'une fois et prend environ 10 ms. Toutes les expérimentations ont été exécutées pour cinq longueurs de séquence de la base de données comme présenté dans la table 2.7. De plus, 3 séquences de requête avec 3 longueurs différentes ont été utilisées comme indiqué dans la table 2.8.

Database	Longueur de la séquence de la base de données
D1	714
D2	1428
D3	2747
D4	3570
D5	5100

TABLE 2.7 – Séquences de la base de données.

Requête	Longueur de la séquence de requête
Q1	11
Q2	20
Q3	32

TABLE 2.8 – Séquences Requêtes.

Calcul de la matrice des différences

Dans la première expérimentation, nous avons mesuré les temps de calcul des matrices des différences en utilisant les séquences de requêtes et de la base de données présentées dans les tables 2.7 et 2.8 respectivement. L'évaluation est donnée sur la figure 2.40, pour une taille de requête de 32 images, montrant une accélération très importante de la version parallèle de SeqSLAM basée sur CUDA en fonction de la taille de la base de données. L'accélération moyenne est égale à un facteur 4.

Renforcement du contraste de la matrice des différences

Nous avons ensuite évalué la performance de l'amélioration du contraste SeqSLAM en parallèle par rapport à l'amélioration du contraste séquentiel. Les performances pour les implémentations séquentielles et parallèles sont présentées dans la figure 2.41 pour une taille de séquence de requête de 32 images. L'accélération obtenue dépasse 16× pour D1 et presque 13× pour D5. Dans l'ensemble, une bonne accélération a été obtenue avec une moyenne de 14×.

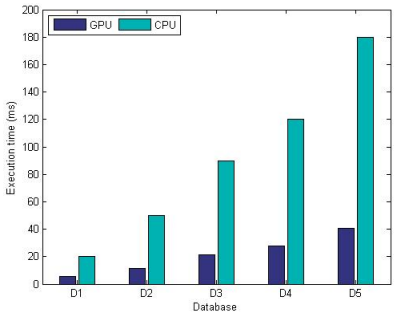


FIGURE 2.40 – Calcul de la matrice des différences.

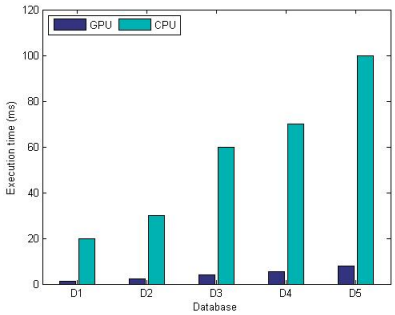


FIGURE 2.41 – Renforcement du contraste de la matrice des différences.

Recherche du chemin optimal

La troisième expérimentation a été consacrée à la comparaison du temps d'exécution de la recherche du chemin optimal pour les implémentations basées sur CPU et CUDA. L'accélération du GPU par rapport au processeur est clairement visible en moyenne à près de 6× pour les jeux de données utilisés dans l'évaluation et est illustrée sur la figure 2.42 pour une taille de séquence de requête de 32 images.

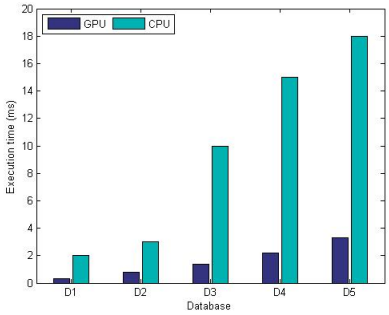
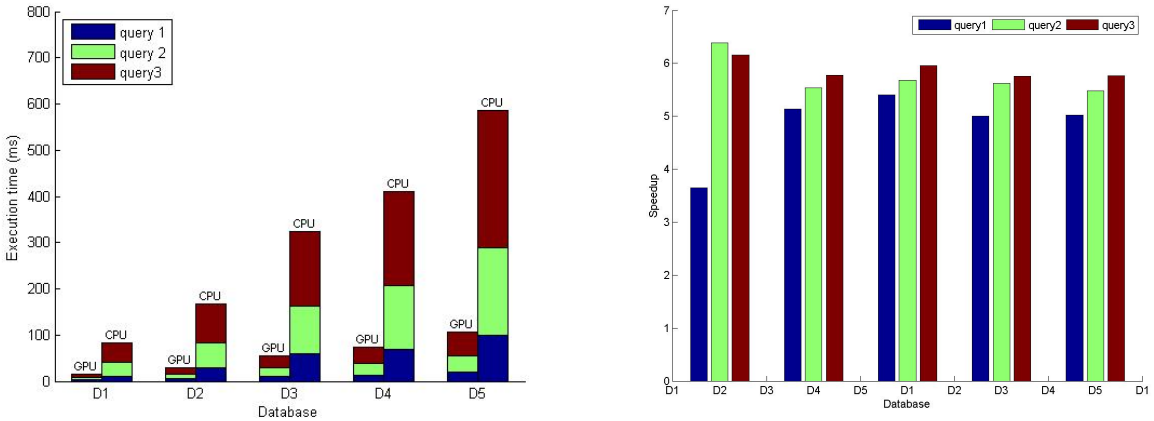


FIGURE 2.42 – Recherche du chemin optimal.

Performance globale de SeqSLAM basé sur CUDA

Dans la figure 2.43, nous montrons la performance de notre implémentation parallèle de Se-

qSLAM en utilisant CUDA. Premièrement, dans la figure 2.43.a, nous comparons les temps d'exécution moyens pour les implémentations CPU et GPU, pour les différentes tailles des séquences requêtes et de la base de données. Le temps de calcul pour la version CPU atteint 298ms pour D5 et Q3, contre 51ms pour le GPU. Dans la figure 2.43, nous avons montré que l'accélération obtenue est d'environ 6× par rapport à l'implémentation CPU.



(a) Temps GPU vs CPU.

(b) Accélération.

FIGURE 2.43 – Performance de SeqSLAM basé sur CUDA.

2.5 Conclusion

Dans ce chapitre, nous avons présenté nos contributions sur la localisation basée vision pour le véhicule autonome. Dans un premier temps, nous avons proposé un nouvel algorithme pour estimer la structure 3D de la scène ainsi que le déplacement à l'échelle d'un réseau de caméras asynchrones. Comme ce type de capteurs est de plus en plus intégré aux véhicules pour des tâches telles que l'assistance au parking, notre but était de démontrer leur utilisation potentielle pour des tâches de plus haut niveau comme la reconstruction 3D autour du véhicule. En posant des hypothèses simples, telles que la linéarité du mouvement entre deux vues consécutives de la même caméra, ainsi que des zones de recouvrement entre deux vues adjacentes, notre méthode permet une estimation simple et rapide de la pose du système à l'échelle. Un ajustement de faisceaux pour optimiser les points 3D et les échelles a également été proposé pour améliorer les estimations initiales sur une fenêtre glissante. Toutes les expérimentations menées sur le jeu de données KITTI et sur notre propre système ont démontré qualitativement et quantitativement la validité de notre approche. Ces travaux, menés dans le cadre de la thèse de Rawia Mhiri [Th1], ont fait l'objet de publications dans [91], [92], [93] et [94].

Pour que les algorithmes de vision puissent fonctionner dans toutes les conditions d'environnement, les véhicules intègrent de plus en plus de caméras aux sensibilités spectrales différentes. Chaque gamme spectrale apporte ses forces et doit donc être choisie en fonction de l'application recherchée. Par exemple, une caméra couleur est adaptée pour la reconnaissance de panneaux routiers, une caméra infrarouge proche pour améliorer la perception en présence de brouillard, une caméra en infrarouge thermique pour la détection de piétons. Nous avons donc mené des travaux visant à développer des méthodes généralisables à plusieurs gammes spectrales, en particu-

lier pour la localisation. Nous avons développé une nouvelle approche de reconnaissance visuelle de lieu entre des données visibles et infrarouges (NIR, SWIR ou LWIR). Nos expérimentations ont été conduites dans des conditions typiquement rencontrées par le véhicule autonome : changements de point de vue ou de résolution, évolution de l'apparence à long terme et variations d'illumination de la scène. Notre méthode a montré des résultats intéressants puisqu'elle surpasse les détecteurs de caractéristiques usuels. Ces travaux ont notamment été menés dans le cadre de la thèse de Fabien Bonardi [Th4] et ont fait l'objet de publications dans [95], [96] et [97].

Une problématique très importante pour l'utilisation d'algorithmes de vision par ordinateur sur des systèmes réels est le temps de calcul. Les algorithmes étant généralement très complexes, il est bien souvent difficile de les faire fonctionner en temps-réel sur des systèmes embarqués. Nous avons donc étudié et proposé, dans le cadre de la thèse de Safa Ouerghi [Th5], des stratégies de parallélisation d'algorithmes d'odométrie visuelle et de reconnaissance de lieu sur des architectures GPU. Cela a nécessité de repenser entièrement les algorithmes originaux et de trouver des solutions pour effectuer certains calculs d'algèbre linéaire qui ne sont pas disponibles sous CUDA. Nous avons obtenu des accélérations d'un facteur 4 à un facteur 76 en fonction des algorithmes utilisés. Ces travaux ont été publiés dans [98], [86], [90], [89] et [87].

Chapitre 3

Solutions Minimales pour l'odométrie visuelle et l'estimation de pose

Sommaire

3.1 Introduction et contexte 76
3.2 Odométrie Visuelle connaissant une direction commune 77
3.2.1 Méthodes d'estimation du mouvement 78
3.2.2 Résultats expérimentaux 86
3.3 Estimation de l'attitude et de l'altitude d'un drone 94
3.3.1 Formulation du problème 95
3.3.2 Une solution géométrique pour l'estimation de l'altitude et de l'attitude 97
3.3.3 Estimations robustes 100
3.3.4 Calibrage du système 107
3.3.5 Résultats expérimentaux 109
3.4 Conclusion 115

3.1 Introduction et contexte

L'odométrie visuelle consiste à estimer le mouvement d'un mobile (véhicule, humain, robot, etc) à partir d'une ou plusieurs caméras embarquées sur celui-ci [10] [11]. L'odométrie visuelle repose sur l'estimation incrémentale des poses du véhicule en analysant les changements que le mouvement a introduits dans les images des caméras embarquées. L'estimation de pose vise quant à elle à localiser la caméra dans un environnement 3D connu, ou à localiser la caméra par rapport à une entité géométrique particulière. Par exemple, il peut être intéressant d'estimer la position d'une caméra par rapport au plan du sol. Ces deux thématiques - odométrie visuelle et estimation de pose - sont très importantes pour les véhicules autonomes, puisque sans une estimation précise du déplacement ou de la pose du véhicule, il est impossible de planifier ou de suivre une trajectoire.

L'estimation de la pose ou du déplacement est généralement obtenue en exploitant la mise en correspondance - ou appariement - de points d'intérêt entre deux images consécutives. Ces couples de points mis en correspondance sont généralement contaminés par la présence d'*outliers*, c'est-à-dire par des couples de points images considérés comme étant la projection d'un même point 3D alors qu'ils ne le sont pas. Sans une détection et un rejet de ces *outliers*, l'estimation de la pose ou du déplacement devient alors totalement erronée. Pour rejeter ces mauvais appariements, des algorithmes d'estimation robuste sont mis en œuvre, parmi lesquels le RANSAC (RANdom SAmple Consensus) [99] s'est imposé comme le standard pour l'estimation d'un modèle en présence d'*outliers*. Le RANSAC est un algorithme itératif qui permet d'estimer les paramètres d'un modèle à partir d'un échantillon minimal de données tiré aléatoirement, puis de vérifier ces paramètres sur le reste des données. Le tirage aléatoire des données est répété plusieurs fois, et les paramètres du modèle ayant obtenu le plus grand consensus, c'est-à-dire correspondant à la plus grande majorité des données, sont alors sélectionnés comme étant la bonne solution. Le nombre d'itérations N nécessaire pour s'assurer, avec une probabilité p, qu'au moins un des tirages aléatoires ne comportera pas d'*outlier* est donné par :

$$N = \frac{\log(1-p)}{\log(1-(1-\epsilon)^s)}, \tag{3.1}$$

où s est le nombre minimal de points nécessaires pour calculer les paramètres du modèle et ϵ est le pourcentage d'*outliers* dans les données [21]. Habituellement, p est fixé à $p = 0.99$ pour assurer une très forte probabilité de succès. Comme le montrent l'équation (3.1) et la figure 3.1, le nombre d'itérations N est exponentiel par rapport au nombre minimal s de données nécessaire pour trouver les paramètres du modèle. C'est pour cette raison qu'avoir une paramétrisation minimale d'un modèle est très avantageux, à la fois en termes de temps de calcul et de robustesse. A titre d'exemple, avec $p = 0.99$ et $\epsilon = 0.5$, un algorithme utilisant 5 points nécessitera 146 itérations, alors qu'une méthode basée sur l'utilisation de 3 points n'en nécessitera que 35.

Devant les enjeux de robustesse et de temps de calcul de la détermination de solutions minimales, je me suis intéressé à cette problématique aux travers de plusieurs travaux qui ont été réalisés en collaboration avec des chercheurs d'autres laboratoires nationaux et internationaux.

Le premier travail porte sur l'odométrie visuelle d'une caméra basée sur le calcul d'une homographie en connaissant le vecteur de gravité entre deux images calibrées. Obtenir une direction commune avec le vecteur de gravité est très courant dans le cas de systèmes mobiles (smartphones, etc) grâce à l'utilisation d'une centrale inertielle (IMU). Sans IMU, une direction commune peut également être obtenue facilement grâce à l'extraction dans les images de

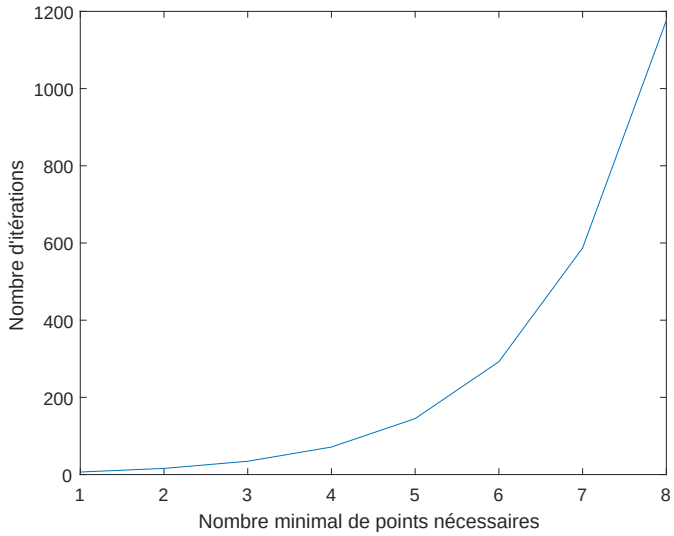


FIGURE 3.1 – Nombre d'itérations nécessaire, pour assurer avec une probabilité $p = 0.99$, qu'au moins un des tirages ne comportera pas d'*outlier* lorsque la proportion d'*outliers* est de 50%.

points de fuite, de l'horizon, etc. Dans ces travaux, nous avons donc proposé différentes solutions minimales dans des environnements structurés (bâtiments, etc) grâce à l'estimation d'une homographie. Ces travaux sont le fruit d'une collaboration internationale (ETH Zürich, LITIS, IRSEEM, LE2I, TU Graz) et ont fait l'objet d'une publication dans [2].

Le second travail sur les solutions minimales porte sur l'estimation de la pose d'un drone basé sur la projection d'un laser circulaire au sol (cf figure 3.13). L'objectif de ce système n'est pas de retrouver la pose totale du drone (l'angle de lacet ne peut pas être estimé) mais obtenir l'attitude et l'altitude du drone par rapport au sol présente un grand intérêt pour la phase d'atterrissage. Un tel système a également l'avantage de fonctionner dans des environnements très sombres, là où une caméra seule, c'est-à-dire sans projection de lumière, aurait échoué. La projection du laser circulaire au sol est une conique, ce qui est un outil mathématique très intéressant en géométrie projective [100]. Nous avons développé une approche géométrique qui permet une estimation à l'échelle de la pose du drone, relative au plan sur lequel la forme est projetée. Trois solutions ont été proposées, incluant deux solutions minimales. Ces travaux sont le fruit d'une collaboration nationale entre quatre laboratoires (INRIA, LITIS, LE2I, IRSEEM) et ont fait l'objet d'une publication dans [3].

Les expérimentations réalisées dans le cadre de ces deux travaux reposent sur les moyens d'essais du CISE présentés dans la section 1.4.1.

3.2 Odométrie Visuelle connaissant une direction commune

Dans cette section, nous explorons différentes solutions minimales pour l'estimation du mouvement d'une caméra basée sur une homographie en ayant la connaissance du vecteur de gravité entre des images calibrées. Ces solutions dépendent des connaissances *a priori* sur le plan de référence utilisé par l'homographie. Nous démontrons ensuite que le nombre de points appariés peut varier entre deux et trois, et qu'une solution analytique directe ou une solution basée sur

les bases de Gröbner peuvent être formulées à partir de ce plan. Les performances de notre approche, en terme de précision et de robustesse, ont été évaluées sur de nombreuses séquences en simulation et sur des données réelles, aussi bien en intérieur qu'en extérieur.

3.2.1 Méthodes d'estimation du mouvement

La connaissance de la direction verticale dans les images simplifie l'estimation du mouvement de la caméra. Il est alors possible d'aligner chaque système de coordonnées associé à la caméra avec la direction verticale mesurée de sorte que l'axe z de la caméra soit parallèle à la direction verticale et que le plan x-y de la caméra soit orthogonal à la direction verticale comme illustré sur la figure 3.2. De plus, cela signifie que le plan x-y de la caméra est maintenant parallèle au plan du sol et que l'axe z est parallèle aux murs verticaux.

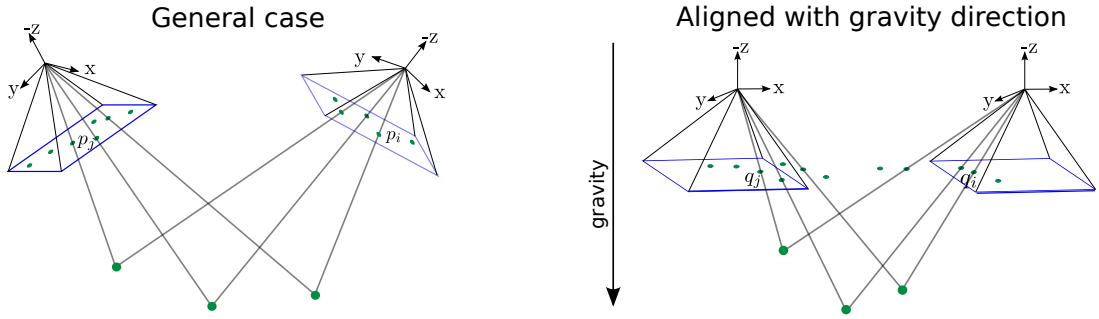


FIGURE 3.2 – Alignement de la caméra avec la direction de la gravité.

Cet alignement peut simplement être effectué comme une transformation des coordonnées pour les algorithmes d'estimation de mouvement, mais il est aussi possible de déformer l'image pour que les méthodes d'extraction de primitives puissent en bénéficier. Le mouvement relatif entre deux caméras alignées se réduit alors à un mouvement à 3 degrés de liberté, consistant en une rotation restante et un vecteur de translation à 2 ddl (c'est-à-dire, un vecteur de translation 3D à un facteur d'échelle près).

Les algorithmes d'estimation de la pose relative sont dérivés d'une homographie, où un plan est observé dans deux images. L'homographie est ensuite décomposée en une rotation et une translation relatives entre les deux images. En incorporant la direction verticale connue, la paramétrisation du problème d'estimation de pose est considérablement réduite de 5 dll à 3 ddl. Cette simplification conduit à un algorithme à 2 points et un algorithme à 2.5 points pour calculer l'homographie. En supprimant l'hypothèse de structures strictement verticales ou horizontales, et en utilisant la direction connue de la gravité, la formulation de l'homographie aboutit à une solution analytique ne nécessitant que 3 points.

Dans les sous-sections suivantes, nous dérivons l'algorithme 2pt pour les cas où les normales des plans sont connues (plan du sol et plan vertical), puis nous présentons des variantes 2.5pt et 3pt pour une direction de gravité connue avec une orientation de plan inconnue.

Estimation de la pose relative à partir de 2 points appartenant au sol

La relation homographique générale pour des points appartenant à un plan 3D et projetés dans deux vues différentes est définie comme suit :

$$\mathbf{q}_j = \mathbf{H}\mathbf{q}_i, \tag{3.2}$$

avec $\mathbf{q}_i = [x_i, y_i, w_i]^\top$ et $\mathbf{q}_j = [x_j, y_j, w_j]^\top$ les coordonnées projectives des points entre les vues i et j. $\mathbf{H}$ est donnée par :

$$\mathbf{H} = \mathbf{R} - \frac{1}{d}\mathbf{t}\mathbf{n}^\top, \tag{3.3}$$

où $\mathbf{R}$ et $\mathbf{t}$ sont respectivement la rotation et la translation entre les vues i et j, et d est la distance entre la caméra i et le plan 3D défini par la normale $\mathbf{n}$.
Dans notre cas, nous supposons que les paramètres intrinsèques de la caméra sont connus et que les points $\mathbf{q}_i$ et $\mathbf{q}_j$ sont normalisés. Nous considérons également que l'attitude des caméras pour les deux vues sont connues et que ces mesures d'attitude ont été utilisées pour aligner le repère de la caméra avec celui du plan du sol. De cette façon, seul l'angle de lacet θ (yaw) entre les deux vues est inconnu. Par conséquent, l'équation (3.3) peut être exprimée comme :

$$\mathbf{H} \;=\; \mathbf{R}_z - \frac{1}{d}\mathbf{t}\mathbf{n}^\top, \tag{3.4}$$

où $\mathbf{R}_z$ est la rotation inconnue due au lacet (autour de l'axe z). De même, puisque nous considérons que le plan du sol est le plan 3D visible pendant le mouvement de la caméra, nous pouvons noter que $\mathbf{n} = [0, 0, 1]^\top$.
Par conséquent, l'équation (3.4) peut être écrite

$$\mathbf{H} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{\mathbf{t}}{d}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^\top, \tag{3.5}$$

d étant inconnue, la translation ne peut être connue qu'à un facteur d'échelle près. Par conséquent, la distance caméra-plan d est fixée à 1 et absorbée par $\mathbf{t}$. Nous obtenons alors :

$$\mathbf{H} \;=\; \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^\top, \tag{3.6}$$

$$\;=\; \begin{bmatrix} \cos(\theta) & -\sin(\theta) & -t_x \\ \sin(\theta) & \cos(\theta) & -t_y \\ 0 & 0 & 1 - t_z \end{bmatrix}. \tag{3.7}$$

De façon générale, l'homographie peut être paramétrisée par

$$\mathbf{H} = \begin{bmatrix} h_1 & -h_2 & h_3 \\ h_2 & h_1 & h_4 \\ 0 & 0 & h_5 \end{bmatrix}. \tag{3.8}$$

Le problème consiste à estimer les cinq éléments de l'homographie $\mathbf{H}$. Nous considérons la relation suivante :

$$\mathbf{q}_j \times \mathbf{H}\mathbf{q}_i = \mathbf{0}, \tag{3.9}$$

où $\times$ est le produit vectoriel. En ré-écrivant cette équation, nous obtenons :

$$\begin{bmatrix} x_j \\ y_j \\ w_j \end{bmatrix} \times \begin{bmatrix} h_1 & -h_2 & h_3 \\ h_2 & h_1 & h_4 \\ 0 & 0 & h_5 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ w_i \end{bmatrix} = \mathbf{0}. \tag{3.10}$$

Cela nous donne trois équations, où deux d'entre-elles sont linéairement indépendantes. Nous développons l'équation ci-dessus et considérons seulement les deux premières équations linéairement indépendantes, ce qui aboutit à :

$$\begin{bmatrix} -w_j y_i h_1 - w_j x_i h_2 - w_i w_j h_4 + w_i y_j h_5 \\ w_j x_i h_1 - w_j y_i h_2 + w_i w_j h_3 - w_i x_j h_5 \end{bmatrix} = \mathbf{0}. \tag{3.11}$$

Le système d'équations peut être ré-écrit comme :

$$\begin{bmatrix} -w_j y_i & -w_j x_i & 0 & -w_i w_j & w_i y_j \\ w_j x_i & -w_j y_i & w_i w_j & 0 & -w_i x_j \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix} = \mathbf{0}. \tag{3.12}$$

Les équations ci-dessus représentent un système d'équations de la forme $\mathbf{Ah} = 0$. Il est important de noter que $\mathbf{A}$ est de rang 4. Puisque chaque correspondance de points donne naissance à deux équations indépendantes, nous avons besoin de deux correspondances de points pour trouver $\mathbf{h}$ à un facteur d'échelle inconnu près. Le vecteur singulier de $\mathbf{A}$ qui a la plus petite valeur singulière couvre un espace de solutions à une dimension (à l'échelle près). Nous choisissons la solution $\mathbf{h}$ telle que $||\mathbf{h}|| = 1$. Ensuite, pour obtenir des paramètres de rotation valides, nous renforçons la contrainte trigonométrique $h_1^2 + h_2^2 = 1$, en divisant le vecteur de solution par $\pm\sqrt{h_1^2 + h_2^2}$. Les paramètres de mouvement de la caméra peuvent être directement déduits de l'homographie :

$$\mathbf{t} = \begin{bmatrix} -h_3, & -h_4, & 1 - h_5 \end{bmatrix}^\top, \tag{3.13}$$

$$\mathbf{R} = \begin{bmatrix} h_1 & -h_2 & 0 \\ h_2 & h_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.14}$$

A cause de l'ambiguïté sur le signe dans $\pm\sqrt{h_1^2 + h_2^2}$, nous obtenons deux solutions possibles pour $\mathbf{R}$ et $\mathbf{t}$.

Estimation de la pose relative à partir de 2 points avec une normale de plan vertical connue

L'algorithme suivant est capable de calculer la pose relative étant données deux correspondances de points et la normale du plan sur lequel sont situés les points. La formulation est décrite pour un plan vertical mais peut fonctionner de façon similaire pour des plans paramétrisés autour d'autres axes.

L'homographie pour un plan vertical peut être donnée par :

$$\mathbf{H} = \mathbf{R}_z - [t_x, t_y, t_z]^\top [n_x, n_y, 0], \tag{3.15}$$

où $\mathbf{R}_z$ est la matrice de rotation autour de l'axe z.
En développant l'expression dans (3.15), nous obtenons :

$$\mathbf{H} = \begin{bmatrix} \cos(\theta) - n_x t_x & -\sin(\theta) - n_y t_x & 0 \\ \sin(\theta) - n_x t_y & \cos(\theta) - n_y t_y & 0 \\ -n_x t_z & -n_y t_z & 1 \end{bmatrix}, \tag{3.16}$$

$$= \begin{bmatrix} h_1 & h_2 & 0 \\ h_3 & h_4 & 0 \\ h_5 & \frac{n_y}{n_x} h_5 & 1 \end{bmatrix}. \tag{3.17}$$

Cela mène à 5 éléments de $\mathbf{H}$ à estimer. Chaque correspondance de points donne deux équations linéairement indépendantes de la forme $\mathbf{Ah} = \mathbf{b}$. En utilisant l'équation (3.9), nous obtenons :

$$\begin{bmatrix} -d - h_3 a - h_4 b + h_5 x_i y_j + h_5 y_i c \\ -e + h_1 a + h_2 b - h_5 x_i x_j - h_5 x_j c \end{bmatrix} = \mathbf{0}, \tag{3.18}$$

$$\begin{bmatrix} 0 & 0 & -a & -b & x_i y_j + y_i c \\ a & b & 0 & 0 & -x_i x_j - x_j c \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix} = \begin{bmatrix} d \\ e \end{bmatrix}, \tag{3.19}$$

avec :

$$a = w_j x_i,\ b = w_j y_i,\ c = y_j \frac{n_y}{n_x},\ d = -w_i y_j,\ e = w_i x_j.$$

En utilisant deux correspondances de points, cela donne 4 équations qui forment un système de rang incomplet. La solution est $\mathbf{h} = \mathbf{Vy} + \lambda \mathbf{v}$ (voir [21]) où $svd(\mathbf{A}) = \mathbf{UDV}^\top$ et $\mathbf{v}$ est le dernier vecteur colonne de $\mathbf{V}$. Le vecteur $\mathbf{y}$ est donné par $\mathbf{y}_i = \mathbf{b'}_i / d_i$ où d_i est le i$^{\text{ème}}$ élément diagonal de $\mathbf{D}$ et $\mathbf{b'} = \mathbf{U}^\top \mathbf{b}$.

Cela laisse le scalaire inconnu λ qui peut être calculé à partir des contraintes trigonométriques additionnelles $(\cos^2(\theta) + \sin^2\theta) - 1 = 0)$. La contrainte trigonométrique peut être entièrement exprimée en fonction des variables h_1, h_2, h_3, h_4 :

$$\cos(\theta)^2 + \sin(\theta)^2 - 1 = 0, \tag{3.20}$$

$$(h_1 + n_x t_x)^2 + (-h_2 - n_y t_x)^2 - 1 = 0, \tag{3.21}$$

avec :

$$t_x = n_x(h_4 - h_1) - n_y(h_2 + h_3). \tag{3.22}$$

Substituer $\mathbf{h} = \mathbf{Vy} + \lambda \mathbf{v}$ dans l'équation (3.21) donne une équation quadratique fonction de l'inconnue restante λ (l'équation développée n'est pas affichée en raison de sa longueur excessive). Le calcul de λ donne deux solutions pour les paramètres h_1, h_2, h_3, h_4, h_5.

Une fois l'homographie $\mathbf{H}$ estimée, elle peut être décomposée en paramètres de rotation et de translation relatives. Nous substituons les éléments de $\mathbf{H}$ de (3.17), c'est-à-dire h_1, h_2, h_3, h_4

et h_5 dans (3.16). Connaissant n_x et n_y, les paramètres de translation peuvent être directement calculés en utilisant les relations suivantes :

$$t_z = \frac{-h_5}{n_x}, \tag{3.23}$$
$$t_x = n_x(h_4 - h_1) - n_y(h_2 + h_3), \tag{3.24}$$
$$t_y = n_y(h_1 - h_4) - n_x(h_2 + h_3). \tag{3.25}$$

Et le paramètre de rotation est alors obtenu grâce à :

$$\cos(\theta) = h_1 + n_x t_x. \tag{3.26}$$

Estimation de la pose relative à partir de 2.5 points avec une normale de plan vertical inconnue

L'algorithme à 2.5 points est une extension de l'algorithme à 2 points décrit précédemment. L'homographie est paramétrisée comme dans (3.15). Cependant, quand la normale du plan $\mathbf{n}$ n'est pas connue, il n'est pas possible d'utiliser les mêmes contraintes linéaires, et par conséquent la totalité des 6 paramètres de $\mathbf{H}$ doivent être estimés. Pour cela, une équation supplémentaire est requise et peut être obtenue d'un troisième point. Les équations des contraintes des 2 points et les équations du troisième point sont stockées dans un système d'équation de la forme $\mathbf{Ah} = \mathbf{b}$. Pour une correspondance de points, deux équations peuvent être obtenues comme suit. D'abord, l'homographie est définie comme :

$$\mathbf{H} = \begin{bmatrix} \cos(\theta) - n_x t_x & -\sin(\theta) - n_y t_x & 0 \\ \sin(\theta) - n_x t_y & \cos(\theta) - n_y t_y & 0 \\ -n_x t_z & -n_y t_z & 1 \end{bmatrix}, \tag{3.27}$$
$$= \begin{bmatrix} h_1 & h_2 & 0 \\ h_3 & h_4 & 0 \\ h_5 & h_6 & 1 \end{bmatrix}. \tag{3.28}$$

Calculer $\mathbf{q}_j \times \mathbf{H}\mathbf{q}_i$ mène à :

$$\begin{bmatrix} -c - h_3 a - h_4 b + h_5 x_i y_j + h_6 y_i y_j \\ -d + h_1 a + h_2 b - h_5 x_i x_j - h_6 x_j y_i \end{bmatrix} = \mathbf{0}, \tag{3.29}$$

$$\begin{bmatrix} 0 & 0 & -a & -b & x_i y_j & y_i y_j \\ a & b & 0 & 0 & -x_i x_j & -x_j y_i \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \end{bmatrix} = \begin{bmatrix} c \\ d \end{bmatrix}, \tag{3.30}$$

avec :

$$a = w_j x_i,\ b = w_j y_i,\ c = -w_i y_j,\ d = w_i x_j.$$

Comme dans la section précédente, la solution à ce système est de la forme $\mathbf{h} = \mathbf{V}\mathbf{y} + \lambda\mathbf{v}$. Le scalaire inconnu λ peut être calculé en utilisant la contrainte trigonométrique sur le vecteur normal :

$$\cos(\theta)^2 + \sin(\theta)^2 - 1 = 0, \tag{3.31}$$

$$n_x^2 + n_y^2 = 1. \tag{3.32}$$

En partant de l'équation (3.31), la contrainte peut être obtenue en substituant $cos(\theta)$ et $sin(\theta)$ avec les expressions de h_1 et h_2. L'étape suivante est de substituer t_x avec l'équation (3.22). La relation $n_x^2+n_y^2 = 1$ peut être utilisée pour annuler beaucoup de termes dans l'équation et obtenir :

$$\begin{aligned} &h_1^2 n_y^2 + h_2^2 n_x^2 + h_3^2 n_y^2 + h_4^2 n_x^2 \\ &-2(h_1 h_2 + h_3 h_4) n_x n_y - 1 = 0. \end{aligned} \tag{3.33}$$

En utilisant (3.35-3.37), l'équation peut être réécrite en termes de h_1, h_2, h_3, h_4, h_5 :

$$\begin{aligned} &h_1^2 h_6^2 - 2h_1 h_2 h_5 h_6 + h_2^2 h_5^2 + h_3^2 h_6^2 \\ &-2h_3 h_4 h_5 h_6 + h_4^2 h_5^2 - h_5^2 - h_6^2 = 0. \end{aligned} \tag{3.34}$$

Substituer symboliquement les éléments de $\mathbf{h} = \mathbf{V}\mathbf{y} + \lambda\mathbf{v}$ dans l'équation (3.34) aboutit à un polynôme de degré 4 fonction de l'inconnue λ (l'équation développée n'est pas présentée à cause de sa longueur excessive). Trouver les racines pour λ donne 4 solutions pour le jeu de paramètres h_1, h_2, h_3, h_4, h_5.

La décomposition de l'homographie en paramètres de translation et de rotation du mouvement relatif se déroule de la même façon que dans la section consacrée à l'algorithme 2-pt. Cependant, cela diffère car les normales n_x et n_y ne sont pas données et nécessitent d'être calculées. Nous substituons les éléments de $\mathbf{H}$ à partir de (3.28) dans (3.27). Nous calculons en premier t_z en utilisant la relation $n_x^2 + n_y^2 = 1$,

$$t_z = \pm\sqrt{h_5^2 + h_6^2}. \tag{3.35}$$

Cela donne deux solutions pour t_z qui diffèrent par leur signe et qui conduisent à deux ensembles supplémentaires de solutions. La normale inconnue peut maintenant être calculée :

$$n_x = \frac{-h_5}{t_z}, \tag{3.36}$$

$$n_y = \frac{-h_6}{t_z}. \tag{3.37}$$

Après cela, la procédure de la section précédente peut être de nouveau suivie pour calculer les paramètres supplémentaires restants, avec les équations suivantes, en utilisant cependant les deux solutions pour t_z, n_x et n_y :

$$t_x = n_x(h_4 - h_1) - n_y(h_2 + h_3), \tag{3.38}$$

$$t_y = n_y(h_1 - h_4) - n_x(h_2 + h_3). \tag{3.39}$$

L'angle θ peut être calculé à partir de la relation

$$\cos(\theta) \quad = \quad h_1 + n_x t_x. \tag{3.40}$$

Un fait intéressant dans ce cas est que seule l'une des deux équations disponibles pour le troisième point est utilisée. Bien que pour cette méthode il soit encore nécessaire de tirer 3 points dans la boucle RANSAC, il est maintenant possible de faire un contrôle de cohérence sur la correspondance du troisième point. Pour être une hypothèse d'homographie sans *outlier*, l'équation restante doit aussi être vérifiée. Cela peut facilement être testé et si elle n'est pas vérifiée, l'hypothèse est rejetée prématurément. Cela donne un avantage de temps de calcul par rapport à la méthode basée sur la matrice essentielle à 3 points standard [101], car des échantillons incohérents peuvent être détectés sans test sur toutes les autres correspondances de points.

Estimation de la pose relative à partir de 3 points en utilisant la contrainte de l'homographie

Dans cette section, nous discutons d'une formulation à 3 points de l'estimation de pose d'une caméra avec une direction verticale connue. Elle diffère des algorithmes précédents car elle n'a pas besoin de la connaissance de plans dans la scène. Un algorithme à 3 points a déjà été présenté par [49] mais en utilisant une formulation à base de matrice essentielle. Avec cet algorithme à 3 points, nous proposons une alternative à l'algorithme basé sur la matrice essentielle en utilisant une formulation homographique. Nous partons de (3.15) et, au lieu de supposer que le plan est parallèle au vecteur de gravité, nous ne faisons aucune hypothèse sur l'orientation du plan et utilisons donc 3 paramètres n_x, n_y, n_z, pour le plan normal entièrement inconnu, ce qui conduit à :

$$\mathbf{H} = \mathbf{R}_z - [t_x, t_y, t_z]^\top [n_x, n_y, n_z]. \tag{3.41}$$

La distance caméra-plan est intégrée dans $\mathbf{t}$ de la même façon que les sections précédentes. La matrice homographique est alors définie par :

$$\mathbf{H} = \begin{bmatrix} \cos(\theta) - t_x n_x & -\sin(\theta) - t_x n_y & -t_x n_z \\ \sin(\theta) - t_y n_x & \cos(\theta) - t_y n_y & -t_y n_z \\ -t_z n_x & -t_z n_y & 1 - t_z n_z \end{bmatrix}. \tag{3.42}$$

Les inconnues que nous essayons de trouver sont les paramètres du mouvement $\cos(\theta)$, $\sin(\theta)$, t_x, t_y, t_z et la normale $[n_x, n_y, n_z]$ du plan défini par les correspondances des trois points. Rappelons que l'algorithme à 3 points basé sur la matrice essentielle n'estime que le mouvement de la caméra, alors que l'algorithme homographique à 3 points fournit le mouvement de la caméra et la normale du plan avec le même nombre de correspondances.

Pour estimer les inconnues, nous formons une système d'équations de la forme : $\mathbf{q}_j \times \mathbf{H}\mathbf{q}_i = 0$ et nous développons les relations pour obtenir les deux équations polynomiales suivantes :

$$a t_y - b t_z - w_j x_i \sin(\theta) - w_j y_i \cos(\theta) + y_j w_i = \quad 0, \tag{3.43}$$

$$-a t_x + c t_z + w_j x_i \cos(\theta) - w_j y_i \sin(\theta) - x_j w_i = \quad 0, \tag{3.44}$$

où :

$$\begin{aligned} a &= w_j x_i n_x + w_j y_i n_y + w_j n_z w_i, \\ b &= y_j w_i n_z + y_j n_x x_i + y_j n_y y_i, \\ c &= x_j n_x x_i + x_j w_i n_z + x_j n_y y_i. \end{aligned} \tag{3.45}$$

La troisième équation obtenue grâce à $\mathbf{q}_j \times \mathbf{H}\mathbf{q}_i = 0$ est omise puisqu'elle est une combinaison linéaire des deux autres équations. Par conséquent, chaque correspondance de points donne deux équations linéairement indépendantes et il y a deux contraintes quadratiques additionnelles, la contrainte trigonométrique et la longueur unitaire du vecteur normal, qui peuvent être utilisées :

$$\sin^2(\theta) + \cos^2(\theta) = 1, \tag{3.46}$$

$$n_x^2 + n_y^2 + n_z^2 = 1. \tag{3.47}$$

Il y a au total 8 inconnues ; les deux contraintes quadratiques combinées avec les équations des 3 correspondances de points donnent un total de 8 équations polynomiales fonction des inconnues. Une façon établie de trouver une solution algébrique à un tel système d'équations polynomiales est d'utiliser une base de Gröbner [102]. En calculant la base de Gröbner, un polynôme univarié peut être trouvé, ce qui permet de trouver l'inconnue par calcul des racines. Les variables restantes peuvent ensuite calculées par substitution. Pour résoudre notre problème, nous utilisons le solveur automatique de bases de Gröbner de Kukelova [103] qui peut être téléchargé sur la page web de l'auteur. Ce logiciel génère automatiquement le code Matlab qui calcule une solution au système d'équations polynomiales donné (dans notre cas, les 8 équations spécifiées ci-dessus). Le code Matlab produit consiste en 299 lignes et ne peut par conséquent pas être donné ici. L'analyse des solutions de la base de Gröbner montre que le polynôme univarié final est de degré 8, ce qui signifie qu'il y a jusqu'à 8 solutions réelles à notre problème.

Configurations dégénérées

Dans cette section, nous étudions les configurations dégénérées pour les algorithmes proposés. Dans [101] et [104], les configurations dégénérées pour la méthode 3-points standard à partir de la matrice essentielle ont été étudiées en détail.

Dans ces articles, plusieurs configurations dégénérées sont identifiées. Il est également souligné qu'une configuration colinéaire de points 3D n'est en général pas une configuration dégénérée pour la méthode 3-points, alors qu'elle en est une pour la méthode 5-points. Les configurations dégénérées pour l'algorithme standard à 3-points sont cependant des points colinéaires parallèles à la direction de la translation et des points coplanaires au vecteur de translation. Nous avons étudié si ces scénarios étaient également des configurations dégénérées pour nos algorithmes (la méthode 2-points, 2.5-points et 3-points homographie) en effectuant des expérimentations sur des données simulées. Les configurations dégénérés peuvent être identifiés par une perte de rang de la matrice du système d'équations ou, pour le cas de la base de Gröbner, par une perte de rang de la matrice d'action. Pour le cas de l'homographie à 3 points, l'étude a révélé que la méthode proposée partage les configurations dégénérées de la méthode 3-points standard, mais elle a en plus une configuration dégénérée pour le cas de points colinéaires. Cela est compréhensible puisque la méthode homographique à 3 points estime également la normale du plan qui a alors un degré de liberté indéfini autour de l'axe des points colinéaire. Pour les méthodes à 2 points

TABLE 3.1 – Comparaison des configurations dégénérées ("oui" signifie "dégénérée") pour la méthode 3-points standard (basée sur la matrice essentielle), la méthode proposée à 3 points basée sur l'homographie, et les méthodes à 2 et 2.5 points.

	3-points standard	3-points homographie	2-points	2.5-points
Points colinéaires	non	oui	non	non
Points colinéaires parallèles à la direction de la translation	oui	oui	non	non
Points coplanaires au vecteur de translation	oui	oui	non	non

et à 2.5 points, ces cas spéciaux ne posent cependant pas de configuration dégénérée. Les informations supplémentaires dans le cas de la connaissance totale ou partielle des paramètres du plan permettent d'éviter la dégénérescence dans les cas critiques contrairement aux méthodes 3-points plus générales. Les résultats de la comparaison sont résumés dans la table 3.1.

3.2.2 Résultats expérimentaux

Expérimentations sur des données simulées

Pour évaluer les algorithmes sur des données synthétiques, nous avons choisi le protocole suivant. La distance moyenne de la scène au premier centre de la caméra est fixée à 1. La scène se compose de deux plans, un plan du sol et un plan vertical qui est parallèle au plan image de la première caméra. Les deux plans sont constitués de 200 points échantillonnés aléatoirement. La *baseline* entre deux caméras est fixée à 0.2, c'est-à-dire 20% de la distance moyenne de la scène, et la distance focale est fixée à 1000 pixels, avec un champ de vue de 45 degrés. Chaque algorithme est évalué en fonction d'un bruit image variable et en augmentant le bruit des données de l'IMU. Chacune des configurations est évaluée avec une translation longitudinale, vers la scène (le long de l'axe z) et une translation latérale (le long de l'axe x) de la deuxième caméra. De plus, la seconde caméra subit une rotation autour de chaque axe.

Pour évaluer la robustesse des algorithmes, nous comparons les translations et les rotations relatives séparément. La mesure d'erreur compare la différence d'angle entre la rotation réelle et la rotation estimée. La translation n'étant connue qu'à l'échelle près, nous comparons l'angle entre la translation réelle et la translation estimée. Les erreurs sont calculées comme suit :

- Différence angulaire pour $\mathbf{R}$:
  $\xi_R = \arccos((\mathrm{Tr}(\mathbf{R}\dot{\mathbf{R}}^\top) - 1)/2)$
- Différence de direction pour $\mathbf{t}$:
  $\xi_t = \arccos((\mathbf{t}^\top \dot{\mathbf{t}})/(\|\mathbf{t}\|\|\dot{\mathbf{t}}\|))$

où $\mathbf{R}$, $\mathbf{t}$ sont les transformations réelles et $\dot{\mathbf{R}}$, $\dot{\mathbf{t}}$ sont les transformations correspondantes estimées.

Chaque point dans les courbes représente le 5-quantile [1] (quintile) de 1000 mesures.

1. Le k-quantile est la valeur limite du $k^{\text{ème}}$ intervalle qui sépare des données ordonnées en k intervalles réguliers. Pour $k = 2$, le 2-quantile est la valeur médiane.

Poses relatives

Les figures 3.3 et 3.4 comparent l'algorithme à 2 points à l'algorithme général à 5 points (basé sur la matrice essentielle) [42], à l'algorithme 4pt-homographie [21] et 3pt-matrice essentielle [101]. Notez que, dans ces expérimentations, les poses des caméras ont été calculées à partir de points tirés au hasard sur le plan du sol. Comme les poses des caméras estimées à partir de points coplanaires ne fournissent pas une solution unique pour les algorithmes 5pt-matrice essentielle, 4pt et 3pt-matrice essentielle, nous évaluons chaque hypothèse avec tous les points provenant des deux plans. La solution fournissant le plus d'*inliers* est choisie comme étant la solution correcte. Cette évaluation est utilisée dans toutes nos expérimentations en simulation. De même, les figures 3.5 et 3.6 montrent une comparaison de l'algorithme 2.5pt avec l'algorithme général 5pt, l'algorithme 4pt et l'algorithme 3pt-matrice essentielle. Dans ce cas, les poses des caméras sont calculées à partir de points échantillonnés au hasard sur le plan vertical uniquement.

L'évaluation montre que la connaissance de la direction verticale et l'exploitation de la planéité de la scène améliore l'estimation du mouvement. Les algorithmes 2pt et 2.5pt surpassent les algorithmes 5pt et 4pt en termes de précision. Avec des mesures parfaites de l'IMU, les algorithmes sont robustes au bruit d'image et surpassent les algorithmes 5pt et 4pt. Avec l'augmentation du bruit de l'IMU, leurs performances restent comparables à celles de l'algorithme 5pt et supérieures à l'algorithme 4pt.

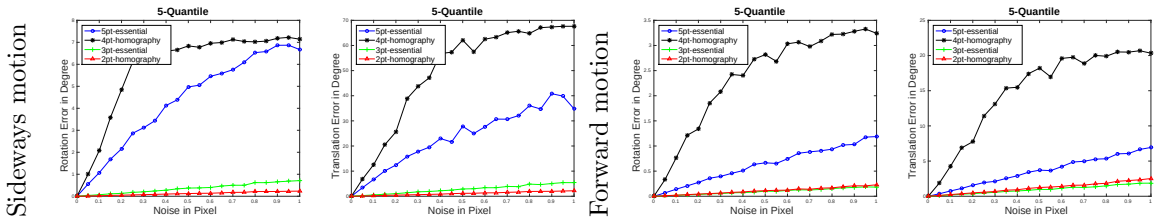


FIGURE 3.3 – Evaluation de l'algorithme à 2 points lors de mouvements latéraux et longitudinaux avec du bruit image variable.

Homographie à 3 points

Les figures 3.7 et 3.8 comparent l'algorithme 3pt-homographie à l'algorithme général à 5 points [42] et à l'algorithme 3pt-matrice essentielle [101]. Cette évaluation montre que la méthode proposée surpasse l'algorithme 5pt en termes de précision. Avec des mesures IMU parfaites, notre algorithme est robuste au bruit image et surpasse l'algorithme 5pt, tout en faisant aussi bien que l'algorithme 3pt basé sur la matrice essentielle. Avec une augmentation du bruit IMU, les performances des algorithmes 3pt-matrice essentielle et 3pt-homographie sont toujours comparables à l'algorithme 5pt.

Temps de calcul

Nous avons évalué le temps d'exécution de tous les algorithmes sur une architecture Intel i7-2600K à 3.4GHz en utilisant Matlab. Pour fournir une comparaison équitable, tous les algorithmes ont été implémentés en Matlab. Aucun fichier mex n'a été utilisé, à l'exception de la fonction pour obtenir une matrice échelonnée en ligne réduite *rref*, qui est requise par les algorithmes 3pt-matrice essentielle et 3pt-homographie. Tous les temps ont été moyennés sur 1000 exécutions. La table 3.2 résume les temps d'exécution pour chacun des six algorithmes. Le temps

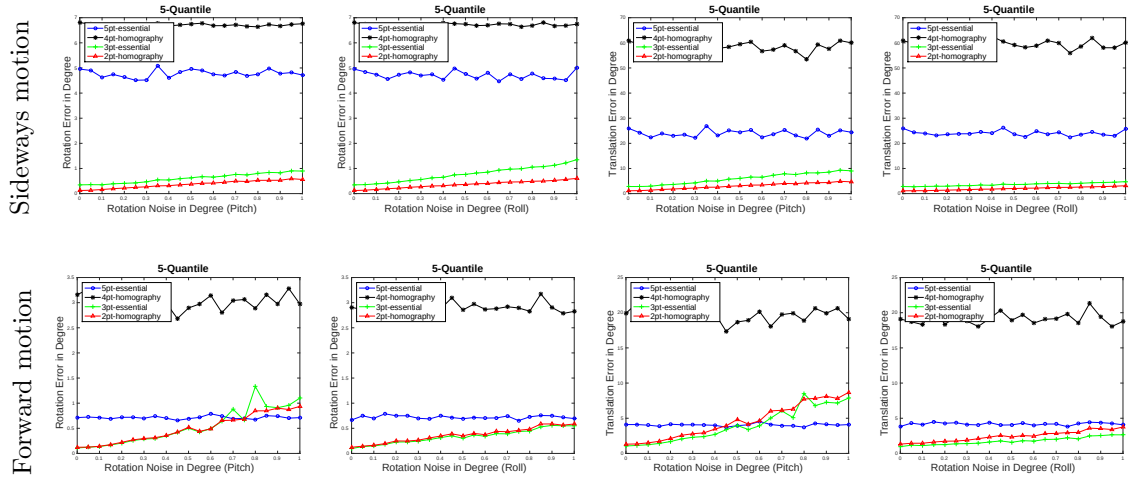


FIGURE 3.4 – Evaluation de l'algorithme à 2 points avec un bruit IMU variable et un bruit image constant de 0.5 pixel d'écart-type. Première ligne, mouvement latéral, seconde ligne, mouvement longitudinal.

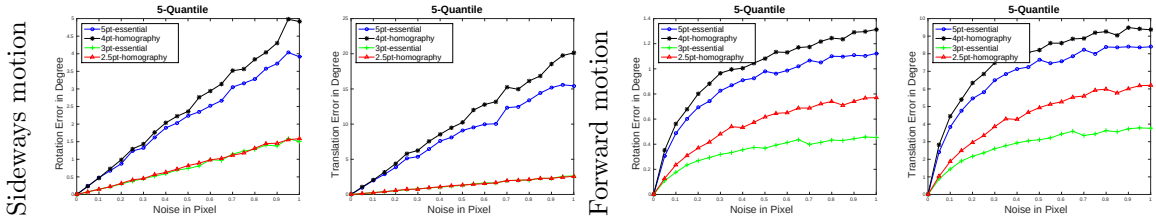


FIGURE 3.5 – Evaluation de l'algorithme 2.5pt lors de mouvements latéraux et longitudinaux sous un bruit image variable.

d'exécution élevé de l'algorithme 3pt-homographie est dû à la complexité de la solution par la base de Gröbner, qui doit effectuer une élimination de Gauss-Jordan sur la matrice d'élimination de taille 443×451.

Pour une itération RANSAC, les temps peuvent varier considérablement d'un algorithme à l'autre. Cela est dû aux différents espaces de solutions fournis par les algorithmes. Pour avoir la même mesure d'erreur pour tous les algorithmes, nous choisissons d'utiliser l'erreur de reprojection pour sélectionner la pose correcte parmi l'ensemble de poses possibles. Par exemple, l'algorithme 2pt fournit une pose de caméra unique, tandis que l'algorithme 5pt peut fournir jusqu'à 10 matrices essentielles différentes. De plus, pour chaque matrice essentielle, il existe 4 poses de caméra possibles qui doivent être vérifiées pour trouver la pose correcte, ce qui peut donner un total de 40 poses possibles. Les formulations basées sur l'homographie fournissent quant à elles directement des ensembles de poses. Même si l'estimation de l'hypothèse de l'algorithme 3pt-homographie a une plus grande complexité en temps, comparée à sa contrepartie basée sur la matricielle essentielle, une itération RANSAC est moins coûteuse car nous devons

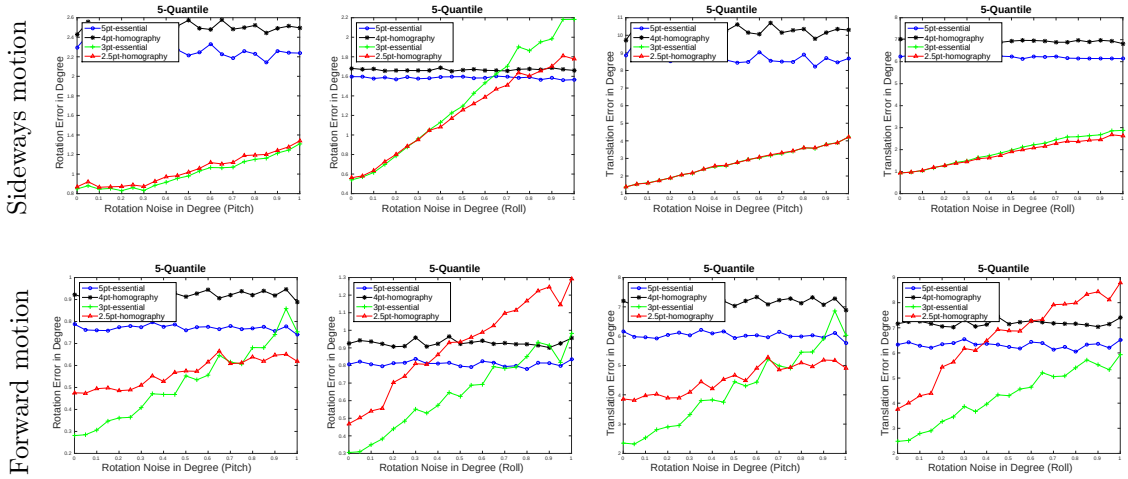


FIGURE 3.6 – Evaluation de l'algorithme 2.5pt sous un bruit IMU variable et un bruit image constant de 0.5 pixel d'écart-type. Première ligne, mouvement latéral, seconde ligne, mouvement longitudinal.

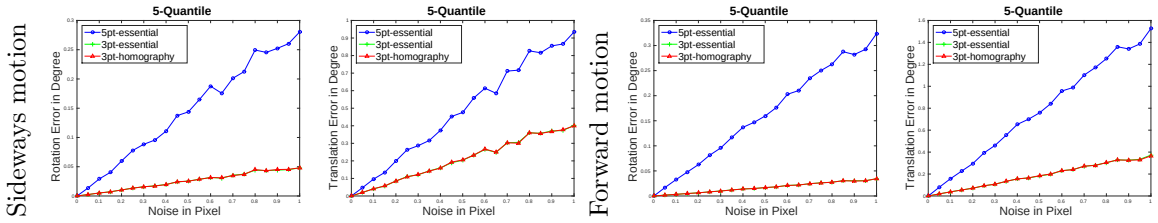


FIGURE 3.7 – Evaluation de l'algorithme 3pt-homographie lors de mouvements latéraux et longitudinaux avec un bruit image variable.

évaluer moins de poses potentielles. La table 3.2 montre clairement que le temps de calcul est dominé par la sélection des hypothèses (calcul de l'erreur de re-projection), et non par le solveur. Dans toutes les expérimentations, nous avons utilisé un ensemble de 200 points mis en correspondance.

Expérimentations sur des données réelles

Dans cette section, nous évaluons les algorithmes proposés dans des environnements intérieurs et extérieurs.

Mesure de l'erreur

Afin de comparer les poses estimées de la caméra avec la vérité de terrain, nous avons utilisé la mesure de l'erreur de pose relative (RPE, *Relative Pose Error*) telle que proposée par Sturm [105]. La RPE compare la précision locale de la trajectoire sur un intervalle de temps fixe Δ, qui correspond à la dérive de la trajectoire. La RPE au pas de temps i peut être définie par :

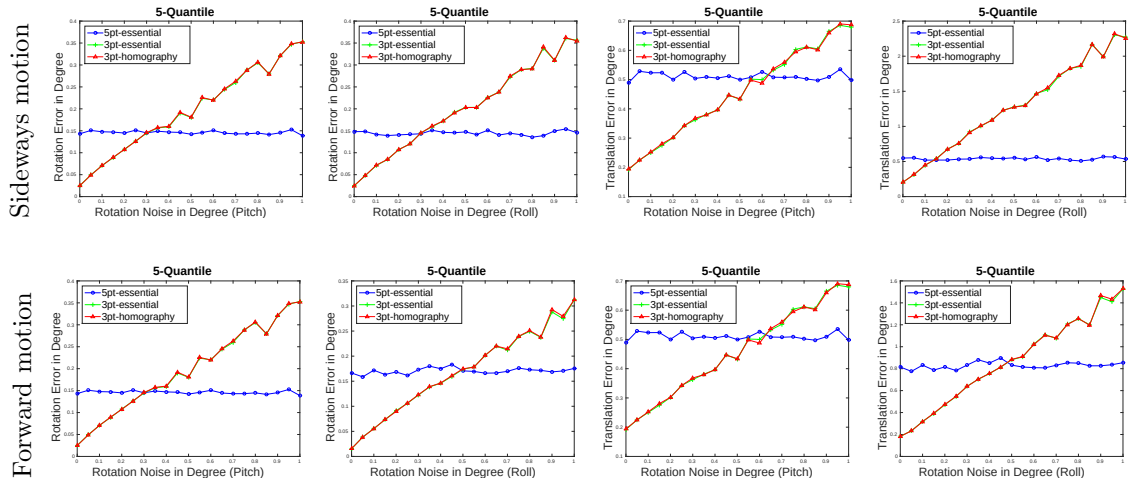


FIGURE 3.8 – Évaluation de l'algorithme 3pt-homographie sous différents bruits d'IMU et bruit d'image constant d'écart-type 0.5 pixel. Première ligne : mouvement latéral de la caméra avec un bruit variable sur l'angle de tangage (à gauche) et un bruit variable sur l'angle de roulis (à droite). Deuxième ligne : mouvement vers l'avant de la caméra avec un bruit variable sur l'angle de tangage (à gauche) et avec un bruit variable sur l'angle de roulis (à droite).

TABLE 3.2 – Comparaison des temps d'exécution des différents algorithmes d'estimation de pose. La seconde colonne donne les temps pour estimer une hypothèse. La troisième colonne fournit les temps pour une itération de RANSAC, ce qui inclut la sélection de la bonne solution parmi l'ensemble des hypothèses. La dernière colonne montre le nombre moyen de solutions réelles (poses de la caméra) fournies par les algorithmes respectifs.

Méthode	Hypothèse Estimation(ms)	RANSAC 1 Itération(ms)	Nombre moyen de solutions
2pt	0.09	8.31	2
2.5pt	0.22	33.45	8
3pt-homographie	27.28	55.17	6.85
3pt-essentielle	0.49	25.02	6.18
4pt-homographie	0.18	8.65	2
5pt-essentielle	0.42	64.33	16.02

$$\mathbf{E}_i = (\mathbf{Q}_i^{-1}\mathbf{Q}_{i+\Delta})^{-1}(\mathbf{P}_i^{-1}\mathbf{P}_{i+\Delta}), \tag{3.48}$$

où $\mathbf{Q}_i, \mathbf{P}_i \in SE(3)$ représentent respectivement la vérité terrain et les poses estimées. $\mathbf{E}_i$ représente alors l'erreur relative. Pour une séquence de n poses de caméra, $m = n - \Delta$ erreurs de poses relatives sont alors estimées. De ces erreurs, nous proposons de calculer l'erreur quadratique moyenne (RMSE) sur tous les indices de temps de la composante translationnelle par :

$$RMSE(\mathbf{E}_{1:n}, \Delta) = \sqrt{\frac{1}{m}\sum_{i=1}^{m} \|trans(E_i)\|^2}, \tag{3.49}$$

où $trans(\mathbf{E}_i)$ correspond à la composante translationnelle de l'erreur de pose relative $\mathbf{E}_i$.

Jeux de données VICON

Afin d'avoir une évaluation pratique des algorithmes 2-pt, 2.5pt et 3pt, plusieurs jeux de données réelles ont été collectés avec une vérité terrain fiable (cf figure 3.9). Les données de vérité terrain ont été obtenues en réalisant les expérimentations dans dans une salle équipée d'un système de capture de mouvement Vicon composé de 20 caméras T40S. Avec un tel système, nous pouvons assurer une localisation à 6 ddl de notre système avec une précision submillimétrique et à une fréquence élevée (500fps) comme démontré dans [106] et [107]. Nous avons utilisé les données du Vicon comme mesures inertielles et pour obtenir le facteur d'échelle dans les différentes expérimentations. Les séquences ont été acquises soit avec une caméra perspective montée sur un robot mobile Segway (cf figure 3.9), soit avec un système tenu à la main, afin d'avoir des trajectoires à la fois dans le plan et en 3D. Dans les deux cas, les caméras sont synchronisées avec le système Vicon. La résolution des images est de 1624×1234 pixels. La longueur de ces trajectoires est comprise entre 20 et 50 mètres et le nombre d'images est compris entre 150 et 350 par séquence. La vitesse de déplacement du robot est d'environ 1 m/s. Deux jeux de données différents ont été acquis, le premier montrant le plan de sol de façon dominante, le second montrant les murs de manière dominante.

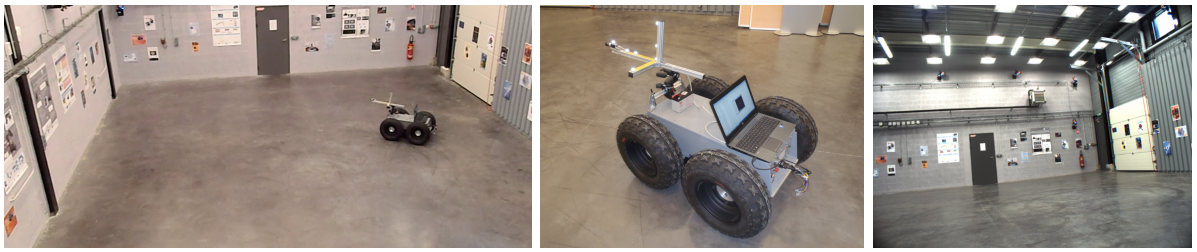


FIGURE 3.9 – A gauche, la salle Vicon utilisée pour enregistrer les jeux de données avec la vérité terrain. Au centre, le robot Segway télé-opéré enregistrant les données. A droite, un exemple d'image capturée par le robot.

Nous effectuons une comparaison des algorithmes 2pt, 2.5pt et 3pt-homographie avec l'algorithme 5pt afin de montrer l'efficacité des méthodes proposées. Tout d'abord, nous utilisons [108] pour extraire et mettre en correspondance les descripteurs SIFT [72]. Les mêmes ensembles de points caractéristiques sont utilisés pour les différents algorithmes et sont les données d'entrée du RANSAC [99] afin de sélectionner les *inliers*. Pour les paramètres du RANSAC, nous utilisons un nombre fixe de 100 itérations dans toutes nos expérimentations.

La figure 3.10 montre l'évaluation de l'algorithme à 2 points basé sur le plan du sol. Les trajectoires obtenues avec le 2pt (courbe rouge) et le 5pt (courbe noire) sont comparées à la vérité terrain (courbe bleue) du Vicon. Dans toutes ces expérimentations, même si les deux approches proposent des trajectoires ayant une forme globalement similaire à la réalité terrain, nous pouvons noter que l'algorithme 2pt donne de meilleurs résultats que la méthode 5pt. Dans le cas des trajectoires dans le plan, c'est-à-dire pour les séquences I, II et IV sur la figure

TABLE 3.3 – Erreur quadratique moyenne (RMSE) pour les 5 séquences.

Séquence	2pt (mm)	5pt (mm)
Ground Sequence I	8.94	48.55
Ground Sequence II	9.28	56.66
Ground Sequence III	18.46	65.39
Ground Sequence IV	14.25	93.30
Ground Sequence V	25.61	34.46

TABLE 3.4 – Erreur quadratique moyenne pour les séquences avec un mur vertical.

Méthode	Wall Sequence I	Wall Sequence II
2.5pt	60.65	41.60
5pt-matrice essentielle	24.97	27.84
3pt-matrice essentielle	27.44	64.65
3pt-homographie	27.26	65.96

3.10, il faut noter que l'algorithme 2pt a une très faible dérive selon l'axe vertical alors que le 5pt accumule une erreur significative. Sur les cinq séquences, l'erreur angulaire moyenne en translation est égale à 0.1883 radians pour le 2pt et 0.3380 pour le 5pt. L'erreur quadratique moyenne définie dans l'équation (3.49) est donnée pour les 5 séquences dans la table 3.3. Le 2pt surpasse clairement l'algorithme 5pt.

La figure 3.11 compare les différentes trajectoires obtenues par les algorithmes 2.5pt (courbe rouge), 3pt-homographie (courbe verte), 3pt-matrice essentielle (courbe magenta), et 5pt (courbe noire) à la vérité de terrain (courbe bleue) obtenue par le système Vicon. Dans la table 3.4, nous comparons l'erreur RMS des différents algorithmes. L'algorithme 2.5pt présente des performances similaires à celles de l'algorithme standard 5pt et des deux algorithmes 3pt, mais il présente l'avantage d'une résolution plus simple.

Ces expérimentations ont également démontré que les hypothèses posées pour l'algorithme 2pt (plan du sol horizontal) et pour l'algorithme 2.5pt (murs verticaux) sont vérifiées dans des situations pratiques et peuvent être utilisées dans des applications réelles.

L'algorithme 2pt dans un processus de SfM

Dans cette expérimentation finale, nous démontrons l'utilisation de l'algorithme 2pt dans un processus de SfM incrémental en remplaçant l'algorithme 5pt initial par notre algorithme 2pt. L'algorithme de SfM MAVMAP [109] a été adapté pour comparer l'algorithme 2pt à l'algorithme 5pt. L'estimation de la pose entre deux vues est réalisée à chaque nouvelle image. Pour calculer la pose relative entre deux images consécutives, nous estimons la matrice essentielle dans le cas de l'algorithme 5pt et l'homographie pour l'algorithme 2pt. Ensuite, un ajustement de faisceaux complet est effectué pour calculer les poses précises de la caméra et les coordonnées des points 3D. Le but principal de cette expérimentation est de montrer que le 2pt peut, en pratique, remplacer les algorithmes standards (comme le 5pt) pour gagner en vitesse tout en maintenant la précision du système.

Pour cette expérimentation, un jeu de données a été réalisé par un drone au-dessus d'un parking. Les images ont été capturées par une caméra montée sur un cardan, de sorte que l'axe z de

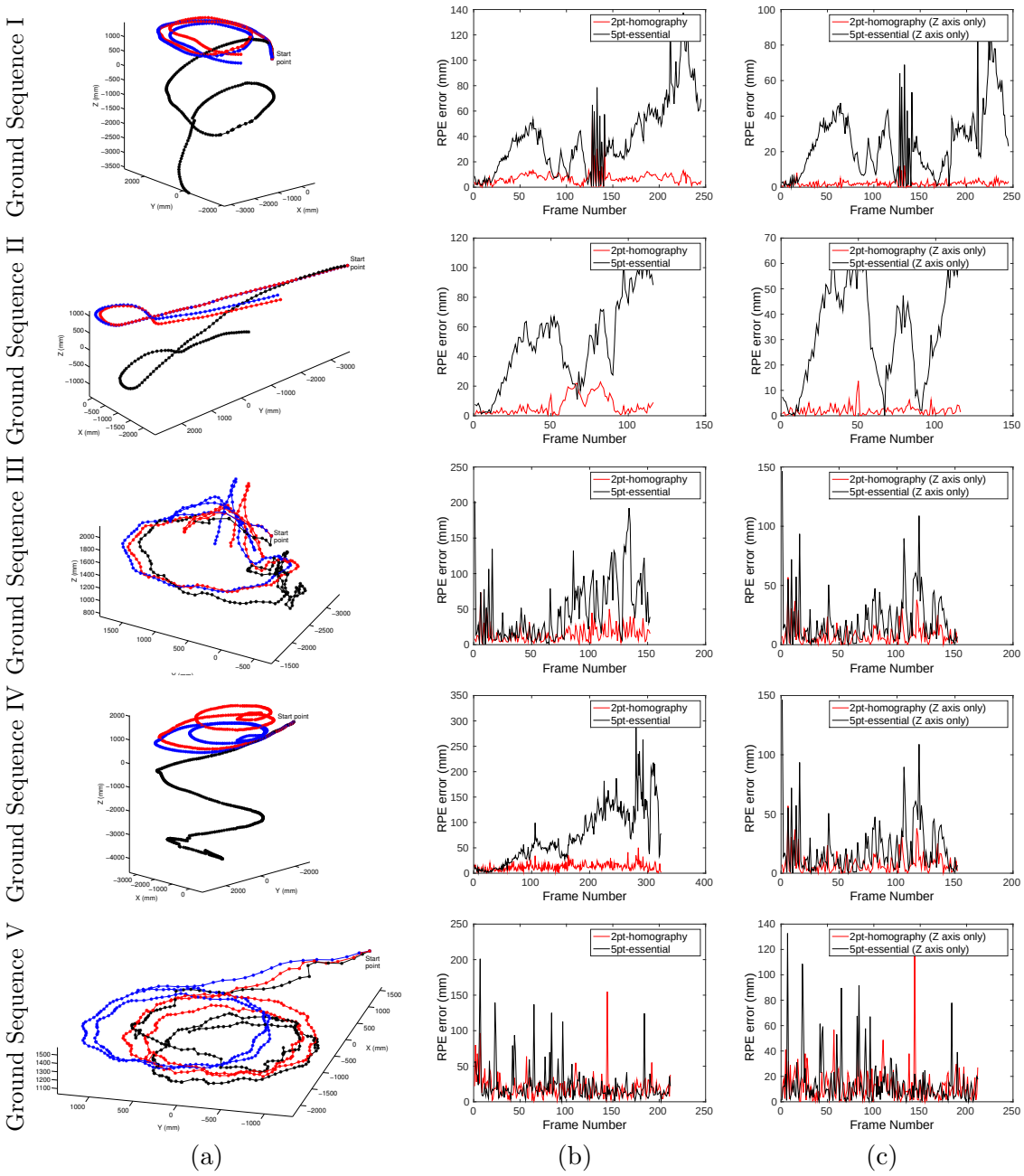


FIGURE 3.10 – Evaluation de l'algorithme 2pt basé sur le plan du sol : (a) Odométrie visuelle estimée en utilisant les algorithmes 2pt (en rouge) et 5pt (en noir) par rapport à la vérité terrain Vicon (en bleu). (b) Erreur de pose relative (RPE) en *mm* à chaque image. (c) Erreur RPE pour l'axe vertical (axe z). Voir [2] pour davantage de résultats.

la caméra soit aligné avec la direction de la gravité. Les images ont été enregistrées à la résolution native de 24MP. Le drone était équipé d'un GPS et la trajectoire GPS est utilisée comme vérité terrain pour la comparaison. La figure 3.12 montre les résultats de cette expérimentation. La

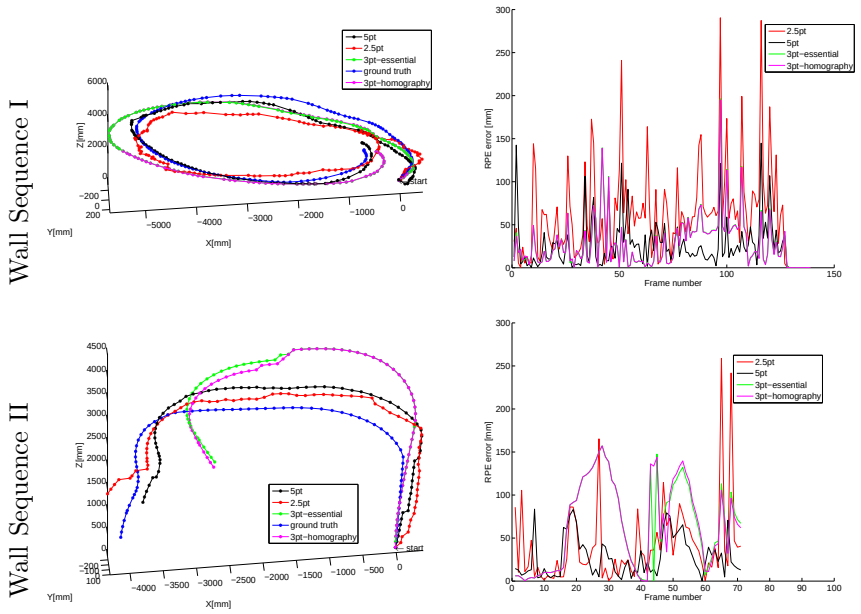


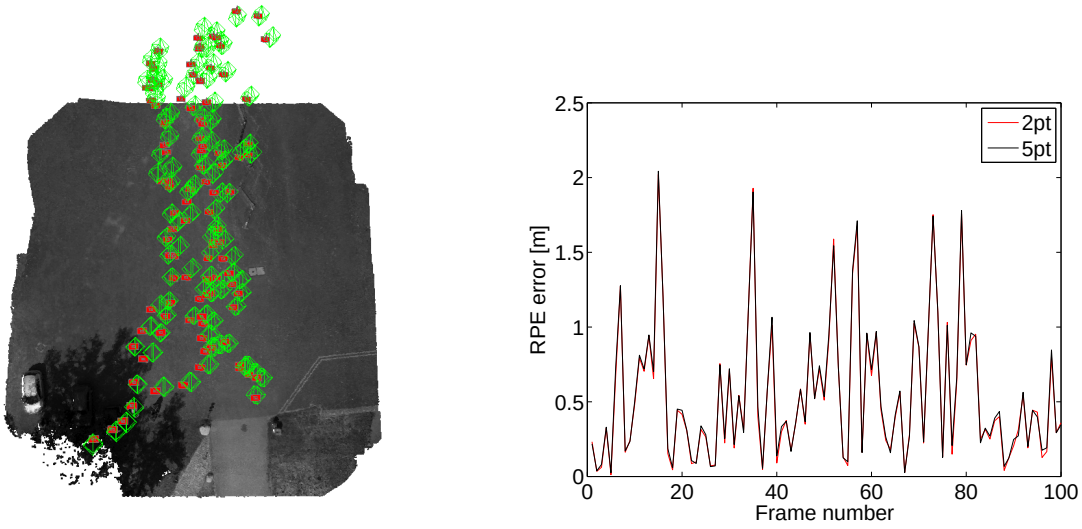
FIGURE 3.11 – Évaluation de l'algorithme 2.5pt basé sur les murs verticaux : trajectoires estimées avec les algorithmes 2.5pt (en rouge), 3pt-homographie (en magenta), 3pt-matrice essentielle (en vert) et 5pt (en noir) comparées à la vérité terrain Vicon (en bleu).

figure 3.12.a montre la sortie du système de SfM, composée du nuage de points 3D (densifié par SURE [110]), des positions des caméras (en rouge) et des positions GPS (en vert). La figure 3.12.b montre l'évolution de l'erreur de position relative (RPE) en utilisant l'algorithme 5pt (en noir) et l'algorithme 2pt (en rouge). Les deux algorithmes mènent à des résultats quasi identiques. La valeur de l'erreur RPE restante est principalement due aux incertitudes des mesures GPS et était attendue. L'erreur de reprojection résultante après l'ajustement de faisceaux est des 0.249 pixels pour l'algorithme 2pt, et 0.246 pixels pour l'algorithme 5pt, c'est-à-dire des valeurs similaires. La justification de ces résultats similaires provient de l'optimisation par ajustement de faisceaux. Cette expérimentation démontre que l'algorithme à 2 points proposé peut remplacer l'algorithme standard à 5 points avec succès et de manière transparente, mais avec l'avantage d'un gain en temps de calcul.

3.3 Estimation de l'attitude et de l'altitude d'un drone

Le second travail sur les solutions minimales porte sur l'estimation de la pose d'un drone basé sur la projection d'un laser circulaire au sol (cf Figure 3.13). L'objectif de ce système n'est pas de retrouver la pose totale du drone (l'angle de lacet ne peut pas être estimé), mais obtenir l'attitude et l'altitude du drone par rapport au sol présente un grand intérêt pour la phase d'atterrissage. Un tel système a également l'avantage de fonctionner dans des environnements très sombres, là où une caméra seule, c'est-à-dire sans projection de lumière, aurait échoué.

La projection du laser circulaire au sol est une conique, ce qui est un outil mathématique très intéressant en géométrie projective [100]. Nous avons développé une approche géométrique



(a) (b)

FIGURE 3.12 – Résultats du SfM incrémental en utilisant l'algorithme à 2 points. (a) Nuage de points résultant, positions des caméras (en rouge) et positions GPS (en vert). (b) Erreur de pose relative (RPE) en utilisant l'algorithme 5pt ou l'algorithme 2pt dans le processus de SfM. Les solutions initiales du 5pt et du 2p sont suffisamment similaires pour que l'ajustement de faisceaux converge quasiment à la même solution finale.

qui permet une estimation à l'échelle de la pose du drone, relative au plan sur lequel la forme est projetée. Trois solutions ont été proposées, incluant deux solutions minimales :

- la première solution proposée, Plane-Pair 5-Point (PP-5), repose sur l'utilisation de 5 points, ce qui est le nombre classique de points utilisés pour estimer une conique,
- la deuxième solution proposée, Plane-Pair 3-Point (PP-3), repose sur l'utilisation de la géométrie épipolaire appliquée aux coniques et qui ne nécessite plus que 3 points,
- la troisième solution, Ground Plane 3-Point (GP-3) permet l'estimation directe du plan du sol (et donc de la pose du drone) avec 3 points sans passer par l'estimation de la conique au sol.

Enfin, nous avons proposé une méthode originale pour calibrer ce système grâce à un ajustement de faisceaux ne nécessitant qu'un paramètre par point.

3.3.1 Formulation du problème

La figure 3.13 présente l'architecture de notre système composé d'une caméra et d'une source laser projetant une forme circulaire. Ce système peut être monté sur une drone pour estimer son altitude et son attitude (roulis, tangage) par rapport au plan du sol sur lequel est projeté le motif laser. La forme circulaire générée par le laser se projette en une conique sur le sol. L'image de cette conique sur le plan image de la caméra est également une conique dont l'extraction permet d'estimer la pose (altitude et attitude) du système caméra/laser. Nous allons maintenant formaliser ce problème mathématiquement.

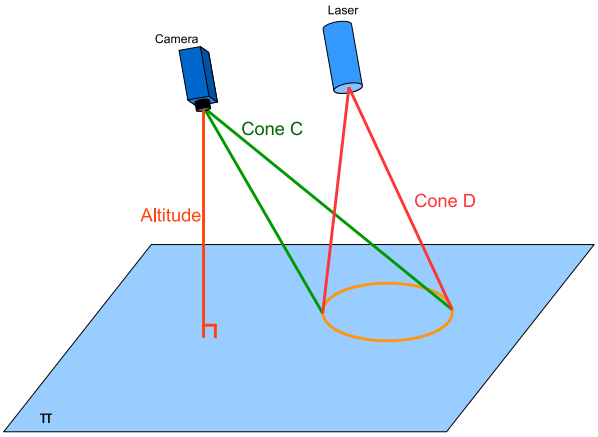


FIGURE 3.13 – Représentation géométrique du système caméra/laser.

Le repère associé à la caméra est notre repère de référence, c'est-à-dire le repère monde. Par conséquent, la matrice de projection de la caméra est définie par :

$$\mathbf{P}_{cam} \sim \begin{bmatrix} \mathbf{K} & \mathbf{0} \end{bmatrix}_{3\times 4}, \tag{3.50}$$

où $\mathbf{K}$ est la matrice 3×3 contenant les paramètres intrinsèques de la caméra.

Nous décrivons aussi la projection réalisée par le laser par une projection perspective.

Soit $\mathbf{P}_{las}$ la "matrice de projection" du laser, alors :

$$\mathbf{P}_{las} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{las} & -\mathbf{R}_{las}\mathbf{t}_{las} \\ 0 & 1 \end{bmatrix}. \tag{3.51}$$

Ici, $\mathbf{R}_{las}$ représente l'orientation et $\mathbf{t}_{las}$ la position du laser par rapport à la caméra. Ces paramètres peuvent être obtenus par calibrage comme expliqué dans la section 3.3.4.

Le motif circulaire généré par le laser peut être représenté par un cercle $\mathbf{d}$ dans le "plan image" du laser :

$$\mathbf{d} \sim \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -tan^2(\theta) \end{bmatrix}, \tag{3.52}$$

où θ est l'angle d'ouverture du cône du laser.

Le cône $\mathbf{D}$ est alors obtenu en rétro-projetant $\mathbf{d}$ en 3D :

$$\mathbf{D} \sim \mathbf{P}_{las}^T \, \mathbf{d} \, \mathbf{P}_{las} \tag{3.53}$$

$$\mathbf{D} \sim \begin{bmatrix} \mathbf{R}_{las}^T \mathbf{d} \, \mathbf{R}_{las} & -\mathbf{R}_{las}^T \mathbf{d} \, \mathbf{R}_{las}\mathbf{t}_{las} \\ -\mathbf{t}_{las}^T \mathbf{R}_{las}^T \mathbf{d} \, \mathbf{R}_{las} & \mathbf{t}_{las}^T \mathbf{R}_{las}^T \mathbf{d} \, \mathbf{R}_{las}\mathbf{t}_{las} \end{bmatrix}. \tag{3.54}$$

Comme l'illustre la figure 3.13, ce cône intersecte le plan du sol en générant une conique, vue sur l'image de la caméra comme une conique également.

Soit $\mathbf{c}$ la matrice symétrique 3×3 représentant cette conique. L'estimation de $\mathbf{c}$ à partir des points de contour extraits dans l'image est décrite dans la section 3.3.3.

Le problème considéré dans ces travaux est par conséquent l'estimation de la pose du système caméra/laser par rapport au plan du sol. Les paramètres d'entrée supposés connus et fixes sont la connaissance de la forme du laser (cercle $\mathbf{d}$ générant le cône $\mathbf{D}$) et le calibrage du système caméra/laser (paramètres intrinsèques $\mathbf{K}$ de la caméra et pose relative caméra/laser représentée par $\mathbf{R}_{las}$ and $\mathbf{t}_{las}$). Une autre entrée supplémentaire est l'image de la conique $\mathbf{c}$ extraite dans l'image courante de la caméra. Cette conique dépend de la pose du système par rapport au plan du sol.

Nous pouvons immédiatement observer qu'avec cette entrée, les 6 degrés de liberté de la pose du système caméra/laser ne peuvent pas être déterminés. Pour les 3 degrés de liberté concernant la translation, la translation du système parallèlement au plan du sol n'affecte aucune des entrées : la conique $\mathbf{c}$ dans l'image reste identique dans ce cas. Il en va de même pour la rotation autour de la normale du plan. Par conséquent, nous pouvons déterminer 3 degrés de liberté pour la pose : l'altitude au dessus du plan et l'attitude relative par rapport à ce plan (2 angles de rotation : roulis et tangage). Nous pouvons noter que cela revient à déterminer la position du plan du sol par rapport au système caméra/laser. Dans les sections suivantes, nous décrivons par conséquent des méthodes pour estimer la pose du plan du sol.

3.3.2 Une solution géométrique pour l'estimation de l'altitude et de l'attitude

Dans la section précédente, le cône $\mathbf{D}$ généré par le laser circulaire, était défini. De la même façon, la rétro-projection de l'image de la conique $\mathbf{c}$ en 3D donnait naissance à un cône $\mathbf{C}$ (cf. figure 3.13). Ce cône peut être calculé par

$$\mathbf{C} \sim \mathbf{P}_{cam}^{T}\, \mathbf{c}\, \mathbf{P}_{cam}. \tag{3.55}$$

Dans notre scénario, ces deux cônes - celui partant du projecteur laser et celui de la caméra - sont générés par les centres optiques respectifs et par la conique sur le plan du sol. Pour résoudre notre problème d'estimation de pose, nous pouvons procéder comme suit. Premièrement, il faut calculer l'intersection des deux cônes. L'intersection doit contenir la conique sur le plan du sol. Ensuite, si cette conique est déterminée de façon unique, elle est alors suffisante pour calculer la position de son plan "support", c'est-à-dire le plan du sol.

Dans la suite, nous présentons une approche analogue, mais qui ne nécessite pas le calcul explicite de l'intersection des deux cônes. Cette approche est basée sur l'étude de la famille linéaire des quadriques générées par les deux cônes, c'est-à-dire la famille des quadriques $\mathbf{Q}$ paramétrées par un scalaire x. $\mathbf{Q}$ est définie par

$$\mathbf{Q} = \mathbf{C} + x\mathbf{D}. \tag{3.56}$$

Nous étudions dans un premier temps les propriétés de cette famille, puis nous proposons une méthode d'estimation basée sur cette étude.

Étude géométrique

Nous étudions en particulier les éléments dégénérés de la famille de quadriques, c'est-à-dire les quadriques avec une déterminant nul : $\det(\mathbf{Q}) = 0$. Le déterminant $\det(\mathbf{Q})$ est en général un polynôme de degré 4 en x. Parmi ses quatre racines possibles, nous avons toujours les racines $x = 0$ et $x \to \infty$, correspondant aux cônes $\mathbf{C}$ et $\mathbf{D}$. Comme pour les deux autres racines, elles peuvent être réelles ou imaginaires, en fonction des cônes $\mathbf{C}$ et $\mathbf{D}$ générant la famille.

Dans notre contexte, nous savons que ces deux cônes s'intersectent en au moins une conique (la conique sur le plan du sol). Dans ce cas, il peut être prouvé que les deux racines restantes sont des nombres réels et égaux [3]. De plus, la quadrique dégénérée associée avec cette racine est de rang 2, et représente par conséquent une paire de plans. Enfin, l'un de ces plans n'est autre que le plan du sol, alors que le second plan de cette paire est le plan qui sépare les centres optiques de la caméra et du laser, ce qui signifie que les deux centres optiques sont situés sur des côtés opposés du plan. Ceci est illustré par la figure 3.14.

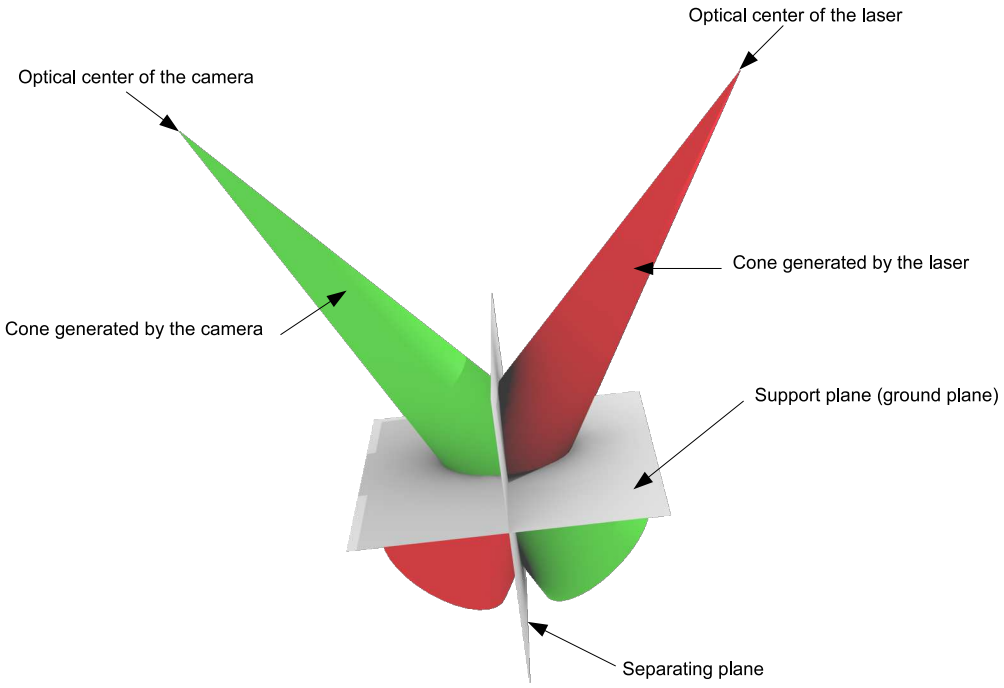


FIGURE 3.14 – Deux cônes générés par la même conique s'intersectent, en général en une deuxième conique.

Méthode d'estimation de la pose

Les propriétés décrites dans la section précédente sont utilisées ici pour concevoir une méthode d'estimation de pose pour notre scénario. Concrètement, nous souhaitons calculer l'emplacement du plan du sol par rapport à la caméra.

Considérons la famille linéaire de quadriques générée par les deux cônes $\mathbf{C}$ et $\mathbf{D}$, c'est-à-dire

$$\mathbf{Q} = \mathbf{C} + x\mathbf{D}. \tag{3.57}$$

Nous calculons d'abord les racines du polynôme $det(\mathbf{Q})$ et considérons ensuite la quadrique $\mathbf{Q}$ associée à la seule racine différente de 0 et ∞. C'est une quadrique de rang 2, autrement dit une paire de plans. Il faut maintenant extraire ces deux plans de $\mathbf{Q}$ et sélectionner ensuite celui qui correspond au plan du sol.

Soient $\mathbf{U}$ et $\mathbf{V}$ les deux plans que nous souhaitons extraire de $\mathbf{Q}$. La matrice 4×4 représentant la paire de plans satisfait [3]

$$\mathbf{Q} \sim \mathbf{V}\mathbf{U}^T + \mathbf{U}\mathbf{V}^T. \tag{3.58}$$

Les deux plans peuvent être extraits de $\mathbf{Q}$, par exemple en lui appliquant une décomposition en valeurs singulières (SVD). Comme $\mathbf{Q}$ est de rang 2 et que les deux valeurs singulières non-nulles sont de signe opposé (voir [3]), sa SVD doit être de la forme

$$\mathbf{Q} = \begin{bmatrix} \mathbf{A} & \mathbf{B} & \vdots & \vdots \end{bmatrix} diag(\sigma_1, \sigma_2, 0, 0) \begin{bmatrix} \pm\mathbf{A}^T \\ \mp\mathbf{B}^T \\ \cdots \\ \cdots \end{bmatrix}. \tag{3.59}$$

Par conséquent, nous pouvons écrire :

$$\mathbf{Q} = \pm(\sigma_1 \mathbf{A}\mathbf{A}^T - \sigma_2 \mathbf{B}\mathbf{B}^T). \tag{3.60}$$

$\mathbf{U}$ et $\mathbf{V}$ satisfont donc :

$$\mathbf{U}\mathbf{V}^T + \mathbf{V}\mathbf{U}^T \sim \sigma_1 \mathbf{A}\mathbf{A}^T - \sigma_2 \mathbf{B}\mathbf{B}^T. \tag{3.61}$$

De (3.58), il est clair que $\mathbf{U}$ et $\mathbf{V}$ forment une base minimale de l'espace ligne de $\mathbf{Q}$ (et, $\mathbf{Q}$ étant symétrique, de son espace colonne également). De (3.59), $\mathbf{A}$ et $\mathbf{B}$ forment également une base minimale de cet espace ligne. Par conséquent, les deux plans $\mathbf{U}$ et $\mathbf{V}$ doivent être des combinaisons linéaires des vecteurs singuliers $\mathbf{A}$ et $\mathbf{B}$, c'est-à-dire

$$\mathbf{U} = u\mathbf{A} + \mathbf{B}, \tag{3.62}$$

$$\mathbf{V} = v\mathbf{A} + \mathbf{B}. \tag{3.63}$$

Nous avons maintenant besoin de déterminer u et v. En introduisant (3.62) et (3.63) dans l'équation (3.61), nous obtenons

$$2uv\mathbf{A}\mathbf{A}^T + 2\mathbf{B}\mathbf{B}^T + (u+v)(\mathbf{A}\mathbf{B}^T + \mathbf{B}\mathbf{A}^T)...$$

$$... \sim \sigma_1 \mathbf{A}\mathbf{A}^T - \sigma_2 \mathbf{B}\mathbf{B}^T. \tag{3.64}$$

Par conséquent, nous pouvons conclure que $u+v = 0$. En introduisant $v = -u$ dans l'équation (3.64), nous obtenons

$$-2u^2 \mathbf{A}\mathbf{A}^T + 2\mathbf{B}\mathbf{B}^T \sim \sigma_1 \mathbf{A}\mathbf{A}^T - \sigma_2 \mathbf{B}\mathbf{B}^T. \tag{3.65}$$

Cette condition est remplie pour

$$u^2 = \frac{\sigma_1}{\sigma_2}. \tag{3.66}$$

Les deux plans peuvent maintenant être calculés pour

$$\mathbf{U} = \sqrt{\frac{\sigma_1}{\sigma_2}}\mathbf{A} + \mathbf{B} \text{ et} \tag{3.67}$$

$$\mathbf{V} = -\sqrt{\frac{\sigma_1}{\sigma_2}}\mathbf{A} + \mathbf{B}. \tag{3.68}$$

Notons que puisque les valeurs singulières σ_1 et σ_2 sont positives, la racine carrée dans ces équations est bien définie.

Nous devons encore déterminer lequel parmi ces deux plans est le plan du sol. Il est évident que les centres optiques de la caméra et du laser se trouvent du même côté du plan du sol.

D'après ce qui a été montré dans [3], les centres optiques doivent se trouver des deux côtés opposés du deuxième plan. Il suffit donc de sélectionner le plan parmi **U** et **V** pour lequel les centres optiques se trouvent du même côté ; c'est le plan du sol.

Normalisons le plan sélectionné de telle sorte qu'il puisse être écrit $\mathbf{\Pi} = [n_x\, n_y\, n_z\, d]^T$, avec $\sqrt{n_x^2 + n_y^2 + n_z^2} = 1$.

L'altitude de notre système est obtenue en calculant la distance orthogonale entre le centre de la caméra et le plan, c'est-à-dire

$$Altitude = |d| \tag{3.69}$$

puisque le centre de la caméra est l'origine de notre repère de référence.

Nous cherchons maintenant l'attitude du système. Nous avons la normale du plan du sol exprimée dans deux repères : dans le repère monde où elle vaut $\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$ et dans le repère caméra, où notre estimation est $\begin{bmatrix}n_x & n_y & n_z\end{bmatrix}^T$. Récupérer l'attitude du système équivaut à trouver la matrice de rotation **R** d'un repère à l'autre, ce qui satisfait

$$\begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \mathbf{R} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{3.70}$$

Comme mentionné précédemment, la rotation autour de la normale du plan du sol (angle de lacet) ne peut pas être récupérée. Nous considérons donc uniquement les angles de tangage et de roulis. La paramétrisation de Denavit-Hartenberg [111] de **R** avec ces deux angles conduit à

$$\begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \begin{bmatrix} cos(\theta) & -sin(\theta)cos(\alpha) & sin(\theta)sin(\alpha) \\ sin(\theta) & cos(\theta)cos(\alpha) & -cos(\theta)sin(\alpha) \\ 0 & sin(\alpha) & cos(\alpha) \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{3.71}$$

De (3.71), α (le roulis) and θ (le tangage) peuvent être retrouvés facilement puisque

$$\begin{cases} \alpha = \arccos(n_z) \\ \theta = \frac{\arcsin(n_x)}{\sqrt{1-\cos^2(\alpha)}} \end{cases}. \tag{3.72}$$

Ils peuvent être réinjectés dans (3.71) pour calculer toute la matrice de rotation **R** qui définit l'attitude du système caméra/laser.

3.3.3 Estimations robustes

La méthodologie présentée dans la section 3.3.2 suppose que le cône associé au projecteur (cône **D** dans la figure 3.13) est connu sans erreur. En effet, pour calculer le cône, nous n'avons pas besoin de faire de traitements d'images. En revanche, le cône associé à la caméra (cône **C** dans la figure 3.13) est calculé en extrayant d'abord une ellipse **c** dans l'image de la caméra. Notez que notre approche est valide dans le cas où **c** est une conique générale ; cependant, dans notre contexte pratique, il s'agit toujours d'une ellipse. Nous considérons donc dans ce qui suit que **c** est une ellipse. Un problème potentiel est que des valeurs aberrantes peuvent affecter l'estimation de l'ellipse. Par exemple, ces valeurs aberrantes peuvent apparaître lorsque le projecteur laser intercepte un plan de sol partiellement obstrué par des objets. Pour continuer à fonctionner dans ce cas, on peut recourir à une approche RANSAC pour calculer l'ellipse **c**.

Dans cette section, nous proposons trois estimations robustes : une basée sur un RANSAC à 5 points pour estimer l'ellipse dans le plan image, une basée sur un RANSAC à 3 points pour estimer l'ellipse en tenant compte de la géométrie épipolaire, et une basée sur une RANSAC à 3 points pour estimer directement le plan du sol (et par conséquent l'altitude et l'attitude de notre système), sans estimer l'ellipse.

L'algorithme Plane-Pair 5-point (PP-5)

La méthode d'estimation de l'altitude et de l'attitude présentée dans la section 3.3.2 nécessite le calcul de l'ellipse $\mathbf{c}$. Dans cette section, nous expliquons comment l'estimer avec tous les points et ensuite avec 5 points en utilisant une approche RANSAC. Cette estimation robuste est dénommée algorithme Plane-Pair à 5 points (PP-5).

Un point $\mathbf{x} = [x \quad y \quad z]^T$ (donné en coordonnées homogènes) appartient à $\mathbf{c}$ si $\mathbf{x}^T\ \mathbf{c}\ \mathbf{x} = 0$. Si on représente $\mathbf{c}$ de manière usuelle par une matrice symétrique

$$\mathbf{c} \sim \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix}, \tag{3.73}$$

l'équation ci-dessus devient

$$ax^2 + bxy + cy^2 + dxz + eyz + fz^2 = 0. \tag{3.74}$$

La représentation matricielle de $\mathbf{c}$ a cinq degrés de liberté : les six éléments de la matrice (3.73) moins un pour l'échelle puisque la multiplication de l'équation (3.74) par un scalaire non-nul n'affecte pas cette équation.

Supposons que nous avons n points ($n >= 5$) appartenant à $\mathbf{c}$. Soit $\mathbf{x}_i = [x_i \quad y_i \quad z_i]^T$ le i$^{\text{ème}}$ point. Nous pouvons construire le système d'équations linéaires

$$\begin{bmatrix} x_1^2 & x_1y_1 & y_1^2 & x_1z_1 & y_1z_1 & z_1^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^2 & x_ny_n & y_n^2 & x_nz_n & y_nz_n & z_n^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \mathbf{0}. \tag{3.75}$$

Les coefficients a, b, c, d, e et f peuvent être obtenus (à l'échelle près) par une décomposition en valeurs singulières de la première matrice de l'équation (3.75).

Les points $\mathbf{x}_i$ sont détectés par une étape de traitement d'images (seuillage et filtrage) où des points aberrants peuvent apparaitre. Une estimation directe comme celle présentée dans cette section conduit généralement à des résultats erronés en présence d'*outliers*. Pour éviter cela, l'algorithme d'estimation d'ellipse est réalisé en utilisant une approche RANSAC pour supprimer les *outliers* potentiels. Ici, 5 points est le nombre minimal requis pour déterminer les coefficients de l'ellipse en utilisant l'équation (3.75).

L'algorithme Plane-Pair 3-point (PP-3)

Trois points ne suffisent pas, en général, pour calculer une ellipse, mais dans notre cas nous avons des informations supplémentaires : nous connaissons la géométrie épipolaire entre

la caméra et le projecteur. Cette géométrie épipolaire fournit des contraintes supplémentaires puisque les deux cônes ($\mathbf{C}$ et $\mathbf{D}$) doivent être tangents aux mêmes plans épipolaires. En considérant la figure 3.15 par exemple, les deux cônes sont tangents au plan formé par les deux centres optiques et les lignes noires sur les cônes. Il y a aussi un deuxième plan épipolaire qui est tangent aux deux cônes, derrière eux.

L'équivalent en 2D est le suivant : considérons le cercle dans le "plan image" du projecteur. Il y a deux lignes épipolaires, c'est-à-dire des lignes qui contiennent l'épipôle, et qui sont tangentes à ce cercle. Les deux lignes épipolaires correspondantes dans l'image de la caméra doivent être tangentes à l'ellipse que nous recherchons dans l'image de la caméra. C'est la contrainte épipolaire pour les images des coniques [112].

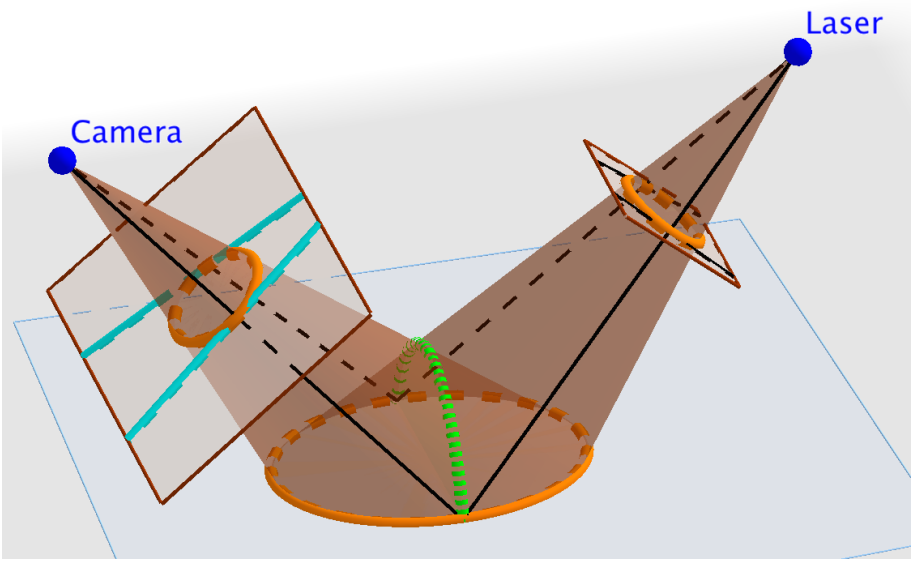


FIGURE 3.15 – Géométrie épipolaire de la caméra et du projecteur laser. Les lignes noires sont tangentes aux cônes par rapport à un plan épipolaire. Leur projection dans l'image de la caméra donne des lignes épipolaires (en bleu) qui doivent être tangentes à la conique c dans l'image.

Comme nous connaissons la pose du laser par rapport à la caméra, nous pouvons calculer directement la matrice fondamentale donnée par

$$\mathbf{F} = \mathbf{K}^{-T}[\mathbf{t}_{las}]_{\times}\mathbf{R}_{las}. \tag{3.76}$$

Les épipôles peuvent alors être déterminées en utilisant la SVD de $\mathbf{F}$. Les épipôles $\mathbf{e}$ dans l'image laser et $\mathbf{e}'$ dans l'image de la caméra sont les vecteurs singuliers à gauche et à droite de $\mathbf{F}$. Il est maintenant possible de calculer les deux lignes tangentes dans "l'image" du laser car nous connaissons l'épipôle par lequel elles passent et l'équation du cercle dans l'image laser. Comme nous connaissons également la matrice essentielle, nous pouvons obtenir l'équation de ces lignes dans l'image de la caméra.

Nous avons donc deux contraintes sur $\mathbf{c}$. Elles ne sont cependant pas triviales à utiliser. Nous proposons la formulation suivante. Soient $\mathbf{u}$ et $\mathbf{v}$ les deux droites épipolaires qui doivent être tangentes à l'ellipse $\mathbf{c}$. En d'autres termes, les deux droites doivent être sur la conique duale à $\mathbf{c}$, ce qui peut être écrit comme

$$\mathbf{u}^T\,\mathbf{c}^{-1}\,\mathbf{u} \;=\; 0 \;\text{ et} \tag{3.77}$$
$$\mathbf{v}^T\,\mathbf{c}^{-1}\,\mathbf{v} \;=\; 0. \tag{3.78}$$

D'autre part, tout point $\mathbf{x}$ qui se trouve sur $\mathbf{c}$, donne la contrainte

$$\mathbf{x}^T\,\mathbf{c}\,\mathbf{x} = 0. \tag{3.79}$$

Si nous considérons 3 points, nous avons donc 3 contraintes linéaires sur $\mathbf{c}$ et 2 contraintes linéaires sur son inverse. La résolution d'un tel système d'équations n'est pas triviale. Pour simplifier les expressions, nous appliquons d'abord une homographie au plan image qui conduit à des coordonnées simples pour les points considérés. Soient $\mathbf{x}_i, i = 1, 2, 3$ les trois points situés sur l'ellipse et $\mathbf{x}_4$ le point d'intersection des deux droites tangentes $\mathbf{u}$ et $\mathbf{v}$, c'est-à-dire l'épipôle $\mathbf{e}'$ dans l'image de la caméra. Soit $\mathbf{H}$ l'homographie qui transforme ces quatre points en

$$\mathbf{x}'_1 = \begin{bmatrix}0\\0\\1\end{bmatrix},\;\; \mathbf{x}'_2 = \begin{bmatrix}1\\0\\1\end{bmatrix},\;\; \mathbf{x}'_3 = \begin{bmatrix}0\\1\\1\end{bmatrix} \;\text{ et }\; \mathbf{x}'_4 = \begin{bmatrix}1\\1\\0\end{bmatrix}. \tag{3.80}$$

Cette homographie est calculée à partir d'équations linéaires de type $(\mathbf{H}\mathbf{x}_i) \times \mathbf{x}'_i = \mathbf{0}$. Pour chacune des quatre paires de points $\mathbf{x}_i = \begin{bmatrix}x_i & y_i & z_i\end{bmatrix}^T$ et $\mathbf{x}'_i = \begin{bmatrix}x'_i & y'_i & z'_i\end{bmatrix}^T$, nous pouvons constuire le système d'équations suivant et le résoudre par SVD :

$$\begin{bmatrix} \mathbf{0}^T & z'_i\mathbf{x}_i^T & -y'_i\mathbf{x}_i^T \\ -z'_i\mathbf{x}_i^T & \mathbf{0}^T & -x'_i\mathbf{x}_i^T \\ y'_i\mathbf{x}_i^T & -x'_i\mathbf{x}_i^T & \mathbf{0}^T \end{bmatrix} \begin{bmatrix}\mathbf{h}_1\\\mathbf{h}_2\\\mathbf{h}_3\end{bmatrix} = \mathbf{0}. \tag{3.81}$$

Dans cette équation, les $\mathbf{h}_i$ correspondent aux lignes de $\mathbf{H}$. Après avoir calculé $\mathbf{H}$, nous l'utilisons pour transformer les deux lignes tangentes comme suit

$$\mathbf{u}' = \mathbf{H}^{-T}\mathbf{u} \tag{3.82}$$
$$\mathbf{v}' = \mathbf{H}^{-T}\mathbf{v}. \tag{3.83}$$

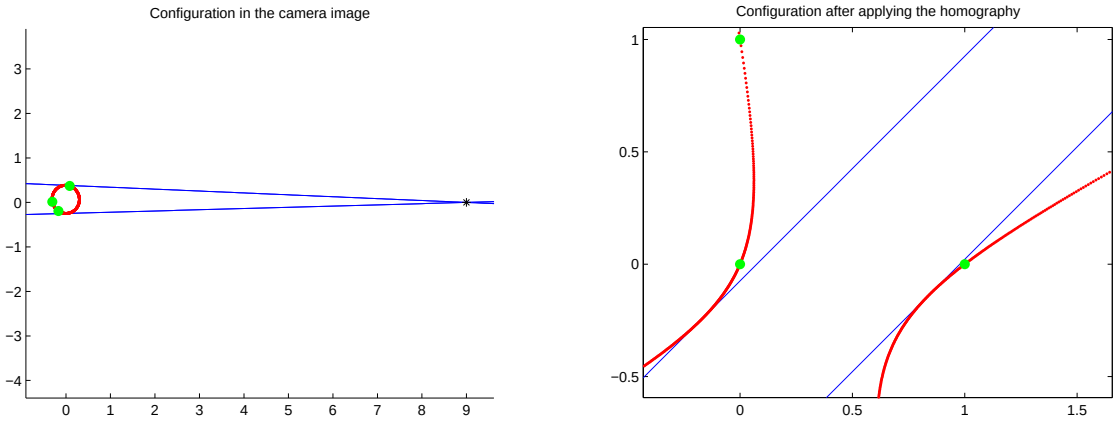
Cette transformation est illustrée par la figure 3.16.

$\mathbf{u}'$ et $\mathbf{v}'$ contiennent le point $[1, 1, 0]^T$, et doivent par conséquent être de la forme

$$\mathbf{u}' \;\sim\; \begin{bmatrix}1\\-1\\r\end{bmatrix} \;\text{ et} \tag{3.84}$$
$$\mathbf{v}' \;\sim\; \begin{bmatrix}1\\-1\\s\end{bmatrix} \tag{3.85}$$

où r et s peuvent être extraits des équations (3.82) et (3.83).

Nous passons maintenant à l'estimation de la conique $\mathbf{c}'$. Premièrement, comme elle contient $\mathbf{x}'_i, i = 1, 2, 3$, avec les coordonnées particulières décrites dans l'équation (3.80), elle doit être de la forme spéciale



(a) La configuration caméra.

(b) La configuration après homographie.

FIGURE 3.16 – Notre problème dans les deux configurations : dans l'image de la caméra et après application de l'homographie. Les 3 points sélectionnés pour estimer $\mathbf{c}$ sont représentés en vert, l'épipôle en noir, les deux tangentes en bleu et la conique en rouge. Notez que sur la figure de droite, obtenue en appliquant la transformation projective $\mathbf{H}$, la conique peut ne pas être une ellipse.

$$\mathbf{c}' \sim \begin{pmatrix} 2v & u & -v \\ u & 2t & -t \\ -v & -t & 0 \end{pmatrix}. \tag{3.86}$$

Sans perte de généralité, on peut fixer le facteur d'échelle homogène pour $\mathbf{c}'$ en fixant $v = 1$ (le seul cas où cela n'est pas autorisé serait si $v = 0$, mais dans ce cas, $\mathbf{c}'$ serait dégénérée, ce cas peut être exclu en toute sécurité dans notre application où, en pratique, nous observerons toujours une ellipse non dégénérée dans l'image de la caméra). Par conséquent, nous définissons

$$\mathbf{c}' = \begin{pmatrix} 2 & u & -1 \\ u & 2t & -t \\ -1 & -t & 0 \end{pmatrix}. \tag{3.87}$$

L'inverse de $\mathbf{c}'$ est, à un facteur d'échelle près, égale à

$$\mathbf{c}'^{-1} \sim \begin{pmatrix} -t^2 & t & -t(u-2) \\ t & -1 & 2t-u \\ -t(u-2) & 2t-u & 4t-u^2 \end{pmatrix}. \tag{3.88}$$

Pour déterminer les deux inconnues t et u, nous utilisons

$$\mathbf{u}'^T \, \mathbf{c}'^{-1} \, \mathbf{u}' = 0 \text{ et} \tag{3.89}$$
$$\mathbf{v}'^T \, \mathbf{c}'^{-1} \, \mathbf{v}' = 0. \tag{3.90}$$

Rendre ces équations explicites donne deux équations quadratiques en t et u :

$$-(t+1)^2 + 2ru(1-t) + r^2(4t-u^2) = 0, \tag{3.91}$$
$$-(t+1)^2 + 2su(1-t) + s^2(4t-u^2) = 0. \tag{3.92}$$

Nous pouvons soustraire les deux équations l'une de l'autre pour obtenir

$$0 = 2ru(1-t) + r^2(4t-u^2) - 2su(1-t) - s^2(4t-u^2), \tag{3.93}$$

$$0 = 2u(1-t)(r-s) + (4t-u^2)(r^2-s^2), \tag{3.94}$$

$$0 = t\left(2u(s-r) + 4(r^2-s^2)\right) + \left(2u(r-s) + u^2(s^2-r^2)\right). \tag{3.95}$$

Cette équation est linéaire en t et nous pouvons la résoudre comme suit :

$$t = \frac{2u(s-r) + u^2(r^2-s^2)}{2u(s-r) + 4(r^2-s^2)}. \tag{3.96}$$

En réinjectant t dans l'équation (3.91) ou (3.92), et en extrayant le numérateur, nous obtenons l'équation de degré 4 en u suivante :

$$\begin{aligned}(r-s)^2u^4 + 8(r+s)(rs-1)u^3 + 8(r^2+4rs+s^2+2)u^2 \\ -32(r+s)(rs+1)u + 16(r+s)^2 = 0.\end{aligned} \tag{3.97}$$

La résolution de l'équation (3.97) conduit à quatre solutions réelles pour u. Pour chaque solution, nous pouvons alors calculer t à partir de l'équation (3.96) et donc une solution potentielle pour $\mathbf{c}'$ à partir de l'équation (3.88). Nous avons alors seulement besoin de retransformer chaque solution dans le plan image d'origine avec

$$\mathbf{c} = \mathbf{H}^T\, \mathbf{c}'\, \mathbf{H}. \tag{3.98}$$

Il est possible d'exclure des solutions parasites pour $\mathbf{c}$, en éliminant les coniques qui ne sont pas des ellipses. Une conique est une ellipse si et seulement si les valeurs propres de la sous-matrice supérieure 2×2 de $\mathbf{c}$ sont toutes deux non-nulles et ont le même signe. Néanmoins, nous pouvons tout de même obtenir plusieurs solutions qui sont des ellipses. Pour obtenir une solution unique, au moins un point de plus est nécessaire. Soit $\mathbf{x}_5$ ce point, la bonne solution est celle où

$$\mathbf{x}_5^T\, \mathbf{c}\, \mathbf{x}_5 = 0. \tag{3.99}$$

Comme la méthode d'estimation avec trois points expliquée ci-dessus est en pratique intégrée dans une approche RANSAC, la sélection d'un quatrième point n'est pas nécessaire. Nous pouvons simplement évaluer toutes les solutions obtenues pour $\mathbf{c}$ qui sont des ellipses, en utilisant tous les autres points images, dans l'étape sélection des points appartenant au consensus.

Une solution minimale : l'algorithme Ground-Plane 3-point (GP-3)

L'estimation de l'ellipse à partir de 3 points est réalisable, comme indiqué dans la section 3.3.3, mais pas forcément très simple. Il s'avère qu'il est plus simple de résoudre directement le problème qui nous intéresse : l'estimation du plan du sol. L'intersection de deux cônes en 3D donne, comme le montrent les figures 3.14 et 3.15, deux coniques en 3D. L'une d'elles est la trace du cercle projeté sur le sol et son plan support est donc le plan du sol, exprimé dans le système de référence dans lequel les cônes sont représentés (le repère de la caméra dans notre cas).

Considérons maintenant 3 points dans l'image de la caméra supposés se trouver sur l'ellipse $\mathbf{c}$. Nous pouvons maintenant rétro-projeter ces 3 points en 3D, c'est-à-dire calculer leurs rayons

correspondants. Nous intersectons ensuite le cône laser $\mathbf{D}$ avec chacune de ces lignes, donnant en général deux points d'intersection chacune. Il y a donc $2^3 = 8$ combinaisons possibles de points 3D associés à nos 3 points images et l'un d'eux doit correspondre à des points situés sur le plan du sol. La sélection de la solution correcte peut être effectuée en utilisant une approche RANSAC, comme expliqué ci-dessous.

Soit $\mathbf{x}$ un point image, supposé appartenir à la conique $\mathbf{c}$, sa rétro-projection donne une ligne en 3D, constituée de points paramétrisés par un scalaire λ. Avec la matrice de projection donnée dans l'équation (3.50), la rétro-projection donne

$$\mathbf{X}(\lambda) = \begin{bmatrix} \lambda \mathbf{K}^{-1}\mathbf{x} \\ 1 \end{bmatrix}. \tag{3.100}$$

Pour trouver les points d'intersection de cette ligne avec le cône du laser, nous devons résoudre l'équation suivante pour λ :

$$\mathbf{X}(\lambda)^T \, \mathbf{D} \, \mathbf{X}(\lambda) = 0$$

où $\mathbf{D}$ est le cône, comme définit par l'équation (3.54). En détail, cela donne l'équation quadratique suivante :

$$\begin{aligned} \lambda^2 & \underbrace{\mathbf{x}^T \mathbf{K}^{-T} \mathbf{R}_{las}^T \mathbf{d} \, \mathbf{R}_{las} \mathbf{K}^{-1} \mathbf{x}}_{c_2} \\ -2\lambda & \underbrace{\mathbf{x}^T \mathbf{K}^{-T} \mathbf{R}_{las}^T \mathbf{d} \, \mathbf{R}_{las} \mathbf{t}_{las}}_{c_1} \\ + & \underbrace{\mathbf{t}_{las}^T \mathbf{R}_{las}^T \mathbf{d} \, \mathbf{R}_{las} \mathbf{t}_{las}}_{c_0} = 0. \end{aligned} \tag{3.101}$$

Soit $\Delta = c_1^2 - c_0 c_2$. Alors,

- si $\Delta < 0$, il n'y a pas de solution réelle et par conséquent pas d'intersection entre le cône et le rayon,
- si $\Delta = 0$, il y a une seule solution réelle ($\lambda = \frac{c_1}{c_2}$), correspondant à une ligne tangente au cône,
- si $\Delta > 0$, il y a deux intersections : $\lambda = \frac{c_1 \pm \sqrt{\Delta}}{c_2}$.

Comme mentionné précédemment, les (jusqu'à) deux points d'intersection par point image rétro-projeté, donnent jusqu'à 8 triplets de points 3D, parmi lesquels un triplet se trouve sur le plan du sol. Pour le déterminer, on peut utiliser des contraintes géométriques (comme celles déjà utilisées auparavant, c'est-à-dire le fait que les centres optiques de la caméra et du laser doivent être du même côté par rapport au plan du sol) et des points images supplémentaires.

Les avantages de cette approche RANSAC basée sur 3 points sont multiples :

- temps de calcul inférieur que l'approche générale basée sur 5 points (moins d'itérations sont nécessaires comme expliqué dans la section 3.3.3),
- une plus grande robustesse comme illustrée dans la section 3.3.5,
- la solution calculée à partir de 3 points satisfait toutes les contraintes géométriques (les contraintes épipolaires) ; cela signifie que l'intersection des cônes sera exacte. Au contraire, si l'on estime d'abord une ellipse générale dans l'image de la caméra, puis qu'on intersecte son cône avec le cône du projecteur, le problème est sur-contraint et la solution ne sera pas une intersection exacte des cônes. La solution numérique obtenue avec une telle méthode à 5 points est alors plus mauvaise que la méthode à 3 points.

3.3.4 Calibrage du système

Le calibrage est une étape nécessaire pour exécuter nos algorithmes sur des données réelles. Dans notre système, nous avons trois éléments à calibrer : le projecteur, la caméra, et la pose relative entre la caméra et le laser.

En ce qui concerne le projecteur, nous supposons que nous connaissons l'angle d'ouverture du cône laser puisqu'il est donné par le fabricant ou qu'il peut être facilement mesuré. La caméra est étalonnée par une méthode conventionnelle, en utilisant une mire de calibrage [113]. Le problème principal réside donc dans l'estimation de la pose relative entre le laser et la caméra. La pose consiste normalement en trois paramètres de translation et trois paramètres de rotation. Puisque le cône laser est circulaire, la rotation autour de son axe est sans importance dans notre application. Par conséquent, seulement deux paramètres de rotation ont besoin et peuvent être déterminés.

Notre procédé utilise une surface plane avec une texture connue, c'est-à-dire une mire de calibrage de type damier. Dans ce cas, la pose de la mire par rapport à la caméra peut être calculée [114]. Il est théoriquement possible d'effectuer le calibrage à partir d'une seule image. Néanmoins, pour de meilleurs résultats, il convient de combiner toutes les images disponibles, en utilisant un ajustement du faisceaux. Nous proposons la méthode suivante dans cet objectif. Nous devons optimiser la pose du cône laser par rapport à la caméra et, pour cela, nous devons définir une fonction coût. Une possibilité consiste à échantillonner les points des ellipses du plan du sol et à minimiser la somme des distances au carré entre les points échantillonnés et les ellipses générées par l'intersection du cône avec le plan du sol, où l'équation du cône est une fonction des paramètres de pose à optimiser. La minimisation de cette somme de distances au carré permet donc d'optimiser les paramètres du cône.

L'optimisation de la fonction de coût proposée peut être réalisée de plusieurs manières ; nous décrivons ici une solution analogue à celle proposée pour ajuster les coniques aux points décrite dans [115]. Elle nécessite d'optimiser, outre les paramètres du cône, un paramètre par point, qui exprime la position de chaque point sur le cône.

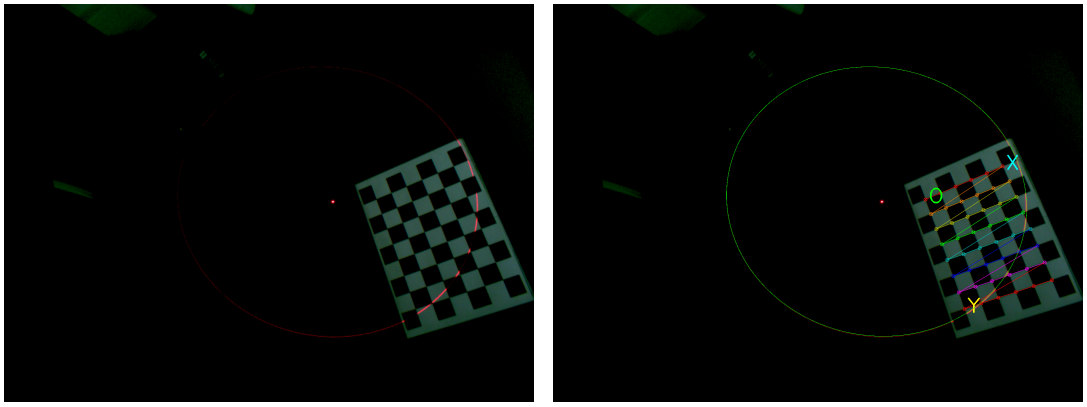
La formulation est la suivante. Considérons un cône en position canonique, avec le sommet à l'origine et avec l'axe Z comme axe de révolution. Les directions des lignes sur le cône peuvent être paramétrées par un angle γ_i telles que

$$\mathbf{D}_i = \begin{bmatrix} \cos\gamma_i & -\sin\gamma_i & 0 \\ \sin\gamma_i & \cos\gamma_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \sin\frac{\Theta}{2} \\ 0 \\ \cos\frac{\Theta}{2} \end{bmatrix}. \tag{3.102}$$

Les inconnues du problème d'estimation de pose sont l'orientation et la position du cône par rapport à la caméra. L'orientation est donnée à une rotation autour de l'axe Z près, ce qui signifie que la rotation globale peut-être exprimée par une rotation atour de Y, suivie par une rotation autour de X. La position peut être représentée simplement comme la position du sommet, donnée par un vecteur $\mathbf{v} = [v_x \; v_y \; v_z]^T$.

Comme pour l'orientation, la direction $\mathbf{D}_i$ est transformée en une direction $\mathbf{D}'_i$ dans le référentiel de la caméra par

$$\begin{aligned} \mathbf{D}'_i &= \mathbf{R}_{las,X}\mathbf{R}_{las,Y}\mathbf{D}_i \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \mathbf{D}_i \end{aligned}. \tag{3.103}$$



(a) Une des images utilisées pour le calibrage.

(b) Détection de la conique sur la mire de calibrage.

FIGURE 3.17 – Notre méthode de calibrage est basée sur l'utilisation d'une mire plane posée au sol. L'image source (a) est traitée pour extraire la mire de calibrage et l'équation de la conique (b).

Enfin, pour une image j, la pose de la caméra par rapport à la mire de calibrage sur le plan du sol est donnée par une matrice de rotation $\mathbf{S}_j$ et un vecteur $\mathbf{t}_j$ tels que les points dans le référentiel de la caméra sont exprimés dans le référentiel de la mire par

$$\mathbf{Q}^g = \mathbf{S}_j \mathbf{Q}^c + \mathbf{t}_j. \tag{3.104}$$

Maintenant, dans le référentiel de la mire, la direction est donnée par

$$\mathbf{D}^g_{ij} = \mathbf{S}_j \mathbf{D}'_i \tag{3.105}$$

et le sommet du cône par

$$\mathbf{t}^g_j = \mathbf{S}_j \mathbf{v} + \mathbf{t}_j. \tag{3.106}$$

Nous devons trouver l'intersection de la ligne donnée par le sommet et la direction, avec le plan du sol (considéré comme le plan $Z = 0$ pour le processus de calibrage). Ceci est simplement donné par le point

$$\mathbf{S}_j \mathbf{v} + \mathbf{t}_j + \lambda_{ij} \mathbf{S}_j \mathbf{D}'_i \tag{3.107}$$

tel que

$$\lambda_{ij} = -\frac{(\mathbf{S}_j \mathbf{v} + \mathbf{t}_j)_3}{(\mathbf{S}_j \mathbf{D}'_i)_3}. \tag{3.108}$$

Les coordonnées XY de ce point sont données par

$$\begin{array}{rcl} X_{ij} & = & (\mathbf{S}_j \mathbf{v} + \mathbf{t}_j)_1 - \frac{(\mathbf{S}_j \mathbf{v} + \mathbf{t}_j)_3}{(\mathbf{S}_j \mathbf{D}'_i)_3} (\mathbf{S}_j \mathbf{D}'_i)_1 \text{ et} \\ Y_{ij} & = & (\mathbf{S}_j \mathbf{v} + \mathbf{t}_j)_2 - \frac{(\mathbf{S}_j \mathbf{v} + \mathbf{t}_j)_3}{(\mathbf{S}_j \mathbf{D}'_i)_3} (\mathbf{S}_j \mathbf{D}'_i)_2. \end{array} \tag{3.109}$$

La fonction coût est la somme des différences au carré entre ces coordonnées XY prédites et celles mesurées (pour les points échantillonnés mentionnés ci-dessus). Pour optimiser cette fonction, nous utilisons l'algorithme de Levenberg-Marquardt [45] qui nécessite de calculer ses dérivées partielles par rapport aux inconnues qui sont : $\alpha, \beta, \mathbf{v}$ et les γ_i, comme le montre la figure 3.18. Pour assurer la convergence de l'algorithme, l'optimisation est réalisée en deux étapes : on n'optimise d'abord que les γ_i avant la ré-estimation de tous les paramètres $(\alpha, \beta, \mathbf{v}, \gamma_i)$.

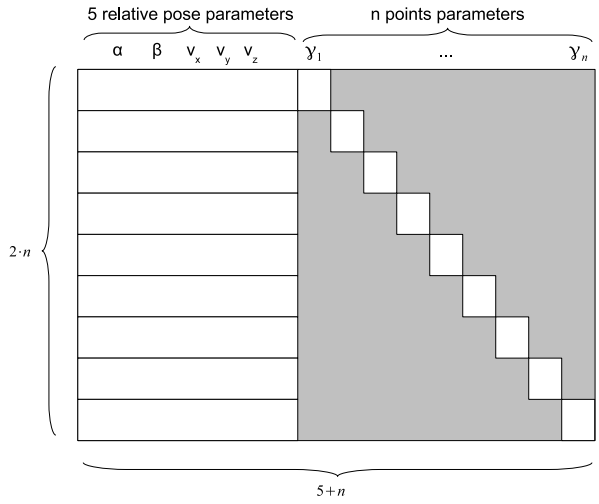


FIGURE 3.18 – Forme de la matrice Jacobienne pour notre formulation qui consiste en 5 paramètres de pose relative et n points. Les valeurs nulles de la matrice sont représentées en gris.

3.3.5 Résultats expérimentaux

Pour vérifier la validité des méthodes proposées, nous avons effectué des expérimentations en utilisant à la fois des données simulées et des images réelles. Ces dernières ont été acquises avec le système caméra/laser et un système de capture de mouvement permettant d'obtenir la vérité terrain pour des comparaisons quantitatives.

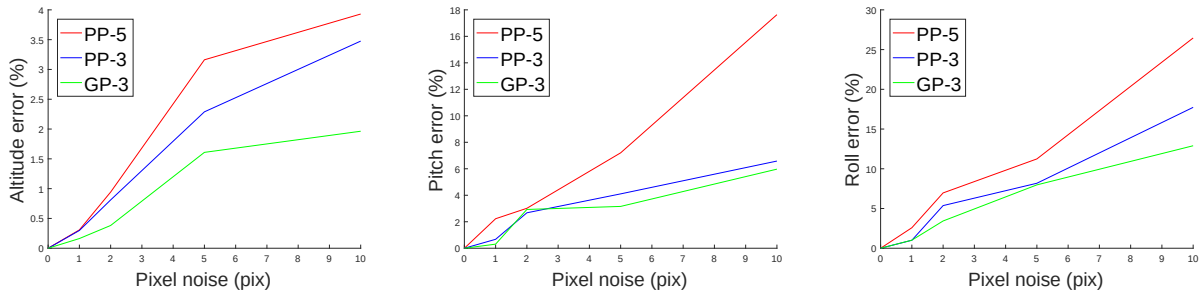
Expérimentations sur des données simulées

Dans ces expérimentations, nous générons un ensemble de points laser sur le plan du sol, compte tenu des paramètres intrinsèques de la caméra et du laser ainsi que de leur pose relative. Nous avons ensuite introduit différents bruits dans les données simulées tels que du bruit image, des valeurs aberrantes, du bruit sur les paramètres intrinsèques et extrinsèques, etc. Les performances des trois algorithmes proposés sont évaluées en comparant l'erreur moyenne d'altitude, et l'erreur des angles de roulis et de tangage estimés lors d'un millier d'essais.

Évaluation en fonction du bruit image

Afin d'évaluer la robustesse des trois algorithmes en présence de bruit image, nous avons ajouté différents niveaux de bruit aux coordonnées des pixels des points images. Nous proposons ensuite de comparer l'erreur moyenne de l'altitude estimée et des angles de roulis et de tangage obtenus à partir des trois méthodes sur un millier d'essais. Les résultats sont donnés sur la figure 3.19.

L'algorithme GP-3 donne les meilleurs résultats pour l'estimation d'altitude alors que pour l'estimation d'attitude (roulis et tangage), les PP-3 et GP-3 ont des performances similaires. Nous pensons que l'algorithme 5 points est le plus sensible car il utilise moins de contraintes que les deux autres approches.



(a) Altitude (b) Pitch (c) Roll

FIGURE 3.19 – Évaluation des algorithmes proposés en fonction du bruit image.

	PP-5	PP-3	GP-3
Proportion d'*outliers*	75 %	86 %	85 %

TABLE 3.5 – Proportion d'*outliers* à partir de laquelle les algorithmes échouent.

Évaluation par rapport au pourcentage de valeurs aberrantes (*outliers*)

Dans cette deuxième expérimentation, nous générons une proportion de données aberrantes (*outliers*) dans l'ensemble de l'image de la caméra. La comparaison n'est pas basée sur des courbes d'erreur car l'estimation conduit à une solution exacte (aucun bruit n'est ajouté aux points *inliers*). Les résultats sont résumés dans la table 3.5 où sont indiquées les proportions de valeurs aberrantes qui provoquent l'échec des algorithmes. Les deux algorithmes PP-3 et GP-3 semblent avoir une robustesse similaire aux valeurs aberrantes.

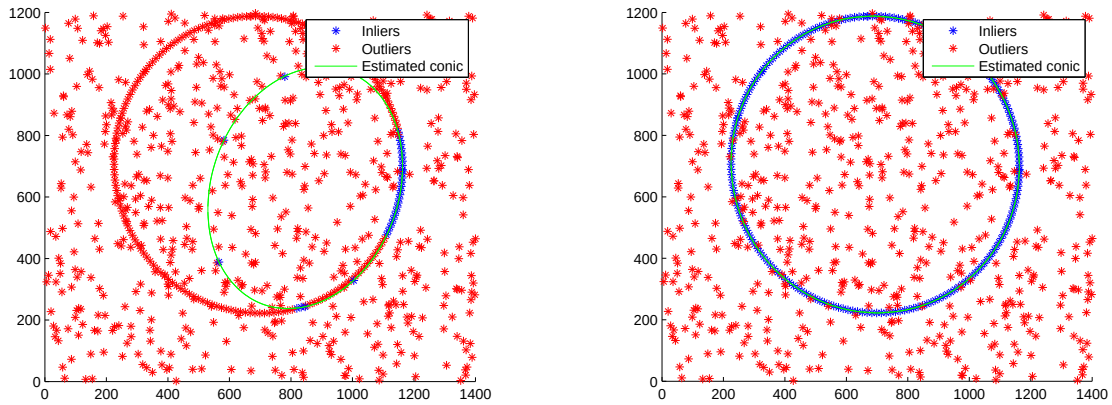
Des exemples d'estimation d'ellipses basées respectivement sur le PP-5 et le PP-3 sont représentés sur la figure 3.20. Ce type de résultat n'est pas proposé pour l'algorithme GP-3 car il n'estime pas une ellipse mais directement le plan du sol. Le principal avantage de notre algorithme PP-3 est qu'il prend en compte les contraintes géométriques (la géométrie épipolaire de notre système) pour estimer l'ellipse. L'introduction de ces contraintes supplémentaires augmente la robustesse de cet algorithme lorsque le nombre de valeurs aberrantes devient très important. Comme le montre la figure 3.20, dans les mêmes conditions de nombre d'itérations et de seuil, l'algorithme PP-3 fournit une bonne estimation de l'ellipse alors que l'algorithme PP-5 conventionnel échoue.

Évaluation en fonction du bruit sur les données de calibrage intrinsèque

Pour cette expérimentation, nous avons introduit le bruit dans les paramètres intrinsèques. Les résultats sont donnés sur la figure 3.21. Comme illustré sur cette figure, les algorithmes PP-3 et GP-3 donnent de meilleurs résultats pour l'estimation d'altitude que le PP-5. Pour l'estimation d'attitude, les trois algorithmes fournissent des résultats similaires.

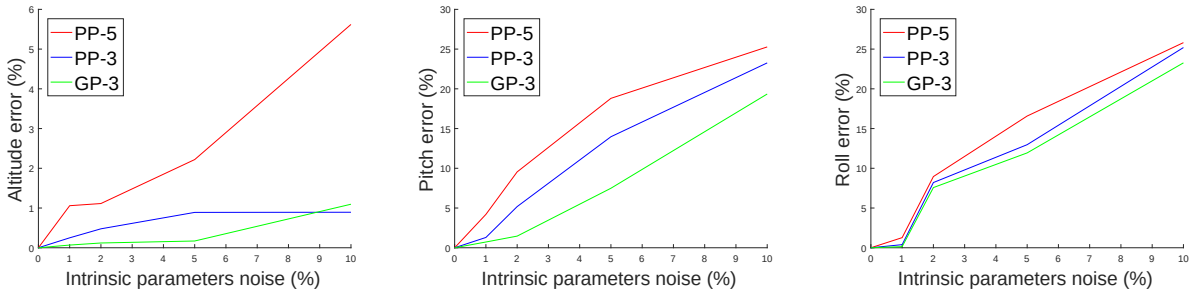
Évaluation en fonction du bruit sur les données de calibrage extrinsèque

Dans cette expérimentation, nous avons introduit du bruit sur la distance entre la caméra et le laser (c'est-à-dire sur la *baseline*). Les résultats sont donnés dans la figure 3.22. Comme illustré sur cette figure, la *baseline* a une influence plus forte sur l'estimation de l'altitude que sur l'attitude. Tous les algorithmes proposés semblent réagir de la même manière pour l'estimation



(a) PP-5 (b) PP-3

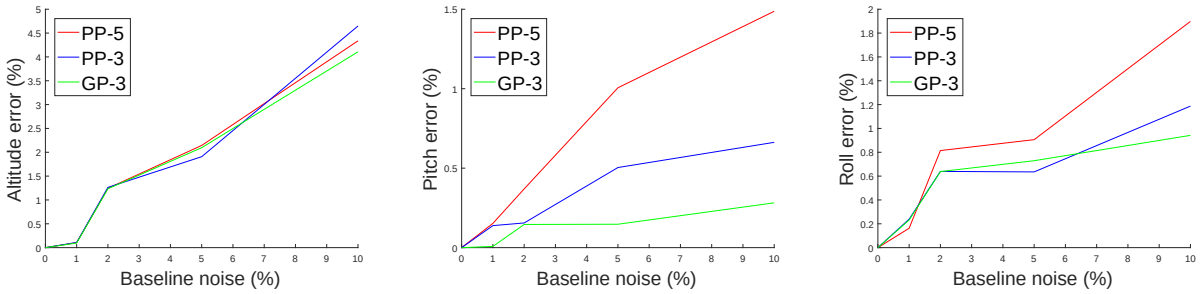
FIGURE 3.20 – Les ellipses estimées en utilisant les algorithmes PP-5 (a) et and PP-3 (b). Proportion d'*outliers* = 75%, Nombre d'itérations RANSAC = 100, Seuil du RANSAC = 0.01.



(a) Altitude (b) Pitch (c) Roll

FIGURE 3.21 – Évaluation des algorithmes proposés en fonction du bruit sur les paramètres intrinsèques.

de l'altitude. Les algorithmes PP-3 et GP-3 donnent de meilleurs résultats pour l'estimation d'attitude que le PP-5.

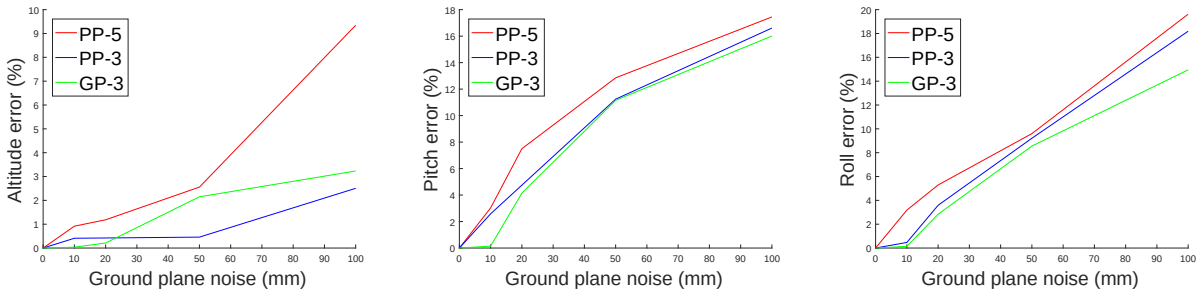


(a) Altitude (b) Pitch (c) Roll

FIGURE 3.22 – Évaluation des algorithmes proposés en fonction du bruit sur la *baseline*.

Évaluation en fonction du bruit sur le plan du sol

En complément des valeurs aberrantes précédemment traitées, nous avons également introduit du bruit dans les coordonnées des points du plan du sol. L'objectif est de simuler ce qui se passerait avec un sol non-uniforme (présence de gravier ou d'herbe). Les résultats sont donnés dans la figure 3.23. Comme illustré sur cette figure, un plan non-uniforme a une forte influence sur les estimations d'altitude et d'attitude. Les algorithmes PP-3 et GP-3 donnent les meilleurs résultats, notamment pour l'estimation de l'altitude.



(a) Altitude (b) Pitch (c) Roll

FIGURE 3.23 – Évaluation des algorithmes proposés en fonction du bruit sur le plan du sol.

Expérimentation sur des données réelles

Afin d'avoir une évaluation en situation réelle de nos algorithmes, un jeu de données a été collecté avec une vérité terrain précise obtenue par un système de capture de mouvement. Les expérimentations ont été menées dans une salle équipée d'un système de capture de mouvement Vicon composé de 20 caméras Vicon T40S. Avec un tel système, nous pouvons assurer une localisation à 6 ddl de notre système avec une précision submillimétrique et à une fréquence élevée (500fps) comme démontré dans [106] et [107].

La caméra utilisée dans ces expérimentations est une caméra couleur uEye d'IDS avec une résolution de 1600x1200 pixels et une fréquence de 60fps. La couleur est utile pour la segmentation du laser dans l'image puisque celui-ci produit une lumière rouge. Le laser est un Z5M18B-F-635-c34 de Z-Laser qui fournit une lumière rouge (635nm) avec une puissance de 5mW. Il est équipé d'une optique circulaire avec un angle d'ouverture de 34°.

Pour l'évaluation de la précision de nos algorithmes, nous avons utilisé un système portatif présenté dans la figure 3.24. La caméra et le laser sont montés sur un trièdre pour faciliter le positionnement des marqueurs du système de capture de mouvement.

En raison de la faible puissance du laser et de la couleur sombre du sol, les expérimentations sont menées dans un environnement sombre. Les lumières ne sont cependant pas totalement éteintes puisque la caméra doit observer la mire de calibrage. La chaîne de traitements pour détecter les points de la conique dans l'image est simple. L'image couleur est d'abord convertie de l'espace RVB vers l'espace HSV. Un seuil fixe est ensuite appliqué uniquement sur le canal H car il contient les informations colorimétriques et nous recherchons la lumière rouge du laser. Il n'y a pas de traitement supplémentaire, les valeurs aberrantes sont directement supprimées en utilisant les trois algorithmes proposés.

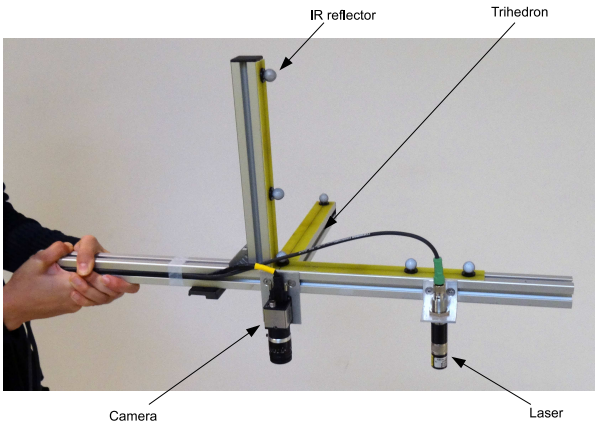
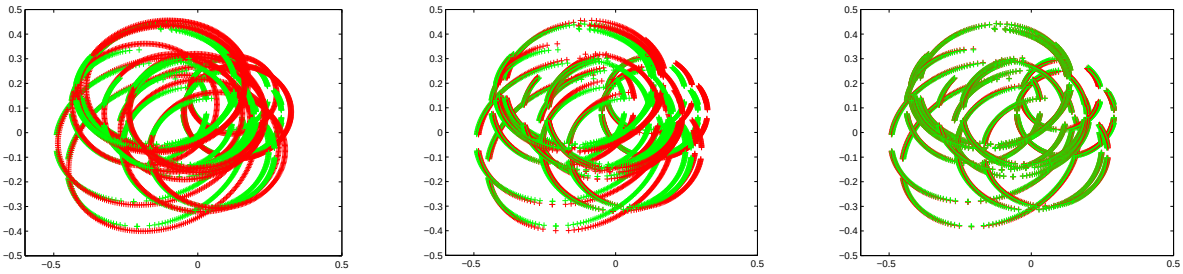


FIGURE 3.24 – Le système portatif utilisé pour l'expérimentation avec le Vicon.

Un premier jeu de données est réalisé pour le calibrage du système, comme expliqué dans la section 3.3.4. Ce jeu de données est composé de 16 images où la projection du laser et une mire de calibrage sont visibles comme le montre la figure 3.17. La pose relative du laser par rapport à la caméra est initialisée en la mesurant grossièrement. Cette première estimation est représentée dans la figure 3.25.a. Des estimations intermédiaires et finales après la convergence de l'algorithme sont montrées respectivement dans la figure 3.25.b et dans la figure 3.25.c. L'erreur moyenne après le calibrage est inférieure à 1,6mm par point.



(a) Projection estimée (en rouge) et points réels (en vert) de la conique sur le plan du sol avec les valeurs initiales.

(b) Projection estimée (en rouge) et points réels (en vert) de la conique sur le plan du sol après la convergence des γ_i.

(c) Projection estimée (en rouge) et points réels (en vert) de la conique sur le plan du sol après la convergence de tous les paramètres (points et pose relative).

FIGURE 3.25 – Résultats des trois étapes du processus de calibrage.

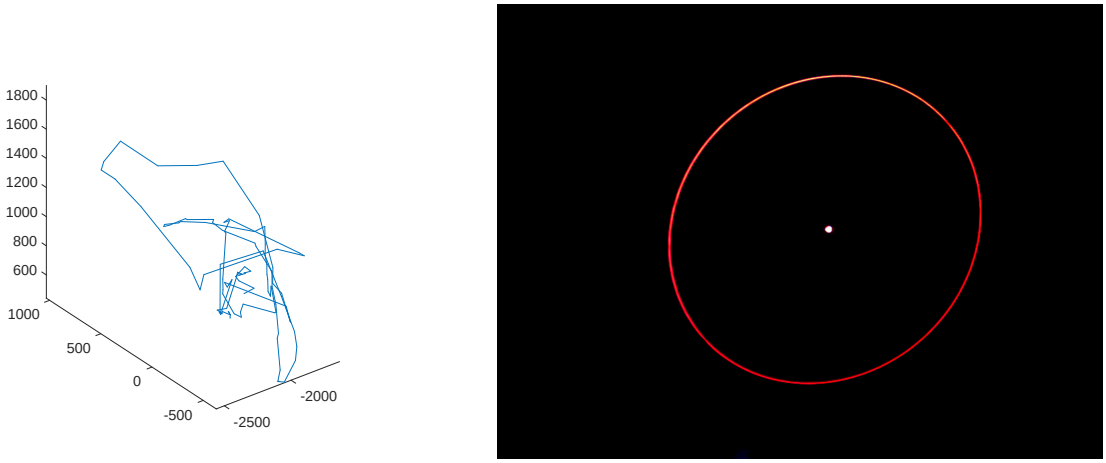
Un deuxième jeu de données composé de 106 images a ensuite été réalisé sans la mire de calibrage. La trajectoire de ce deuxième jeu de données est représentée dans la figure 3.26. La vérité terrain est donnée par le système Vicon. Les résultats de nos algorithmes sont donnés dans la figure 3.27 et dans la table 3.6.

Comme nous pouvons le voir, les trois algorithmes fournissent une estimation fiable de l'altitude et de l'attitude de notre système. Les algorithmes PP-3 et GP-3 ont des performances

	PP-5	PP-3	GP-3
Erreur d'altitude (mm)	11.28 ± 5.91	7.90 ± 4.51	7.52 ± 4.12
Erreur de tangage (pitch) (°)	1.19 ± 0.86	0.67 ± 0.39	0.66 ± 0.37
Erreur de roulis (roll) (°)	1.25 ± 0.89	0.78 ± 0.41	0.76 ± 0.36

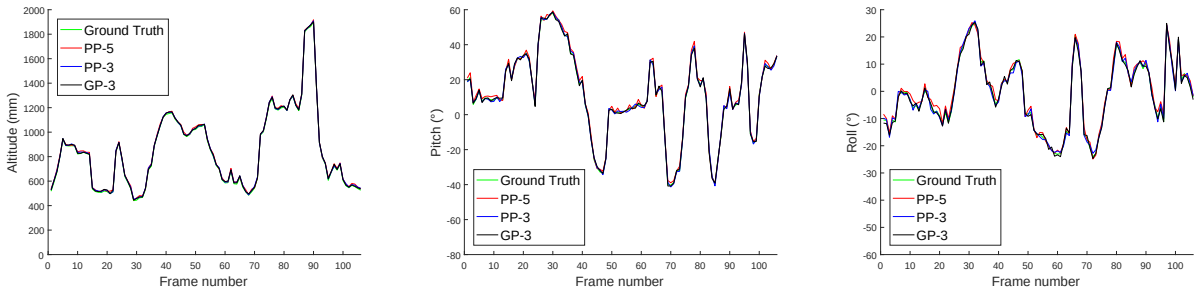
TABLE 3.6 – Erreurs d'altitude, de tangage (pitch) et de roulis (roll) de l'expérimentation réelle.

similaires et offrent une meilleure précision que l'algorithme PP-5.



(a) La trajectoire réalisée dans l'expérimentation réelle (b) Un exemple d'image acquise par la caméra.

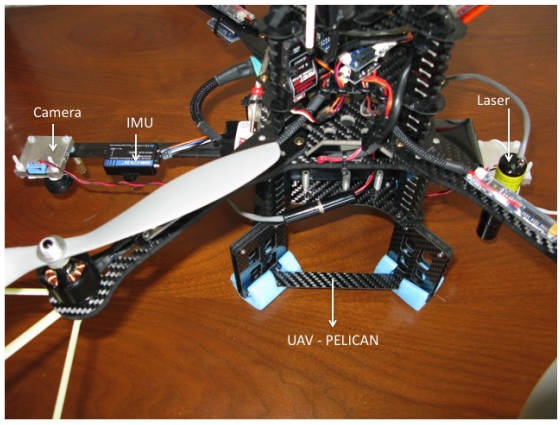
FIGURE 3.26 – L'expérimentation réelle.



(a) Altitude estimée (b) Tangage (pitch) estimé (c) Roulis (roll) estimé

FIGURE 3.27 – Résultats de l'expérimentation réelle.

Comme indiqué précédemment dans [116], notre système peut également être monté sur un drone avec une *baseline* similaire au système utilisé dans l'expérimentation réelle. Cette expérimentation visait à démontrer la faisabilité d'une application de positionnement d'un drone comme le montre la figure 3.28.



(a) Notre système monté sur un drone Pelican.

(b) Le drone Pelican dans la salle VICON. L'espace disponible pour la navigation est de 15x10x5m^3 équipé avec 20 caméras Vicon T40S.

FIGURE 3.28 – L'expérimentation avec le drone (voir [116] pour plus de détails).

3.4 Conclusion

Dans ce chapitre, nous avons présenté deux travaux originaux sur l'estimation de l'odométrie visuelle et de poses à partir de solutions minimales. L'enjeu principal de ces approches est d'améliorer la robustesse et le temps de calcul de la solution correcte dans un processus de type RANSAC.

Le premier travail est basé sur le calcul d'une homographie en connaissant le vecteur de gravité entre deux images calibrées et en posant des hypothèses sur l'environnement. Cela a permis de formuler différents algorithmes nécessitant moins de points que les méthodes de l'état de l'art. Ces algorithmes permettent d'améliorer la performance du RANSAC associé, ce dernier étant une brique logicielle fondamentale dans n'importe quel système d'odométrie visuelle ou de SfM. Dans ces travaux, nous avons mené de nombreuses expérimentations avec des données simulées et des données réelles, ce qui a permis de prouver que cette méthode est parfaitement utilisable. Nos expérimentations réelles démontrent en particulier que les hypothèses sur le plan du sol ou sur les murs verticaux sont parfaitement plausibles dans des scénarios réels. Nos algorithmes ont été utilisés avec succès pour de la navigation robotique en environnement intérieur aussi bien que pour de la reconstruction 3D à partir d'images aériennes prises par un UAV. Ces travaux sont le fruit d'une collaboration internationale (ETH Zürich, LITIS, IRSEEM, LE2I, TU Graz) et ont fait l'objet d'une publication dans [2].

Le second travail porte sur l'estimation de la pose relative d'un drone par rapport au plan du sol. Différentes approches ont été proposées pour estimer l'altitude et l'attitude d'un système mobile équipé d'un laser circulaire et d'une caméra. Nous avons proposé une formulation géométrique et trois méthodes robustes pour estimer la pose à partir de 3 ou 5 points. Les résultats des expérimentations sur les données simulées et sur les données réelles montrent que les deux approches basées sur 3 points sont plus robustes car elles utilisent des contraintes additionnelles pour résoudre le problème. Une nouvelle méthode de calibrage, basée sur un ajustement de faisceaux avec un paramètre par point a également été proposée pour estimer la pose relative entre le laser et la caméra. Ces travaux sont le fruit d'une collaboration nationale entre

quatre laboratoires (INRIA, LITIS, LE2I, IRSEEM) et ont fait l'objet d'une publication dans [3].

Chapitre 4

Localisation 6 ddl basée lidar

Sommaire

4.1 Introduction et contexte 118
4.2 État de l'art de la localisation en milieu industriel 119
4.2.1 SLAM et localisation 119
4.2.2 Technologies en milieu industriel 120
4.2.3 Travaux analogues basés lidar 120
4.2.4 Contraintes de l'environnement par rapport aux solutions existantes . . 121
4.3 Méthodologie 121
4.3.1 Notions sur la théorie de la localisation 121
4.3.2 Méthode de filtrage retenue 122
4.3.3 Champ de vraisemblance 3D 124
4.3.4 Stockage du champ de vraisemblance 126
4.4 Résultats expérimentaux 127
4.4.1 Performance du stockage du champ de vraisemblance par l'octree hybride 129
4.4.2 Caractéristiques de la fonction de vraisemblance 132
4.4.3 Etude de la convergence de l'algorithme 134
4.4.4 Localisation en simulation 136
4.4.5 Evaluation des performances sur robots réels 137
4.5 Conclusion 139

4.1 Introduction et contexte

Ce chapitre est consacré à la localisation pour le véhicule autonome en utilisant des capteurs lidar (*LIght Detection and Ranging*). Cette technologie s'est massivement diffusée ces dernières années dans le domaine du véhicule autonome [117] [118] car elle a des avantages indéniables par rapport à la vision par ordinateur : les lidars fournissent directement une information sur la profondeur de la scène, ils sont moins sensibles aux variations de luminosité que les systèmes de vision et ils fournissent pour certains un nuage de points sur 360° avec des portées pouvant aller jusqu'à 800m.

Mes premiers travaux sur la localisation basée lidar ont été réalisés dans le cadre du projet NAVALIS. L'objectif de ce projet était le développement d'un système de stabilisation de mouvements d'une passerelle de navire pour les opérations de maintenance en mer. Dans ce contexte, nous avons développé un algorithme de localisation robuste de la passerelle du navire par rapport à une éolienne. Ces travaux ont fait l'objet de publications dans [119] et [120]. Je me suis ensuite consacré à la localisation par filtrage particulaire à partir d'un lidar mononappe dans le cadre des projets VIATIC et VIATIC^2 où nous avions besoin de piloter une flotte de robots mobiles.

Mes principaux travaux sur la localisation lidar sont sans nul doute ceux sur la localisation dans un milieu complexe (cf figure 4.1) à partir d'un lidar multi-nappes. Ces travaux ont été menés dans le cadre de la thèse de Pierre Merriaux [Th2] et de notre participation au challenge international de robotique ARGOS [1, 2] (*Autonomous Robot for Gas and Oil Sites*) organisé par Total, en partenariat avec l'ANR. Cinq équipes originaires d'Autriche et d'Allemagne (ARGONAUTS), d'Espagne et du Portugal (FOXIRIS), de France (VIKINGS), du Japon (AIR-K) et de Suisse (LIO) se sont confrontées pour créer des robots capables d'effectuer des tâches d'inspection, détecter des anomalies et intervenir en situation d'urgence. L'IRSEEM s'est allié avec la PME Sominex pour répondre à ce challenge en formant le consortium VIKINGS. Notre consortium a remporté les deux premières manches de cette compétition en trois manches qui s'est déroulée de septembre 2014 à mars 2017.

L'environnement industriel (plateforme pétrolière) dans lequel devait se déplacer le robot était complexe à plusieurs titres : le milieu ne correspondait ni aux environnements intérieurs, ni aux environnements extérieurs de la littérature, les installations étaient complexes (nombreux tuyaux), et l'espace d'évolution comportait plusieurs niveaux, avec des escaliers pour passer d'un niveau à l'autre. Toutes ces contraintes nous ont poussés à développer un nouvel algorithme de localisation à 6 degrés de liberté qui devait être robuste et temps-réel. Nos travaux ont donné lieu à deux principales contributions :

- la conception et la mise en œuvre d'une solution de localisation précise d'un robot mobile pour le challenge de robotique Argos. Cette solution de localisation à 6 degrés de liberté, robuste et rapide, est basée sur l'extension du concept de champ de vraisemblance à la 3D,
- la proposition d'une solution pour un stockage efficace de la carte, appelée "Octree Hybride".

Ces travaux sur la localisation avaient pour but de mener au développement d'un robot autonome d'inspection étant capable d'effectuer différentes mesures :

1. `http://argos-challenge.com`
2. `https://www.youtube.com/watch?v=kdx-DFI1VuA`

FIGURE 4.1 – Le robot Viking évoluant dans son environnement complexe.

— mesures de pression sur des manomètres,
— lecture de la position de vannes,
— détection et localisation de sources anormalement chaudes
— mesures de températures,
— analyses sonores des pompes,
— détection d'inconsistance par rapport à la carte 3D,
— détection de fuites de gaz...

Les différentes missions se déroulaient à travers des modes télé-opérés ou autonomes. En terme de sécurité, le robot devait pouvoir éviter toute collision avec la structure ainsi qu'avec les obstacles dynamiques, et réagir à l'alarme sonore d'évacuation ou encore aux coupures de réseau. D'un point de vue mobilité, il devait également être capable de contourner ou de négocier les obstacles rencontrés et de franchir les escaliers. Tous ces points constituent un aperçu des 87 pages du règlement émis par Total et l'ANR.

Après une présentation de l'état de l'art de la localisation en milieu industriel, nous exposerons la solution mise en œuvre à partir de la mesure d'un lidar multi-nappes.

4.2 État de l'art de la localisation en milieu industriel

4.2.1 SLAM et localisation

Nous pouvons commencer par séparer deux cas : celui où les environnements sont inconnus, et celui pour lequel nous disposons d'une carte *a priori*.

Si l'environnement est inconnu, la problématique est connue sous le nom de SLAM (*Simultaneous Localization And Mapping*). Le robot doit simultanément construire la carte de l'environnement et se localiser dans celle-ci. Les premiers travaux autour du *SLAM* proposent des solutions pour des déplacements plans dans des environnements simples et structurés tels que des bureaux et des couloirs [121]. Les surfaces planes ne nécessitent que 3 ddl pour réprésenter l'état du robot (x,y,θ). Les applications industrielles et en extérieur nécessitent plus de degrés de liberté. Par exemple, le SLAM 6 ddl a été étudié entre autres par [122] pour l'exploration de mines abandonnées. La principale problématique du SLAM est la déformation des cartes résultant de l'accumulation des erreurs de localisation et de cartographie. Des fermetures de boucle sont par conséquent nécessaires pour obtenir une carte correcte de l'environnement. Un

état de l'art récent des techniques de SLAM peut être trouvé dans [123] .

Si l'environnement est connu *a priori*, la problématique est celle de la localisation uniquement. La plupart des sites industriels sont numérisés avec précision pour des raisons opérationnelles de maintenance, de formation ou d'évolution. Il est donc intéressant de tirer partie de ces cartes qui sont généralement plus précises que celles pouvant être obtenues par le SLAM.

Nous n'abordons pas dans ces travaux le domaine du SLAM, mais celui de la localisation basée sur une connaissance *a priori* de l'environnement 3D. Cette localisation doit déterminer les 6 degrés de liberté et être adaptée aux milieux complexes avec de fortes contraintes de robustesse et d'embarquabilité.

4.2.2 Technologies en milieu industriel

Dans le cas d'environnements simples caractérisés par un sol plan et en milieu intérieur, il existe de nombreuses solutions pour localiser un mobile en milieu industriel. Une solution simple consiste à suivre une ligne optique ([124]) ou magnétique ([125]). En plus de nécessiter l'installation d'une infrastructure, ces méthodes restreignent les déplacements des mobiles à des tracés spécifiques, ce qui ne permet pas de tirer complètement parti des apports de la robotique mobile, comme par exemple le contournement d'obstacles.

D'autres solutions ont alors émergé pour se libérer de cette trajectoire qu'il n'était pas possible de quitter. [126] et [127] proposent une localisation laser avec des balises connues qu'il faut répartir dans l'usine. Les balises ont cependant deux inconvénients : elles doivent être visibles par le capteur et il est nécessaire d'instrumenter l'environnement.

Dans leur feuille de route pour favoriser le développement des AGV (*Automated guided vehicle*) dans l'industrie, [128] mentionne le fait que l'utilisation de balises artificielles à installer est un frein à l'expansion de ces applications robotiques. Une localisation basée sur des amers naturels permettrait une bien plus grande souplesse.

La localisation basée sur les balises n'est pas possible dans notre cas de figure car il serait extrêmement complexe et coûteux d'installer des balises dans un environnement pétrochimique de ce type, et les 6 ddl nécessaires à la navigation nécessiteraient un nombre très important de balises pour s'assurer d'être vues par le lidar.

4.2.3 Travaux analogues basés lidar

Pour pallier ces inconévients, [129] propose une localisation basée sur l'extraction d'amers naturellement présents dans les bâtiments (lignes, coins). Cependant, les hypothèses fortes sur les propriétés des amers rencontrés, associées au champ de vue limité des lidars mononappes, réduisent très nettement le potentiel de telles méthodes sur des sites pétrochimiques comme le montre la figure 4.1.

La communauté s'est beaucoup intéressée ces derniers temps aux environnements simples mais dynamiques comme [130] qui propose de répondre à la localisation d'un robot en 2D sur un parking entouré de bâtiments tout en étant robuste à son taux de remplissage.

Les scènes complexes statiques sont également un sujet d'intérêt : [131] et [132] localisent des robots à vocation agricole en extérieur dans des vergers. Les scènes sont complexes d'un point de vue microscopique, mais macroscopiquement, les rangées d'arbres bien taillés pourraient s'apparenter à des couloirs végétaux.

Les travaux de [133] se déroulent en intérieur ; il propose de localiser un mobile dans une laiterie avec un lidar HDL32. L'environnement est nettement plus complexe que des bureaux ou

un parking, mais il est tout de même composé de grands plans formés par les murs du bâtiment et le déplacement se fait en 2D. Sur ce point, les environnements d'ARGOS nous semblent plus compliqués, avec leurs multiples niveaux et la densité des équipements présents dans la scène.

4.2.4 Contraintes de l'environnement par rapport aux solutions existantes

Par rapport à la littérature, nous ne pouvons pas dire que nous sommes en milieu intérieur. L'environnement n'est pas constitué majoritairement de grands plans que constituent généralement les murs et les sols. Nous ne nous situons pas non plus dans ce qui est communément appelé environnement extérieur qui représente typiquement soit des scènes routières, soit des robots tout-terrain en milieu naturel. Le fait que l'environnement ARGOS soit constitué majoritairement de tubes, de poteaux, et d'autres surfaces arrondies ne facilite pas la tâche du lidar. Sur une surface courbe, comme un tube, l'écho est généralement plus faible que sur des plans à cause de l'angle d'incidence du rayon laser.

Dans la littérature, de nombreux travaux sur l'acquisition de données 3D se font en utilisant un lidar motorisé sur un axe comme [122] ou encore [134] qui s'intéressent à des scènes dynamiques intérieures. L'utilisation de ce type de capteur - sauf si une correction des scans est apportée comme le fait [135] ou que nous proposons dans [136] - impose d'arrêter le robot le temps de l'acquisition ; ce qui est incompatible avec la notion de temps-réel visée par les applications d'ARGOS. De plus, la plupart des méthodes d'appariement de nuages de points que nous trouvons sont basées sur l'ICP (*Iterative Closest Point*). La version de base de cet algorithme est lente et sensible aux variations de la scène. Malgré les améliorations apportées, il est peu adapté à une utilisation temps-réel dans un robot aux ressources calculatoires limitées.

4.3 Méthodologie

Pour pallier tous les problèmes que nous venons de mentionner, nous avons proposé une nouvelle méthode de localisation basée lidar. Notre principale contribution consiste en l'extension du concept de champ de vraisemblance à un environnement en 3 dimensions. Nous avons également proposé une méthode originale pour son stockage en mémoire, ainsi que des solutions d'implémentation afin de respecter les contraintes d'embarquabilité. Il est intéressant de noter que notre méthode a démontré sa performance dans un milieu industriel pétrochimique. Des systèmes de localisation, fonctionnant dans des environnements de complexité similaire, sont à ce jour peu développés dans la littérature.

4.3.1 Notions sur la théorie de la localisation

Le rôle de la localisation est de déterminer la transformation entre un repère fixe global associé à la connaissance de l'environnement, et un repère mobile sur le robot (figure 4.2). Cette transformation est exprimée par un vecteur d'état $\mathbf{X_t}$. Nous utilisons une carte $\boldsymbol{\mathcal{M}}$ comme connaissance *a priori* de l'environnement. Le capteur nous retourne des mesures $\mathbf{Z_t}$ de cet environnement. Le vecteur d'état est défini par :

$$\mathbf{X_t} = \begin{bmatrix} x & y & z & \psi & \theta & \varphi \end{bmatrix}^T \tag{4.1}$$

où :

x, y, z : est la position en mètres (m),
ψ, θ, φ : est l'orientation en radians (rad).

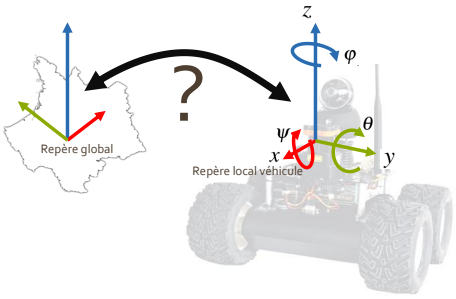


FIGURE 4.2 – Le rôle de la localisation est de déterminer la transformation entre un repère fixe global et un repère mobile sur le robot.

Les méthodes de localisation, quelles que soient leurs modalités, visent à identifier le vecteur d'état le plus probable parmi plusieurs hypothèses. La fonction de vraisemblance exprime la probabilité $\mathbb{P}(\mathbf{Z}|\mathbf{X}, \boldsymbol{\mathcal{M}})$ d'obtenir une mesure $\mathbf{Z}$ pour un état donné $\mathbf{X}$ avec notre connaissance *a priori* de l'environnement $\boldsymbol{\mathcal{M}}$. Elle permet de vérifier que l'hypothèse de position $\mathbf{X}$ est vraisemblable avec la mesure actuelle $\mathbf{Z}$.

Idéalement, la fonction de vraisemblance devrait être :

— discriminante entre plusieurs hypothèses de vecteur d'état, afin de ne conserver que les plus vraisemblables,
— monotone et sans maxima locaux, afin de converger vers la bonne solution,
— avec un gradient modéré en direction de la solution, afin de faciliter la convergence des hypothèses proches de la solution,
— peu coûteuse en temps de calcul, afin de tester un grand nombre d'hypothèses rapidement,
— indépendante entre les états, afin de pouvoir faire converger chaque dimension séparément.

Avant d'étudier la fonction de vraisemblance, nous allons aborder la méthode de filtrage retenue.

4.3.2 Méthode de filtrage retenue

Il existe dans la littérature de la localisation plusieurs types de filtres bayésiens (et surtout plusieurs variantes de chacun) afin de sélectionner les hypothèses émises :

— les filtres de Kalman et Kalman étendu, mais les modèles d'incertitudes et surtout d'états doivent forcément être gaussiens,
— le filtre d'histogramme ; sa complexité de calcul évolue quadratiquement en fonction du nombre de dimensions. Pour des questions de temps de calcul, il est donc délicat de le porter sur un problème à 6 ddl,
— le filtre particulaire n'est pas limité à une distribution unimodale de probabilité du vecteur d'état. L'espace du vecteur d'état est continu contrairement au filtre d'histogramme. Le nombre de particules par rapport aux dimensions du problème tend à être exponentiel plutôt que quadratique pour le filtre d'histogramme ([137]).

Nous avons donc implémenté un filtre particulaire standard ([137]), pour estimer le vecteur d'état $\hat{\mathbf{X}}_\mathbf{t}$. Son rôle est d'émettre des hypothèses de $\hat{\mathbf{X}}_\mathbf{t}$, appelées particules, et de les sélectionner

à l'aide de la fonction de vraisemblance pour converger vers une solution. $\hat{\mathbf{X}}_\mathbf{t}$ est le résultat du barycentre des particules.

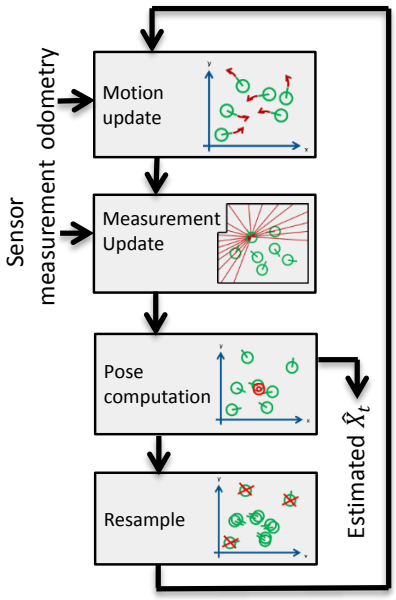


FIGURE 4.3 – Étapes principales du filtre particulaire pour la localisation.

Les principales étapes du filtre particulaire illustrées sur la figure 4.3, sont les suivantes :

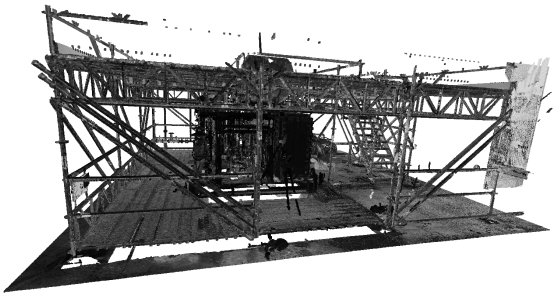
Prédiction/*Motion update* : nous utilisons un modèle cinématique à 3 dimensions pour déplacer les particules (et donc émettre de nouvelles hypothèses de $\mathbf{X_t}$). Nous considérons que la mesure du mouvement du mobile est contaminée par un bruit gaussien proportionnel au déplacement qui est modélisé par une distribution normale $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_p)$. Les erreurs de modélisation de la cinématique sont quant à elles modélisées par un bruit additif $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_a)$. Nous considérons que ces bruits sont non-corrélés entre les différentes variables du vecteur d'état du robot. Par conséquent, Σ_p et Σ_a sont des matrices diagonales définies respectivement par les vecteurs σ_p^2 et σ_a^2.

Correction/*Measurement update* : une mesure du capteur permet à la fonction de vraisemblance d'évaluer la probabilité de chaque hypothèse (le poids de chaque particule). Dans notre cas, la probabilité $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\mathcal{M}})$ est déterminée en utilisant l'équation (4.5) à partir de la mesure lidar.

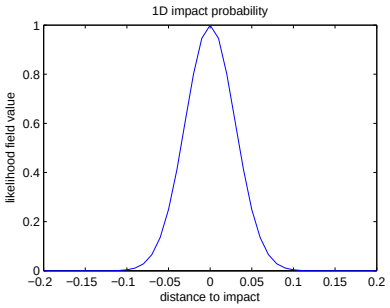
Calcul de la pose : $\hat{\mathbf{X}}_\mathbf{t}$ est le résultat du calcul du barycentre de $\mathbf{X_t}$ pour chaque particule.

Ré-échantillonnage/*Resample* : les meilleures particules sont sélectionnées en fonction de leur poids puis dupliquées en utilisant une méthode de ré-échantillonnage systématique.

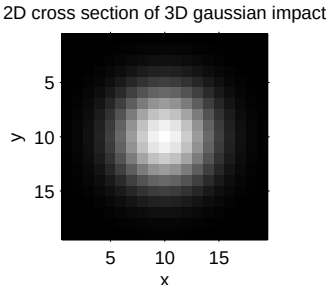
La section suivante présente la fonction de vraisemblance retenue ainsi que son implémentation en proposant l'extension du concept de champ de vraisemblance (*Likelihood field*) en 3 dimensions.



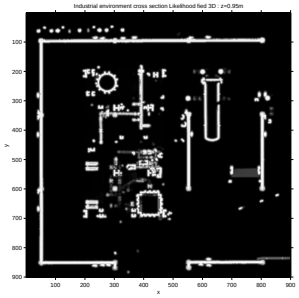
(a) Nuage de points issu de la numérisation du rez-de-chaussée du site industriel de la compétition. ©Total S.A.



(b) Champ de vraissemblance 1D d'impact lidar.



(c) Coupe 2D d'un champ de vraissemblance gaussien 3D issu d'un impact lidar local.



(d) Coupe du résultat du champ de vraissemblance de l'environnement complet.

FIGURE 4.4 – Exemple de construction d'une carte avec le champ de vraisemblance 3D : à partir du nuage de points issu de la numérisation de l'environnement (a), nous utilisons la variance du lidar (b), pour calculer un champ de vraisemblance local 3D (c), puis nous fusionnons l'ensemble des points dans un champ de vraisemblance global (d).

4.3.3 Champ de vraisemblance 3D

Il existe de nombreuses méthodes pour corréler un scan lidar avec une carte. Par principe, ces méthodes sont très dépendantes de la manière dont est représenté le monde dans ladite carte ; la réciproque étant également vraie. Par exemple, [138] utilise l'information 3D et la réflectivité infrarouge lidar pour localiser un véhicule dans une carte contenant une "image de réflectivité". D'autres méthodes, appelées "*Multi-Level Surface Maps*" ([139]), stockent dans leurs cartes les hauteurs de la surface du terrain qui seront comparées avec les impacts lidar.

Nous avons démarré ces travaux avec une approche similaire à ce que propose [140]. Une simulation en trois dimensions de l'environnement permet de calculer une erreur entre la mesure lidar et l'intersection de facettes dans la scène. C'est-à-dire que pour chaque raie lidar, une opération de lancé de rayon (*ray-tracing*) est effectuée, ce qui permet de déterminer l'erreur de distance entre la mesure du capteur et la représentation dans la carte. L'opération est à effectuer pour chaque hypothèse de pose, la combinaison de ces erreurs sur chaque raie donnant une image de la probabilité recherchée $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\mathcal{M}})$. Deux problèmes sont récurrents dans ce type d'approche :

— Comme l'environnement est constitué principalement d'une multitude de tuyaux de faible

diamètre, il faut un nombre très conséquent de facettes pour représenter correctement la scène. La simulation de lancé de rayons déjà gourmande par principe en temps de calcul s'en retrouve encore plus ralentie. A titre d'exemple, [140] n'a conservé que les plans supérieurs à $1m^2$ de l'environnement. Il a ensuite sous-échantillonné le capteur RGB-D Kinect à 20×15 mesures de distance. Malgré cela, la fréquence de rafraîchissement ne dépasse pas 10Hz.

— Un autre point problématique est le fait que l'environnement soit constitué essentiellement d'un grand nombre de petits objets et non de surfaces continues importantes. Nous obtenons une fonction de vraisemblance qui est très sensible à une variation de $\mathbf{X}$. Les rayons lancés vont impacter à côté de l'élément de structure voulu conduisant à une erreur de distance mesurée importante. Par exemple, un très faible écart en angle entraînera une très grande variation du résultat de vraisemblance. Par conséquent, une hypothèse proche de la solution sera rejetée du fait de son faible score. [137] explique ce phénomène en deux dimensions avec un lidar placé dans une salle de classe au niveau des pieds de chaises. Une faible variation d'angle entraîne une forte variation de la scène perçue au sens des distances d'impact raie par raie.

Pour faire face à ces deux problèmes, nous proposons d'étendre l'utilisation du champ de vraisemblance (*likelihood field*) proposé par [137], de deux dimensions à trois dimensions. L'environnement est discrétisé et représenté par une grille 3D de voxels, et pour chaque objet constituant la scène, nous calculons la probabilité d'impact lidar en utilisant une distribution normale :

$$p_{hit}(\mathbf{z}|m, O^k_{\boldsymbol{\mathcal{M}}}) = \frac{1}{\sigma_{map}\sqrt{2\pi}} \ \mathrm{e}^{-\frac{1}{2}\left(\frac{d}{\sigma_{map}}\right)^2} \tag{4.2}$$

où :

m	: Voxel,
d	: Distance euclidienne du voxel m au $k^{\text{ème}}$ obstacle $O^k_{\boldsymbol{\mathcal{M}}}$ dans $\boldsymbol{\mathcal{M}}$,
$\boldsymbol{\sigma}_{map}$	: Incertitude de la carte.

Chaque voxel m de la grille 3D de la carte constitue le champ de vraisemblance de la manière suivante :

$$p_{hit}(m) = \max_{0 \leq k \leq n_{obstacle}} p_{hit}(\mathbf{z}|m, O^k_{\boldsymbol{\mathcal{M}}}) \tag{4.3}$$

où :

$n_{obstacle}$	: nombre d'obstacles dans l'environnement.

Le champ de vraisemblance $\mathcal{L}(\boldsymbol{\mathcal{M}})$, pour la carte $\boldsymbol{\mathcal{M}}$, est une grille 3D qui stocke $p_{hit}(m)$ pour tout m résultant d'une discrétisation de $\boldsymbol{\mathcal{M}}$.

La résolution angulaire du lidar détermine le nombre n de raies mesurées. Seul un sous-ensemble $\mathbb{L}$ de ces faisceaux sera valide et deviendra des points 3D. En fait, les objets peuvent être trop loin, leur surface trop absorbante ou bien l'angle d'incidence trop faible.

Grâce au calcul du champ de vraisemblance donné par l'équation (4.3), la probabilité $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\mathcal{M}})$ est très aisée à calculer. Dans un premier temps, nous projetons les points lidar $\mathbf{P}_{\boldsymbol{\mathcal{L}}}$ dans le repère du robot $\boldsymbol{\mathcal{R}}$ puis dans le repère de la carte $\boldsymbol{\mathcal{M}}$:

$$\mathbf{P}_{\boldsymbol{\mathcal{M}}} = \begin{bmatrix} \mathbf{R}_{\boldsymbol{\mathcal{R}}\to\boldsymbol{\mathcal{M}}} & \mathbf{T}_{\boldsymbol{\mathcal{R}}\to\boldsymbol{\mathcal{M}}} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{\boldsymbol{\mathcal{L}}\to\boldsymbol{\mathcal{R}}} & \mathbf{T}_{\boldsymbol{\mathcal{L}}\to\boldsymbol{\mathcal{R}}} \\ 0 & 1 \end{bmatrix} \cdot \mathbf{P}_{\boldsymbol{\mathcal{L}}} \tag{4.4}$$

Dans un second temps, les positions obtenues dans la carte $p_{\boldsymbol{\mathcal{M}}}$ permettent de sélectionner des voxels m. Enfin, nous déterminons la vraisemblance de chaque impact $\mathbb{P}(\mathbf{Z}|\mathbf{X}, \boldsymbol{\mathcal{M}})$ directement en lisant le champ de vraisemblance $\mathcal{L}(\boldsymbol{\mathcal{M}})$:

$$p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\mathcal{M}}) = \frac{(\sum_{i=\mathbb{L}} p_{hit}(m_i|\mathrm{p}_{i,\boldsymbol{\mathcal{M}}} \in m_i))^2}{n} \tag{4.5}$$

où :
$\mathrm{p}_{i,\boldsymbol{\mathcal{M}}}$: $i^{ème}$ raie de la mesure lidar $\mathbf{P}_{\boldsymbol{\mathcal{M}}}$,
m_i : voxel de $\mathcal{L}(\boldsymbol{\mathcal{M}})$ correspondant à la position 3D de $\mathrm{p}_{i,\boldsymbol{\mathcal{M}}}$.

Comme les impacts lidar sont considérés comme indépendants, la combinaison des vraisemblances de chaque raie devrait prendre la forme d'un produit et non d'une somme. Mais avec le nombre d'impacts obtenus, nous dépassons très rapidement les capacités de codage des nombres, même sur 64 bits. [141] présente différents types de sommes pour pallier ce problème.

Le fait d'utiliser une grille 3D pour stocker notre carte n'est pas très favorable à une représentation compacte en mémoire de l'environnement. Pour pallier cet inconvénient et respecter nos contraintes embarquées, nous proposons l'utilisation d'un octree hybride, spécialement adapté pour le stockage de ce champ de vraisemblance 3D.

4.3.4 Stockage du champ de vraisemblance

Les octrees sont des structures récursives de type arbre qui permettent entre autres de représenter efficacement un environnement 3D. Entre chaque niveau de profondeur de nœud, l'espace représenté est divisé en 8 sous-espaces égaux. Ils sont très utilisés pour stocker des cartes sous forme de nuages de points [142]. Ils permettent un gain de place dû au fait que seul l'espace occupé sera codé en mémoire, et non l'intégralité du volume comme dans le cas d'un simple tableau 3D. Un autre avantage est que l'identification des voisins, de par leur organisation géométrique de l'espace, est nettement plus efficace que dans une structure de type liste de points.

Pour stocker des données géométriques en mémoire, il faut stocker leur position et leur valeur. La position peut être décrite de plusieurs manières :

- par un tableau 3D : la structure géométrique est décrite implicitement à travers les indices du tableau. Cela ne consomme pas à proprement parler d'espace mémoire,
- par un octree : la structure des liens entre les nœuds organise les coordonnées géométriques des données. Ce sont donc des pointeurs qui occupent de l'espace mémoire pour cette description,
- par une liste de points : les trois coordonnées sont directement présentes en mémoire pour chaque point.

Dans le cadre de notre champ de vraisemblance, chaque probabilité d'impact $p_{hit}(m)$ est stockée sur 8 bits ; la plus faible valeur supérieure à 0 qui peut être codée est donc $1/255$. En utilisant un $\sigma_{carte} = 3$cm, notre probabilité d'un impact est égale à zéro à partir de 19cm de distance d'un obstacle (cf figure 4.4b). Cela veut dire qu'à proximité de tous les objets de la scène, nous aurons une vraisemblance s'étalant sur 19cm comme illustré sur la figure 4.5. Or, si nous utilisons par exemple une résolution de carte de 1cm, nous obtenons un rayon de 19 voxels à stocker autour de chaque objet de la carte.

D'un point de vue de la mémoire occupée, un simple tableau 3D est très efficace si le taux d'occupation est très important. Un octree deviendra intéressant si les données à stocker sont

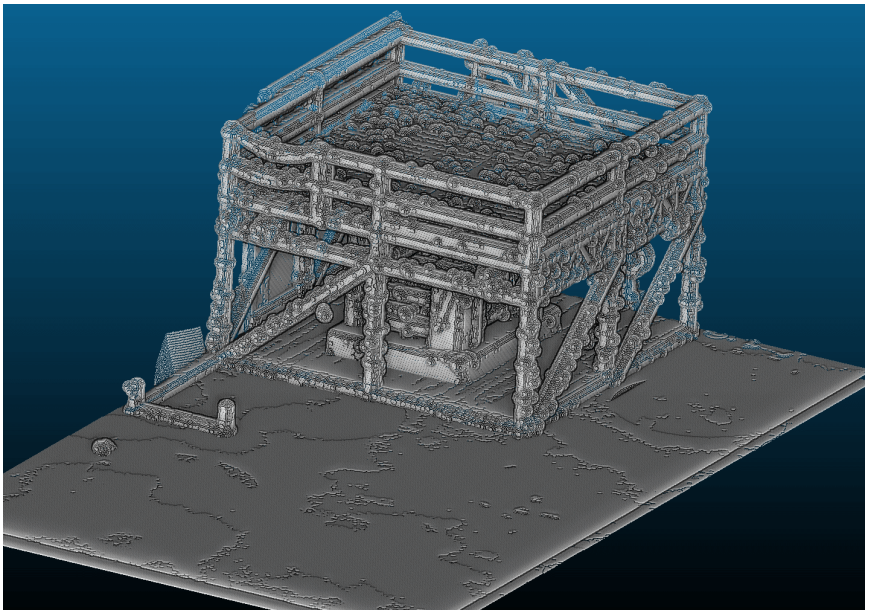


FIGURE 4.5 – Enveloppe de l'octree de la structure présente dans notre laboratoire : représentation des frontières où le champ de vraisemblance s'annule avec une représentation sur 8 bits et $\sigma_{carte} = 3cm$.

relativement éparses. En effet, il faudra également stocker les pointeurs utilisés pour représenter sa structure en plus des données.

Comme notre champ de vraisemblance s'étale autour des objets, nous aurons une forte densité de voxels occupés autour de la matière et du vide ailleurs. Nous sommes donc dans un cas intermédiaire, et c'est pour cela que nous proposons un "octree hybride" : une fois que le niveau de profondeur de l'arbre de l'octree atteint une certaine résolution R_o, un tableau 3D code l'information jusqu'à la résolution finale R_a comme le montre la figure 4.6.

D'un point de vue du temps d'accès, seule la lecture nous importe. En effet la construction de la carte s'effectuant hors ligne, le temps d'écriture a une moindre importance :

- tableau 3D : accès à 3 pointeurs quelle que soit la dimension de l'environnement,
- octree : accès à n pointeurs, n évoluant en log_2 de la dimension de l'environnement,
- octree hybride : accès à $n - log_2(R_o)$ pointeurs + 1 calcul de d'index linéaire dans le tableau 3D, de même n évoluant en log_2 de la dimension de l'environnement.

La consommation mémoire et le temps d'accès de l'octree hybride dépendent donc de la dimension du tableau 3D final, et du type de champ de vraisemblance qu'il stocke.

4.4 Résultats expérimentaux

Cette section présente les résultats obtenus avec cette localisation 6 dll en environnement complexe. Nous disposons de deux sites d'évaluation :

- notre laboratoire équipé d'une structure représentant un process pétrochimique industriel et numérisé avec un lidar Leica C10 (cf figure 4.7). Nous avons équipé la structure du système de capture de mouvement Vicon, afin de disposer d'une vérité terrain et de pouvoir qualifier la précision de l'algorithme,

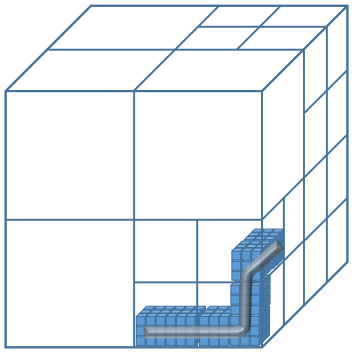
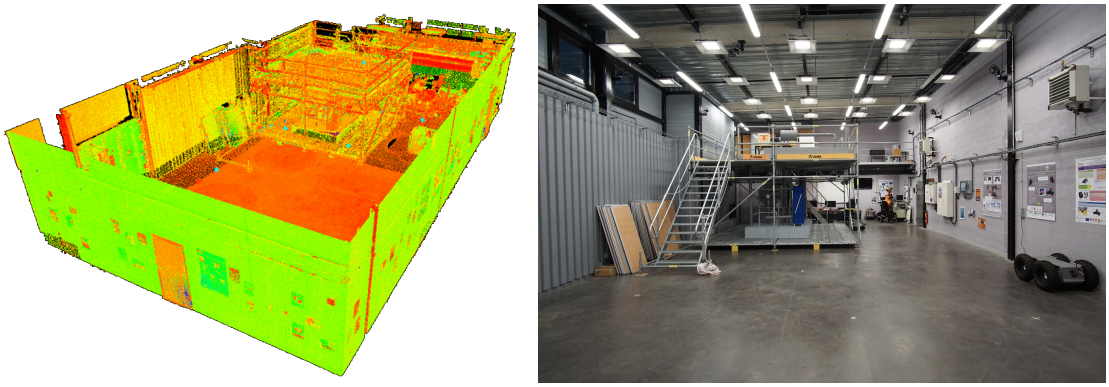


FIGURE 4.6 – Octree hybride optimisé pour stocker le champ de vraisemblance : afin de stocker un champ de vraisemblance comme le tuyau gris sur la figure, l'octree sera calculé jusqu'à une résolution R_o puis un tableau 3D permettra d'atteindre la résolution finale R_a. Comme le montre la table 4.1, cette méthode permet d'importants gains de place mémoire par rapport à un octree standard dans le cas d'objets relativement denses comme les champs de vraisemblance.

— le site de la compétition ARGOS, l'UMAD à Lacq (cf figure 4.8). Il n'est pas équipé de moyen de mesure de la vérité terrain en terme de localisation, mais il possède plusieurs autres avantages : il est nettement plus vaste, plus réaliste, sur cinq niveaux et en extérieur. Un sous-ensemble de cette structure (le premier niveau) nous a servi pour réaliser des simulations dans le but de vérifier les caractéristiques de la fonction de vraisemblance et de convergence du filtre particulaire.



(a) Nuage de points acquis par le Leica C10.

(b) Photo de notre laboratoire.

FIGURE 4.7 – Plateforme de navigation autonome de l'IRSEEM. (a) Exemple de nuage de points obtenu avec le *Leica C10*. Les couleurs correspondent à l'intensité de la réflectance. (b) Plateforme de navigation autonome de l'IRSEEM équipée avec la maquette du *Challenge Argos*.

Nous avons également investigué deux technologies de lidar afin de caractériser l'apport d'un capteur multi-nappes ML (*multi layers*) par rapport au classique mono-nappe SL (*single layer*). Les deux lidars testés durant ces expérimentations sont les suivants :

— *SL* : single layer, basé sur le Sick LMS511 [3],

3. `http://www.sick.com/instruments/EN/home/products/flow_sensors/Pages/BulkscanLMS511.aspx`

FIGURE 4.8 – Site d'entraînement UMAD (Unité de Mise A Disposition) à Lacq où se déroulent les épreuves du challenge ARGOS.

— *ML* : multi layers, basé sur le Velodyne VLP16 [4].

Dans la suite de ce document nous dénommerons le LM511 comme SL (*Single Layer*) et le VLP16 comme ML (*Multi Layer*) Nous avons évalué les performances sur cinq aspects principaux :

— le stockage et l'exploitation de la carte,
— l'appariement des données et les caractéristiques de la fonction vraisemblance,
— la convergence de l'algorithme en simulation,
— la localisation en simulation,
— la localisation embarquée réelle sur le robot dans les deux sites.

4.4.1 Performance du stockage du champ de vraisemblance par l'octree hybride

Afin de déterminer l'utilisation mémoire de l'octree hybride, nous sommes partis du nuage de points de la structure de notre laboratoire. Nous n'avons pas pris en compte les murs du laboratoire pour être le plus proche possible d'une situation réaliste en extérieur avec essentiellement des installations dans la scène. Il occupe un volume de $9.8 \times 7 \times 3.8m$, soit $261m^3$ (figure 4.5).

Consommation mémoire

Le calcul du champ de vraisemblance est effectué avec une résolution de 1cm et $\sigma_{carte} = 3$cm. Cela donne environ 32.7 millions de points de probabilité à stocker.

Nous avons fait varier la dimension du tableau final de l'octree hybride selon les puissances de 2, de 1 à 32 voxels. Pour la dimension 1, nous sommes dans le cas d'un octree standard. Plus

4. `http://velodynelidar.com/lidar/hdlproducts/vlp16.aspx`

la dimension augmente, plus nous nous rapprochons d'un simple tableau 3D. Une comparaison est également effectuée avec un tableau 3D classique. Comme nous l'avons déjà évoqué dans la section 4.3.4, l'espace occupé en mémoire par l'octree hybride dépend de la structure de la matière dans l'environnement et des caractéristiques du champ de vraisemblance (résolution spatiale et σ_{carte}).

Les résultats sont présentés dans la table 4.1. Le taux de remplissage correspond au rapport du nombre de tableaux présents dans l'octree hybride par le nombre de tableaux stockables dans le volume couvert par la structure. Nous constatons un minimum d'espace occupé en mémoire pour une dimension de tableau entre 4 et 8 voxels (figure 4.9). Si nous diminuons la dimension du tableau final, nous stockons beaucoup de pointeurs (phénomène encore accentué par la compilation en 64 bits au lieu de 32 bits). Les données du champ de vraisemblance ne sont pas assez éparses. Au contraire, si nous augmentons la dimension du tableau final, un bonne partie des tableaux se retrouve faiblement remplie. Dans ce cas, la matière n'est pas assez dense pour la dimension du tableau choisie.

Dans le cas d'une dimension de tableau de 4 à 8 voxels, nous pouvons noter que par rapport à l'octree hybride "idéal" avec un tableau final de 4 voxels, nous économisons 87.2% de l'espace mémoire utilisé par rapport à un tableau 3D classique.

Dimension tableau 3D (R_o)	**Nombre de nœuds de l'arbre**	**Nombre de tableaux 3D**	**Nombre de tableaux 3D stockable**	**Taux de remplissage** (%)	**Taille Mémoire** (Mo)
1^3	5105966	32747775	259940879	12.6	381.8
2^3	728712	4377254	32492610	13.5	83.4
4^3	110909	617803	4061576	15.2	45.3
8^3	18336	92573	507697	18.2	45.9
16^3	3468	14868	63462	23.4	58.3
32^3	727	2741	7933	34.6	85.7
tableau 3D complet	–	–	–	–	247.8

TABLE 4.1 – Consommation mémoire de l'octree hybride compilé en 64 bits en fonction de la dimension du tableau 3D final : test effectué sur une sous-partie du champ de vraisemblance de la maquette de la structure installée dans notre laboratoire (figure 4.7) $9.8 \times 7 \times 3.8m$ avec une résolution de $R_a = 1cm$ comportant 32.7 millions de points.

Si nous prenons la situation réelle du concours d'avril 2016, avec son site industriel de $43.8 \times 44.5 \times 6.1m$ soit $11890m^3$ couverts (cf figure 4.8), l'octree hybride généré occupe 595.2Mo et contient 471.6 millions de points. Un tableau 3D équivalent occuperait 11.1Go, soit une réduction de la place occupée de 94.6%. Nous pouvons constater que plus les scènes sont importantes, plus la méthode proposée est avantageuse en terme d'espace nécessaire en mémoire.

Temps d'accès

Comme nous l'avons déjà évoqué, nous nous intéressons au temps d'accès de l'octree hybride essentiellement en lecture, puisque la phase de construction de la carte est calculée hors ligne. Le temps d'accès théorique dans l'octree dépend de la profondeur de l'arbre. Il faut parcourir le

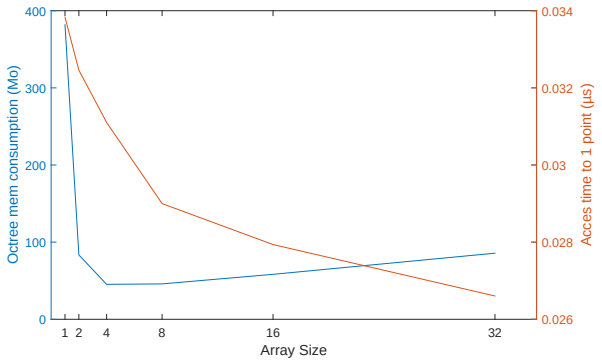


FIGURE 4.9 – Consommation mémoire et temps d'exécution de l'octree hybride en fonction de la dimension du tableau final. L'espace mémoire est minimal pour une dimension de tableau de 4 ou 8 éléments de côté.

même nombre de nœuds donc de pointeurs pour arriver sur les données stockées dans le tableau final. La profondeur de l'octree P_{octree} évolue d'une part avec le log_2 du rapport de la dimension de l'environnement sur la résolution voulue pour le représenter, et d'autre part avec la dimension du tableau final utilisé comme le montre l'équation (4.6) :

$$P_{octree} = \lceil log_2 \frac{\max_{\forall i} \mathrm{Dim}_i}{R_a} \rceil - log_2 R_o, \tag{4.6}$$

où :

R_a	: résolution finale de l'octree hybride,
R_o	: dimension du tableau 3D,
Dim	: dimensions de l'environnement (X,Y,Z).

Nous préférons parler de temps d'accès théorique, car avec les systèmes de mémoire cache modernes des processeurs actuels, il est très délicat de faire une prédiction de temps d'accès. Les performances peuvent être très dépendantes de l'implémentation. La mise en place d'une métrique et d'essais est la seule garantie pour valider des idées d'optimisations.

Pour cette évaluation, nous avons utilisé le même environnement que pour la consommation mémoire (cf section 4.4.1). Nous mesurons le temps nécessaire à la lecture de 19.2 millions de points 3D dans l'octree. Les coordonnées de ces points sont issues d'impacts lidar obtenus en simulation dans la structure du laboratoire. Si nous avions utilisé simplement des coordonnées 3D aléatoires, il y aurait eu un biais favorable dans l'évaluation. En effet, comme la scène est plutôt creuse, nous aurions majoritairement accédé à des points n'existant pas dans l'octree, c'est-à-dire des espaces vides de l'environnement. L'octree n'étant pas parcouru entièrement, le temps d'accès serait nettement plus rapide. Dans le cas extrême d'un premier pointeur *NULL*, le parcours se terminerait alors à la première étape.

Cette mesure de temps d'accès est effectuée 25 fois de manière à lisser d'éventuels problèmes de variance dus au système d'exploitation.

Les résultats en terme de temps d'accès sont présentés dans la table 4.2 et sur la figure 4.9. Comme prévu, plus l'octree est profond, plus le temps d'accès est long. Une comparaison par rapport à un tableau 3D simple est proposée sur la dernière ligne. Nous pouvons remarquer que si nous reprenons la configuration donnant les meilleurs résultats en terme d'occupation mémoire

Dimension tableau 3D (R_o)	**Profondeur de l'arbre**	**Temps d'accès total** (ms)	**Temps d'accès par point** (ns)
1^3	11	64.96	33.8
2^3	10	62.32	32.4
4^3	9	59.72	31.1
8^3	8	55.68	29.0
16^3	7	53.64	27.9
32^3	6	51.08	26.6
tableau 3D complet	–	25.56	13.3

TABLE 4.2 – Performance en temps d'accès à l'octree hybride pour 19.2 millions d'impacts lidar en fonction de la dimension du tableau final sur 1 core d'un CPU *intel i7-4700MQ @ 2.4GHz*.

(dimension de tableau final entre 4 et 8), le temps d'accès est environ 2.2 fois plus long que dans un tableau 3D. Ce résultat est raisonnable au vu de la relative complexité algorithmique pour accéder à l'octree hybride. Le choix d'une dimension de 8^3 semble être un bon compromis d'un point de vue du temps d'accès du fait qu'il soit placé dans le point d'inflexion de la courbe de la figure 4.9.

Dans le cadre d'un scan lidar complet, en supposant que tous les impacts reviennent au capteur, le temps d'accès serait de :

— 11 µs pour les 381 points d'un LMS511, soit un choix de résolution sur l'angle d'azimut de 0.5° sur 190°,

— 167µs pour les 5760 points d'un VLP16, soit un choix de résolution sur l'angle d'azimut de 1° sur 360°.

Les performances obtenues en terme de temps d'accès et de consommation mémoire sont compatibles avec l'application embarquée visée.

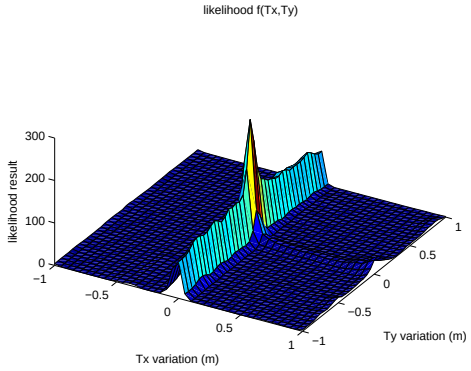
4.4.2 Caractéristiques de la fonction de vraisemblance

Les propriétés idéales de la fonction de vraisemblance que nous avons énoncées dans la section 4.3.1 ne dépendent pas que de la formule utilisée pour définir la fonction elle-même. L'environnement et le capteur jouent également un rôle important.

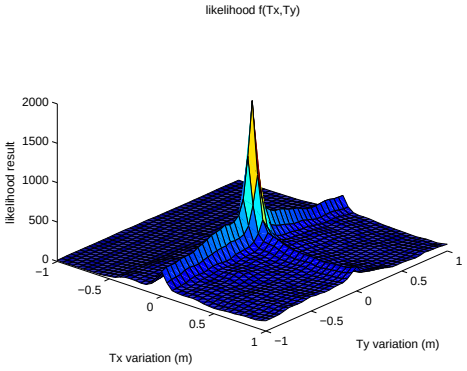
Afin de mettre ces points en évidence, nous avons utilisé le site industriel de la compétition. Nous avons simulé un scan lidar $S_{\mathbf{X}_{ref}}$ pour une pose $\mathbf{X}_{ref}$ donnée. Le robot se trouvait sur une passerelle du site, avec la longueur de la coursive alignée sur l'axe y de la carte. Comme il n'est pas facile de représenter des données en plus de 3 dimensions sur une figure, nous avons fait varier cette pose X_{ref} selon 2 dimensions sur un intervalle $\mathbf{I}$ (de $[-1m, 1m]$ pour les translations T_i et $[-20°, 20°]$ pour les rotations R_i). Nous avons ensuite tracé le score de vraisemblance sur l'intervalle I correspondant à $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\mathcal{M}}) = p(\mathbf{S_{Xref}}|\mathbf{X_{ref}} + \mathbf{i}, \boldsymbol{\mathcal{M}}, i \in \mathbf{I})$. La figure 4.10 montre le résultat de cette étude.

La figure 4.10 met en évidence les caractéristiques de la fonction de vraisemblance et l'apport du lidar ML sur le SL. Rappelons que les lidars sont simulés pour une pose dans un couloir du site de Lacq, avec l'axe longitudinal du robot aligné avec la coursive.

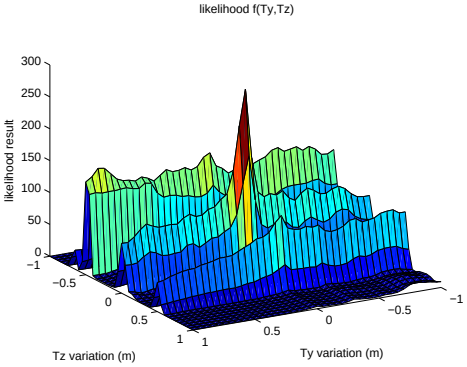
Dans le cas de translations purement dans le plan (figures 4.10a et 4.10b) avec le SL, nous obtenons une forte similarité dans l'alignement de la coursive. Dans ce cas de figure, le ML a une dynamique plus grande entre la pose réelle et les poses environnantes. Dans le cas de la variance



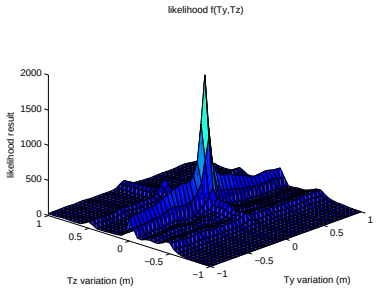
(a) SL translations horizontales



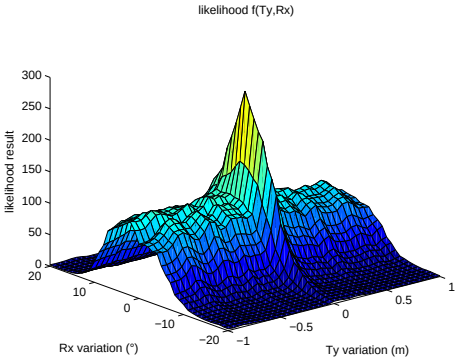
(b) ML translations horizontales



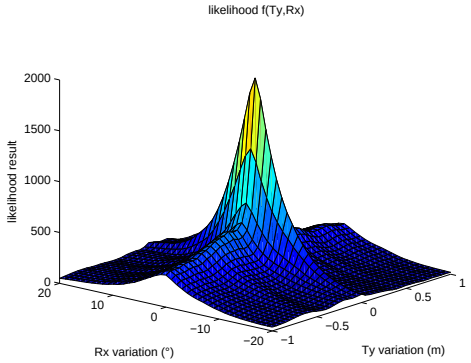
(c) SL translations verticale et horizontale



(d) ML translations verticale et horizontale



(e) SL translation et rotation



(f) ML translation et rotation

FIGURE 4.10 – Évaluation de la fonction de vraisemblance des deux lidars (*SL*) et *ML*). Les lidars sont positionnés dans un couloir du site industriel de la compétition à Lacq. L'hypothèse de pose est testée avec une variation sur deux axes.

de hauteur (figures 4.10c et 4.10d), le SL produit des "franges de similarité" importantes sur l'axe z alors que le ML discrimine nettement mieux. C'est tout-à-fait normal puisqu'il perçoit le sol dans ses nappes. Sur un mouvement de roulis (figures 4.10e et 4.10f), c'est toujours le même constat, le ML est plus discriminant que le SL. Nous pouvons donc en conclure que la fonction de vraisemblance est plus discriminante et robuste avec un lidar ML.

4.4.3 Etude de la convergence de l'algorithme

Pour évaluer la convergence de notre méthode, nous nous mettons dans la situation du "robot perdu" : nous initialisons le filtre particulaire sur une pose donnée X_{init} avec une erreur (ε_{pos}) par rapport à la position réelle du robot X_{truth}, puis nous observons le comportement de l'algorithme de localisation :

$$X_{init} = X_{truth} + \varepsilon_{pos}. \tag{4.7}$$

Dans ce cas, le robot étant immobile, il n'y a pas d'information en provenance de l'étape de *motion update*. Seul le bruit additif assure la convergence vers la position réelle du robot X_{truth}. Nous initialisons les particules du filtre en les tirant aléatoirement selon une loi normale $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_i)$. Σ_i est une matrice diagonale de covariance définie comme ceci : $\sigma_i = abs(\varepsilon_{pos})$. La variance du bruit additif est donc fixée à un dixième de celle de la répartition initiale des particules. Les bruits proportionnels et additifs de l'étape *motion update* sont donc fixés à $\sigma_p = \mathbf{0}$ et $\sigma_a = \sigma_i/10$.

Pour comparaison entre la pose réelle $\mathbf{X}_t$ et la pose estimée par le filtre particulaire $\hat{\mathbf{X}}_\mathbf{t}$, nous évaluons séparément les dimensions position et orientation du vecteur d'état (équation (4.1)). Nous calculons la distance euclidienne pour comparer les positions et l'angle solide 3D pour les orientations. L'angle solide 3D Ω est défini de la manière suivante :

$$\Omega = \arccos \frac{\mathbf{X}_t \cdot \hat{\mathbf{X}}_t}{\|\mathbf{X}_t\| \cdot \|\hat{\mathbf{X}}_t\|}. \tag{4.8}$$

Les résultats de cette convergence pour les deux types de lidar sont présentés sur la figure 4.11. Nous pouvons constater que la pente de convergence est plus raide pour le ML que pour le SL. L'erreur finale est également en faveur du ML.

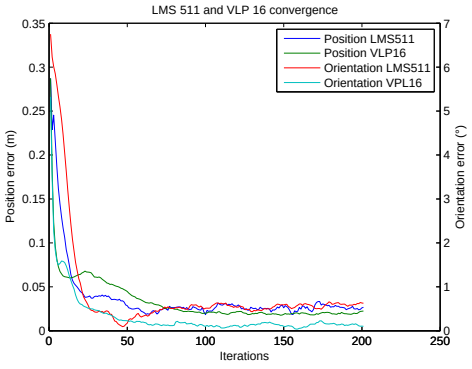


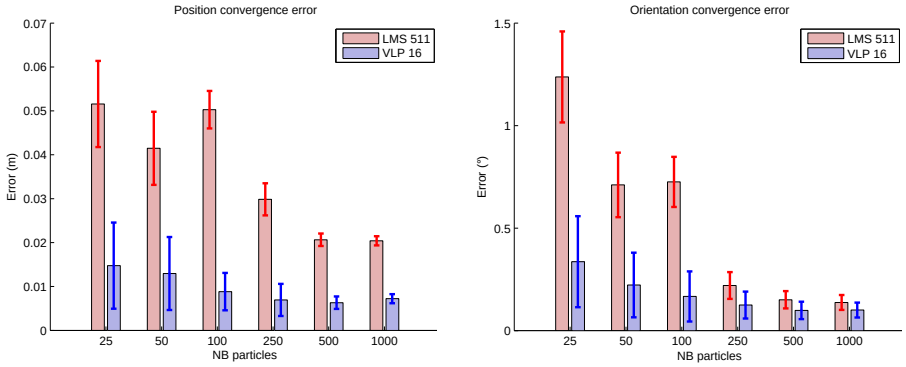
FIGURE 4.11 – Comparaison de la convergence en position et en orientation avec un lidar mono-nappe et multi-nappes : *1000 particules,* σ_{lidar}*=1cm.*

Toujours en simulation, nous avons ensuite étudié l'influence du nombre de particules sur la précision de cette convergence pour une erreur de pose initiale :

$$\varepsilon_{pos} = \begin{bmatrix} 0.2 & -0.2 & 0.05 & 2° & -2° & 5° \end{bmatrix}^T.$$

La convergence est calculée sur 200 itérations de l'étape de *measurement update*, le résultat est défini par la moyenne et l'écart-type sur les 50 derniers échantillons. Afin de s'affranchir de la variance du processus stochastique, surtout quand le nombre de particules est faible, cette

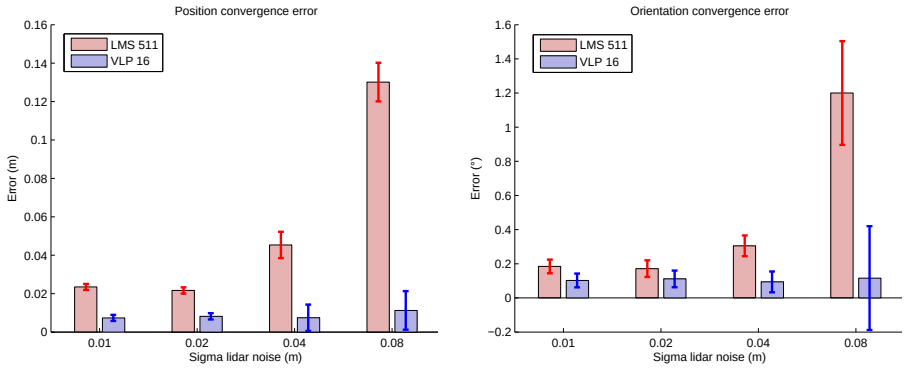
évaluation a été répétée 8 fois. C'est donc la moyenne de chaque métrique qui est représentée sur la figure 4.12. Le bruit lidar était tiré aléatoirement selon $\mathcal{N}(0, \sigma_{lidar})$ avec $\sigma_{lidar} = 0.01m$. Nous pouvons en conclure que le nombre idéal de particules se situe entre 250 et 500.



(a) Erreur de convergence en position. (b) Erreur de convergence en orientation.

FIGURE 4.12 – Moyenne et écart-type de la précision de convergence en fonction du nombre de particules.

La même étude est proposée figure 4.14, en fixant le nombre de particules à 500, et en faisant varier le bruit lidar. La méthodologie est en tout point semblable. La précision de notre méthode avec le multi-nappes est peu influencée par le bruit du capteur, seule la variance augmente. Par contre le mono-nappe qui n'a pas une vision globale de l'environnement est fortement perturbé.



(a) Erreur de convergence en position. (b) Erreur de convergence en orientation.

FIGURE 4.13

FIGURE 4.14 – Moyenne et écart-type de la précision de convergence en fonction du bruit lidar σ_{lidar}.

La robustesse à une initialisation plus erronée a également été évaluée en altérant fortement ε_{pos}. Par exemple, avec $\varepsilon_{pos} = \begin{bmatrix} 1 & 1 & 0.1 & 5° & -5° & 5° \end{bmatrix}$, le ML converge toujours alors que le SL ne converge plus. Avec une erreur de position plus modérée mais la même erreur d'orientation, $\varepsilon_{pos} = \begin{bmatrix} 0.5 & 0.5 & 0.05 & 5° & -5° & 5° \end{bmatrix}$, le SL converge correctement pour l'orientation ou la position mais rarement les deux à la fois. Alors que pour une forte erreur de position et une erreur

d'orientation plus modeste, $\varepsilon_{pos} = \begin{bmatrix}0.5 & 0.5 & 0.05 & 1° & -1° & 5°\end{bmatrix}$, il converge systématiquement. Le lidar mono-nappe est beaucoup plus sensible que le lidar multi-nappes, par rapport aux degrés de liberté sur lesquels il ne perçoit que partiellement l'environnement : t_z, r_x and r_y.

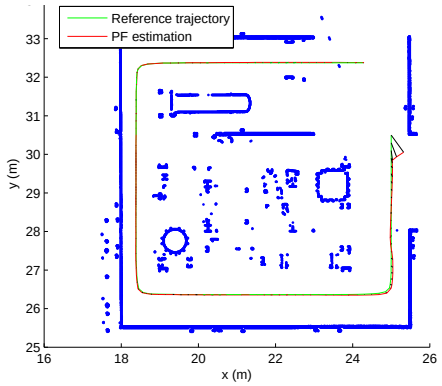
4.4.4 Localisation en simulation

Pour tester le processus de localisation complet, nous avons généré une trajectoire constituée de 160 poses lidars et de l'odométrie correspondante (figure 4.15a) sur le site de la compétition (figure 4.8 et 4.4a). Sa longueur est de 22.1m. Le bruit lidar est de $\sigma_{lidar} = 0.01m$. L'odométrie est bruitée proportionnellement aux déplacements effectués, 10% en linéaire et 20% en rotation (figure 4.15b). Les résultats sont obtenus avec 500 particules. La position initiale est altérée de $\varepsilon_{pos} = \begin{bmatrix}0.5 & -0.5 & 0.05 & 1° & -1° & 5°\end{bmatrix}$ (figure 4.15a).

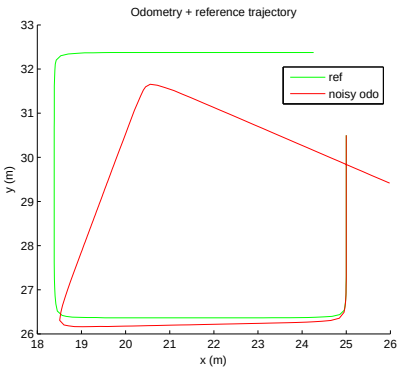
Malgré une erreur initiale de position de 71cm et d'orientation de 5.1 °, l'erreur de position moyenne est inférieure à 3cm et l'erreur d'orientation moyenne est inférieure au degré comme le montre la table 4.3. La performance atteinte par le lidar ML est supérieure à celle du lidar SL aussi bien pour la position que pour l'orientation.

TABLE 4.3 – Performance de localisation en simulation sur une trajectoire de 22.1m avec 500 particules et $\sigma_{lidar} = 1cm$.

	Erreur de position (m)		Erreur d'orientation (°)		Temps d'exécution
	Moyenne	Écart-type	Moyenne	Écart-type	Intel i2640M-2.8GHz (ms)
SL	0.0223	0.0145	0.45	0.81	0.9 (381 LiDAR impacts)
ML	0.0157	0.0120	0.31	0.44	2.2 (5760 LiDAR impacts)



(a) Trajectoire de référence (en vert), trajectoire estimée par le filtre (en rouge), coupe de l'environnement (en bleu).



(b) Trajectoire de référence (en vert) et intégration de l'odométrie bruitée (en rouge).

FIGURE 4.15 – Étude de la localisation en simulation dans l'environnement de la compétition de Lacq, à l'aide du lidar SL, 500 particules et avec $\sigma_{lidar} = 1cm$.

Les temps de calcul de la fonction de vraisemblance sous Matlab sont également donnés dans la table 4.3. Le temps d'exécution n'étant pas linéaire par rapport au nombre d'impacts lidar,

nous pouvons en déduire que le temps d'accès à la mémoire du champ de vraisemblance 3D n'est pas prédominant devant l'appel des diverses fonctions Matlab.

4.4.5 Evaluation des performances sur robots réels

Les expérimentations sont découpées en deux parties ; tout d'abord dans notre laboratoire où nous disposons d'une vérité terrain, puis sur le site de la compétition, où nous devons nous contenter d'une évaluation qualitative.

Résultats avec vérité terrain

Les expérimentations en laboratoire ont été menées dans la maquette représentative du process pétro-chimique (figure 4.7) équipée de 14 caméras Vicon T40S. Les 6 autres caméras T40S du laboratoire couvrent la zone d'entrée dans la structure. Sur des robots à chenilles, la précision de l'estimation des déplacements angulaires par l'odométrie peut s'avérer extrêmement médiocre. Dans ce cadre, nous avons intégré une centrale inertielle à bas coût dans notre robot. Elle nous permet de corriger dans une certaine mesure ces dérives. Nous l'utilisons également pour obtenir une première approximation du roulis et du tangage (r_x et r_y), qui sont au même titre que t_z, des degrés de liberté ne pouvant pas être estimés par l'odométrie.

Nous avons commencé les expérimentations par des trajectoires planes et un lidar SL, le Sick LMS511. Trois séquences (figure 4.16) ont été exécutées. L'algorithme détermine la pose avec 500 particules. Il est initialisé avec une erreur par rapport à la position réelle du robot. Les résultats sont présentés dans la table 4.4.

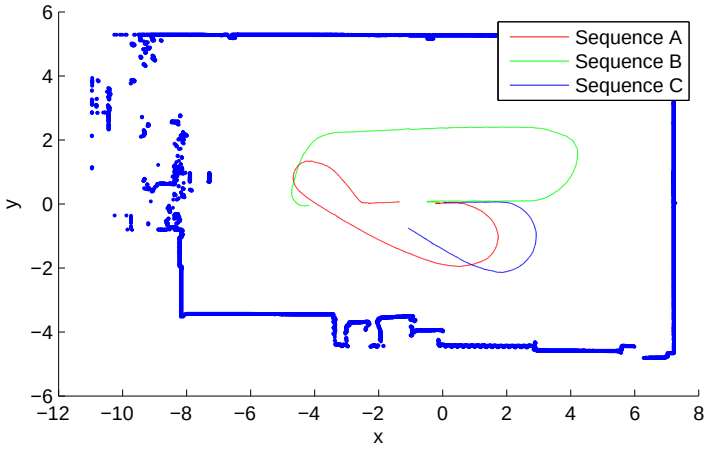


FIGURE 4.16 – Trois trajectoires dans le plan avec un lidar LMS511 (SL), la vérité terrain est obtenue par le système Vicon.

Avec le lidar SL *LMS511*, la localisation est calculée à la vitesse de rafraîchissement du capteur, c'est-à-dire 25Hz (trajectoire A-C). Les résultats que nous avons trouvés (table 4.4) sont plutôt légèrement meilleurs que ceux obtenus en simulation (table 4.3). Cela est dû au fait que la trajectoire utilisée en simulation est échantillonnée en distance tous les 13cm, alors que le système réel est échantillonné à 25Hz, ce qui correspond à une distance de 3 à 7mm entre chaque scan lidar pour les trajectoires A-C. Pour effectuer une comparaison avec les simulations,

la trajectoire A a été rééchantillonnée tous les 13cm et devient la trajectoire D. Les résultats obtenus sont alors sensiblement les mêmes qu'en simulation.

Séquence et longueur (m)	Type d'échantillonnage	Erreur de position (m)			Erreur d'orientation ($^\circ$)	
		Initiale	Moyenne	Écart-type	Moyenne	Écart-type
A : 14.97	25Hz	0.57	0.0106	0.0054	0.963	0.567
B : 17.71		0.82	0.0192	0.0128	0.804	0.389
C : 8.71		0.28	0.0178	0.0096	0.890	0.435
D : 14.97	0.13m	0.57	0.0279	0.0185	0.959	1.699
E : 22.5	20Hz	0.15	0.0236	0.0113	0.292	0.162

TABLE 4.4 – Résultats des expérimentations réelles avec vérité terrain Vicon dans notre laboratoire, 500 particules.

Nous avons également effectué la trajectoire E avec un lidar ML, le VLP16. Il s'agit d'une trajectoire 6 ddl dans le sens où le robot monte la marche de la structure de notre laboratoire, fait un aller-retour sur la plateforme en franchissant un obstacle, puis redescend. Le VLP16 est orienté vers l'arrière du robot, et 110 $^\circ$ de ses 360 $^\circ$ de son champ de vue horizontal sont masqués par le mât du robot. Ce lidar a une fréquence de rafraîchissement de 20Hz, la localisation est donc calculée à cette cadence. Les résultats sont également présentés dans la table 4.4. Nous pouvons noter que l'erreur d'orientation est nettement plus faible que pour le lidar SL.

En terme de ressources CPU, la localisation est calculée à 20Hz, avec 18084 points lidar évalués à chaque seconde et 500 particules. La fonction de vraisemblance est donc calculée environ 9 millions de fois par seconde. Le processeur embarqué est un *intel i7-4600U* à 2.1GHz, le composant de localisation utilise 16% du temps CPU qui se répartit ainsi : 3.7% pour le *motion update* et 96.3% pour le *measurement update*. L'implémentation du code fait appel à la parallélisation. Les deux étapes *motion update* et *measurement update* sont multithreadées par rapport au vecteur de particules. La consommation CPU obtenue laisse de la latitude pour les autres fonctions du robot.

Résultats sur le site *UMAD* de la compétition

La méthode proposée dans ce chapitre a été utilisée pour le Challenge Argos sur l'UMAD, à Lacq (figure 4.8). Lors de la première compétition en juin 2015, les missions se déroulaient toutes au rez-de-chaussée de la plateforme. Le robot devait tout de même franchir la marche pour accéder aux installations. En avril 2016, les missions se déroulaient sur les deux premiers niveaux, avec franchissement des escaliers et d'obstacles positifs ou négatifs (figure 4.17). La localisation a parfaitement rempli son rôle dans toutes les situations sans défaillance majeure et est en grande partie responsable de nos victoires aux deux premières manches du Challenge Argos. Sur le site de Lacq, le robot a parcouru un total de 7.8km et a été correctement localisé dans les installations pendant 16h10 à une vitesse moyenne de 2km/h (figure 4.18).

FIGURE 4.17 – Exemples de trajectoires 6 ddl exécutées par le robot Viking : négociation des escaliers sur les installations du site de la compétition (en haut), franchissement d'obstacles (en bas).

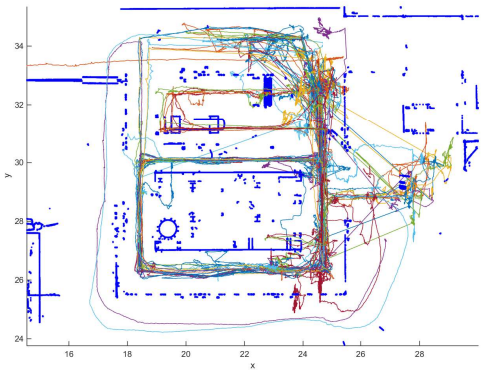


FIGURE 4.18 – Vue de dessus du cumul des trajectoires effectuées sur le site de Lacq. Bleu épais, superposition de 3 coupes de la carte 3D à 0.3, 0.5 et 0.9m. En couleur : cumul des trajectoires 3D. Le robot a été correctement localisé pendant 16h10 et a parcouru 7.8km, sa vitesse moyenne est de 2km/h. Il est souvent arrêté pour faire des mesures.

4.5 Conclusion

Dans ce chapitre, nous avons présenté notre méthode de localisation 6 ddl basée lidar en environnement complexe. Cette méthode a été développée dans le cadre de la participation de notre équipe VIKINGS au challenge ARGOS et de la thèse de Pierre Merriaux [Th2]. Notre principale contribution consiste en l'extension du concept de champ de vraisemblance à un environnement en 3 dimensions. Nous avons également proposé une méthode originale pour son stockage en mémoire, ainsi que des solutions d'implémentation afin de respecter les contraintes

d'embarquabilité. Cette méthode a démontré sa performance lors des différentes manches de la compétition ARGOS et ces travaux ont fait l'objet de publications dans [143], [144] et [145].

Des travaux sont encore en cours sur cette thématique dans le but de proposer une localisation lidar pour la voiture autonome sur une route ouverte. Les prémices de ces travaux sont exposés dans [136] et [146] ; nous y reviendront dans le chapitre 5 consacré à la présentation de mes perspectives de recherche.

Chapitre 5

Bilan et Perspectives

Sommaire

5.1 Bilan 142
5.2 Perspectives de Recherche 142
5.2.1 Approches non-conventionnelles pour la localisation 142
5.2.2 Solutions minimales pour l'estimation de pose 143
5.2.3 Localisation Lidar 144
5.2.4 Localisation multi-modale 144
5.2.5 Utilisation d'informations sémantiques pour la localisation 145
5.3 Moyens mis en œuvre pour développer le projet 145
5.3.1 Utilisation des moyens issus des projets existants 145
5.3.2 Renforcement des contrats industriels 145
5.3.3 Montage de nouveaux projets 145
5.3.4 Renforcement des collaborations 146

5.1 Bilan

Dans ce document, j'ai donné un aperçu de mes activités pédagogiques, administratives et de mes travaux de recherche pour la période 2010-2018.

Mes recherches sont principalement focalisées sur la localisation pour le véhicule autonome. Je me suis intéressé aux différentes facettes de la localisation : odométrie visuelle, *Structure from Motion*, *place recognition* et leur implémentation sur des architectures embarquées. Dans ces différents travaux, nous avons étudié et proposé de nouveaux algorithmes pour faire face à des contraintes supplémentaires que celles habituellement proposées dans l'état de l'art : non-synchronisation d'un réseau de caméras, utilisation de capteurs avec des gammes spectrales différentes (visible-infrarouge) ou encore utilisation d'un nombre minimal de points. Je me suis ensuite tourné vers une technologie alternative, le lidar, toujours dans une problématique de localisation. Ce capteur est de plus en plus intégré aux véhicules autonomes et est promis à un bel avenir. Mon objectif n'est pas de renoncer aux méthodes basées sur la vision par ordinateur, mais plutôt de fusionner ces deux technologies pour tirer parti des avantages qu'elles proposent.

D'un point de vue quantitatif, ces travaux ont fait l'objet de publications dans dix revues internationales avec comité de lecture, dans une vingtaine de conférences et ont conduit à quatre soutenances de thèses que j'ai co-encadrées. Ces recherches ont été menées au travers d'une dizaine de projets, qu'ils soient régionaux, nationaux ou encore internationaux. Par ailleurs, certaines de mes recherches ont été menées en dehors de tout projet, sous forme de travail collaboratif avec d'autres chercheurs. J'ai également eu l'opportunité de transférer mes recherches vers l'industrie au travers de trois contrats de développement portant directement sur mes domaines de compétences : un contrat pour Areva [1] pour le développement d'un système de localisation d'un ROV par vision *fisheye*, et deux contrats dédiés au véhicule autonome (PSA et Transdev).

J'ai intégré le pôle IIS (Instrumentation, Informatique et Systèmes) de l'IRSEEM en 2006 en tant que doctorant, puis en tant que permanent en 2009. Cela a été pour moi une formidable aventure et une véritable source de motivation car tout était à construire. En effet, ma thèse fut l'une des premières thèses soutenues dans ce pôle. Mon arrivée en tant que permanent de ce pôle de recherche en 2009 a coïncidé avec la mise en place du CISE (Campus Intégration des Systèmes Embarqués) pour lequel j'ai apporté ma contribution à la définition des équipements du laboratoire de navigation autonome, à leur achat, et à leur mise en place. Cette plateforme est maintenant utilisée dans de nombreux projets et thèses, ce qui montre que nous avons fait les bons choix.

Dans ce qui suit, je présenterai le bilan de ces différents travaux et dresserai les perspectives de recherche qui me semblent intéressantes.

5.2 Perspectives de Recherche

5.2.1 Approches non-conventionnelles pour la localisation

Plusieurs de mes travaux ont été consacrés à des approches d'imagerie non-conventionnelles pour la localisation des systèmes mobiles : je me suis intéressé dans un premier temps aux capteurs catadioptriques, notamment par mes travaux de thèses, puis aux systèmes de vision

1. Orano depuis janvier 2018

fisheye. J'ai ensuite élargi ce champ à l'imagerie infrarouge au travers de la thèse de Fabien Bonardi [Th4] sur la localisation visuelle multimodale (visible-infrarouge) à long terme.

Je souhaite maintenant poursuivre ces recherches en proposant des méthodes d'odométrie visuelle basées sur les caméras plénoptiques. Ces capteurs peuvent être vus comme des capteurs bio-inspirés qui permettent l'observation de la scène depuis une multitude de points de vues, ce qui est une propriété très intéressante pour la navigation autonome. Ces caméras combinent par exemple les avantages de la vision monoculaire (utilisation d'un seul capteur) à ceux des systèmes stéréoscopiques, comme l'obtention des poses et du modèle 3D à l'échelle.

Il existe quelques méthodes d'odométrie visuelle pour les caméras plénoptiques ([147] [148]) mais ces méthodes ont été développées pour des matrices de caméras avec de "grandes" *baselines*. Nous proposons dans nos travaux d'utiliser de véritables caméras plénoptiques, c'est-à-dire disposant d'une matrice de micro-lentilles. Dans ce cadre, nous avons commencé à exploiter une caméra Raytrix R8 [2] pour ce problème d'odométrie visuelle. Ces recherches seront menées en partie dans le cadre de la thèse de Oumaima Ait-Boutargante [Th7]. Des travaux intéressants sont apparus dans la littérature depuis le lancement de cette thèse ; citons en particulier les travaux de Zeller [149]. Ces travaux prouvent qu'il est possible d'utiliser des caméras plénoptiques pour la localisation, bien que ce type de caméra ne soit à la base pas prévu pour ce type d'application, notamment à cause de leur faible profondeur de champ.

En plus des contributions scientifiques que nous pourrions apporter sur cette thématique, il me semble intéressant de proposer publiquement un jeu de données avec vérité terrain pour que les différents chercheurs puissent comparer leurs algorithmes. En effet, à l'heure actuelle, les quelques publications sur le sujet ne permettent pas d'avoir une évaluation objective des performances des méthodes proposées puisqu'un tel jeu de données n'existe pas. Nous avons au sein de notre laboratoire tout le matériel nécessaire pour réaliser ce type d'expérimentation, aussi bien en intérieur avec le système Vicon, qu'en extérieur avec le système GPS-RTK/IMU dont nous disposons. L'importance d'une telle contribution n'est plus à démontrer depuis l'engouement pour les bases de données KITTI [52] ou EuRoC MAV [150].

5.2.2 Solutions minimales pour l'estimation de pose

Mes travaux de recherche menés sur les solutions minimales ont permis d'aboutir à des résultats intéressants pour l'estimation de poses à partir de points ou de coniques dans les images. Je souhaite poursuivre les recherches dans ce domaine en travaillant cette fois sur l'exploitation de lignes. Des travaux ont d'ores et déjà débuté sur ce sujet notamment au travers de la thèse de Louis le Crosnier [Th6]. Cette thèse, inscrite dans le cadre du projet COPTER, a notamment pour objectif d'étudier des solutions minimales pour le problème de PnL (*Perspective-n-Line*). L'objectif final est de permettre à un drone de se localiser à partir de projections de lignes dans les images et de leurs correspondances 3D obtenues par un lidar embarqué sur un robot terrestre. Dans ces travaux, nous utilisons les informations sur l'attitude du drone qui peuvent être obtenues par exemple grâce à une centrale inertielle pour réduire le nombre d'inconnues de ce problème de PnL. Dans la littérature, les méthodes de PnL dans le cas général nécessitent de 9 lignes pour les formulations linéaires [151] à 3 lignes pour les formulations polynomiales [152]. Nos premiers travaux ont pour le moment permis d'aboutir à une solution basée sur 2 lignes lorsque les angles de roulis et tangage sont connus.

2. https ://raytrix.de

5.2.3 Localisation Lidar

Les travaux développés dans le cadre de la localisation lidar ont donné de très bons résultats dans des environnements de dimensions modérées et connus au préalable, les objectifs initiaux étant de développer un système de localisation en milieu industriel. Je souhaite maintenant étendre ces travaux à la problématique de la localisation du véhicule autonome. Cette évolution nécessitera le développement de nouvelles méthodes de création et de stockage des cartes, ainsi que des adaptions du module de localisation. En effet, de par les dimensions de l'environnement, il ne sera plus possible de le numériser par un scanner 3D "statique", mais il faudra utiliser les capteurs disponibles sur le véhicule et exploiter le déplacement de celui-ci pour construire ces cartes. Nous avons récemment réussi à réaliser toute la chaîne de traitement, de la construction de la carte à la localisation du véhicule dans le cas où le signal GPS-RTK est disponible lors de la cartographie. Cependant, il existe des configurations où ce signal n'est plus disponible (passage sous un pont, dans un tunnel ou dans une forêt par exemple). Nos prochains travaux porteront donc sur le développement d'algorithmes de SLAM permettant de combler les défaillances du GPS-RTK lors de la cartographie. Ces travaux sont très importants puisque la phase de cartographie conditionne énormément les performances du système : il est impossible de se localiser précisément dans une carte si celle-ci n'est pas précise.

A plus long terme, si nous envisageons d'exploiter les véhicules autonomes sur de longues distances, il ne sera plus possible de stocker toute la carte en mémoire et des optimisations devront être trouvées. Le filtre particulaire utilisé pour la localisation devra également être adapté en terme de modèle cinématique, de nombre de particules, de nombre de dimensions à estimer, etc, par rapport à notre algorithme de localisation en milieu industriel complexe.

Une autre thématique très importante concerne la mise à jour dynamique de la carte. En effet, après une première construction de carte, celle-ci pourrait être mise à jour de façon collaborative par tous les véhicules l'utilisant. Cela permettrait par exemple de mettre à jour la carte suite à des changements (construction d'un nouveau bâtiment, modification du tracé de la route, etc), de retirer des objets non-fixes de la carte (par exemple de véhicules stationnés au moment de la première cartographie), ou encore d'améliorer la résolution et la précision de la carte suite à de multiples passages avec des lidars.

5.2.4 Localisation multi-modale

Actuellement, nous développons des solutions de localisation basées vision ou lidar. A l'avenir, j'envisage de coupler les deux approches pour le véhicule autonome. Plusieurs voies sont envisageables. La première consiste à créer une carte 3D avec un lidar et à se localiser dans celle-ci grâce à des capteurs de vision embarqués sur le véhicule. Cette approche a plusieurs avantages : les lidars sont des capteurs parfaitement adaptés pour la phase de cartographie puisqu'ils fournissent directement une information sur la structure de la scène (nuage de points 3D) et avec une grande précision. La localisation pourrait ensuite se faire grâce aux caméras qui ont l'avantage d'être bien moins onéreuses et donc plus facilement embarquées sur des véhicules. La seconde approche serait de créer des cartes multi-modales, c'est-à-dire des cartes intégrant à la fois des données issues du lidar et des données visuelles. Cette approche aurait l'avantage d'augmenter la robustesse globale du système de localisation puisque ces deux technologies sont complémentaires. Le lidar fournit en effet un nuage de points précis autour du véhicule mais il a une portée limitée. Au contraire, la caméra permet d'observer des points caractéristiques plus lointains et facilite la compréhension de la scène, mais celle-ci est très sensible aux conditions

d'illumination.

Nous avons également démarré une thèse CIFRE [Th8] sur la localisation précise en intérieur et la capture simultanée du geste par fusion d'une localisation *Ultra Wide Band*, d'une mesure de mouvement par capteurs inertiels, et par vision. L'idée sous-jacente est encore une fois d'exploiter les avantages de toutes ces technologies pour une estimation robuste et précise de la localisation.

5.2.5 Utilisation d'informations sémantiques pour la localisation

Les approches de localisation que nous développons, qu'elles soient basées lidar ou vision, reposent principalement sur des approches géométriques avec l'utilisation de points d'intérêt. Il serait intéressant de créer des cartes avec des informations sémantiques afin d'améliorer la robustesse des algorithmes de localisation. Cela est devenu possible grâce à l'émergence récente du *Deep Learning*, par exemple avec l'arrivée de SegNet [153]. L'idée serait de stocker à l'intérieur des cartes des informations de plus haut niveau, par exemple en attribuant des labels aux différents objets de la carte (panneau de signalisation, marquage au sol, bâtiment, etc) afin d'avoir des informations plus discriminantes qu'un simple nuage de points.

5.3 Moyens mis en œuvre pour développer le projet

Pour développer le projet de recherche décrit précédemment, j'envisage d'utiliser et de développer les moyens suivants.

5.3.1 Utilisation des moyens issus des projets existants

Un certain nombre de travaux à court terme, par exemple sur la localisation vision et lidar, pourront se faire dans le cadre de projets existants (LOCADYN, COPTER). Dans le cadre du projet LOCADYN, un post-doctorant que j'encadrerai nous rejoindra début octobre pour travailler sur la localisation visuelle et embarquée en environnement industriel.

5.3.2 Renforcement des contrats industriels

Comme je l'ai décrit dans la section 1.4.3, de nombreux travaux que j'ai menés ont été rendus possibles grâce à des contrats de valorisation avec des industriels. Certains de ces contrats sont encore en cours et permettront de continuer quelques uns de mes axes de recherche (localisation lidar pour le véhicule autonome notamment). Je souhaite renforcer nos activités contractuelles en trouvant de nouveaux contrats ou de nouveaux partenaires afin de disposer de moyens plus conséquents pour mener ces recherches.

5.3.3 Montage de nouveaux projets

J'ai pu mener mes recherches grâce aux nombreux projets dans lesquels j'ai été impliqué, ou que j'ai montés et coordonnés. Ces projets ont été financés par des sources très variées (ANR, DGA, FEDER, Région Normandie, etc). J'envisage à l'avenir de monter un projet ANR à l'appel à projets générique sur la localisation visuelle multimodale pour le véhicule autonome. De plus, avec un partenaire industriel rencontré dans le cadre d'un autre projet, nous envisageons de monter un projet ANR ASTRID.

5.3.4 Renforcement des collaborations

Pour mener à bien mon projet de recherche, je souhaite renforcer mes collaborations existantes mais également trouver de nouvelles collaborations. Certains de mes travaux de recherche ont été menés en dehors de tout projet, grâce à un travail collectif avec des chercheurs d'autres laboratoires (LITIS, Le2i, INRIA notamment). Je souhaite donc pérenniser et renforcer nos échanges pour continuer à travailler ensemble. Cela pourrait se faire notamment au travers de nouveaux projets ou d'encadrements communs. Je souhaite également développer de nouvelles collaborations, notamment avec le laboratoire GREYC à Caen qui travaille sur des thématiques proches des nôtres.

Bibliographie

[1] National Highway Traffic Safety Administration [en ligne]. Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey, février 2015 [consulté le 02 août 2018]. Disponible sur : https ://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115.

[2] O. Saurer, P. Vasseur, R. Boutteau, C. Demonceaux, M. Pollefeys, and F. Fraundorfer. Homography based egomotion estimation with a common direction. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(2) :327–341, February 2017.

[3] R. Boutteau, P. Sturm, P. Vasseur, and C. Demonceaux. Circular laser/camera-based attitude and altitude estimation : minimal and robust solutions. *Journal of Mathematical Imaging and Vision (JMIV)*, 60(3) :382–400, March 2018.

[4] A. Belbachir, R. Boutteau, P. Merriaux, J.M. Blosseville, and X. Savatier. From autonomous robotics toward autonomous cars. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 1362–1367, Gold Coast, Australia, June 2013.

[5] J.C. McCall and M.M. Trivedi. Video-based lane estimation and tracking for driver assistance : survey, system, and evaluation. *IEEE Transactions on Intelligent Transportation Systems*, 7(1) :20–37, March 2006.

[6] D. Dooley, B. McGinley, C. Hughes, L. Kilmartin, E. Jones, and M. Glavin. A blind-zone detection method using a rear-mounted fisheye camera with combination of vehicle detection methods. *IEEE Transactions on Intelligent Transportation Systems*, 17(1) :264–278, 2016.

[7] R. Timofte, K. Zimmermann, and L. Van Gool. Multi-view traffic sign detection, recognition, and 3d localisation. *Machine Vision and Applications (MVA)*, 25(3) :633–647, 2014.

[8] M. Mathias, R. Timofte, R. Benenson, and L. Van Gool. Traffic sign recognition—how far are we from the solution ? In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Dallas, TX, USA, August 2013.

[9] S. Lüke, O. Fochler, T. Schaller, and U. Regensburger. Traffic jam assistance and automation. *In Handbook of Driver Assistance Systems*, pages 1287–1302, 2016.

[10] D. Scaramuzza and F. Fraundorfer. Visual odometry : Part i - the first 30 years and fundamentals. *IEEE Robotics and Automation Magazine*, 18(4) :80 – 92, December 2011.

[11] F. Fraundorfer and D. Scaramuzza. Visual odometry : Part ii - matching, robustness, and applications. *IEEE Robotics and Automation Magazine*, 19(2) :78–90, June 2012.

[12] M. H. Mirabdollah and B. Mertsching. Fast techniques for monocular visual odometry. In *German Conference on Pattern Recognition*, pages 297–307, 2015.

[13] I. Cvisic and I. Petrovic. Stereo odometry based on careful feature selection and tracking. In *European Conference on Mobile Robots (ECMR)*, pages 1–6, 2015.

[14] R. Mur-Artal, J.M.M. Montiel, and J.D. Tardos. Orb-slam : a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5) :1147–1163, 2015.

[15] S. Lowry, N. Sünderhauf, P. Newman, J.J. Leonard, D. Cox, P. Corke, and M.J. Milford. Visual place recognition : A survey. *IEEE Transactions on Robotics*, 32(1) :1–19, 2016.

[16] R. Siegwart, I.R. Nourbakhsh, and D. Scaramuzza. *Introduction to Autonomous Mobile Robots*. MIT Press, second edition edition, 2004.

[17] S. Baker and S.K. Nayar. A theory of single-viewpoint catadioptric image formation. *International Journal of Computer Vision (IJCV)*, 35(2) :175–196, November 1999.

[18] L. Delahoche, C. Pégard, B. Marhic, and P. Vasseur. A navigation system based on an omnidirectional vision sensor. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 718–724, 1997.

[19] R. Boutteau, X. Savatier, and J.Y. Ertaud. A dynamic programming algorithm applied to omnidirectional vision for dense 3d reconstruction. In *IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 927–931, Naha, Okinawa, Japan, 5-8 November 2013.

[20] M. Schönbein, B. Kitt, and M. Lauer. Environmental perception for intelligent vehicles using catadioptric stereo vision systems. In *E*, pages 189–194, 2011.

[21] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition edition, 2004.

[22] A. Comport, E. Malis, and P. Rives. Real-time quadrifocal visual odometry. *International Journal of Robotics Research (IJRR)*, 29(2-3) :245–266, February 2010.

[23] M. Sons, M. Lauer, C.G. Keller, and C. Stiller. Mapping and localization using surround view. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 1158–1163, Los Angeles, USA, June 2017.

[24] V. Appia, H. Hariyani, S. Sivasankaran, S. Liu, K. Chitnis, M. Mueller, U. Batur, and G. Agarwa. Surround view camera system for adas on ti's tdax socs. Technical report, Texas Instrument, 2015.

[25] B. Zhang, V. Appia, I. Pekkucuksen, Y. Liu, A. Umit Batur, P. Shastry, S. Liu, S. Sivasankaran, and K. Chitnis. A surround view camera solution for embedded systems. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 662–667, Columbus, USA, June 2014.

[26] A. Ramirez, E. Ohn-Bar, and M.M. Trivedi. Panoramic stitching for driver assistance and applications to motion saliency-based risk analysis. In *IEEE International Conference on Intelligent Transportation Systems ITSC*, pages 597–601, The Hague, The Netherlands, October 2013.

[27] M. Dhome, L. Mennillo, E. Royer, F. Mondot, and J. Mousain. Multibody reconstruction of the dynamic scene surrounding a vehicle using a wide baseline and multifocal stereo system. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, Vancouver, Canada, September 2017.

[28] M. Pollefeys, J. Nikolic, L. Kneip, T. Kazik, and R. Siegwart. Real-time 6d stereo visual odometry with non-overlapping fields of view. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1529–1536, Providence, Rhode Island, June 2012.

[29] Y. Wang and L. Kneip. On scale initialization in non-overlapping multi-perspective visual odometry. In *International Conference on Computer Vision Systems (ICVS)*, pages 144–157, Shenzhen, China, July 2017.

[30] R. Pless. Using many cameras as one. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 587–593, June 2003.

[31] B. Clipp, J.H. Kim, J.M. Frahm, M. Pollefeys, and R. Hartley. Robust 6dof motion estimation for non-overlapping, multi-camera systems. In *IEEE Workshop on Applications of Computer Vision*, pages 1–8, 2008.

[32] J.H. Kim, R. Hartley, J.M. Frahm, and M. Pollefeys. Visual odometry for non-overlapping views using second-order cone programming. In *Asian Conference on Computer Vision (ACCV)*, pages 353–362, Tokyo, Japan, November 2007.

[33] P. Furgale, U. Schwesinger, M. Rufli, W. Derendarz, H. Grimmett, P. Muhlfellner, S. Wonneberger, J. Timpner, S. Rottmann, B. Li, B. Schmidt, Nguyen ; T.N., E. Cardarelli, S. Cattani, Bruning ; S., S. Horstmann, M. Stellmacher, H. Mielenz, K. Koser, M. Beermann, C. Hane, L. Heng, G.H. Lee, F. Fraundorfer, R. Iser, R. Triebel, I. Posner, P. Newmann, L. Wolf, M. Pollefeys, S. Brosig, J. Effertz, C. Pradalier, and R. Siegwart. Toward automated driving in cities using close-to-market sensors : An overview of the v-charge project. In *IEEE Intelligent Vehicles Symposium (IV)*, Gold Coast, QLD, Australia, October 2013.

[34] L. Heng, B. Li, and M. Pollefeys. Camodocal : Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1793–1800, Tokyo, Japan, January 2013.

[35] C. Hane, L. Heng, G.H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys. 3d visual perception for self-driving cars using a multi-camera system : Calibration, mapping, localization, and obstacle detection. *Image and Vision Computing (IVC)*, 2018.

[36] T.T. Nguyen and M. Lhuillier. Self-calibration of omnidirectional multi-cameras including synchronization and rolling shutter. *Computer Vision and Image Understanding (CVIU)*, 162 :166–184, 2017.

[37] 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 (IVC)*, 27(8) :1178–1193, July 2009. ISSN :0262-8856.

[38] D.W. Pooley, M.J. Brooks, A.G. Van den Hengel, and W. Chojnacki. A voting scheme for estimating the synchrony of moving-camera videos. In *International Conference on Image Processing (ICIP)*, volume 1, 2003.

[39] L. Wolf and A. Zomet. Sequence-to-sequence self calibration. In *European Conference on Computer Vision (ECCV)*, pages 370–382, Copenhagen, Denmark,, May 2002.

[40] Y. Caspi and M. Irani. A step towards sequence-to-sequence alignment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 682–689 vol.2, 2000.

[41] M. Svedman. 3-d structure from stereo vision using unsynchronized cameras. In *Masters thesis, Royal Institute of Technology (KTH)*, 2005.

[42] D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26 :756–770, June 2004.

[43] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision (ECCV)*, pages pp 430–443, Graz, Austria, May 2006.

[44] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief : Binary robust independent elementary features. In *European Conference on Computer Vision (ECCV)*, volume 6314, pages pp 778–792, Heraklion, Crete, Greece, September 2010.

[45] K. Levenberg. A method for the solution of certain problems in least squares. *Quarterly of Applied Mathematics*, 2 :164–168, 1944.

[46] C. Engels, H. Stewénius, and D. Nistér. Bundle adjustment rules. *Photogrammetric computer vision*, 2(2006), 2006.

[47] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. *Vision Algorithms : Theory and Practice*, 1883 :298–372, 2000.

[48] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 363–370, 2006.

[49] F. Fraundorfer, D. Scaramuzza, and M. Pollefeys. A constricted bundle adjustment parameterization for relative scale estimation in visual odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1899–1904, 2010.

[50] D. Scaramuzza, F. Fraundorfer, and R. Siegwart. Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4293–4299, 2009.

[51] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving ? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, Rhode Island, USA, June 16-21 2012.

[52] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics : The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[53] C. McManus, W. Churchill, W. Maddern, A.D. Stewart, and P. Newman. Shady dealings : Robust, long-term visual localisation using illumination invariance. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 901–906, Hong Kong, China, May 2014.

[54] M.J. Milford and G. F. Wyeth. Seqslam : Visual route-based navigation for sunny summer days and stormy winter nights. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1643–1649, St Paul, USA, May 2012.

[55] T. Naseer, L. Spinello, W. Burgard, and C. Stachniss. Robust visual robot localization across seasons using network flows. In *AAAI International Conference on Artificial Intelligence*, pages 2564–2570, Quebec City, Canada, July 2014.

[56] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(10) :1615–1630, 2005.

[57] J. Heinly, E. Dunn, and J.M. Frahm. Comparative evaluation of binary features. In *European Conference on Computer Vision (ECCV)*, Firenze, Italy, October 2012.

[58] E. Garcia-Fidalgo and A. Ortiz. Vision-based topological mapping and localization methods : A survey. *Robotics and Autonomous Systems*, 64 :1–20, 2015.

[59] Yan F. & Mikolajczyk K. (2013). 117(5) 479-492. Koniusz, P. Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection. *Computer vision and image understanding (CVIU)*, 117(5) :479–492, 2013.

[60] N. Sunderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford. On the performance of convnet features for place recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4297–4304, Hamburg, Germany, 2015.

[61] N. Sunderhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, B. Upcroft, and M. Milford. Place recognition with convnet landmarks : Viewpoint-robust, condition-robust, training-free. In *Robotics : Science and Systems (RSS)*, 2015.

[62] T. Weyand, I. Kostrikov, and J. Philbin. Planet-photo geolocation with convolutional neural networks. In *European Conference on Computer Vision (ECCV)*, pages 37–55, 2016.

[63] C.A. Aguilera, F.J. Aguilera, A.D. Sappa, C. Aguilera, and R. Toledo. Learning cross-spectral similarity measures with deep convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1–9, 2016.

[64] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. Monoslam : Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(6) :1052–1067, April 2007.

[65] F. Abrate, B. Bona, and M. Indri. Experimental ekf-based slam for mini-rovers with ir sensors only. In *European Conference on Mobile Robots (ECMR)*, September 2007.

[66] G. Caron, A. Dame, and E. Marchand. Direct model based visual tracking and pose estimation using mutual information. *Image and Vision Computing (IVC)*, 32(1) :54–63, 2014.

[67] M. Magnabosco and T.P. Breckon. Cross-spectral visual simultaneous localization and mapping (slam) with sensor handover. *Robotics and Autonomous Systems*, 61(2) :195–208, 2013.

[68] W. Maddern, M. Milford, and G. Wyeth. Cat-slam : probabilistic localisation and mapping using a continuous appearance-based trajectory. *International Journal of Robotics Research (IJRR)*, 31(4) :429–451, April 2012.

[69] P. Ricaurte, C. Chilán, C.A. Aguilera-Carrasco, B.X. Vintimilla, and A.D. Sappa. Feature point descriptors : Infrared and visible spectra. *Sensors*, 14(2) :3690–3701, 2014.

[70] M. Firmenichy, D. andBrown and S. Süsstrunk. Multispectral interest points for rgb-nir image registration. In *IEEE International Conference on Image Processing (ICIP)*, pages 181–184, Brussels, Belgium, September 2011.

[71] T. Mouats and N. Aouf. Multimodal stereo correspondence based on phase congruency and edge histogram descriptor. In *International Conference on Information Fusion (FUSION)*, pages 1981–1987, 2013.

[72] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2) :91–110, November 2004. ISSN :0920-5691.

[73] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision (IJCV)*, 65(1-2) :43–72, 2005.

[74] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors : a survey. *Foundations and trends in computer graphics and vision*, 3(3) :177–280, 2008.

[75] C. Harris and M. Stephens. A combined corner and edge detector. In *Fourth Alvey Vision Conference*, pages 147–151, 1988.

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

[77] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Surf : Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 110(3) :346–359, 2008.

[78] C.A. Aguilera, A.D. Sappa, and R. Toledo. Lghd : A feature descriptor for matching across non-linear intensity variations. In *IEEE International Conference on Image Processing*, pages 178–181, Québec City, Canada, September 2015.

[79] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2911–2918, Rhode Island, USA, June 2012.

[80] J. Sivic and A. Zisserman. Video google : A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1470–1477, Nice, France, October 2003.

[81] M. Brown and S. Süsstrunk. Multi-spectral sift for scene category recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 177–184, Colorado Springs, USA, June 2011.

[82] D. Mishkin, M. Perdoch, and J. Matas. Place recognition with wxbs retrieval. In *International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Boston, USA, June 2015.

[83] M. Herlihy and N. Shavit. *The art of multiprocessor programming.* Elsevier, 2012.

[84] R. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 19(6) :580–593, June 1997.

[85] H. Stewenius, C. Engels, and D. Nister. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60(4) :284–294, 2006.

[86] S. Ouerghi, R. Boutteau, X. Savatier, and F. Tlili. Cuda accelerated visual egomotion estimation for robotic navigation. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, pages 107–114, Porto, Portugal, February 2017.

[87] S. Ouerghi, R. Boutteau, X. Savatier, and F. Tlili. *Communications in Computer and Information Science*, chapter CUDA-accelerated Feature-based Egomotion Estimation. Springer, 2018.

[88] L. Kneip and P. Furgale. Opengv : A unified and generalized approach to real-time calibrated geometric vision. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, Hong Kong, China, May 2014.

[89] S. Ouerghi, R. Boutteau, F. Tlili, and X. Savatier. Visual odometry and place recognition fusion for vehicle position tracking in urban environments. *Sensors*, 18(4) :939, 2018.

[90] S. Ouerghi, R. Boutteau, F. Tlili, and X. Savatier. Cuda-based seqslam for real-time place recognition. In *Conference on Computer Graphics, Visualization and Computer Vision (WSCG)*, Plzen, Czech Republic, May 29 - June 2 2017.

[91] R. Mhiri, P. Vasseur, S. Mousset, R. Boutteau, and A. Bensrhair. Estimation à l'échelle du mouvement d'un réseau multi-caméras non synchronisées. In *Reconnaissance des Formes et Intelligence Artificielle (RFIA)*, 2014.

[92] R. Mhiri, P. Vasseur, S. Mousset, R. Boutteau, and A. Bensrhair. Visual odometry with unsynchronized multi-cameras setup for intelligent vehicle application. In *IEEE Intelligent Vehicles Symposium (IV)*, 2014.

[93] R. Mhiri, P. Vasseur, S. Mousset, R. Boutteau, and A. Bensrhair. Estimation du mouvement et de la structure à l'échelle absolue à partir d'un réseau multi-caméras non synchronisées. In *Congrès des jeunes chercheurs en vision par ordinateur (ORASIS)*, Amiens, France, June 15-19 2015.

[94] R. Mhiri, P. Vasseur, S. Mousset, R. Boutteau, and A. Bensrhair. Accurate scale estimation based on unsynchronized camera network. In *IEEE International Conference on Image Processing (ICIP)*, Quebec City, Canada, September, 27-30 2015.

[95] F. Bonardi, S. Ainouz, R. Boutteau, Y. Dupuis, X. Savatier, and P. Vasseur. Phrog : A multimodal feature for place recognition. *Sensors*, 17 :1167, 2017.

[96] F. Bonardi, S. Ainouz, R. Boutteau, Y. Dupuis, X. Savatier, and P. Vasseur. Localisation visuelle multimodale à long terme. In *Colloque Gretsi (GRETSI)*, Juan-Les-Pins, France, September 2017.

[97] F. Bonardi, S. Ainouz, R. Boutteau, Y. Dupuis, X. Savatier, and P. Vasseur. A novel global image description approach for long term vehicle localization. In *European Signal Processing Conference (EUSIPCO)*, Kos Island, Greece, 2017.

[98] S. Ouerghi, R. Boutteau, P. Merriaux, N. Ragot, X. Savatier, and P. Vasseur. Absolute localization using visual data for autonomous vehicles. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, Rome, Italy, February 2016.

[99] M.A. Fischler and R.C. Bolles. Random sample consensus : A paradigm for model fitting with applications to image analysis and automated cartography. In *Communications of the ACM*, volume 24, pages 381–395, June 1981.

[100] J.S. Kim, P. Gurdjos, and I.S. Kweon. Euclidean structure from confocal conics : Theory and application to camera calibration. *Computer Vision and Image Understanding (CVIU)*, 114(7) :803–812, July 2010.

[101] F. Fraundorfer, P. Tanskanen, and M. Pollefeys. A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles. In *European Conference on Computer Vision (ECCV)*, pages 269–282, Hersonissos, Greece, September 2010.

[102] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms : An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 2007.

[103] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In *European Conference on Computer Vision (ECCV)*, pages 302–315, Marseille, France, October 2008.

[104] O. Naroditsky, X.S. Zhou, J. Gallier, S.I. Roumeliotis, and K. Daniilidis. Two efficient solutions for visual odometry using directional correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 34(4) :818–824, 2012.

[105] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 573–580, Vilamoura, Portugal, October 2012.

[106] A. Manecy, N. Marchand, F. Ruffier, and S. Viollet. X4-mag : A low-cost open-source micro-quadrotor and its linux-based controller. *International Journal of Micro Air Vehicles (IJMAV)*, 7(2) :89–110, June 2015.

[107] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier. A study of vicon system positioning performance. *Sensors*, 17 :1591, 2017.

[108] A. Vedaldi and B. Fulkerson. Vlfeat : An open and portable library of computer vision algorithms. In *ACM International Conference on Multimedia*, pages 1469–1472, October 2010.

[109] J.L. Schönberger, F. Fraundorfer, and J.M. Frahm. Structure-from-motion for mav image sequence analysis with photogrammetric applications. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40(3) :305, 2014.

[110] M. Rothermel, K. Wenzel, D. Fritsch, and N. Haala. Sure : Photogrammetric surface reconstruction from imagery. In *LC3D Workshop*, volume 8, page 29, Berlin, Germany, December 2012.

[111] R.S. Hartenberg and J. Denavit. A kinematic notation for lower pair mechanisms based on matrices. *Journal of Applied Mechanics*, 77(2) :215–221, 1955.

[112] F. Kahl and A. Heyden. Using conic correspondences in two images to estimate the epipolar geometry. In *Proceedings of the International Conference on Computer Vision*, pages 761–766, 1998.

[113] J.Y. Bouguet. *Visual methods for three-dimensional modeling.* PhD thesis, Thèse de doctorat, California Institute of Technology, May 1999.

[114] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp : An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision (IJCV)*, 81(2) :155–166, 2009.

[115] P. Sturm and P. Gargallo. Conic fitting using the geometric distance. In *Asian Conference on Computer Vision (ACCV)*, pages 784–795, Tokyo, Japan, November 18-22 2007.

[116] A. Natraj, P. Sturm, C. Demonceaux, and P. Vasseur. A geometrical approach for vision based attitude and altitude estimation for uavs in dark environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4565 – 4570, Vilamoura, Portugal, October 2012.

[117] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J.Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun. Towards fully autonomous driving : Systems and algorithms. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 163–168, 2011.

[118] J.M. Anderson, K. Nidhi, K.D. Stanley, P. Sorensen, C. Samaras, and O.A. Oluwatola. *Autonomous vehicle technology : A guide for policymakers.* Rand Corporation, 2014.

[119] P. Merriaux, R. Boutteau, P. Vasseur, and X. Savatier. Algorithme de positionnement d'une passerelle à mouvements compensés à partir de mesures inertielles et lidar pour les opérations de maintenance des parcs éoliens offshore. In *Reconnaissance des Formes et Intelligence Artificielle (RFIA)*, 2014.

[120] P. Merriaux, R. Boutteau, P. Vasseur, and X. Savatier. Imu/lidar based positioning of a gangway for maintenance operations on wind farms. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4213–4219, Chicago, USA., September 2014.

[121] S.B. Thrun. Exploration and model building in mobile robot domains. In *IEEE International Conference on Neural Networks*, pages 175–180, 1993.

[122] A. Nuchter, H. Surmann, K. Lingemann, J. Hertzberg, and S. Thrun. 6d slam with an application in autonomous mine mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1998–2003, 2004.

[123] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J.J. Leonard. Past, present, and future of simultaneous localization and mapping : Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6) :1309–1332, 2016.

[124] M.A. Olivares-Mendez, I. Mellado, P. Campoy, I. Mondragon, and C. Martinez. A visual agv-urban car using fuzzy control. In *International Conference on Automation, Robotics and Applications (ICARA)*, pages 145–150, 2011.

[125] F. Taghaboni and J.M.A. Tanchoco. A lisp-based controller for free-ranging automated guided vehicle systems. *International Journal of Production Research*, 26(2) :173–188, 1988.

[126] I. Loevsky and I. Shimshoni. Reliable and efficient landmark-based localization for mobile robots. *Robotics and Autonomous Systems*, 58(5) :520–528, 2010.

[127] D. Ronzoni, R. Olmi, C. Secchi, and C. Fantuzzi. Agv global localization using indistinguishable artificial landmarks. In *IEEE International Conference on Robotics and Automation*, pages 287–292, 2011.

[128] L. Sabattini, V. Digani, C. Secchi, G. Cotena, D. Ronzoni, M. Foppoli, and F. Oleari. Technological roadmap to boost the introduction of agvs in industrial applications. In *Intelligent Computer Communication and Processing (ICCP)*, pages 203–208, 2013.

[129] C. Reinke and P. Beinschob. Strategies for contour-based self-localization in large-scale modern warehouses. In *IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 223–227, 2013.

[130] G.D. Tipaldi, D. Meyer-Delius, and W. Burgard. Lifelong localization in changing environments. *International Journal of Robotics Research (IJRR)*, 32(14) :1662–1678, 2013.

[131] G. Jagbrant, J.P. Underwood, J. Nieto, and S. Sukkarieh. Lidar based localisation in almond orchards. In *Conference on Field and Service Robotics*, 2013.

[132] J.P. Underwood, G. Jagbrant, J.I. Nieto, and S. Sukkarieh. Lidar-based tree recognition and platform localization in orchards. *Journal of Field Robotics (JFR)*, 32(8) :1056–1074, 2015.

[133] T. Stoyanov, J. Saarinen, H. Andreasson, and A.J. Lilienthal. Normal distributions transform occupancy map fusion : Simultaneous mapping and tracking in large scale dynamic environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4702–4708, 2013.

[134] Y. Zhuang, N. Jiang, H. Hu, and F. Yan. 3-d-laser-based scene measurement and place recognition for mobile robots in dynamic indoor environments. *IEEE Transactions on Instrumentation and Measurement*, 62(2) :438–450, 2013.

[135] J. Zhang and S. Singh. Loam : Lidar odometry and mapping in real-time. *Robotics : Science and Systems (RSS)*, 2 :9, 2014.

[136] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier. Correction de nuages de points lidar embarqué sur véhicule pour la reconstruction d'environnement 3d vaste. In *Reconnaissance des Formes et Intelligence Artificielle (RFIA)*, Clermont-Ferrand, France, June 2016.

[137] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT Press, 2005.

[138] J. Levinson and S. Thrun. Robust vehicle localization in urban environments using probabilistic maps. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4372–4378. IEEE, 2010.

[139] P. Pfaff, R. Kümmerle, D. Joho, C. Stachniss, R. Triebel, and W. Burgard. Navigation in combined outdoor and indoor environments using multi-level surface maps. *IROS Workshop on Safe Navigation in Open and Dynamic Environments*, 7, 2007.

[140] M. Fallon, H. Johannsson, and J. Leonard. Efficient scene simulation for robust monte carlo localization using an rgb-d camera. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1663–1670. IEEE, 2012.

[141] O. El Hamzaoui. *Localisation et cartographie simultanées pour un robot mobile équipé d'un laser à balayage : CoreSLAM*. PhD thesis, Ecole Nationale Supérieure des Mines de Paris, 2012.

[142] K.M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. Octomap : A probabilistic, flexible, and compact 3d map representation for robotic systems. In *ICRA Workshop on best practice in 3D perception and modeling for mobile manipulation*, volume 2, 2010.

[143] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier. Localisation robuste en milieu industriel complexe. In *Colloque Gretsi (GRETSI)*, Lyon, France, September 2015.

[144] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier. Robust robot localization in a complex oil and gas industrial environment. *Journal of Field Robotics (JFR)*, 2017.

[145] P. Merriaux, R. Rossi, R. Boutteau, V. Vauchey, L. Qin, P. Chanuc, F. Rigaud, F. Roger, B. Decoux, and X. Savatier. Vikings autonomous inspection robot for the argos challenge. *IEEE Robotics and Automation Magazine (IEEE-RAM), Special Issue on Robots for in-situ Interventions*, 2018.

[146] P. Merriaux, R. Boutteau, R. Rossi, G. Coru, V. Vauchey, and X. Savatier. Synchronisation et calibrage entre un lidar 3d et une centrale inertielle pour la localisation précise d'un véhicule autonome. In *Journées scientifiques d'URSI-France (URSI)*, 2018.

[147] D.G. Dansereau, I. Mahon, O. Pizarro, and S.B. Williams. Plenoptic flow : Closed-form visual odometry for light field cameras. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4455–4462, San Francisco, USA, September 2011.

[148] F. Dong, S. H. Ieng, X. Savatier, R. Etienne-Cummings, and R. Benosman. Plenoptic cameras in real-time robotics. *International Journal of Robotics Research (IJRR)*, 32(2) :206–217, 2013.

[149] N. Zeller, F. Quint, and U. Stilla. From the calibration of a light-field camera to direct plenoptic odometry. *IEEE Journal of Selected Topics in Signal Processing*, 11(7) :1004–1019, 2017.

[150] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M.W. Achtelik, and R. Siegwart. The euroc micro aerial vehicle datasets. *International Journal of Robotics Research (IJRR)*, 35(10) :1157–1163, 2016.

[151] B. Přibyl, P. Zemčík, and M Čadík. Camera pose estimation from lines using plücker coordinates. arXiv preprint arXiv :1608.02824.

[152] C. Xu, L. Zhang, L. Cheng, and R. Koch. Pose estimation from line correspondences : A complete analysis and a series of solutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(6) :1209–1222, 2017.

[153] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet : A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(12) :2481–2495, 2017.