Approches 2D/2D pour le SFM à partir d'un réseau de caméras asynchrones

Rawia Mhiri

Institut National des Sciences Appliquées de Rouen

Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes

Thèse de Doctorat

Approches 2D/2D pour le SFM à partir d'un réseau de caméras asynchrones

présentée par :

Rawia MHIRI

Pour obtenir le titre de Docteur de l'INSA de Rouen

le 14 Décembre 2015

Soutenue devant le jury composé de :

Samia Bouchafa, Professeur des Universités, IBISC, Université d'Evry Val d'Essonne	Rapporteur
David Fofi, Professeur des Universités, Le2i, Université de Bourgogne	Rapporteur
Didier Aubert, Directeur de Recherche, Lepsis (IFSTTAR)	Examinateur
Abdelaziz Bensrhair, Professeur des Universités, LITIS, INSA de Rouen	Directeur de thèse
Pascal Vasseur, Professeur des Universités, LITIS, Université de Rouen	Co-directeur de thèse
Stéphane Mousset, Maitre de Conférence, LITIS, Université de Rouen	Encadrant de thèse
Rémi Boutteau, Enseignant chercheur, IRSEEM, ESIGELEC	Co-encadrant de thèse

The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one– and preferably only one –obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea – let's do more of those!

Pour Mahmoud,

Pour ma famille,

pour maman, pour papa,
pour Wafa, pour Zied et pour Edam

Remerciments

La plupart des gens dit que la thèse n'est jamais quelque chose de simple à faire. D'autres pensent que la thèse est une dualité complémentaire entre la souffrance, le plaisir et l'enrichissement. Personnellement, je pense que j'ai été vraiment contente et très épanouie dans les trois années de mon doctorat. Les témoignages de mes proches le confirment. J'ai adoré cette étape de ma vie et je me suis sentie complètement transformée et grandie par cette expérience. Je souhaite donc adresser mes plus vifs remerciements à des merveilleuses personnes qui ont fait de ces trois années de thèse une aventure superbe, un véritable enrichissement personnel et professionnel. J'adresse un grand merci aux personnes qui ont cru en moi et m'ont supportée pour arriver au bout de cette thèse.

Je tiens à remercier en premier lieu les membres du jury de m'avoir fait l'honneur de bien vouloir prendre connaissance de ce travail et de l'intérêt qu'ils ont bien voulu porter à ma thèse. Je voudrais remercier monsieur **David Fofi** et madame **Samia Bouchafa** pour leurs lectures très attentives ainsi que leurs remarques précieuses. Je remercie également monsieur **Didier Aubert** d'être président du jury.

A l'issue de la rédaction de ce travail doctoral, je suis convaincue que je n'aurais pu le réaliser sans l'aide précieuse de mes très chers encadrants et directeurs de thèse. Je vous remercie pour le temps que vous m'avez consacré pour toutes nos réunions et pour vos réponses qui m'ont permis de progresser dans cette recherche.

Je tiens tout d'abord à adresser mes remerciements les plus sincères à mon directeur de thèse monsieur **Abdelaziz Bensrhair**, le co-directeur du LITIS, pour sa disponibilité et son soutien tout au long de ces années que je travaille sous sa direction. Aziz, je vous remercie d'avoir dirigé cette thèse et m'avoir permis de la réaliser dans les meilleures conditions. Je n'oublie jamais que vous m'avez donné la chance de rejoindre l'équipe STI et je vous en suis très reconnaissante. Je vous remercie de m'avoir accueillie dans le laboratoire et d'avoir cru en moi. Depuis que je suis venue en France, en 2011, j'ai été toujours extrêmement impressionnée par vos qualités humaines, d'écoute, compréhension ainsi que par vos manières de gérer toute sorte de situations. Ce fut un grand plaisir de travailler avec vous.

Je souhaite exprimer ma très grande gratitude à mon co-directeur de thèse monsieur **Pascal Vasseur**, le directeur de l'équipe STI, pour l'aide inconditionnelle qu'il m'a apportée durant ces trois années, mais aussi pour ses encouragements répétés, pour ses relectures, pour ses réponses, pour son écoute et tout le reste. Pascal, je voudrais vous dire merci pour avoir piloté mon travail avec beaucoup de patience et pour m'avoir transmis et appris un peu de votre savoir scientifique et de votre savoir faire. Votre aide

scientifique m'a été énormément valorisante et m'a beaucoup guidée pour les aspects géométriques de même que pour les concepts que j'ai trouvé ambiguës. Je vous dois une grande majorité des connaissances que j'ai acquises. Merci infiniment pour vos conseils avisés, vos relectures minutieuses, et votre écoute qui ont été la clés de la réussite de ma thèse. J'espère avoir été digne de la confiance que vous m'avez accordée et que ce travail est finalement à la hauteur de vos espérances. J'espère également que vous oublierez toutes les petites bêtises (comme mes oublies des dates ou mes confusions) que j'ai faites pendant ces trois années et que vous garderez une bonne image de moi. La manière dont vous avez dirigé ma thèse restera pour moi un modèle. J'ai été extrêmement ravie de travailler avec vous.

J'associe à ces remerciements, mon encadrant monsieur **Stéphane Mousset** pour m'avoir offert un encadrement de qualité depuis mon stage de projet de fin d'étude jusqu'à ce aujourd'hui. Sa rigueur, sa capacité d'analyse des problèmes et ses très nombreuses connaissances m'ont permis de progresser et de m'améliorer. Stéphane, merci pour votre aide, pour votre disponibilité, pour votre bonne humour et vos encouragements tout au long de ces années. Merci de m'avoir formée et de m'avoir donnée l'envie de continuer dans la recherche. Je suis très honorée de vous avoir eu pour encadrant.

J'exprime ma reconnaissance à mon encadrant monsieur **Rémi Boutteau** de m'avoir prodigué maints conseils et de l'aide durant la phase d'acquisition, mais aussi pour ses encouragements répétés au cours de la thèse et surtout au cours de la rédaction de ce manuscrit. Rémi, un grand merci pour toutes vos relectures et pour votre disponibilité. Je vous remercie vivement de m'avoir aidée pour résoudre mes problèmes de développement surtout dans la partie de l'ajustement de faisceaux. Nos échanges m'ont beaucoup aidée. Je suis très enchantée de vous avoir eu pour encadrant.

Je souhaite remercier monsieur **In So Kweon**, le directeur du laboratoire Rcv lab au KAIST en Corée du sud, pour son accueil dans son laboratoire et pour nos échanges scientifiques fructueux. Le séjour d'un mois en Corée du sud a été une expérience humaine très enrichissante.

J'adresse mes remerciements aux membres de l'équipe ≪Systèmes de Transport Intelligents≫ pour les discussions que j'ai pu avoir durant les réunions d'équipe ou en dehors qui m'ont apporté beaucoup. Je tiens à remercier particulièrement monsieur **Sébastien Kramm** pour toutes les réponses à mes questions concernant les notions de géométrie et madame **Samia Ainouz** pour ses encouragements ainsi que pour les bons moments partagés. Je remercie également **Elsa**, **Sandra** et **Brigitte** pour leurs aides administratives et techniques tout au long de cette période.

Je souhaite remercier le stagiaire, **Hichem Maïza**, avec qui j'ai eu la chance de travailler. L'ambiance de travail à été excellente et nous avons pu ainsi allier bons moments et séances de travail qui se sont prolongées tard le soir et même pendant les week-ends. Hichem, je te remercie pour ton aide durant les longues journées d'acquisition, pour la mise en place du système et répéter le fastidieux protocole de calibration.

Je remercie toutes les personnes formidables que j'ai rencontrées par le biais de cette thèse. Je pense particulièrement à tous les thésards de l'équipe et en particulier à ma petite Vanee, monsieur Bonardi, Alina, Aymeric, Pierre, René Emmanuel, Dina, Hamza et Nadine pour le climat sympathique et la bonne ambiance de travail mais aussi pour l'aide précieuse qu'ils m'ont fournie pour la campagne d'acquisition des données. J'espère que vous n'oublierez pas ma présence qui n'a pas toujours été discrète. Mention spéciale à ma complice Vanee, madame Fan Wang Lian, pour l'amitié partagée, les bons et les mauvais moments qui finissent toujours par un fou rire. Vanee, merci pour tous les bon plans partagés, n'oublie jamais la personne qui te comprend le plus au monde. Je voudrais exprimer particulièrement toute mon amitié à Fabien, monsieur Bonardi, pour ses relectures et pour nos échanges scientifiques et personnelles. Fabien, merci d'être aussi bavard que moi, n'oublie jamais la personne qui t'a taquiné le plus au monde. Je profite pour remercier Alina pour tous son aide scientifique et technique depuis le début de ma thèse. Alina, merci pour tous les gâteaux.

Il me sera très difficile de remercier tout le monde car c'est grâce à l'aide de nombreuses personnes que j'ai pu mener cette thèse à son terme.

Enfin, ma reconnaissance va à ceux qui ont plus particulièrement assuré le soutien affectif de ce travail : ma famille ainsi que la famille Hakim, en particulier à mes parents, mes frères, ma sœur et mes beaux parents. Malgré mon éloignement depuis de nombreuses années, leur confiance et leur amour m'encouragent encore plus tous les jours. Vous êtes les piliers fondateurs de ce que je suis aujourd'hui. Je remercie également ma petite belle-sœur Asma pour m'avoir supportée et encouragée dans mes moments de force et de faiblesse. Finalement, je remercie ma très chère amie Rihab pour sa présence, ses encouragements et ses conseils.

Mes derniers remerciements vont à mon cher époux pour son soutien quotidien indéfectible. Chéri, le plus grand merci s'adresse à toi pour m'avoir aidée, pour tes relectures, pour tes conseils et surtout pour m'avoir supportée et encouragée du début jusqu'à la fin. Cette thèse et moi te devons beaucoup. Merci. J'avais cette thèse comme bonne excuse, désormais je ne sais pas quelle sera la prochaine... ...

Résumé

Les systèmes d'aide à la conduite et les travaux concernant le véhicule autonome ont atteint une certaine maturité durant ces dernières années grâce à l'utilisation de technologies avancées. Une étape fondamentale pour ces systèmes porte sur l'estimation du mouvement et de la structure de l'environnement (Structure From Motion) pour accomplir plusieurs tâches, notamment la détection d'obstacles et de marquage routier, la localisation et la cartographie. Pour estimer leurs mouvements, de tels systèmes utilisent des capteurs relativement chers. Pour être commercialisés à grande échelle, il est alors nécessaire de développer des applications avec des dispositifs bas coûts. Dans cette optique, les systèmes de vision se révèlent une bonne alternative. Une nouvelle méthode basée sur des approches 2D/2D à partir d'un réseau de caméras asynchrones est présentée afin d'obtenir le déplacement et la structure 3D à l'échelle absolue en prenant soin d'estimer les facteurs d'échelle. La méthode proposée, appelée méthode des triangles, se base sur l'utilisation de trois images formant un triangle : deux images provenant de la même caméra et une image provenant d'une caméra voisine. L'algorithme admet trois hypothèses : les caméras partagent des champs de vue communs (deux à deux), la trajectoire entre deux images consécutives provenant d'une même caméra est approximée par un segment linéaire et les caméras sont calibrées. La connaissance de la calibration extrinsèque entre deux caméras combinée avec l'hypothèse de mouvement rectiligne du système, permet d'estimer les facteurs d'échelle absolue. La méthode proposée est précise et robuste pour les trajectoires rectilignes et présente des résultats satisfaisants pour les virages. Pour affiner l'estimation initiale, certaines erreurs dues aux imprécisions dans l'estimation des facteurs d'échelle sont améliorées par une méthode d'optimisation : un ajustement de faisceaux local appliqué uniquement sur les facteurs d'échelle absolue et sur les points 3D. L'approche présentée est validée sur des séquences de scènes routières réelles et évaluée par rapport à la vérité terrain obtenue par un GPS différentiel. Une application fondamentale dans les domaines d'aide à la conduite et de la conduite automatisée est la détection de la route et d'obstacles. Pour un système asynchrone, une première approche pour traiter cette application est présentée en se basant sur des cartes de disparité éparses.

Mots-clés : Structure From Motion, odométrie visuelle, réseau multi-caméra non synchronisées, méthode des triangles, ajustement de faisceaux local, détection d'obstacles.

Abstract

Driver assistance systems and autonomous vehicles have reached a certain maturity in recent years through the use of advanced technologies. A fundamental step for these systems is the motion and the structure estimation (Structure From Motion) that accomplish several tasks, including the detection of obstacles and road marking, localization and mapping. To estimate their movements, such systems use relatively expensive sensors. In order to market such systems on a large scale, it is necessary to develop applications with low cost devices. In this context, vision systems is a good alternative. A new method based on 2D/2D approaches from an asynchronous multi-camera network is presented to obtain the motion and the 3D structure at the absolute scale, focusing on estimating the scale factors. The proposed method, called Triangle Method, is based on the use of three images forming a triangle shape : two images from the same camera and an image from a neighboring camera. The algorithm has three assumptions : the cameras share common fields of view (two by two), the path between two consecutive images from a single camera is approximated by a line segment, and the cameras are calibrated. The extrinsic calibration between two cameras combined with the assumption of rectilinear motion of the system allows to estimate the absolute scale factors. The proposed method is accurate and robust for straight trajectories and present satisfactory results for curve trajectories. To refine the initial estimation, some errors due to the inaccuracies of the scale estimation are improved by an optimization method : a local bundle adjustment applied only on the absolute scale factors and the 3D points. The presented approach is validated on sequences of real road scenes, and evaluated with respect to the ground truth obtained through a differential GPS. Finally, another fundamental application in the fields of driver assistance and automated driving is road and obstacles detection. A method is presented for an asynchronous system based on sparse disparity maps.

key-words : Structure From Motion, visual odometry, asynchronous multi-camera system, Triangle-based Method, local bundle adjustment, obstacle detection.

Table des matières

Liste des figures

Liste des Tableaux

Notations

Règles générales :	
a	Scalaire
$\mathbf{A}$,$\mathbf{a}$	Vecteur ou matrice
Mathématiques :	
$\mathbf{A}^\top$	Transposée de A
$\mathbf{A}^{-1}$	Inverse de A
$\mathbf{K}$	Matrice des paramètres intrinsèques
$\mathbf{T}$	Matrice de pose
$\mathbf{R}$	Matrice de rotation
$\mathbf{t}$	Vecteur unitaire de translation
$\mathbf{T}_i^j$	Matrice de transformation du repère i au repère j
$\mathbf{R}_i^j$	Matrice de rotation du repère i au repère j
$\mathbf{t}_i^j$	Vecteur unitaire de translation du repère i au repère j
$\mathbf{X}$	Point 3D
$\mathbf{x}$	Point 2D
$\widetilde{x}$	Point 2D normalisé
$\widehat{x}$	Point 2D projeté dans l'image à partir du point 3D
$\mathbf{x}_i$	Point 2D dans l'image i
$\mathbf{x}^i$	Point 2D représenté dans le repère i
$\mathbf{X}^i$	Point 3D représenté dans le repère i
$\mathbf{J}$	Matrice jacobienne
$[\mathbf{A}]_\times$	Matrice antisymétrique
$\mathbf{F}$	Matrice fondamentale
$\mathbf{E}$	Matrice essentielle
t	Instant d'une prise de vue
I	Image
I_i	Image i
C_i	Caméra i
$\mathbf{I}$	Matrice identité
$\mathbb{P}$	Espace projectif

Introduction

Grâce à l'utilisation des technologies avancées, certains véhicules de nos jours sont équipés de systèmes d'aide à la conduite avancés (Advanced driver assistance systems, ADAS). Les ADAS aident le conducteur à manœuvrer son véhicule de façon sûre et précise afin d'éviter les situations dangereuses et les risques d'accidents. Les systèmes d'assistance à la conduite ont été largement étudiés pour assister le conducteur dans sa perception de l'environnement. Plusieurs systèmes ont été commercialisés comme les systèmes d'assistance au parking, les systèmes de détection de piétons et d'obstacles, les systèmes d'alerte de franchissement involontaire de ligne, les avertisseurs de déviation de trajectoire, etc.

Grâce à la sophistication et à la multiplication croissante des aides à la conduite, la conduite automatisée a atteint une certaine maturité durant ces dernières années. Plusieurs démonstrations augurent la réussite de la conduite automatisée à moyen terme : la Google car, le projet SARTRE, le grand challenge DARPA (Defence Advanced Research Projects Agency), VeDeCoM (Véhicule Décarboné Communicant et sa Mobilité), le challenge du laboratoire Vislab et le projet Cybercars de l'INRIA. Ces différents projets ont démontré la faisabilité de véhicules complètement autonomes.

Les systèmes d'assistance à la conduite et les véhicules autonomes ont besoin d'avoir une parfaite connaissance de leur environnement proche afin d'être efficaces et d'assurer la sécurité. Ces domaines nécessitent encore un travail important de recherche et de tests. En effet, avant une mise à disposition pour le grand public, les systèmes utilisés nécessitent encore une amélioration de leur fiabilité et de leur robustesse. Ces systèmes peuvent être utilisés pour accomplir plusieurs tâches, notamment la détection d'obstacles et de marquage routier, la localisation et la cartographie. Toutefois, une étape commune à ces applications porte sur l'estimation du déplacement, appelée aussi odométrie visuelle.

Pour estimer leurs mouvements, la plupart des véhicules intelligents utilisent des capteurs proprioceptifs et extéroceptifs relativement chers, comme l'utilisation des lidars 3D (par exemple le Velodyne), ou des GPS différentiels pour certaines applications etc. Ces dispositifs ne pouvant être commercialisés à grande échelle, développer des systèmes

bas coût est alors nécessaire. Dans cette optique, les systèmes de vision se révèlent une bonne alternative. Ces systèmes peuvent être mis en place suivant plusieurs configurations : mono-caméra, stéréovision et réseaux de caméras multiples.

Pour accomplir la tâche d'odométrie visuelle, l'estimation des poses des caméras du système en déplacement représente l'étape cruciale. Il s'agit de déterminer la rotation, la translation et le facteur d'échelle entre deux positions consécutives du système. Lorsque l'échelle métrique absolue (l'échelle réelle du déplacement) est désirée, l'odométrie visuelle doit avoir une connaissance de la structure 3D de la scène (à partir de la stéréo vision ou à partir d'une connaissance a priori). Dans le cas de la configuration mono caméra, une étape d'initialisation suivie du maintien du facteur d'échelle durant le mouvement est nécessaire. Pour une configuration stéréo ou multiple, l'estimation du mouvement et de la structure (Structure From Motion, SFM) se base sur la géométrie épipolaire pour déduire l'échelle métrique. Cependant, l'estimation du mouvement à l'échelle n'est possible que si les caméras sont synchronisées et si la scène est prise sous différents angles de vue au même instant.

Un système multi-caméras offre une vision parfaite de l'environnement du véhicule mais la synchronisation devient un inconvénient majeur. En effet, elle impose l'ajout d'un circuit électronique et d'un câblage supplémentaire. De plus, un système synchrone utilise en général des périphériques coûteuses. En outre, l'acquisition d'images à partir d'un système asynchrone peut être facilement implémentée. Un réseau de caméras asynchrones présente plusieurs avantages. Tout d'abord, un tel système peut être développé avec des dispositifs à faible coût, ce qui est souvent souhaitable pour les applications réelles. Dans ce cas, l'acquisition ne dépend plus de la caméra la plus lente ou du dispositif de synchronisation lui-même. Les images peuvent être acquises en continu à partir de chaque caméra séparément. La contrainte de bande passante qui apparaît dans le cas des caméras synchronisées peut être également contournée. Enfin, le réseau peut être facilement modifié : l'ajout ou la suppression des caméras ne nuisent pas au fonctionnement du système.

Le SFM (Structure From Motion) est désormais un problème bien maîtrisé dans le cas d'un réseau multi-caméras synchronisées. Jusqu'à présent, les caméras asynchrones ont été rarement étudiées dans le cadre de l'estimation du mouvement et de la structure. C'est dans ce cadre que nous présentons de nouvelles approches pour l'estimation du mouvement et de la structure à partir d'un réseau de caméra multiples non synchronisées.

Contexte de la thèse

Ce travail de recherche a été mené pendant la période 2012-2015 à l'INSA de Rouen au sein du laboratoire d'Informatique, du Traitement de l'Information et des Systèmes (LITIS). Il s'inscrit dans le cadre d'un projet ANR blanc international (ANR-11-IS03-0003) intitulé DrAACaR (Driver Assistance by Asynchronous Camera Ring). Ce projet est une collaboration entre le laboratoire LITIS, le laboratoire d'Électronique, Informatique et Image (Le2I), et les laboratoires USRG Lab et RCV Lab (Robotics & Computer Vision Lab) du KAIST (Korean Advanced Institute of Sciences and Technology) en Corée du sud. Ce projet porte sur le développement de méthodes et d'algorithmes permettant de modéliser l'environnement direct d'un véhicule équipé d'un réseau de caméras asynchrones.

Ce projet, DrAACaR, s'intéresse au développement d'un réseau asynchrone pour un système de conduite assistée. Pour résoudre ce problème, il est nécessaire de développer des méthodes de mises en correspondance sur les séquences obtenues à partir des différentes caméras. Trois types d'approches de mises en correspondance sont envisagées : des approches 2D-2D, des approches 2D-3D et des approches 3D-3D. Dans cette thèse, nous avons choisi de développer un réseau asynchrone basé sur des approches 2D-2D pour un système de conduite assistée. Les autres partenaires du projet étudient les autres approches proposées 3D-3D et 2D-3D.

Objectifs

L'objectif de ce travail est de développer une méthode pour estimer le déplacement d'un véhicule et reconstruire son environnement proche avec un réseau de caméras asynchrones afin de détecter la route et les obstacles. En utilisant des approches 2D-2D, les caractéristiques communes entre les séquences obtenues des différentes caméras peuvent être mises en correspondance. Le problème dans notre cas est que les séquences n'étant pas synchronisées, le processus d'appariement devient particulièrement critique et les méthodes existantes ne peuvent pas le résoudre. Il faut développer de nouvelles techniques en prenant en compte la non-synchronisation. Il est donc important de fixer des hypothèses sur le système : les caméras sont montées sur un système rigide, se chevauchent deux à deux et leur calibration géométrique est effectuée hors ligne. Le recouvrement de deux caméras voisines doit être relativement important pour pouvoir effectuer des appariements entre des images non synchronisées quand le véhicule est en circulation. En effet, les images non synchronisées sont décalées dans l'espace et dans le temps. Pour avoir des objets communs issus de la même scène entre les images, il

faut que le recouvrement entre les caméras voisines soit suffisant. Dans cette thèse, nous avons choisi une configuration avec angle de chevauchement aux alentours de 30 degrés, à l'arrêt du véhicule, en utilisant des caméras ayant des angles de 60 degrés.

Contributions

Les principales contributions de notre méthode se résument dans ces aspects de non-synchronisation afin d'estimer la trajectoire d'un véhicule et la structure 3D de son environnement proche à l'échelle absolue sans aucune calibration temporelle. Cette méthode est basée sur l'hypothèse que le mouvement entre deux vues consécutives est rectiligne (linéaire ou lisse).

Les travaux effectués durant cette thèse ont donné lieu à plusieurs publications scientifiques :

1. Dans Mhiri et al. [16][17], une nouvelle méthode d'estimation de mouvement à l'échelle a été présentée. Nous appelons cette méthode la méthode des triangles. En se basant sur un triplet d'images provenant de deux caméras, nous estimons les poses relatives entre les images avec les algorithmes classiques de SFM. Les facteurs d'échelle absolue sont ensuite estimés en intégrant une pose virtuelle d'une des caméras dans la forme du triangle formé par le triplet d'images.
2. Dans Mhiri et al. [18][19], une étape d'optimisation a été présentée afin d'améliorer la précision de l'estimation initiale obtenue en appliquant la méthode des triangles. Un ajustement de faiseaux a été développé pour réajuster les facteurs d'échelle et la structure 3D.
3. Dans Mhiri et al. [20], une application de détection d'obstacles à partir d'images non synchronisées a été présentée. Ayant le mouvement et la structure à l'échelle, nous calculons une carte de U-V-disparité éparse à partir de laquelle nous détectons les obstacles présents sur la route. Nous utilisons la transformée de Hough pour détecter les droites qui représentent les obstacles dans la carte U-V-disparité. Le plan de la route est suivi et prédit par un filtre de Kalman.

Organisation du manuscrit

Ce manuscrit est divisé en cinq principaux chapitres :

1. Dans le premier chapitre, nous présentons quelques notions de la géométrie projective et de la reconstruction 3D, une étude bibliographique sur les méthodes

de l'estimation du mouvement et de la structure et leurs rapports avec la vision omnidirectionnelle et la vision asynchrone.

2. Dans le deuxième chapitre, une évaluation de plusieurs détecteurs et descripteurs de l'état de l'art en vu des images non synchronisées est présentée. Nous avons sélectionné le détecteur FAST et le descripteur BRIEF qui ont surpassé les autres algorithmes en terme de performance.
3. Dans le troisième chapitre, la méthode d'estimation du mouvement et de la structure à l'échelle, que nous appelons la méthode des triangles, est présentée. Il s'agit d'une méthode qui se base sur trois images provenant de deux caméras non synchronisées pour estimer les facteurs d'échelle correspondant à chaque transformation.
4. Dans le quatrième chapitre, une étape d'optimisation a été introduite pour optimiser la trajectoire obtenue par la méthode des triangles. Nous supposons que l'estimation de la rotation et de la translation est acceptable. La contribution figure dans l'optimisation du facteur d'échelle et de la structure 3D uniquement.
5. Dans le cinquième chapitre, la méthode présentée dans le chapitre 3 est étendue pour un réseau de N caméras et une application de détection du plan de la route et des obstacles est introduite.

Liste des publications

— Rawia Mhiri, Pascal Vasseur, Stéphane Mousset, Rémi Boutteau, and Abdelaziz Bensrhair. "Visual odometry with unsynchronized multi-cameras setup for intelligent vehicle application". Intelligent Vehicles Symposium Proceedings (IV), pages 1339-1344. IEEE, 2014.

— Rawia Mhiri, Pascal Vasseur, Stéphane Mousset, Rémi Boutteau, and Abdelaziz Bensrhair. "Estimation à l'échelle du mouvement d'un réseau multi-caméras non synchronisées". Reconnaissance de Formes et Intelligence Artificielle (RFIA) 2014, 2014.

— Rawia Mhiri, Pascal Vasseur, Stéphane Mousset, Rémi Boutteau, and Abdelaziz Bensrhair. "Accurate scale estimation based on unsynchronized camera network". International Conference on Image Processing (ICIP), IEEE, 2015.

— Rawia Mhiri, Pascal Vasseur, Stéphane Mousset, Rémi Boutteau, and Abdelaziz Bensrhair. "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), 2015.

— Rawia Mhiri, Hichem Maiza, Stéphane Mousset, Khaled Taouil, Pascal Vasseur, and Abdelaziz Bensrhair. "Obstacle detection using unsynchronized multi-camera

network". International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), 2015.

Chapitre 1

État de l'art

Introduction

Le problème de recouvrement des poses relatives des caméras et de la structure tridimensionnelle à partir d'un ensemble d'images est connue sous le nom de l'estimation du mouvement et de la structure (Structure From Motion, SFM). L'odométrie visuelle est un cas particulier de SFM [1]. Le SFM est plus général dans le sens où la structure 3D et le mouvement ont le même degré d'importance dans ce processus tandis que l'odométrie visuelle (VO) se focalise sur le mouvement 3D d'une caméra [1]. Pour chaque processus d'estimation, une étape d'optimisation est nécessaire. Le SFM dépend de la taille de l'ensemble des images à traiter. Une optimisation globale est généralement coûteuse en terme de temps de calcul. Cependant, l'odométrie visuelle est plus adaptée pour les applications temps réel et l'optimisation de l'estimation est faite localement.

Après son apparition en 1996 [21] et en 2001 [22], le terme odométrie visuelle est devenu très connu en 2004 suite à son introduction par Nister [23]. Par définition, l'odométrie visuelle concerne l'estimation du mouvement d'un système par lui-même à partir d'images acquises par une ou plusieurs caméras embarquées [1]. Ce processus consiste à estimer le mouvement relatif entre deux caméras. Il existe deux types d'approches : les approches denses et les approches éparses. Dans cette thèse, nous étudions que les approches éparses qui se basent sur les points d'intérêt.

Les transformations relatives T entre les positions adjacentes d'une caméra (ou d'un réseau de caméras) sont calculées à partir des points caractéristiques appariés entre les images. La trajectoire du système est obtenue par concaténation des transformations T. La concaténation des transformations permet d'obtenir la pose de la caméra par rapport à un repère initial (la première position ou un repère monde), figure 1.1.

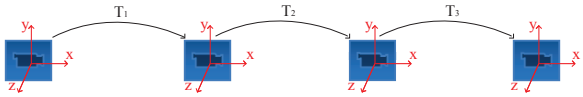


FIGURE 1.1: Odométrie visuelle : les transformations relatives **T** entre les positions adjacentes d'une caméra (ou d'un réseau de caméras) sont calculées à partir des points caractéristiques appariés. La trajectoire du système est obtenue par concaténation des transformations **T** pour obtenir la pose de la caméra par rapport à un repère initial (la première position ou un repère monde)

Le processus d'odométrie visuelle commence par la détection des points caractéristiques de la scène dans chaque image par des algorithmes de détection de points d'intérêt. Ensuite, les points d'intérêt identifiés sont caractérisés par des descripteurs locaux. Pour déterminer la transformation entre deux poses adjacentes d'une caméra, les points d'intérêt des deux images sont mis en correspondance, ensuite la transformation géométrique entre les images est estimée à partir des points appariés. Une estimation itérative provoque toujours des erreurs qui seront accumulées au fil du temps et conduisent à des dérives que même les algorithmes robustes ne peuvent pas éviter. Parmi les méthodes qui permettent de remédier à ce problème, un processus d'optimisation tel que l'ajustement de faisceaux peut être appliqué pour optimiser l'estimation du mouvement et de la structure 3D. Ces étapes sont résumées dans la figure 1.2 qui traduit le diagramme présenté par Scaramuzza et Fraundorfer dans leurs tutoriel sur l'odométrie visuelle [1].

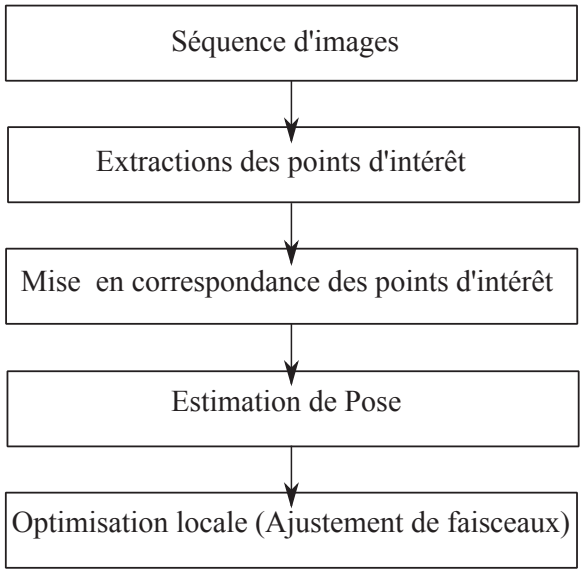


FIGURE 1.2: Les principales étapes d'odométrie visuelle [1]

Les algorithmes d'estimation de mouvement ont été largement étudiés [24] [25] [5]. Les travaux de l'état de l'art sont capables d'estimer le mouvement à l'échelle absolue. Néanmoins, ils utilisent des caméras synchrones. Pour relâcher cette contrainte coûteuse, un système composé de dispositifs non synchronisés peut être utilisé.

Ce chapitre bibliographique détaille quelques travaux de l'état de l'art qui sont capables d'estimer le mouvement à l'échelle absolue ainsi que les travaux qui relâchent la contrainte de la synchronisation. La première partie de ce chapitre présente quelques notions de géométrie projective et de reconstruction 3D. La deuxième partie étudie le processus de SFM et de l'odométrie visuelle. La troisième partie introduit la vision omnidirectionnelle avec les différentes configurations de caméras. La dernière partie présente les méthodes de la vision asynchrone dans la littérature. Finalement, nous discutons les techniques possibles pour résoudre le problème de l'estimation de mouvement en utilisant un système asynchrone.

1.1 Notion de géométrie : géométrie projective et reconstruction 3D

1.1.1 Transformation euclidienne

Une transformation euclidienne $\mathbf{T}$ est une transformation 3D rigide (ou une transformation de pose) entre deux repères orthonormés appartenant à l'espace Euclidien : un repère de référence et un repère courant. $\mathbf{T}$ est une matrice homogène de dimension 4x4 qui appartient au groupe Spécial Euclidien $\mathbb{SE}(3) \subset \mathbb{R}^{4\times4}$. La transformation $\mathbf{T}$ est composée d'une matrice de rotation $\mathbf{R}$ de dimension 3x3 qui appartient au groupe Spécial Orthogonal $\mathbb{SO}(3) \subset \mathbb{R}^{3\times3}$ et d'un vecteur de translation $s \times \mathbf{t}$ de dimension 3x1 qui appartient à $\mathbb{R}^3$, s est le facteur d'échelle et $\mathbf{t}$ est le vecteur directeur de la translation, $\mathbf{T}$ est exprimé dans l'équation 1.1. Une matrice de rotation est une matrice orthogonale de déterminant 1, ce qui peut s'exprimer par $\mathbf{R}^\intercal\mathbf{R} = \mathbf{I}$.

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & s \times \mathbf{t} \\ 0^\intercal & 1 \end{bmatrix} \tag{1.1}$$

$\mathbf{T}$ est une matrice homogène et son inverse est :

$$\mathbf{T}^{-1} = \begin{bmatrix} \mathbf{R}^\intercal & -\mathbf{R}^\intercal \times s \times \mathbf{t} \\ 0^\intercal & 1 \end{bmatrix} \tag{1.2}$$

Un point 3D $\mathbf{X}$ de l'espace Euclidien, défini par rapport à un repère R, peut être exprimé dans un repère R' par $\mathbf{X}'$ tel que $\mathbf{X}' = \mathbf{T}\mathbf{X}$ (figure 1.3).

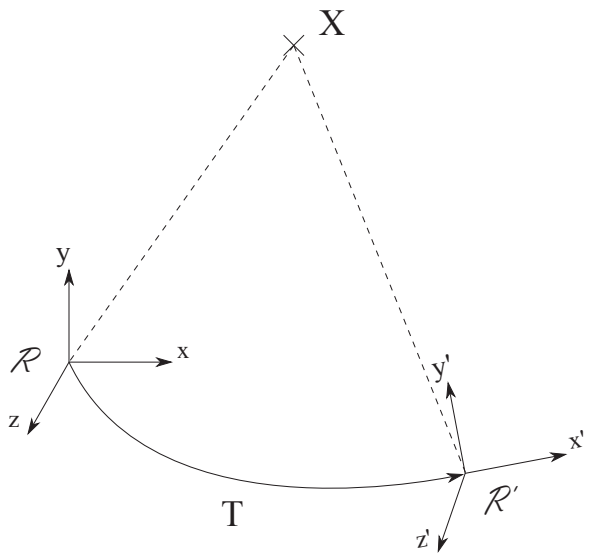


FIGURE 1.3: Transformation euclidienne 3D **T** entre les deux repères R et R'

1.1.2 Géométrie projective

Une caméra perspective classique permet de passer de l'environnement 3D à une représentation 2D de la scène : l'image. L'image offre des informations photométriques et des informations sur la géométrie de la scène. Afin de pouvoir exploiter ces informations géométriques, nous avons besoin d'un modèle qui définit le passage du monde 3D à l'image 2D. En vision par ordinateur, le modèle le plus utilisé est le modèle sténopé, figure 1.4. C'est un modèle simple qui représente la projection perspective en supposant que les rayons lumineux passent par un unique point avant de se projeter sur le capteur numérique.

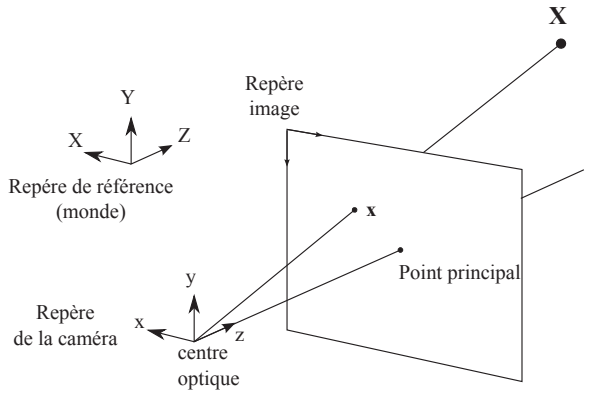


FIGURE 1.4: Le modèle sténopé (« pinhole » en anglais)

Paramètres intrinsèques

Les paramètres intrinsèques modélisent les caractéristiques du capteur de la caméra : les paramètres internes, les paramètres de distorsion et la forme des pixels. Les paramètres

internes d'une caméra définissent la focale et le point principal. La matrice de calibrage $\mathbf{K}$ regroupe ces paramètres :

$$\mathbf{K} = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{1.3}$$

fx et fy représentent la distance focale en direction horizontale et verticale exprimée en pixels, u_0 et v_0 sont les coordonnées du point principal dans le repère image et s, le "skew ratio", modélise le fait que les axes du repère image ne soient pas parfaitement perpendiculaires. En général, s est considéré nul car le défaut de la forme des pixels est souvent négligeable.

La transformation associée aux paramètres intrinsèques est la transformation des coordonnées du repère caméra (l'espace 2D métrique, ou la rétine) au repère image (coordonnées 2D en pixels) :

$$\mathbf{x}^{image} = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}^{caméra,rétine} \tag{1.4}$$

Paramètres extrinsèques

La pose de la caméra par rapport à un repère de référence représente ses paramètres extrinsèques : sa position et son orientation. Comme pour la matrice de transformation euclidienne, les paramètres extrinsèques sont donc définis par une matrice de rotation et un vecteur de translation.

La transformation du repère monde au repère caméra est la transformation inverse de celle associée aux paramètres extrinsèques. On note $\mathbf{T}_{monde}^{caméra}$ la transformation du repère monde au repère caméra avec la matrice de rotation associée $\mathbf{R}$ et le vecteur de translation associé $s \times \mathbf{t}$ (s est le facteur d'échelle et $\mathbf{t}$ est le vecteur directeur de la translation,), équation 1.5.

$$\mathbf{X}^{caméra} = \mathbf{T}_{monde}^{caméra}\mathbf{X}^{monde} \Longrightarrow \mathbf{X}^{caméra} = \begin{bmatrix} \mathbf{R} & s \times \mathbf{t} \\ 0^\intercal & 1 \end{bmatrix}_{monde}^{caméra} \mathbf{X}^{monde} \tag{1.5}$$

Transformation projective

Une transformation projective permet de projeter un point 3D de la scène de l'espace projectif 3D $\mathbb{P}^3$ vers un point 2D dans l'image (de l'espace projectif 2D $\mathbb{P}^2$). Cette transformation permet de modéliser le processus effectué par une caméra lors de l'acquisition d'images. Tous les points de l'image $\mathbf{x}$ sont calculés à partir des points 3D $\mathbf{X}$ de la scène en passant par une matrice de projection P. Un point $\mathbf{X}$ de l'espace tridimensionnel est représenté par un vecteur de taille 4x1 en coordonnées homogènes.

Un point 3D de coordonnées $(X, Y, Z)^\intercal$ dans un repère de référence est représenté par $(X, Y, Z, 1)^\intercal$ en coordonnées homogènes. Un point homogène avec la quatrième coordonnée nulle est un point à l'infini. De même, un point 2D $\mathbf{x}$ de coordonnées $(x, y)^\intercal$ dans le repère image est représenté par $(x, y, 1)^\intercal$ en coordonnées homogènes. La projection du point 3D dans l'image est définie par l'équation suivante :

$$\begin{aligned} \mathbf{x}^{image} &= \mathbf{P}^{image}_{monde} \quad \mathbf{X}^{monde} \\ \text{ou } \mathbf{x} &= \mathbf{P}\mathbf{X} \end{aligned} \tag{1.6}$$

La projection centrale est la transformation de l'espace 3D du repère caméra $\mathbb{P}^3$ vers l'espace 2D (rétine) du repère caméra $\mathbb{P}^2$. Cette transformation est modélisée par une projection linéaire :

$$\mathbf{x}^{caméra,rétine} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{X}^{caméra} \tag{1.7}$$

La transformation projective permettant le passage de l'espace 3D vers l'image est résumée par l'équation suivante :

$$\mathbf{x}^{image} = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}^{image}_{rétine} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}^{caméra,rétine}_{caméra} \begin{bmatrix} \mathbf{R} & s \times \mathbf{t} \\ 0^\intercal & 1 \end{bmatrix}^{caméra}_{monde} \mathbf{X}^{monde} \tag{1.8}$$

Pour la suite de ce manuscrit, nous considérons la fonction de projection suivante :

$$
\begin{aligned}
F : \mathbb{P}^3 &\longrightarrow \mathbb{P}^2 \\
\mathbf{X}^{monde} \longmapsto \mathbf{x}^{image} &= \mathbf{K} \begin{bmatrix} \mathbf{R} & s \times \mathbf{t} \\ 0^\intercal & 1 \end{bmatrix}_{monde}^{caméra} \mathbf{X}^{monde}
\end{aligned} \tag{1.9}
$$

1.1.3 Calcul de Pose

L'estimation de pose d'une caméra consiste à estimer sa position et son orientation par rapport à un repère de référence. Le repère de référence est en général un repère fixe dit "repère monde". Dans cette thèse, comme nous estimons le mouvement des caméras, nous considérons le repère de la première pose de la trajectoire (la première caméra qui acquiert une image) comme le repère monde ou de référence. Comme le système est se déplace et que les images ne sont pas synchronisées, nous calculons la pose entre deux images consécutives (provenant de la même caméra ou de deux caméras voisines) pour pouvoir estimer la pose du réseau de caméras.

L'estimation de pose peut être effectuée à partir des approches éparses basées sur les correspondances 3D-3D, 2D-3D ou 2D-2D.

Dans le cas des méthodes basées sur des correspondances 3D-3D, il est nécessaire de trianguler les points 3-D à chaque instant en utilisant un système stéréoscopique. Pour ce faire, il est nécessaire que les caméras soient parfaitement synchronisées.

Les méthodes basées sur les correspondances entre les points 2D et des points 3D de positions connues cherchent à déterminer la pose de la caméra dans le repère défini par ces points 3D. Pour ce faire, ces méthodes nécessitent au moins 3 points 3D dans le cas des caméras calibrées et 6 points 3D dans le cas des caméras non calibrées. Ces méthodes sont rapides et elles n'ont pas d'ambiguïté sur l'échelle absolue mais pour pouvoir les utiliser il faut avoir une connaissance sur les positions des points 3D. Pour l'approche proposée dans cette thèse, nous ne disposons que des informations 2D et ne posons aucune contrainte sur la scène 3D.

À partir des points 2D observés dans deux images, l'estimation de pose peut être effectuée à partir des correspondances 2D-2D. La méthode basée sur les correspondances 2D-2D est connue par l'estimation de pose relative. Le mot "relative" vient du fait que le calcul de pose est relatif à une position de référence de la caméra. En général, un ensemble de primitives appariées entre les deux images permettent de calculer la translation et la rotation 3D. Comme nous nous basons sur les techniques de SFM à partir d'images 2D, cette thèse utilise les techniques de calcul de pose relative.

Les primitives mises en correspondance entre deux images permettent de définir une relation géométrique entre les repères des deux caméras. Cette relation géométrique est modélisée par la matrice de transformation projective définie par la contrainte de géométrie épipolaire. Quand la caméra n'est pas calibrée (ses paramètres internes sont inconnus), la relation géométrique est obtenue par la matrice fondamentale F (matrice 3x3). Quand la caméra est déjà calibrée (ses paramètres internes sont connus), la relation géométrique est obtenue par la matrice essentielle E (matrice 3x3). Les matrices fondamentale et essentielle définissent la relation entre deux points 2D images d'un même point 3D.

Pour estimer la pose relative entre deux prises de vue avec des techniques 2D-2D, plusieurs algorithmes se basent sur les correspondances pour déterminer la matrice définissant la relation géométrique entre les images.

Lorsque les caméras ne sont pas calibrées, au moins 8 points sont nécessaires pour estimer la pose relative entre deux images [2]. Beaucoup de recherches au sein de la communauté de vision par ordinateur ont été menées pour réduire le degré de liberté de l'estimation. Il est maintenant bien connu que, si les caméras sont calibrées, seulement 5 points mis en correspondances peuvent être suffisants. À partir de 5 points mis en correspondance, l'algorithme de Nister [26] permet de calculer la matrice essentielle pour ensuite la décomposer et en extraire la pose. Pour des caméras non calibrées, la matrice fondamentale peut être calculée à partir de 7 points ou plus. La réduction du nombre de points peut être très intéressant afin de réduire le temps de calcul et d'augmenter la robustesse.

Certains algorithmes ajoutent des contraintes sur la géométrie de la scène ou sur le déplacement ou utilisent des capteurs supplémentaires pour réduire les paramètres à estimer et ainsi réduire le nombre de points nécessaires pour l'estimation. Par exemple, si tous les points 3D appartiennent à un plan, le nombre de points est réduit à 4 points [2]. D'autre part, si le déplacement du véhicule (du système de caméras) est planaire, il suffit d'avoir 2 points mis en correspondance pour estimer le mouvement [27]. Si le déplacement d'un mobile est modélisé avec des contraintes non-holonomes (mouvement circulaire), un seul point est suffisant [28]. Pour le cas où le déplacement des caméras est considéré appartenant à un plan perpendiculaire à la gravité, Troiani et al. [29] ont également montré qu'un seul point est suffisant.

Le nombre de points nécessaires pour l'estimation d'une pose relative peut également être réduit si quelques informations sur la rotation relative entre les deux poses sont fournies par un capteur additionnel. Li et al. [30] utilisent un IMU et réduisent le nombre de points à 4 points, même si l'étalonnage extrinsèque entre l'IMU et la caméra n'est pas connu.

De même, il existe d'autres algorithmes récemment proposés qui permettent d'estimer la pose relative entre deux caméras en connaissant une direction commune. Si les angles de roulis et de tangage de la caméra sont connues, il suffit d'avoir trois points pour récupérer l'angle de lacet et la translation [31][32][33].

Le calcul de pose possède un deuxième intérêt majeur autre que l'estimation de la relation géométrique entre deux points de vue. En effet, le calcul de la matrice fondamentale ou de la matrice essentielle permet également la gestion des valeurs aberrantes des appariements (les "outliers") quand une méthode robuste, de type RANSAC (RANdom SAmple Consensus [34]) ou M-estimateur, est employée. Cela permet d'éliminer la plupart des mauvaises correspondances 2D-2D.

1.1.3.1 Matrice Fondamentale

Définition La relation géométrique entre un ensemble de primitives mises en correspondance entre deux images I_1 et I_2 est algébriquement définie par la matrice fondamentale $\mathbf{F}$. La matrice fondamentale décrit la relation de la géométrie projective entre les points mis en correspondance d'une façon générale et fondamentale. $\mathbf{F}$ est une matrice carrée d'ordre 3 et de rang 2. La matrice F est déterminée à un facteur d'échelle près. Le terme "matrice fondamentale" a été établie par Quan-Tuan Luong dans sa thèse de doctorat [35] et la relation entre deux points mis en correspondance (l'équation 1.10) qui définit la matrice fondamentale a été publiée en 1992 par Faugeras [36] et Hartley [37]. Pour deux points 2D appariés, $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ (en coordonnées homogènes), images du même point 3D de la scène, la contrainte épipolaire s'écrit :

$$\mathbf{x}_2^{\intercal}\mathbf{F}\mathbf{x}_1 = 0 \tag{1.10}$$

Calcul de la matrice fondamentale

La matrice fondamentale peut être déterminée à partir d'au moins 7 points mis en correspondance. Il existe plusieurs algorithmes dans la littérature qui permettent d'estimer la matrice fondamentale. Les algorithmes les plus classiques et les plus utilisés sont l'algorithme des 7-points, l'algorithme normalisé des 8-points [38], 7-points LMEDS (Least Median of Squares) et 7-points RANSAC (Random Sample Consensus).

Dans le cas de l'algorithme de 7 points, le calcul de $\mathbf{F}$ peut retourner jusqu'à 3 solutions. La meilleure façon de calculer la matrice fondamentale est d'utiliser un algorithme de 8 points ou plus pour trouver une solution unique. L'algorithme de 8 points a été présenté par Christopher Longuet-Higgins en 1981 pour le calcul de la matrice essentielle. Cet

algorithme peut être également utilisé pour le calcul de la matrice fondamentale. Richard Hartley décrit l'algorithme normalisé des 8-points [38] de façon mieux adaptée pour le calcul de la matrice fondamentale.

Malgré la présence des correspondances erronées, LMEDS et RANSAC gardent de bonnes performances pour estimer la matrice fondamentale. En effet, ces algorithmes se basent sur la distance entre un point et la ligne épipolaire pour considérer l'appariement comme un bon ou mauvais appariement. Au-delà d'une distance maximale, les appariements sont rejetés. Le choix de cette valeur influe fortement la fiabilité des algorithmes pour le calcul de **F**. Quand la distance maximale est mal choisie, des appariements aberrants peuvent être considérés dans le calcul de la matrice fondamentale et faussent par la suite l'estimation de la pose.

L'estimation de la matrice fondamentale **F** par l'algorithme des 7 points RANSAC est présentée par l'algorithme 1. En effet, RANSAC permet la sélection aléatoire d'un ensemble de points pour en calculer la pose ce qui permet d'utiliser la redondance de points importante pour s'assurer d'avoir une solution unique et éviter les poses ambigües. La sélection aléatoire est reproduite jusqu'à ce que la pose obtenue ait de faibles erreurs avec le maximum d'appariements et une probabilité de sélection des points pertinents élevée. Pour n ($n \geq 7$) points mis en correspondance, on considère une probabilité p qu'au moins un des tirages ne contiendra aucun outlier et une distance maximale t.

Algorithm 1 Pseudo-code de l'algorithme d'estimation de la matrice fondamentale entre deux images : 7-points RANSAC [2]

Répéter pour N échantillons de m points (m=7) où $N = log(1-p)/log(1-(1-\epsilon)^m)$ avec $\epsilon = 1 -$ nombre d'inliers/nombre total des points :
1) Sélectionner aléatoirement 7 paires de points et calculer la matrice fondamentale. Il y aura une ou trois solutions.
2) Calculer de la distance d_i pour chaque paire de points appariés (pour les i solutions)
3) Calcul du nombre de points ("inliers") k_i considérés comme bons appariements (dont la distance $d_i < t$)
4) S'il existe trois solutions pour F, le nombre d'inliers est calculé pour chaque solution. La solution retenue est celle ayant le plus grand nombre d'inliers et le plus faible écart-type ($\sigma(d_i) < \sigma_{best}$).
Re-estimer F à partir de toutes les correspondances classées comme inliers en minimisant une fonction de coût.

1.1.3.2 Matrice essentielle

Définition La matrice essentielle définit la relation géométrique entre un ensemble de primitives mises en correspondance entre deux images I_1 et I_2. Il s'agit d'un cas particulier de la matrice fondamentale pour les points 2D normalisés. En effet, la matrice essentielle a été initialement présentée par Longuet-Higgins pour les caméras calibrées.

Ensuite pour généraliser la matrice essentielle, la matrice fondamentale a été présentée en éliminant l'hypothèse des caméras calibrées. La relation entre la matrice essentielle et la matrice fondamentale est donnée par l'équation 1.11. $\mathbf{K}_1$ et $\mathbf{K}_2$ sont les matrices de calibrage intrinsèque caractérisant les deux images I_1 et I_2.

$$\mathbf{E} = \mathbf{K}_2^{\mathsf{T}}\, \mathbf{F}\, \mathbf{K}_1 \tag{1.11}$$

Un point 2D normalisé (exprimé en coordonnées normalisées) $\widetilde{x}$ est calculé à partir du point 2D $\mathbf{x}$ multiplié par la matrice inverse de calibrage intrinsèque $\mathbf{K}^{-1}$ [2] (équation 1.12).

$$\begin{aligned} \widetilde{\mathbf{x}} &= \mathbf{K}^{-1}\mathbf{x} \\ \text{Alors, } \widetilde{\mathbf{x}} &= [\mathbf{R}|\mathbf{t}]\, \mathbf{X}. \end{aligned} \tag{1.12}$$

Par définition, de même que pour l'équation 1.10, la matrice essentielle $\mathbf{E}$ est une matrice homogène obtenue par la relation suivante :

$$\widehat{\mathbf{x}}_2^{\mathsf{T}}\, \mathbf{E}\, \widehat{\mathbf{x}}_1 = 0 \tag{1.13}$$

$\widehat{\mathbf{x}}_1$ et $\widehat{\mathbf{x}}_2$ sont les points appariés exprimés en coordonnées normalisées $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$.

calcul de la matrice essentielle

La matrice essentielle peut être obtenue soit par le calcul direct à partir des points mis en correspondance exprimés en coordonnées normalisées (équation 1.13), soit à partir de la matrice fondamentale (équation 1.11).

Pour estimer les 9 éléments de la matrice essentielle (3x3), 8 points suffisent pour une estimation à une échelle près. En effet, en termes de calcul, la matrice essentielle suppose que la calibration intrinsèque des caméras est connue, ce qui ajoute 3 degrés de liberté de plus à l'estimation de l'orientation relative qui a 5 degrés de liberté (d'où les 8 points). Pour réduire le degré de liberté de la matrice essentielle au strict minimum, le nombre de points peut être réduit. Dans cette optique, plusieurs algorithmes ont été présentés : l'algorithme des 8 points [39] [38], 7 points [2], 6 points [40] et 5 points [26]. L'algorithme de 5 points représente le nombre minimal possible.

Historiquement, l'algorithme des 5 points a été initialement présenté en 1913 par Kuppra. Plus récemment, en 2004, David Nister [26] a proposé un algorithme pour chercher les solutions possibles pour les poses relatives entre deux positions de caméras calibrées

à partir de 5 points mis en correspondance exprimés en coordonnées normalisées. L'algorithme de Nister a été repris plusieurs fois pour essayer de l'améliorer. En 2006, dans [41], Li et Hartley simplifient l'algorithme de 5-points en éliminant toutes les variables inconnues d'un seul coup au lieu de les éliminer une par une. Le principe est de calculer les éléments de la matrice essentielle pour ensuite la décomposer par une méthode classique de type SVD (Singular Value Decomposition) afin d'extraire la rotation et la translation entre les deux prises de vue.

1.1.3.3 calcul de la rotation et de la translation

Une fois calculée, la décomposition de la matrice essentielle fournit la pose relative entre deux images à un facteur d'échelle près. Pour deux images ayant comme matrices de projection $\mathbf{P}_1 = \mathbf{K}_1 [\mathbf{I}|0]$ et $\mathbf{P}_2 = \mathbf{K}_2 [\mathbf{R}|t]$, la matrice essentielle s'écrit sous la forme :

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R} \tag{1.14}$$

Supposons que la décomposition de E en valeurs singulière (SVD) est donnée par $\mathbf{E} = \mathbf{U}\,\mathbf{\Sigma}\,\mathbf{V}^\intercal$. $\mathbf{U}$ et $\mathbf{V}$ sont deux matrices orthogonales de taille 3x3 et $\mathbf{\Sigma}$ est une matrice diagonale donnée par :

$$\mathbf{\Sigma} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{1.15}$$

La rotation et la translation sont définies par :

$$\begin{aligned} [\mathbf{t}]_\times &= \mathbf{U}\,\mathbf{W}\,\mathbf{\Sigma}\,\mathbf{U}^\intercal \\ \mathbf{R} &= \mathbf{U}\,\mathbf{W}^{-1}\,\mathbf{V}^\intercal \end{aligned} \tag{1.16}$$

La matrice $\mathbf{W}$ est définie par :

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.17}$$

$[\mathbf{t}]_\times$ est définie par :

$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{1.18}$$

Une autre alternative pour $[\mathbf{t}]_\times$, $[\mathbf{t}]_\times = \mathbf{U}\,\mathbf{Z}\,\mathbf{U}^\intercal$ avec :

$$\mathbf{Z} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{1.19}$$

Jusqu'ici, la matrice essentielle estimée est $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$. Au moment de la décomposition de E, nous obtenons $\mathbf{R}$ et $\mathbf{t}$. D'un point de vue pratique il existe quatre possibilités pour $\mathbf{R}$ et $\mathbf{t}$. En effet, il existe deux directions opposées pour t et deux rotations différentes ($\mathbf{R} = \mathbf{U}\mathbf{W}\mathbf{V}^\intercal$ ou $\mathbf{R} = \mathbf{U}\mathbf{W}^\intercal\mathbf{V}^\intercal$). Au total, on obtient quatre possibilités. En plus de cela, il y a aussi une mise à l'échelle inconnue pour le sens t choisi. En outre, seulement une des quatre solutions doit être retenue dans la pratique. Pour faire le choix, il faut calculer les positions de points 3D. Trois solutions parmi les quatre possibles donnent une majorité de points 3D "derrière" au moins une des deux caméras et donc ne peuvent pas être retenues. Une seule solution parmi les quatre possibilités donne un nuage de points 3D "devant" les deux caméras. C'est donc la bonne solution. La figure 1.5 illustre les quatre solutions possibles.

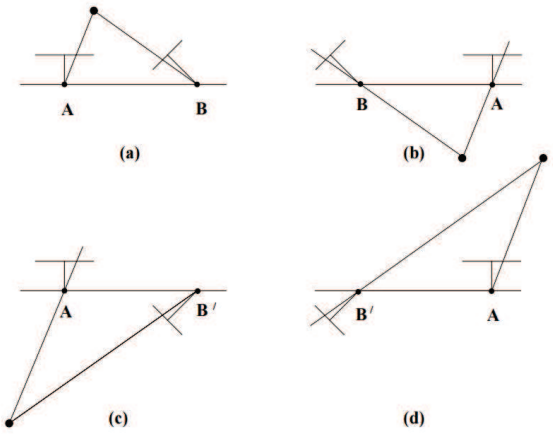


FIGURE 1.5: Les quatre solutions possibles pour $\mathbf{R}$ et $\mathbf{t}$. le point $\mathbf{X}$ est situé devant les deux caméras que dans un seul cas, c'est la bonne solution. L'illustration provient du papier [2]

1.1.4 Reconstruction 3D

Pour reconstruire une scène 3D, il faut qu'elle soit observée par au moins deux caméras ayant des champs de vue recouvrants pour pouvoir extraire les primitives communes. La position d'un point 3D de la scène est calculée à partir de ces observations (au moins deux observations dans deux prises de vue).

Pour la reconstruction de scènes complexes et pour la navigation autonome, il existe différentes méthodes de reconstruction 3D. En utilisant une seule caméra, la structure 3D de la scène peut être obtenue par les techniques de SFM. Cette méthode exploite le mouvement de la caméra dans la scène pour obtenir différents points de vue et ensuite reconstruit la structure 3D de la scène. À partir d'au moins deux caméras, la scène peut être reconstruite par les méthodes de stéréovision. La stéréovision consiste à retrouver la structure géométrique de la scène par le calcul de cartes de disparités ou par le calcul de cartes de profondeurs afin d'avoir une reconstruction 3D dense.

En vision par ordinateur, déterminer la position d'un point 3D est connue sous le nom de triangulation [42]. Ce processus nécessite la connaissance des coordonnées 2D des observations du point 3D dans les images (au moins deux observations) et des matrices de projection de chaque image. Le problème de triangulation est très important dans le SFM et le VO car l'estimation de la structure influe d'une façon directe sur l'estimation du mouvement et réciproquement.

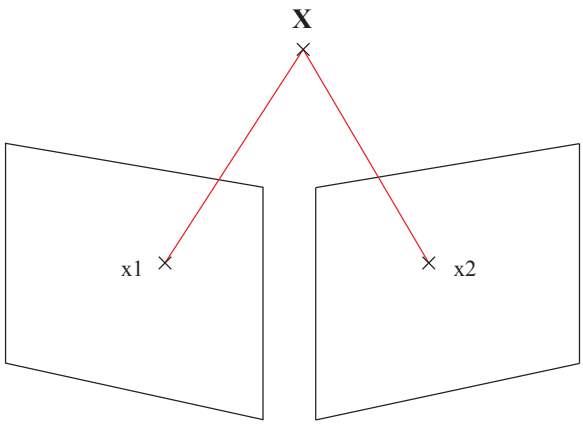


FIGURE 1.6: Triangulation : le cas idéal de la géométrie épipolaire. Calcul des coordonnées du point $\mathbf{X}$ à partir des appariements $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$. Le point 3D est issu de l'intersection des rayons rouges.

En théorie, l'obtention de la position d'un point 3D à partir de ses projections dans les images est triviale. Les paires de points 2D mis en correspondance doivent être les projections des points 3D dans les images. Cette reconstruction est possible lorsque la relation géométrique entre les caméras est connue et lorsque la projection du même point est mesurée dans les images. L'intersection des rayons passant par le centre de

chaque caméra et le point 2D représente la position du point 3D, figure 1.6. Les rayons peuvent être reconstruits à partir des points projetés et le point 3D calculé. Les deux lignes devraient se croiser parfaitement dans l'espace, mais à cause du bruit et d'autres erreurs, les deux rayons ne donnent pas la bonne solution.

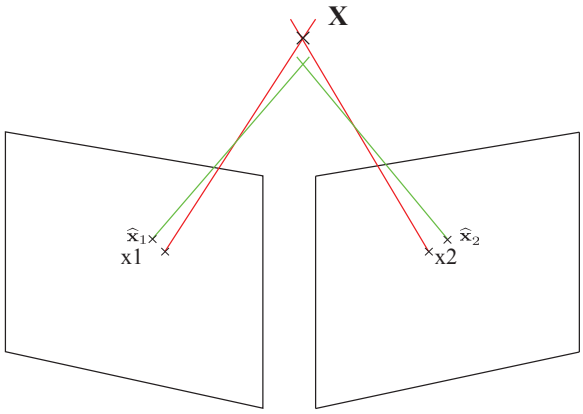


FIGURE 1.7: Triangulation : en pratique, les points d'image ne sont pas mesurées avec une haute précision. Les rayons en vert passant par les points 2D mesurées $\widehat{\mathbf{x}_1}$ et $\widehat{\mathbf{x}_2}$ ne se croisent pas dans le point $\mathbf{X}$ ou peuvent ne pas se croiser dans l'espace 3D

En pratique, les erreurs dans l'estimation de pose et dans le processus d'appariement affectent le calcul de la position 3D. En effet, plusieurs types de bruit peuvent engendrer des calculs erronés. Parfois, la distorsion de la lentille ne respecte pas le modèle sténopé, ce qui provoque des erreurs dans le calcul de pose des caméras. De plus, les points de l'image utilisés pour la triangulation sont souvent obtenus en utilisant divers types d'extracteurs, par exemple les détecteurs de coins ou des points d'intérêt. Ces algorithmes se basent en général sur le voisinage des pixels pour déterminer les points d'intérêt. L'imprécision de localisation dans le processus de détection et les erreurs d'appariement lors de la mise en correspondance des points provoquent des imprécisions dans les coordonnées mesurées dans l'image et dans le processus d'appariement. En général, quand les coordonnées des points 2D sont confus et/ou les matrices de poses des caméras sont imprécises, les rayons passant par les points 2D ne se croisent pas dans l'espace 3D (figure 1.7). Ce problème devient plus critique dans le cas de la géométrie projective et affine pour lesquels il n'y a pas de connaissance métrique sur la scène [42].

Le problème est de trouver les positions optimales des points 3D. Dans la littérature, il existe plusieurs algorithmes pour calculer la structure 3D optimale. La plupart des méthodes définissent une mesure d'erreurs dépendant d'une estimation des points 3D. Les méthodes les plus connues sont la méthode du point milieu [43], la transformation linéaire directe (Direct linear transformation DLT), la triangulation optimale, le "gold standard algorithm" etc. La méthode "gold standard algorithm" [2] permet d'estimer les points 3D et la matrice fondamentale simultanément. Une autre possibilité pour résoudre

le problème de triangulation est d'obtenir l'estimateur du maximum de vraisemblance optimale pour les points 3D.

La méthode du point milieu [43] cherche à calculer le barycentre pour deux points appartenant aux deux rayons passant du centre de la caméra et des points 2D telle que la distance entre ces deux points soit minimale, figure 1.8. Cette méthode ne donne pas toujours des résultats optimaux car l'algorithme utilise plusieurs approximations (les angles des rayons passant par les points images et le point 3D estimé ne sont pas précisément égaux). Cette méthode est applicable dans le cas d'une reconstruction affine et ne marche pas dans le cas d'une reconstruction projective.

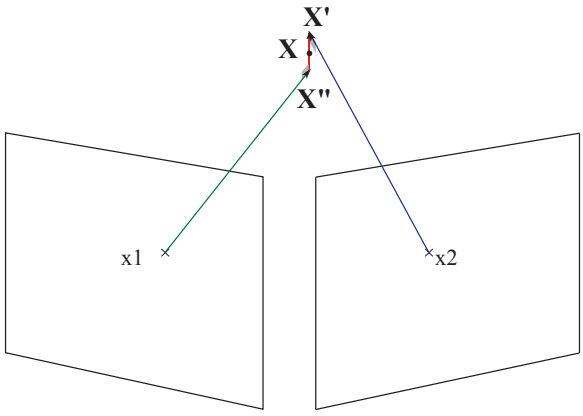


FIGURE 1.8: Triangulation par la méthode du point milieu

La triangulation devient un problème plus critique pour les reconstructions affines et projectives quand aucune connaissance (information) métrique sur l'objet n'est disponible. Une méthode simple de triangulation linéaire a été proposée par Hartley et Zisserman dans leur livre "Multiple View Geometry" [2]. Il existe deux variantes pour la triangulation linéaire : la triangulation linéaire résolue avec la méthode de moindres carrées et la triangulation linéaire résolue avec la méthode de résolution par les vecteurs propres. Comme l'a expliqué Hartley dans [42], avec la méthode des moindres carrées, la triangulation linéaire est invariante à la reconstruction affine. Par contre, en utilisant la résolution par les vecteurs propres, la triangulation linéaire n'est pas invariante à la reconstruction affine [42]. De plus, l'inconvénient de cette méthode, est que pour une reconstruction projective, quelque soit la méthode de résolution, les points reconstruits à l'infini peuvent être erronés. Donc, cette méthode n'est pas une bonne solution pour la reconstruction projective parce qu'elle n'est pas invariante à la reconstruction projective.

Dans [42], Hartley et al. présentent une méthode de triangulation linéaire itérative en changeant le poids des équations du système d'équations linéaires d'une façon adaptée

aux erreurs des coordonnées des points 2D dans les images. Comme pour la triangulation linéaire, il existe deux variantes pour la triangulation linéaire itérative : la triangulation linéaire itérative résolue avec la méthode de moindres carrées et la triangulation linéaire itérative résolue par les vecteurs propres. L'avantage de cette méthode est qu'elle est facile à implémenter en l'adaptant à la triangulation linéaire. En outre, le nombre d'itérations pour que l'algorithme converge est simple à déterminer. L'algorithme s'arrête quand le changement dans les poids des équations devient faible. Hartley et al. montrent que les méthodes itératives sont considérablement plus robustes et précises que les méthodes linéaires non-itératives. La triangulation linéaire itérative est considérée à peu prés invariante à la reconstruction projective et donc une bonne méthode de triangulation.

Hartley et al. proposent également une méthode polynomiale invariante aux transformations projectives de l'espace [42]. Cette méthode suppose que les coordonnées 2D subissent un bruit gaussien. La triangulation est ensuite faite en utilisant la méthode des moindres carrés. Une solution non-itérative est trouvée de façon à chercher un minimum global. Cette méthode donne une solution globale d'une façon optimale pour les reconstructions affines et projectives. Ses performances dépassent les méthodes linéaires, les méthodes linéaires itératives et la méthode du point-milieu mais elle est très couteuse en terme de temps de calcul et donc non adaptée aux applications temps réel.

Pour les travaux dans cette thèse, nous avons utilisé l'algorithme de triangulation linéaire itérative. Cet algorithme de reconstruction 3D éparse est le plus utilisé en vision par ordinateur et plus spécifiquement dans l'odométrie visuelle et le SFM. Dans la suite, nous détaillons l'algorithme de triangulation linéaire itérative comme il a été proposé par Hartley et al [42].

Triangulation linéaire

Dans cette section, nous supposons qu'un point 3D $\mathbf{X}$ appartenant à $\mathbb{R}^3$ est observé dans deux images I_1 et I_2. Pour déterminer les coordonnées du point $\mathbf{X}$, nous supposons que les coordonnées de ses observations appariées $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ et les matrices de projections $\mathbf{P}_1$ et $\mathbf{P}_2$ des deux caméras sont connues. Les matrices de projection $\mathbf{P}_1$ et $\mathbf{P}_2$ peuvent être calculées à partir des appariements $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ en passant par le calcul de la matrice fondamentale. Ce processus sera expliqué dans la section suivante, le calcul de pose.

On note la méthode de triangulation pour calculer les coordonnées du point $\mathbf{X}$ à partir de ses observations $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ et les matrices de projection $\mathbf{P}_1$ et $\mathbf{P}_2$ par τ, equation 1.20.

$$\mathbf{X}_i = \tau(\mathbf{x}_{1i}, \mathbf{x}_{2i}, \mathbf{P}_1, \mathbf{P}_2) \tag{1.20}$$

On suppose que $\mathbf{x}_1 = \mathbf{P}_1\mathbf{X}$, on écrit en coordonnées homogène $\mathbf{x}_1 = w_1(x_1, y_1, 1)^\intercal$, avec (x_1, y_1) les coordonnées du point $\mathbf{x}_1$ et w_1 est le facteur d'échelle à chercher. On note $\mathbf{P}_1^\intercal$ la i ème ligne de la matrice $\mathbf{P}$. L'équation précédente s'écrit donc comme suit :

$$\begin{aligned} w_1 x_1 &= \mathbf{P}_{11}^\intercal \mathbf{X}, \\ w_1 y_1 &= \mathbf{P}_{12}^\intercal \mathbf{X}, \\ w_1 &= \mathbf{P}_{13}^\intercal \mathbf{X} \end{aligned} \tag{1.21}$$

Éliminant w_1 en utilisant la troisième équation, on obtient :

$$\begin{aligned} x_1 \mathbf{P}_{13}^\intercal \mathbf{X} &= \mathbf{P}_{11}^\intercal \mathbf{X} \\ y_1 \mathbf{P}_{13}^\intercal \mathbf{X} &= \mathbf{P}_{12}^\intercal \mathbf{X} \end{aligned} \tag{1.22}$$

A partir de deux images, on obtient quatre équations linéaires qui peuvent être écrites sous la forme de $\mathbf{A}\mathbf{x} = 0$. Ces équations servent à déterminer $\mathbf{x}$ à un facteur d'échelle près. Avec des données bruitées, les équations n'admettent pas une solution précise. Pour avoir une solution optimale de $\mathbf{x}$, il faut résoudre le système $\mathbf{A}\mathbf{x} = 0$ par une méthode de résolution linéaire, comme la méthode de moindres carrées ou par les vecteurs propres.

Triangulation linéaire itérative

La valeur de minimisation des équations du système $\mathbf{A}\mathbf{x} = 0$ (équation 1.22) n'a pas un sens géométrique. L'idée de la méthode linéaire itérative est de changer le poids w des équations 1.22 jusqu'à converger vers la bonne solution afin que les équations pondérées correspondent aux erreurs dans les coordonnées dans les images.

En effet, $\mathbf{X}$ ne satisfait pas l'équation 1.22. Il existe une erreur, cette erreur correspond à ϵ. Pour la première equation du système des équations 1.22, ϵ est définie par :

$$\epsilon = x_1 \mathbf{P}_{13}^\intercal \mathbf{X} - \mathbf{P}_{11}^\intercal \mathbf{X} \tag{1.23}$$

Cette erreur représente l'erreur à minimiser pour retrouver le point $\mathbf{X}$. En plus de cette erreur, il existe une erreur entre le point observé et le point projeté dans l'image. En effet, $\mathbf{x}_1$ représente le point 2D détecté dans l'image alors que la projection de $\mathbf{X}$ ne correspond pas exactement à $\mathbf{x}_1$. Il existe une différence entre les coordonnées du point

2D détecté et les coordonnées du point 2D projeté à partir du point 3D. Cette erreur est appelée l'erreur de reprojection. La différence entre le point observé et le point projeté est donnée par : $(\mathbf{P}_{11}^{\intercal}\mathbf{X})/(\mathbf{P}_{13}^{\intercal}\mathbf{X})$.

L'erreur à minimiser sera donc :

$$\epsilon' = \epsilon/(\mathbf{P}_{13}^{\intercal}\mathbf{X}) \tag{1.24}$$

Remplaçons ϵ par sa valeur de l'équation 1.23, ϵ' devient donc :

$$\begin{aligned} \epsilon' &= (x_1\mathbf{P}_{13}^{\intercal}\mathbf{X} - \mathbf{P}_{11}^{\intercal}\mathbf{X})/(\mathbf{P}_{13}^{\intercal}\mathbf{X}) \\ \epsilon' &= x_1 - (\mathbf{P}_{11}^{\intercal}\mathbf{X})/(\mathbf{P}_{13}^{\intercal}\mathbf{X}) \end{aligned} \tag{1.25}$$

Les équations de de 1.22 sont pondérées par le facteur $1/w_1$ avec $w_1 = \mathbf{P}_{13}^{\intercal}\mathbf{X}$. L'erreur résultante représente précisément l'erreur à minimiser. De même pour la deuxième equation du système d'équations de 1.22, l'erreur est définie de la même façon avec la même pondération $1/w_1$. Pour la deuxième image (le point $\mathbf{x}_2$), l'erreur est également définie de la même manière que pour la première image (le point $\mathbf{x}_1$) avec une pondération $1/w_2$ avec $w_2 = \mathbf{P}_{23}^{\intercal}\mathbf{X}$.

Les poids w_1 et w_2 dépendent de $\mathbf{X}$ et $\mathbf{X}$ n'est connu que après la résolution du système. Alors, pour retrouver $\mathbf{X}$, il faut initialiser les poids : $w_{10} = w_{20} = 1$ et résoudre le système d'équations pour trouver une première estimation du point 3D : $\mathbf{X}_0$. $\mathbf{X}_0$ est la solution de la méthode de triangulation linéaire (résolue par la méthode de moindre carrée ou par les vecteurs propres). Cette méthode est alors utilisée pour trouver la solution initiale $\mathbf{X}_0$. Avec cette première estimation de $\mathbf{X}$, $\mathbf{X}_0$, les poids w_1 et w_2 peuvent être recalculés ($w_{11} = \mathbf{P}_{13}^{\intercal}\mathbf{X}_0$ et $w_{12} = \mathbf{P}_{23}^{\intercal}\mathbf{X}_0$).

Ce processus est répété : à chaque itération, les équations de 1.22 sont multipliées par $1/w_{1i}$ avec $w_{1i} = \mathbf{P}_{13}^{\intercal}\mathbf{X}_{i-1}$. De même pour la deuxième image, les équations sont multipliées par $1/w_{2i}$ avec $w_{2i} = \mathbf{P}_{23}^{\intercal}\mathbf{X}_{i-1}$. Quand l'évolution des poids devient faible, le processus s'arrête et la solution trouvée représente le point $\mathbf{X}$.

Cette méthode converge simplement vers une solution optimale. Néanmoins, dans quelques circonstances instables, cette méthode ne parvient pas à converger. Les points proches des épipoles posent souvent un problème pour la triangulation linéaire itérative (peut-être pour 5% du temps). Pour de telles situations, il faut appliquer une solution de repli pour parvenir à la convergence. Hartley et al. ont utilisé la méthode du polynôme optimal si la convergence n'est pas atteinte au bout de 10 itérations.

Nous avons appliqué cette méthode pour le calcul de la structure 3D à partir des points 2D appariés entre les images non synchronisées.

1.2 Estimation du mouvement et de la structure : SFM & VO

1.2.1 SFM & SLAM visuel

Il est important de rappeler que pour effectuer la reconstruction 3D d'un environnement dynamique autour d'un véhicule en circulation avec un système asynchrone, il faut commencer par l'estimation du mouvement du véhicule. Cela revient à estimer le mouvement des caméras (ego-motion). La problématique de cette thèse se rapproche de plusieurs techniques en vision par ordinateur qui ont fait l'objet de large recherche telles que le SLAM (simultaneous localization and mapping) et le SFM. Comme le système étudié est basé sur la vision, nous nous intéressons plus particulièrement aux problèmes du SLAM visuel (visual simultaneous localization and mapping (V-SLAM)).

Le terme SLAM visuel est un terme utilisé par la communauté robotique et désigne un problème très proche du problème de SFM. Le terme "Structure from Motion" (SFM) est un terme plus utilisé dans la communauté de la vision par ordinateur. La différence majeure réside dans l'existence d'une carte de l'environnement pour le SLAM visuel puisque l'objectif de cette technique est d'effectuer la localisation et la cartographie simultanément. D'un point de vue technique, les techniques de SLAM sont effectuées toujours de manière incrémentale en utilisant des méthodes de prédiction (comme le filtre de Kalman, le filtre à particules etc.) alors que les techniques de SFM nécessitent parfois de disposer de l'ensemble des images pour effectuer certains traitements utilisant des méthodes géométriques comme par exemple l'ajustement de faisceaux global. Les techniques utilisées pour le SFM temps-réel se rapprochent du SLAM en procédant de manière incrémentale mais sans utiliser les techniques de prédiction et de localisation par rapport à des amers.

Entre le SFM et le SLAM visuel, nous avons choisi le SFM pour résoudre le problème de l'estimation de mouvement d'un véhicule et le problème de détection d'obstacle autour de véhicule avec des caméras non synchronisées. Le système développé pendant cette thèse est dédié aux applications de véhicule intelligent et donc pour un fonctionnement temps réel. Plus particulièrement, l'odométrie visuelle est l'algorithme le plus adapté aux applications temps réel afin d'estimer le mouvement du système de caméras et de réaliser une carte représentant la structure de la scène autour du véhicule.

1.2.2 SFM & VO

Les expériences de SFM temps-réel date de 1997 avec les travaux de Tomasi et al [44]. L'odométrie visuelle, un cas particulier de l'SFM, est devenu très populaire dans la communauté robotique et vision par ordinateur depuis 2004 avec l'article "Visual Odometry" présenté par Nister [23]. En 2006, pour les applications de navigation de véhicules terrestres, Nister et al. [45] ont présenté un algorithme d'odométrie visuelle temps réel permettant d'estimer le mouvement d'une paire stéréo ou un système mono-caméra mobile à partir des données visuelles (vidéo).

L'avantage de l'odométrie visuelle par rapport à l'odométrie des roues est que l'odométrie des roues peut être affectée par le patinage des roues en terrain inégal ou d'autres conditions défavorables surtout dans les virages [1]. Il a été démontré que l'estimation de la trajectoire effectuée par des techniques d'odométrie visuelle est plus précise par rapport à l'odométrie de roue.

L'odométrie visuelle est très performante. C'est une technique importante qui peut compléter ou remplacer les autres techniques de navigation tels que l'odométrie des roues, les mesures des centrales inertielles (inertial measurement units : IMUs), l'odométrie à partir des mesures laser et le GPS (global positioning system) pour les robots et les véhicules autonomes (complémentarité). De plus, pour les applications dont l'environnement est non couvert par le signal GPS, comme l'exploration de l'espace, la navigation sous-marine et la navigation des drones, l'odométrie visuelle a une très grande importance.

L'estimation de mouvement par les techniques d'odométrie visuelle peut être obtenu soit par des approches 2D-2D, soit par des approches 3D-3D ou par des approches 3D-2D. Les techniques 2D-2D consistent à estimer le mouvement à partir des primitives 2D mises en correspondance entre les images. Les techniques 3D-3D consistent à estimer le mouvement à partir des primitives 3D mises en correspondance en déterminant la transformation d'alignement entre deux ensembles de primitives 3D. Les techniques 3D-2D consistent à estimer le mouvement à partir de la structure 3D et des primitives 2D des images.

Comme l'a souligné Nister et al. [23], l'estimation à partir des techniques 3D-2D et l'estimation à partir des techniques 2D-2D surpassent l'estimation à partir des techniques 3D-3D. L'estimation à partir des techniques 3D-2D est plus précise que de l'estimation à partir des techniques 3D-3D car elle minimise les erreurs de reprojection 2D dans les images au lieu de la minimisation des erreurs entre les positions 3D pour les techniques 3D-3D. Le principal défi de cette méthode consiste à maintenir un ensemble cohérent et

précis des points 3D calculés par triangulation et de trouver des correspondances 3D-2D pour au moins avec 3 images adjacentes [1].

Pour l'odométrie à partir d'une seule caméra, les approches 2D-2D sont préférables aux approches 3D-2D car l'estimation du mouvement n'inclue pas la triangulation des points 3D. Toutefois, en pratique, les approches 3D-2D sont utilisées plus souvent que les approches 2D-2D car l'association des données est plus rapide [1]. En effet, pour les approches 2D-2D, le mouvement est obtenu à partir de la matrice essentielle estimée à partir de 5 points alors que pour les approches 3D-2D le mouvement est obtenu à partir de 3 points (P3P). Ce nombre inférieur de points entraine une estimation de mouvement plus rapide.

En résumé, la mise en correspondance 3D-3D n'est possible que dans le cas de la vision stéréo. Comme les caméras du système étudié dans cette thèse ne sont pas synchronisées, les techniques de stéréovision ne peuvent pas être appliquées directement. Entre les approches 3D-2D et les approches 2D-2D, nous avons choisi de travailler avec les approches 2D-2D pour obtenir la meilleure estimation de mouvement possible. En effet, comme les images ne sont pas synchronisées, le système peut se rapprocher du cas monoculaire. Les prises de vues des différentes caméras peuvent être schématisées par des prises de vue d'une seule caméras mais qui effectue un grand changement de point de vue entre deux images consécutives.

L'algorithme de l'estimation du mouvement à partir des approches 2D-2D est présenté par l'algorithme 2. À chaque nouvelle image capturée, les primitives 2D sont extraites et appariées aux primitives de l'image précédente. Ensuite, la matrice essentielle est calculée puis décomposée en rotation et translation. Puis, l'échelle absolue entre les deux prises de vue est calculée pour concaténer enfin, la transformation résultante (rotation, translation et facteur d'échelle) avec les précédentes poses de la caméra et ainsi de suite. Pour obtenir l'échelle absolue, elle doit être initialisée puis maintenue durant l'estimation. L'échelle absolue peut être maintenue et calculée de deux manière : soit à partir des points 3D triangulés en calculant la distance entre une paire de points de l'image courante et la même paire de points de l'image précédente soit à partir de la contrainte trifocale entre les points appariés entre trois prises de vue.

Pour les techniques de SFM et d'odométrie visuelle, une estimation itérative à partir des points d'intérêt provoque toujours des erreurs qui seront accumulées au fil du temps et conduisent à des dérives que même les algorithmes robustes ne peuvent pas éviter. Pour remédier à ces erreurs dans le but d'optimiser la géométrie de la scène et les positions des caméras, un processus d'optimisation tel que l'ajustement de faisceaux ou l'optimisation du graphe de poses est généralement appliqué.

Algorithm 2 Pseudo-code de l'algorithme d'estimation du mouvement à partir des approches 2D-2D

1) Acquisition d'une nouvelle image I_k
2) Extraction et mise en correspondances des points 2D entre l'image I_k et l'image I_{k-1}
3) Calcul de la matrice essentielle E à partir des paires de points appariés
4) Décomposition de E en R_k et t_k pour former T_k
5) calcul du facteur d'échelle et mise de t_k à l'échelle
6) Concaténer les transformations pour obtenir la position $k^{ème}$ par rapport au repère de référence
7) Répéter à partir de 1).

1.2.2.1 Optimisation de graphe de poses

L'odométrie visuelle permet le calcul des poses des caméras par la concaténation des transformations relatives. Pour améliorer ces poses, les transformations entre ces dernières peuvent être utilisées comme contrainte supplémentaire dans une optimisation de graphe de poses. Il s'agit d'un graphe où les poses des caméras représentent les nœuds et les transformations entre les caméras représentent les arêtes du graphe. Ces nœuds correspondent à des images clés que l'on garde en mémoire pour la fermeture de boucles ou la localisation.

Le graphe de poses a été initialement proposé en 1997 par Lu and Milios [46]. Cette technique est très utilisée par la communauté robotique [47][48][49][50]. Le principe est d'ajouter un nouveau nœud au graphe chaque fois que la pose du véhicule (robot) subit des changements importants : le robot tourne plus que 0.5 radian ou se déplace plus que 0.5 mètre [3].

Plusieurs travaux ont présenté les techniques de l'optimisation de graphe de poses [47][48][49][50]. Cette technique est récemment étudiée pour les problèmes de l'odométrie et le SLAM. L'optimisation de graphe de poses nécessite deux contraintes : des relations entre les poses (le modèle du mouvement) et les fermetures de boucles. La figure 1.9 présente un graphe de poses pour les problèmes de SLAM [3]. Les nœuds du graphe correspondent aux poses du robot et les arcs (e_{ij}) représentent les mesures de l'odométrie entre une pose et la suivante. Les autres arcs représentent les contraintes spatiales provenant de plusieurs observations de la même scène. Le problème d'optimisation de graphe de poses est un problème difficile et non convexe.

1.2.2.2 Ajustement de faisceaux

L'ajustement de faisceaux, comme l'optimisation du graphe de poses, sert à optimiser les poses des caméras. De plus, l'AF sert à optimiser les points 3D simultanément. Pour les

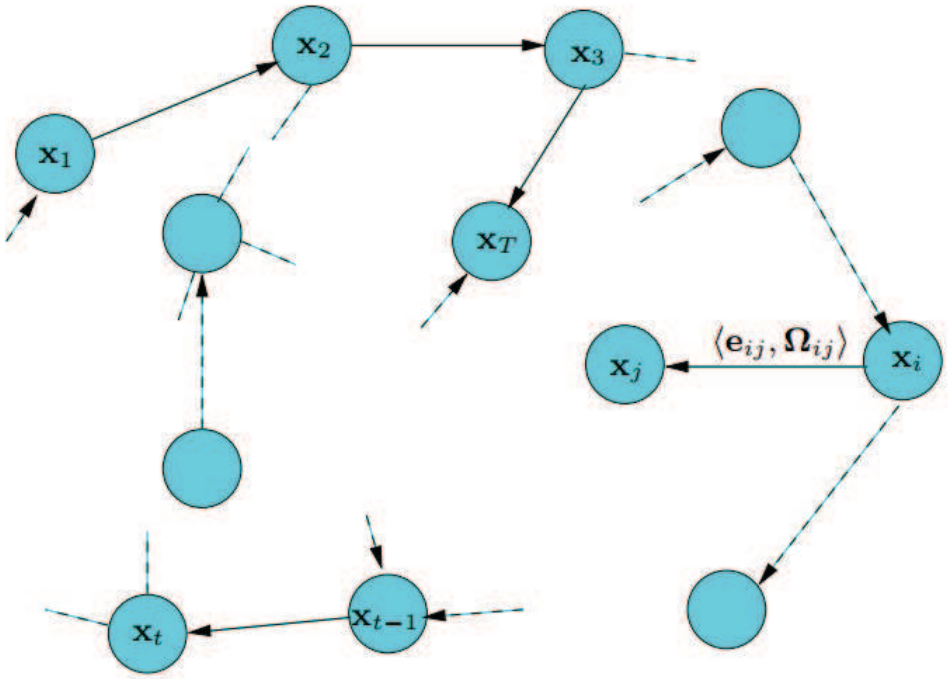


FIGURE 1.9: Une représentation de graphe de poses pour les problèmes de SLAM. L'illustration provient du papier [3]

applications d'odométrie visuelle, une fermeture de boucle suivie par un ajustement de faisceaux global est généralement appliquée [51]. La difficulté de l'AF global, c'est que le nombre des caméras et de points 3D devient très grand. Cette technique est très efficace pour les traitements hors ligne mais difficile à mettre en place dans des applications réelles. Cependant, il est possible d'utiliser un ajustement de faisceaux local sur une fenêtre glissante pour limiter les calculs [52][53][54].

L'ajustement de faisceaux (AF) peut être défini comme étant une méthode d'affinement à la fois des coordonnées 3D de la scène et des paramètres du mouvement relatif d'une/des caméras. L'ajustement de faisceaux est généralement utilisé comme la dernière étape pour les algorithmes de reconstruction 3D à partir des points (primitives) images. Il s'agit d'un problème d'optimisation pour la structure 3D, la pose de la caméra ou même les paramètres intrinsèques et de la distorsion radiale afin d'obtenir une reconstruction optimale [55]. En effet, l'AF affine la première estimation de la structure 3D et de la trajectoire des caméras pour obtenir une estimation optimale à l'aide d'un algorithme d'optimisation non linéaire comme l'algorithme de Levenberg-Marquardt. Il existe deux types d'AF : l'AF calibré (AF euclidien) et l'AF non-calibré (AF projectif). Dans l'AF calibré, les paramètres intrinsèques étant connus, il n'est donc pas nécessaire de les intégrer au processus d'optimisation.

L'ajustement de faisceaux, comme son nom l'indique, consiste à ajuster au mieux deux

ensembles de faisceaux : les faisceaux 3D et les faisceaux 2D. Les faisceaux 3D sont les rayons partant des points de la scène (les primitives 3D) et passant par le centre optique de la caméra. les faisceaux 2D sont les rayons partants du centre optique de la caméra et passant par les points images (la figure 1.10 [4]).

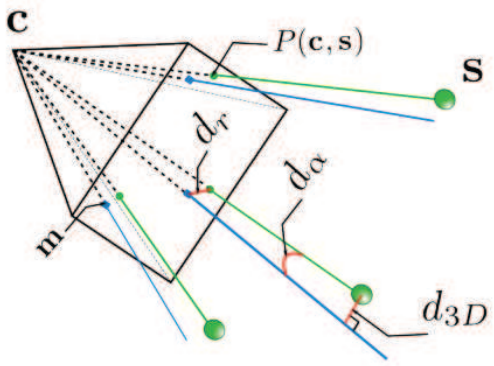


FIGURE 1.10: Le principe de l'ajustement de faisceaux. Les paramètres s, m, c et P sont respectivement un point 3D, un point 2D, une caméra et la fonction de projection. L'image provient du papier [4].

Il existe trois types de mesure de l'erreur entre les faisceaux de rayons 3D et 2D :

- l'erreur angulaire : mesurer l'angle (d_α) entre les deux les faisceaux 3D (en vert) et 2D (en bleu),
- l'erreur 2D : mesurer l'erreur de reprojection (d_r) grâce au modèle de la caméra,
- l'erreur 3D : mesurer l'erreur sur la distance euclidienne (d_{3D}) entre les faisceaux 3D (en vert) et 2D (en bleu) .

Une synthèse complète sur les méthodes et les théories de l'ajustement de faisceaux a été présentée dans le papier de Triggs [55].

Dans notre cas, nous nous intéresserons à minimiser l'erreur de reprojection entre les points images observés dans l'image (2D) et ceux prédits correspondant à la reprojection des point 3D dans l'image (par la fonction de projection initialement estimée). Ainsi, les points 3D et les poses des caméras seront conjointement optimisés. L'erreur de reprojection peut être exprimée dans ce cas par la distance euclidienne entre un point 2D observé (x) et le point prédit ($\widehat{x}$) (reprojection du point 3D par la fonctionnelle de projection estimée) :

$$\forall(\widehat{x}, x) \in I \times I,\ d_{2D} = d(\widehat{x}, x) = \|x - \widehat{x}\|. \tag{1.26}$$

Cette erreur de reprojection doit être minimisée pour toutes les caméras et les points 3D qui ont une observation 2D dans l'image de la caméra correspondante. Pour que le système soit contraint, il faut que chaque point 3D soit au minimum observé par deux caméras et que chaque caméra ait au moins trois observations de trois points 3D. En pratique, ces conditions sont largement satisfaites.

1.3 Vision omnidirectionnelle

L'avantage majeur d'un système multi-caméras est d'obtenir une vue d'ensemble autour du véhicule. Cela nous amène à discuter différentes façons d'obtenir une image omnidirectionnelle et les méthodes associées pour estimer le mouvement à l'échelle absolue.

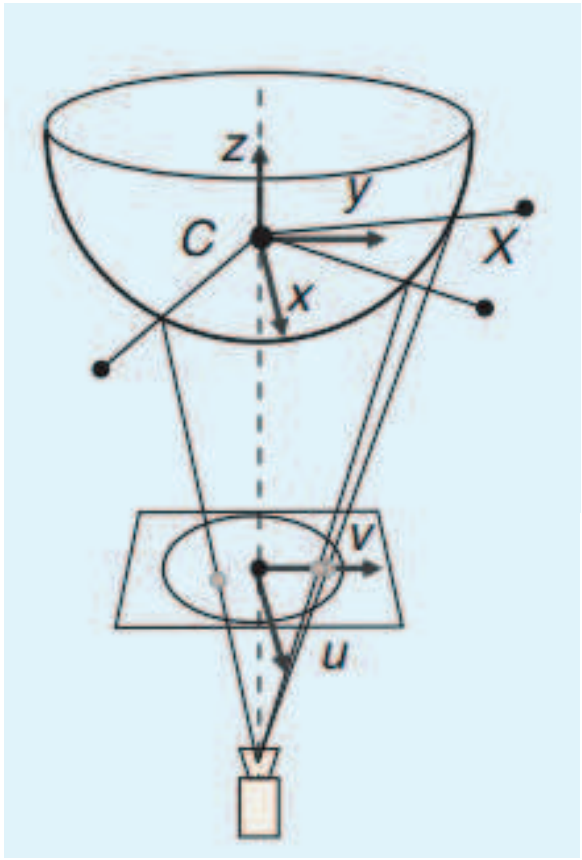
1.3.1 Système monoculaire

Pour avoir une vision omnidirectionnelle autour du véhicule, il est possible d'utiliser une seule caméra offrant une vue complète de l'environnement proche : caméra catadioptrique (figures 1.11 et 1.12), ladybug ou objectif très grand angle dit aussi objectif fish-eye (figure 1.11). Une caméra catadipotrique est composée d'une caméra perspective et d'un miroir, son modèle de projection est illustré par la figure 1.11. Les caméras fish-eye permettent une vision proche de 180 degrés et la ladybug (composé par plusieurs caméras perspective) permet une vision proche de 360 degrés.

En robotique, les caméras omnidirectionnelles sont fréquemment utilisées pour l'odométrie visuelle et le SLAM visuel (la localisation et cartographie simultanées) [56], [57], [58], [6], [59], [5]. Pour l'estimation du mouvement à partir d'images omnidirectionnelles, ces applications sont réalisées par les techniques de flux optique ou par la mise en correspondance des primitives.

Pour les systèmes monoculaires, une connaissance a priori de la structure 3D est nécessaire comme initialisation afin de maintenir l'échelle absolue dans l'estimation de la séquence. En effet, une étape d'initialisation du facteur d'échelle est ensuite suivie du maintien du facteur d'échelle durant le mouvement.

Parmi plusieurs travaux sur les systèmes monoculaires offrants une vision omnidirectionnelle, dans [60], Fraundorfer et al. présentent une méthode d'estimation du mouvement à l'échelle à partir d'une seule caméra omnidirectionnelle (figure 1.12). Après l'estimation de la trajectoire du véhicule à l'échelle près, les auteurs présentent un ajustement de faisceaux contraint pour l'estimation de l'échelle. Pour calculer les poses relatives,



(a) Projection catadioptrique

(b) Ladybug5 360 degrés

(c) Nikon1V1, objectif fish-eye

(d) Canon 8–15mm, image prise à 8mm d'une BMW

FIGURE 1.11: Systèmes de vision omnidirectionnelle : Les illustrations (de a à d) proviennent respectivement du papier de [1], du site du constructeur PointGrey et de Wikipédia (c et d)

FIGURE 1.12: Un véhicule équipé par un système omnidirectionnel composé par un miroir hyperbolique (KAIDAN 360 One VR) et une caméra couleur (SONY XCD-SX910). L'image provient du papier [5]

les auteurs utilisent l'algorithme de 1-point RANSAC présenté dans [28]. Pour calculer l'échelle, Fraundorfer et al. proposent une paramétrisation d'un ajustement de faisceaux global pour résoudre tous les facteurs d'échelle à la fois. Au lieu d'optimiser tous les paramètres du mouvement et de la structure comme l'algorithme classique d'ajustement de faisceaux, ils optimisent seulement les facteurs d'échelle relatifs. La différence entre [60] et notre travail réside dans l'utilisation dans l'utilisation d'une caméra monoculaire alors que nous avons développé un système multi-caméras asynchrone. De plus, leur algorithme est basé sur l'hypothèse de mouvement circulaire alors que le notre est basé sur une hypothèse de mouvement rectiligne.

Les systèmes monoculaires offrants une vision omnidirectionnelle sont certes très utilisés mais présentent quelques limites. Les images obtenues avec un objectif très grand angle sont des images fortement distordues (figure 1.11) et donc ne sont pas très favorables aux approches 2D-2D (nombre de points d'intérêt limité). Pour les système catadioptrique, leur mise en place sur le toit d'un véhicule n'est pas judicieux pour un système grand public (figure 1.13). Quand la caméra est placé très proche du toit, le champ de vue est finalement restreint à cause du toit de la voiture lui-même (figure 1.12).

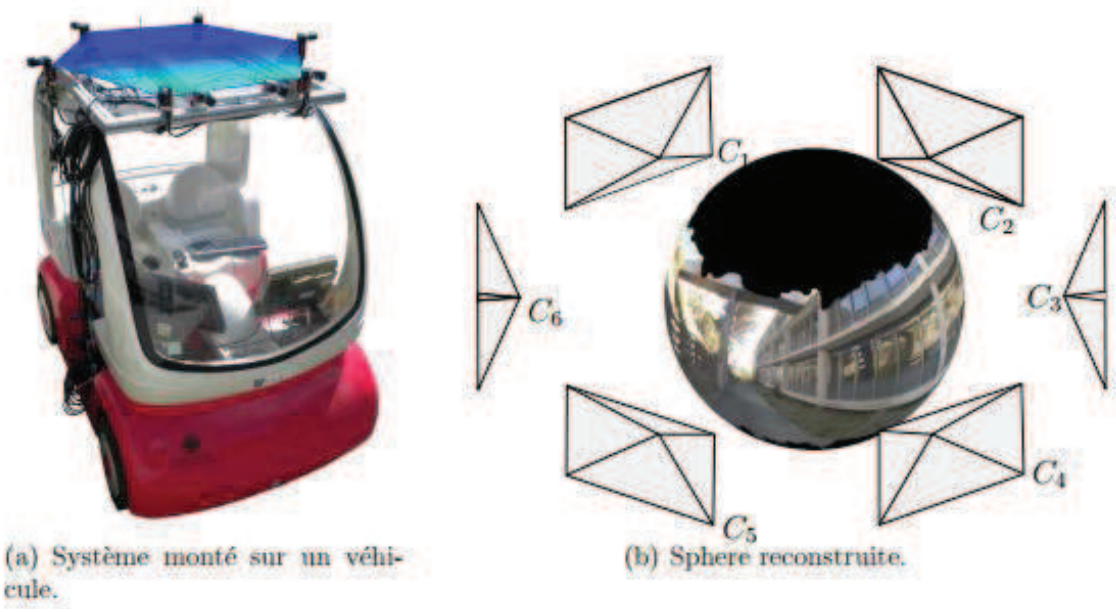
1.3.2 Système multi-caméras

Pour obtenir une vue à 360 degrés, une configuration multiple peut être faite par plusieurs caméras perspectives classiques montées en anneau. Plusieurs systèmes multi-caméras ont été développé pour les applications d'estimation de mouvement et pour le SLAM Visuel.

FIGURE 1.13: Une caméra Ladybug2 (PointGrey) montée sur le toit d'un véhicule. L'image provient du papier [6]

Par exemple, le système développé par Meilland [7] : six caméras grand angle placées sur un cercle. Dans cette configuration, les 360 degrés du champ de vision horizontal sont visibles par le système (figure 1.14).

Un autre exemple d'estimation de mouvement à partir d'un réseau multi-caméras offrant une vision omnidirectionnelle a été présenté dans [24]. Lee et al. utilisent un réseau de quatre caméras synchronisées dans l'estimation de mouvement d'un véhicule autonome. Le réseau a été modélisé par une caméra généralisée. L'idée est de modéliser un système de caméras par une caméra généralisée où une seule contrainte épipolaire est utilisée pour décrire le mouvement relatif de l'ensemble des caméras montées de façon rigide. La principale différence entre le modèle de caméra généralisée et le modèle sténopé est l'absence d'un centre de projection unique. Le mouvement relatif de la caméra généralisée est obtenu entre deux trames consécutives des caméras qui donnent un ensemble d'images synchronisées. En modélisant leur système par une caméra généralisée et le mouvement par la contrainte non-holonome, Lee et al. [24] estiment une matrice essentielle généralisée (de taille 6x6) en se basant sur la contrainte épipolaire généralisée à partir



(a) Système monté sur un véhicule.

(b) Sphere reconstruite.

FIGURE 1.14: Une configuration multiple à 360 degrés : six caméras grand angle placées sur un cercle. L'image provient de la thèse [7]

de 2-points.

Un exemple de système stéréoscopique synchrone offrant une vue autour d'un robot en circulation a été présenté dans [61]. Il s'agit d'un système composé de deux caméras grand angle offrant une vue de 150 degrés chacune. Pour s'affranchir de la contrainte de chevauchement des champs de vision d'un système stéréoscopique, Kazik et al. ont développé un algorithme d'odométrie visuelle en utilisant des caméras synchronisées dans des environnements intérieurs. Chaque caméra a été traitée comme un système monoculaire pour estimer son mouvement à l'échelle près. Une solution linéaire a été utilisée par la suite afin de fusionner les facteurs d'échelle provenant des deux estimations en imposant la transformation statique connue entre les deux capteurs pour retrouver enfin l'échelle métrique absolue.

1.4 Vision asynchrone

Pour une configuration stéréo ou multiple, l'estimation du mouvement et de la structure se base sur la géométrie épipolaire pour déduire l'échelle métrique. De ce fait, l'estimation de mouvement à l'échelle n'est possible que si les caméras sont synchronisées et par conséquent, la scène est prise sous différents angles de vue au même instant. La synchronisation est aussi nécessaire pour étudier les scènes dynamiques avec des traitements à

raisonnement géométrique. En effet, un point de la scène observé par deux caméras non synchronisées est vu de différents points de vue et non pas au même instant t.

Différentes approches ont été présentées pour synchroniser des systèmes de caméras non synchronisées où des flux vidéos non synchronisés. L'accent a été mis sur la recherche de la différence temporelle entre les différents flux de données vidéo [62][63] [64]. 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 la différence de temps. 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 et calculées. Ces approches utilisent une variété de repères visuels et interpolation linéaire pour trouver la différence de temps et pouvoir synchroniser les images des flux vidéo.

Pour les véhicules intelligents, les systèmes de vision embarqués sont généralement de faible coût pour être commercialisés à grande échelle, comme dans [25]. La mise en place de la synchronisation est difficile quand la distance entre les caméra est large. La synchronisation est alors un inconvénient majeur pour les systèmes de vision embarqués sur un véhicule et commercialisés à grande échelle. En effet, la synchronisation impose l'ajout d'un circuit électronique et un câblage supplémentaire. De plus, un système synchrone utilise en général un matériel coûteux. En outre, l'acquisition d'images à partir d'un système asynchrone peut être facilement implémentée sans se baser sur la vitesse d'acquisition de la caméra la plus lente.

Le SFM est désormais un problème bien maîtrisé dans le cas d'un réseau multi-caméras synchronisées. Cependant jusqu'à présent, le SFM asynchrone a été rarement étudié sauf quelques exceptions dans le cadre de l'estimation de mouvement, [65] [66].

La méthode la plus marquante dans le cadre des techniques proches du SFM est la méthode présentée dans [8]. Une configuration multi-caméras non synchronisées a été utilisée avec des techniques basées sur le SLAM. Dans un environnement intérieur, la reconstruction 3D de la scène autour d'un robot est obtenue à l'aide de deux caméras non synchronisées en utilisant l'odométrie des roues. Cette méthode utilise trois images : deux provenant de la caméra à gauche au premier et au troisième instant et une image à partir de la caméra droite au deuxième instant. Les primitives des trois images sont interpolées afin de créer l'image gauche manquante au deuxième instant en supposant que la position de ces primitives change linéairement entre les images. La figure 1.15 illustre l'algorithme proposé par Svedman et al dans [8].

Cette méthode retrouve des caractéristiques 3D en tenant compte des mesures odométriques pour interpoler l'image manquante. En utilisant directement l'odométrie des roues, cette

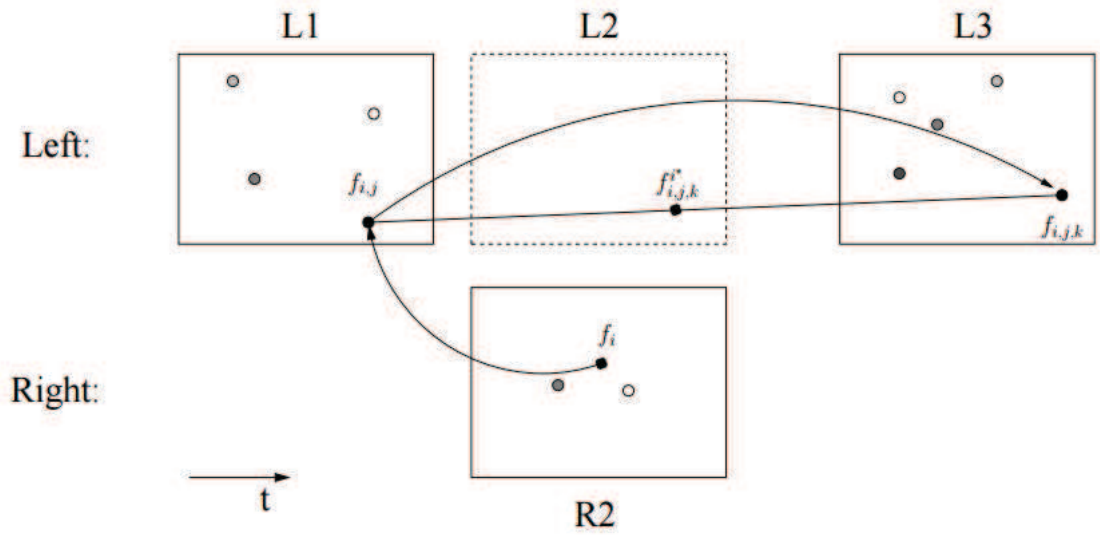


FIGURE 1.15: Simulation de l'image manquante (L2) par interpolations des points détectés dans les images non synchronisées (L1, R2 et L3). L'image provient du papier [8]

approche traite la reconstruction 3D pour un cas asynchrone mais pas des méthodes d'estimation de mouvement d'un système multi-caméras asynchrones.

1.5 Synthèse

Les constructeurs automobiles utilisent de plus en plus les systèmes de vision pour l'aide à la conduite. Les informations provenant des caméras embarquées sur un véhicule peuvent être exploitées pour la navigation, comme par exemple le système AVM (Around View Monitor) de Nissan. En effet il s'agit d'un système composé par quatre caméras et qui offrent une vue à 360 degrés. Les caméras sont positionnées au niveau de la calandre, du hayon et des deux rétroviseurs extérieurs. Les caméras du système AVM fonctionnent séparément et permettent au conducteur une visualisation des obstacles autour de la voiture et la position du véhicule par rapport à son environnement immédiat. Ce système est composé de caméras non synchronisées et il est opérationnel sur la Nissan Note, Nissan Qashqai et Nissan X-Trail.

Pour obtenir une vision omnidirectinnelle autour d'un véhicule, tous les travaux précédemment décrits sont capables d'estimer la structure et le mouvement à l'échelle. Néanmoins, la configuration des systèmes stéréoscopiques et multiples utilisent des caméras synchrones. Pour relâcher cette contrainte coûteuse, un système composé de dispositifs non synchronisés peut être utilisé. Il existe très peu de solutions envisagées et possibles. La configuration asynchrone nécessite un complément d'information pour effectuer l'estimation de mouvement. En effet, l'estimation du mouvement d'un réseau asynchrone peut

être possible en utilisant des approches 3D-3D : chaque caméra effectue une odométrie monoculaire et obtient la structure 3D puis des correspondances 3D-3D peuvent être effectuées. Le problème qui se pose pour cette solution est que les facteurs d'échelles relatifs sont différents. Aussi, cette approche n'est pas très appropriée pour les applications temps réel. Une autre solution est possible en utilisant des approches 3D-2D. Dans ce cas, l'estimation n'est possible que si la structure 3D est connue a priori, cette solution peut être envisagée pour la cartographie, mais pas pour un véhicule en circulation dans un environnement inconnu (sans aucune connaissance a priori).

Conclusion

Dans ce chapitre, les problèmatiques du SFM et de l'odométrie visuelle ont été introduites ainsi que les configurations possibles pour l'obtention d'une vue complète autour d'un véhicule en circulation ont été discutées.

L'odométrie visuelle a été présentée et l'algorithme de ce processus a été détaillé. En utilisant des techniques 2D-2D, le mouvement sera estimé à partir des primitives 2D mises en correspondance entre les images. Le chapitre suivant détaillera le processus de détection et de descriptions des primitives 2D.

Chapitre 2

Extraction des points d'intérêt : évaluation

Introduction

Ce chapitre porte sur le processus de mise en correspondance des primitives entre les images et compare les descripteurs locaux ainsi que leurs performances quand ils sont appliqués aux images non synchronisées. Cette partie de la thèse compare les détecteurs et les descripteurs de points d'intérêt les plus utilisés dans les applications d'odométrie visuelle.

Par définition, une primitive de l'image est un point 2D qui diffère de son environnement immédiat. Elle est généralement associée à un changement d'une propriété d'image ou de plusieurs propriétés en même temps [67]. Les propriétés couramment considérées comme pertinentes dans l'image sont l'intensité, la couleur et la texture. Les primitives locales peuvent être des points, mais aussi des contours ou des motifs d'image. En règle générale, certaines mesures sont prises à partir d'une région centrée sur une fonction locale et converties en descripteurs. Les primitives caractérisées par rapport à leur voisinages peuvent ensuite être utilisées pour diverses applications, notamment l'estimation de la relation géométrique entre deux images.

Le processus de mise en correspondance consiste à extraire les primitives (les points d'intérêt) dans deux images pour ensuite les apparier. Les points d'intérêt sont détectés et décrits dans chaque image puis les descripteurs obtenus pour les deux images sont appariés. Pour la détection de points d'intérêt, les points clés sont détectés par des algorithmes appelés des détecteurs locaux. Les détecteurs sont classés comme des détecteurs de contours, des détecteurs de « blobs », des détecteurs de régions et des détecteurs

basés sur la saillance [67]. Pour la description des points d'intérêt, les points clés détectés sont décrits localement en se basant sur leurs voisinages par des algorithmes appelés des descripteurs locaux. Les descripteurs locaux sont classés en deux catégories : les descripteurs vectoriels et les descripteurs binaires.

De nombreuses évaluations des détecteurs et descripteurs locaux ont été menées pour plusieurs applications différentes notamment la reconnaissance d'objets, le tracking, etc. Selon les applications pour lesquelles la détection et la description des points d'intérêt sont nécessaires, certaines invariances sont importantes. Comme nous utilisons des images asynchrones pour l'odométrie visuelle, plusieurs invariances sont requises. En effet, les images asynchrones sont affectées par le bruit, le flou, le changement de point de vue, l'éclairage, la rotation et les variations d'échelle, etc. Il est donc indispensable que les détecteurs et les descripteurs de points d'intérêt soient invariants à tous ces critères. Le processus d'extraction et d'appariement des points d'intérêt est alors plus sensible dans le cas asynchrone. Pour ces raisons, nous avons évalué les détecteurs et les descripteurs locaux sur des images asynchrones afin de choisir les algorithmes les plus appropriés pour notre cas.

Nous avons testé certains algorithmes de la littérature, les détecteurs et les descripteurs suivants : SIFT, SURF, HARRIS, MSER, STAR, FAST, BRIEF, BRISK, ORB et FREAK. Ces algorithmes sont décrits dans l'annexe A. Le choix de ces algorithmes a été basé sur les évaluations les plus récentes dans la littérature. Nous avons testé ces algorithmes sur des images routières de la base de données KITTI afin de sélectionner les algorithmes qui ont les meilleures performances sur des images non synchronisées. Pour avoir un aspect asynchrone à partir de la banque d'images KITTI, qui contient à la base des images synchronisées, nous prenons une image sur deux pour chaque caméra et nous appliquons le processus d'appariement entre deux images décalées dans le temps (provenant des deux caméras).

2.1 Extraction des primitives dans la littérature

Les détecteurs et les descripteurs locaux ont été comparés par plusieurs travaux dans la littérature [68][69][70] [67] [71][72][73] [74] [75] ce qui permet d'avoir une vue d'ensemble des détecteurs et des descripteurs existants. Les évaluations les plus récentes comparent l'efficacité et la performance des algorithmes sur divers aspects de la détection des caractéristiques et de l'exactitude de l'appariement. Toutes ces évaluations comparent les détecteurs et les descripteurs en les appliquant à des images dites requêtes par rapport à une image de référence. Plusieurs critères sont étudiés indépendamment : l'invariance

à la rotation, l'invariance à la translation, l'invariance au changement d'échelle, l'invariance au changement de luminosité, etc. Ces invariances sont obtenues soit par le détecteur, soit par le descripteur, ou bien par les deux.

Dans la littérature, il existe des détecteurs et des descripteurs très récents, notamment les descripteurs FREAK (2012), BRISK (2011), ORB (2011), BRIEF (2010) et DAISY (2010), et d'autres qui datent de plus longtemps mais qui sont toujours utilisés pour leur fiabilité, notamment SURF (2008), SIFT (1999), HARRIS (1988).

2.1.1 Les détecteurs de points d'intérêt

D'après Fraundorfer et al. [76], il existe deux approches principales pour l'extraction des primitives. La première approche consiste à extraire les primitives dans une image et de les suivre dans les images suivantes en utilisant des techniques de recherche locales, telles que la corrélation. La deuxième approche consiste à détecter les primitives de façon indépendante dans toutes les images et les faire correspondre par la suite en se basant sur la similitude entre leurs descripteurs. Les premières recherches en VO ont opté pour la première approche tandis que les travaux de la dernière décennie se sont basés sur la deuxième approche.

D'après l'étude de Tuytelaars [67], les détecteurs locaux peuvent être classés selon la nature des points d'intérêt en quatre classes : détecteurs de contours, détecteurs de ≪blobs≫ (ou groupe de pixels), détecteurs de régions et détecteurs basés saillance. Les détecteurs de contours se basent sur un changement brutal d'intensité lumineuse pour identifier les points d'intérêt. Harris est le détecteur le plus connu de cette catégorie. Les détecteurs de ≪ blobs ≫ cherchent les régions qui diffèrent dans leurs propriétés (comme la lumière ou la luminosité). Hessien [77] est le détecteur de ≪ blobs ≫ le plus connu. Les détecteurs de régions cherchent à détecter les régions en se basant sur des méthodes de segmentation. MSER [78] est le détecteur de régions le plus connu. Les détecteurs basés saillance cherchent les régions les plus saillantes par rapport à l'attention de l'homme. Itti [79] est le détecteur le plus connu de cette catégorie.

Les détecteurs et les descripteurs locaux peuvent être également distingués en fonction du niveau d'invariance. Plusieurs évaluations ont été menées en se concentrant sur cette propriété [70] [80] [81]. Le processus d'extraction des primitives locales doit satisfaire plusieurs propriétés pour une meilleure performance. L'évaluation de Tuytelaars [67] et le tutoriel d'odométrie visuelle de [76] résument les propriétés les plus importantes pour un bon détecteur : la répétabilité, le caractère distinctif/informatif, la précision de la localisation, la quantité de points détectés et l'efficacité du calcul. La répétabilité est définie comme le pourcentage des caractéristiques détectées sur la partie de la scène commune

à deux prises de vue et mises en correspondance entre deux images. Le caractère distinctif/informatif des algorithmes définit les caractéristiques d'image qui peuvent être distinguées de leurs voisinages et utiles pour l'appariement. La précision de la localisation est importante dans le sens où elle permet d'obtenir l'estimation de la relation géométrique et photométrique la plus cohérente entre deux images prises dans différentes conditions de visualisation. Le nombre de caractéristiques détectées doit être suffisamment grand pour avoir un nombre raisonnable de bons appariements (ou "inliers"). Les caractéristiques détectées doivent être localisées avec précision par rapport à leurs positions dans l'image mais aussi par rapport à l'échelle. L'efficacité du calcul des algorithmes est aussi importante que toutes les autres propriétés car le temps de calcul est une contrainte critique surtout pour les applications temps réel. Les propriétés des détecteurs locaux les plus utilisés en odométrie visuelle ont été étudiées par l'évaluation de [76] et [67]. Une évaluation plus approfondie des détecteurs Harris, Shi Tomasi, SURF, FAST et STAR pour les applications de navigation en robotique a été présentée dans [75].

Pour l'odométrie visuelle, les détecteurs de blobs et les détecteurs de coins sont les plus utilisés pour leur haute précision [76]. Les tableaux 2.1,2.2, 2.3 résument le recensement de [67] et de [76] sur les performances des détecteurs locaux les plus utilisés pour l'odométrie visuelle. Il s'agit d'une étude comparative des propriétés et des performances de quelques détecteurs de points d'intérêt. On remarque que SIFT, SURF, et CENSURE ne sont pas complètement invariants aux transformations affines, mais sont empiriquement invariants à certains changements de point de vue.

TABLE 2.1: comparaison des détecteurs - types des détecteurs

Détecteurs	détecteur de coins	détecteur de blobs	détecteur de régions
Harris	x		
SIFT	(x)	x	
SURF	(x)	x	
CENSURE	x		
MSER			x
FAST	x		

Chaque détecteur possède ses propres avantages et inconvénients. Fraundorfer et al. [76] conclue que les détecteurs de coins sont rapides mais pas très distinctifs, alors que les détecteurs de blob sont distinctifs mais plus lents. En outre, la position des coins dans les images est mieux localisée que les blobs, par contre les blobs sont mieux localisés à l'échelle. Pour ces raisons, le choix du détecteur approprié doit être fait en fonction des contraintes et des exigences de l'application. Pour cela, nous avons testé ces détecteurs pour pouvoir choisir l'algorithme le plus adapté à notre application.

TABLE 2.2: comparaison des détecteurs - Invariances

Détecteurs	la rotation	l'échelle	les transformations affines
Harris	x		
SIFT	x	x	(x)
SURF	x	x	(x)
CENSURE	x	x	(x)
MSER	x	x	x
FAST	x	x	

TABLE 2.3: comparaison des détecteurs - Performances

Détecteurs	répétabilité	précision	robustesse	temps de calcul
Harris	+++	+++	++	++
SIFT	+++	++	+++	+
SURF	+++	++	++	++
CENSURE	+++	++	+++	+++
MSER	+++	+++	++	++
FAST	++	++	++	++++

2.1.2 Les descripteurs de points d'intérêt

En ce qui concerne les descripteurs locaux, la région autour d'un point clé détecté est convertie en un descripteur qui peut être comparé à d'autres descripteurs. La plus simple description d'un point d'intérêt est la description de son voisinage : l'intensité des pixels dans un motif autour du point clé. En pratique, ce n'est pas suffisant car le descripteur basé sur les intensités uniquement n'est plus le même si on change l'orientation du motif ou l'échelle et ne sera pas apparié à son correspondant dans une autre image.

Il existe deux types de descripteurs : les descripteurs de type vecteurs et descripteurs de type chaînes de bits. Les descripteurs vectoriels décrivent un point d'intérêt en se basant sur son voisinage et calculent un vecteur comme descripteur du point clé. SIFT et SURF sont les descripteurs les plus connus de cette catégorie. Les descripteurs binaires décrivent un point d'intérêt en effectuant des tests binaires sur son voisinage et calculent une chaîne de bit comme descripteur du point clé. BRIEF, BRISK, ORB et FREAK sont les descripteurs les plus connus de cette catégorie. Les descripteurs binaires sont plus rapides et plus récent que les descripteurs vectoriels.

En ce qui concerne les critères qui nous intéressent, nous cherchons des algorithmes rapides, de bonne précision et qui ont une répétabilité élevée. Les tableaux 2.4 et 2.5 présentent une comparaison approximative des descripteurs locaux les plus utilisés pour l'odométrie visuelle. Cette étude comparative résume les recensements des comparaisons

présentées par Alahi et al. dans [15], quand ils présentent l'algorithme de FREAK, et par Calonder et al. dans [82], quand ils présentent l'algorithme de BRIEF. Dans [15], les performances de FREAK ont été évaluées par rapport à BRISK, SIFT et SURF et dans [82] celles de BRIEF par rapport à SURF.

TABLE 2.4: comparaison des descripteurs - Types & Invariances

Descripteurs	vecteur	binaire	Invariance à la rotation	Invariance à l'échelle
SIFT	x		x	x
SURF	x		x	x
FREAK		x	x	x
BRISK		x	x	x
ORB		x	x	x
BRIEF		x		

TABLE 2.5: comparaison des descripteurs - Performances

Descripteurs	répétabilité	précision	robustesse	temps de calcul
SIFT	+++	++	+++	++
SURF	+++	++	+++	+++
FREAK	+++	+++	+++	++++
BRISK	+++	+++	+++	+++
ORB	++++	+++	+++	++++
BRIEF	++++	+++	++++	++++

D'après le tableau 2.5, nous remarquons que les performances des descripteurs binaires sont très proches. Nous remarquons également que les descripteurs binaires dépasse les descripteurs de type vecteur en termes de temps de calculs et restent comparables en terme de répétabilité, précision et robustesse.

2.2 Évaluation des détecteurs et descripteurs de points d'intérêt

Dans cette section, nous présentons les résultats expérimentaux obtenus en appliquant la majorité des algorithmes mentionnés dans l'annexe A.

Pour évaluer les performances métriques des détecteurs et des descripteurs de points d'intérêt, Mikolajczyk et al. [70] ont proposé d'utiliser trois paramètres : le rapport de rappel (recall), la répétabilité et la précision. Ces paramètres décrivent la performance des algorithmes et sont très utilisés pour évaluer les détecteurs et les descripteurs. Heinly et al. [72] propose d'autres paramètres pour évaluer les algorithmes de détection et de

description de points d'intérêt : le rapport d'appariement putatif, la précision, le score d'appariement, le rapport de rappel (recall) et l'entropie. Cet ensemble de paramètres évalue les algorithmes par rapport à leur performance métrique, à la distribution des points d'intérêt et à la fréquence des points d'intérêt considérés comme candidats pour l'appariement. Le rapport des appariements putatifs est défini comme étant le rapport entre les appariements putatifs divisés par le nombre de caractéristiques. Ce paramètre traite de la sélectivité du descripteur et décrit la fraction de points d'intérêt initialement identifiés comme candidats à l'appariement (un matching) même si l'appariement est potentiellement incorrect.

Nous voulions utiliser un ensemble complet de paramètres qui nous permettent de choisir judicieusement les algorithmes les plus adéquats à notre application. Nous avons évalué les détecteurs et les descripteurs par des paramètres utilisés par les évaluations de l'état de l'art.

Les critères d'évaluation dépendent fortement de plusieurs facteurs : la baseline entre les images, les conditions de visibilités, la relation géométrique entre les scènes à mettre en correspondance (rotation, translation et échelle). De plus, les critères sont influencés par l'algorithme de mise en correspondance. En se basant sur les évaluations des détecteurs et descripteurs de points d'intérêt dans la littérature, nous avons sélectionné les critères suivant pour évaluer les algorithmes implémentés :

— Nombre de points clés initialement détectés (la moyenne entre les deux images).
— Nombre de bons appariements (noté par goodpt) : il s'agit du nombre de points considérés comme bons appariements (inliers) après l'estimation de la matrice essentielle en utilisant RANSAC avec un seuil de 0.5 et une probabilité de 0.99.
— Pourcentage des bons appariements (noté par % goodpt) : le nombre de bons appariements divisé par le nombre de points clés initialement détectés (le nombre minimum entre les deux images).
— Répétabilité : c'est la capacité de détecter le point clés quelque soit le point vue et même en présence de changements d'illumination et sous réserve de bruit.
— Recall : le nombre des bons appariements divisé par le nombre de points mis en correspondance.
— Efficacité : la Répétabilité multipliée par le Recall.

Nous n'évaluons pas les performances des détecteurs et des descripteurs en terme de temps, comme toutes les implémentations de cette thèse, car nos implémentations évaluent plutôt la performance des algorithmes et non leurs rapidités. Cependant, comme notre méthode est dédiée pour des applications temps réels, nous choisissons des algorithmes adaptés pour des implémentations temps réel. Pour les détecteurs et les descripteurs évalués dans cette section, nous utilisons l'implémentation de la bibliothèque openCV

en $C++$. Nous prenons en considération cette contrainte (la rapidité) dans le choix des algorithmes. En effet, nous nous sommes basé sur ce critère (la rapidité) qui a été déjà évalué par les évaluations de l'état de l'art. Nos implémentations ne sont pas optimisées mais reste proches du temps réel.

2.2.1 Résultats expérimentaux

Nous avons testé les détecteurs et les descripteurs locaux sur une séquence de 100 images de la base de données KITTI [9][10]. Nous avons testé ces détecteurs et descripteurs pour le cas synchrone (entre deux prises de vues synchronisées) ensuite pour le cas asynchrone (entre deux prises de vues non synchronisées avec un saut d'une seule image). Les résultats obtenus montrent que l'extraction a de meilleure performance dans le cas synchrone ce qui est raisonnable en raison du changement de point de vue pour le cas asynchrone.

Afin de comparer les détecteurs, nous avons testé tous les détecteurs avec le même descripteur (BRIEF) pour évaluer leur performance dans les mêmes conditions. Le nombre des bons appariements est obtenu après l'estimation de la pose (calcul de la matrice essentielle) pour éliminer les points aberrants. Nous avons utilisé l'algorithme de 5 points (avec un RANSAC pour la sélection de points) pour estimer la matrice essentielle.

FIGURE 2.1: Détéction de points d'intérêt avec le détecteur FAST - une image de la base KITTI

Les figures 2.1, 2.2, 2.3, 2.4, 2.5, 2.6 et 2.7 montrent les points clés détectés avec différents détecteurs de points d'intérêt. Le détecteur FAST donne le plus grand nombre de points clés, la figure 2.1.

Nous avons fait le test sur 100 images de la base de donnée KITTI. Nous avons commencé par tester tous les détecteurs (FAST, Harris, MSER, STAR, SURF, SIFT et ORB) associés au descripteur FREAK pour des images non synchronisées (tableau 2.6). Nous avons recommencé les mêmes tests de tous les détecteurs associés au descripteur BRIEF pour le cas asynchrone (tableau 2.7). Nous avons également testé tous les détecteurs

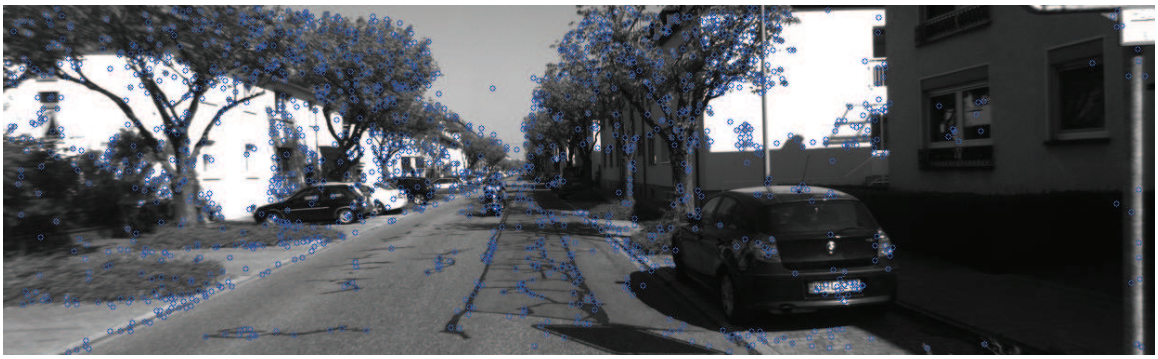


FIGURE 2.2: Détéction de points d'intérêt avec le détecteur SIFT - une image de la base KITTI

FIGURE 2.3: Détection de points d'intérêt avec le détecteur Harris - une image de la base KITTI

FIGURE 2.4: Détection de points d'intérêt avec le détecteur MSER - une image de la base KITTI

associés au descripteur BRIEF pour le cas synchrone afin de comparer les résultats par rapport à ceux du cas asynchrone (tableau 2.8).

Les tableaux 2.6, 2.7 et 2.8 montrent une comparaison quantitative des moyennes des nombres de points clés obtenus, des moyennes des points considérés comme des bons appariements (goodpt), les moyennes des pourcentages des bons appariements (% goodpt) et les répétabilités de chaque détecteur (FAST, HARRIS, MSER STAR, SURF, SIFT et ORB). Dans les trois tableaux, le détecteur FAST dépasse les autres détecteurs en terme de performances. C'est l'algorithme qui a les meilleures répétabilité et efficacité.

FIGURE 2.5: Détection de points d'intérêt avec le détecteur SURF - une image de la base KITTI

FIGURE 2.6: Détection de points d'intérêt avec le détecteur STAR - une image de la base KITTI

FIGURE 2.7: Détection de points d'intérêt avec le détecteur ORB - une image de la base KITTI

Afin de comparer les descripteurs, nous comparons les performances de tous les descripteurs (BRIEF, BRISK, FREAK et ORB) associés au même détecteur. Nous avons utilisé la même technique d'appariement pour tous les tests pour que cela n'influence pas nos résultats. Nous avons comparé les descripteurs en les appliquant sur les points détectés par le détecteur SURF dans le cas asynchrone puis sur les points détectés par le détecteur FAST dans les deux cas synchrone et asynchrone. Les résultats obtenus en appliquant ces algorithmes sur 100 images de la base de données ont été introduits par les tableaux 2.9, 2.10 et 2.11. Le descripteur BRIEF dépasse les autres algorithmes en

TABLE 2.6: Comparaison des détecteurs associés au descripteur FREAK - en mode non synchrone

	points clés	goodpt	% goodpt	répétabilité	recall	efficacité
FAST - FREAK	7863	2493	31.71%	0.784	0.357	0.280
Harris - FREAK	976	379	39.16%	0.547	0.429	0.234
MSER - FREAK	446	117	26.23%	0.190	0.414	0.078
STAR - FREAK	584	211	36.13%	0.448	0.370	0.166
SURF - FREAK	1030	282	27.37%	0.502	0.341	0.171
SIFT - FREAK	2903	879	30.28%	0.356	0.381	0.135
ORB - FREAK	500	107	21.54%	0.488	0.566	0.276

TABLE 2.7: Comparaison des détecteurs associés au descripteur BRIEF - en mode non synchrone

	points clés	goodpt	% goodpt	répétabilité	recall	efficacité
FAST - BRIEF	7863	3525	44.82%	0.822	0.519	0.427
Harris - BRIEF	976	547	56.04%	0.619	0,601	0.372
MSER - BRIEF	446	181	40.58%	0.355	0.504	0.178
STAR - BRIEF	584	285	48.80%	0.481	0.482	0.232
SURF - BRIEF	1030	414	40.19%	0.518	0.425	0.220
SIFT - BRIEF	2903	1122	36.64%	0.372	0.491	0.182
ORB - BRIEF	500	284	56.80%	0.584	0.569	0.332

TABLE 2.8: Comparaison des détecteurs associés au descripteur BRIEF - en mode synchrone

	points clés	goodpt	% goodpt	répétabilité	recall	efficacité
FAST - BRIEF	7867	4451	56.57%	0.829	0.656	0.544
Harris - BRIEF	964	637	66.02%	0.648	0.699	0.453
MSER - BRIEF	365	207	57.66%	0.433	0.567	0.246
STAR - BRIEF	592	353	60.50%	0.535	0.596	0.319
SURF - BRIEF	1051	477	45.36%	0.530	0.496	0.262
SIFT - BRIEF	2589	1335	51.57%	0.406	0.585	0.238
ORB - BRIEF	500	321	64.39%	0.632	0.643	0.407

donnant les meilleures répétabilité et efficacité.

Nous avons comparé les résultats obtenus par FAST-BRIEF pour le cas synchrone et le cas asynchrone, ces résultats sont présentés par les figures 2.8, 2.9 et 2.10. Nous avons conclu que la détection, la description et l'appariement sont plus sensibles et plus difficiles pour le cas asynchrone à cause des objets dynamiques dans la scène. En effet, les images non synchronisés sont décalés dans l'espace et dans le temps (le système est

TABLE 2.9: Comparaison des descripteurs associés au détecteur SURF - en mode non synchrone

	points clés	goodpt	% goodpt	répétabilité	recall	efficacité
SURF - BRIEF	1030	414	40.19%	0.505	0.434	0.219
SURF - BRISK	1030	328	31.84%	0.481	0.377	0.181
SURF - FREAK	1030	282	27.30%	0.502	0.341	0.171
SURF - ORB	1030	322	31.25%	0.461	0.342	0.158

TABLE 2.10: Comparaison des descripteurs associés au détecteur FAST - en mode non synchrone

	points clés	goodpt	% goodpt	répétabilité	recall	efficacité
FAST - BRIEF	7863	3525	44.82%	0.822	0.519	0.427
FAST - BRISK	7863	3369	42.84%	0.795	0.453	0.361
FAST - FREAK	7863	2493	31.71%	0.784	0.357	0.280
FAST - ORB	7863	3288	41.81%	0.820	0.494	0.405

TABLE 2.11: Comparaison des descripteurs associés au détecteur FAST - en mode synchrone

	points clés	goodpt	% goodpt	répétabilité	recall	efficacité
FAST - BRIEF	7867	4451	56.57%	0.829	0.656	0.544
FAST - BRISK	7867	4193	53.29%	0.818	0.564	0.461
FAST - FREAK	7867	3224	40.97%	0.823	0.462	0.380
FAST - ORB	7867	4284	54.45%	0.824	0.643	0.537

en mouvement) ce qui rend les variances de la rotation, de l'échelle et de l'illumination plus importantes.

Nous avons également évalué FAST-BRIEF en fonction du décalage temporel entre les images quand le véhicule est en circulation. La figure 2.11 illustre le nombre de bon appariements, le pourcentage des appariements, le recall et la répétabilité en fonction du décalage temporel entre les images : décalage 1 désigne un décalage $t + 1$, décalage 2 pour $t + 2$, etc. Nous en déduisons que plus le décalage temporel et spatial entre les images est grand, plus les performances de l'extraction et de l'appariement des points d'intérêt sont dégradées ce qui rend l'estimation du mouvement à partir des images non synchronisées plus délicat et relève donc plus de challenges.

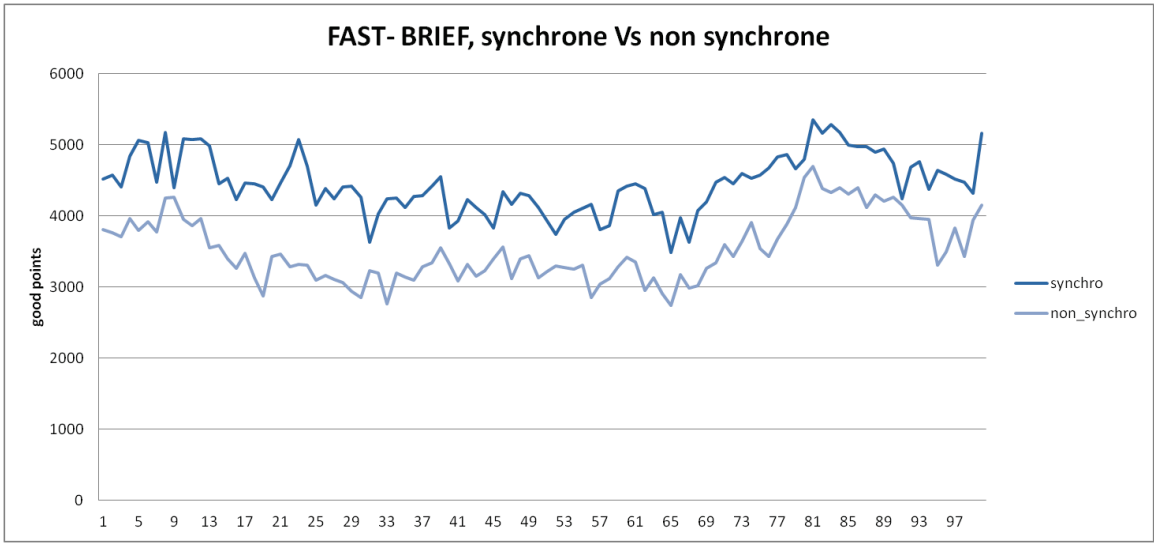


FIGURE 2.8: FAST - BRIEF appliqués à 100 images de la base KITTI, dans le cas synchrone et asynchrone : nombre de bons appariements

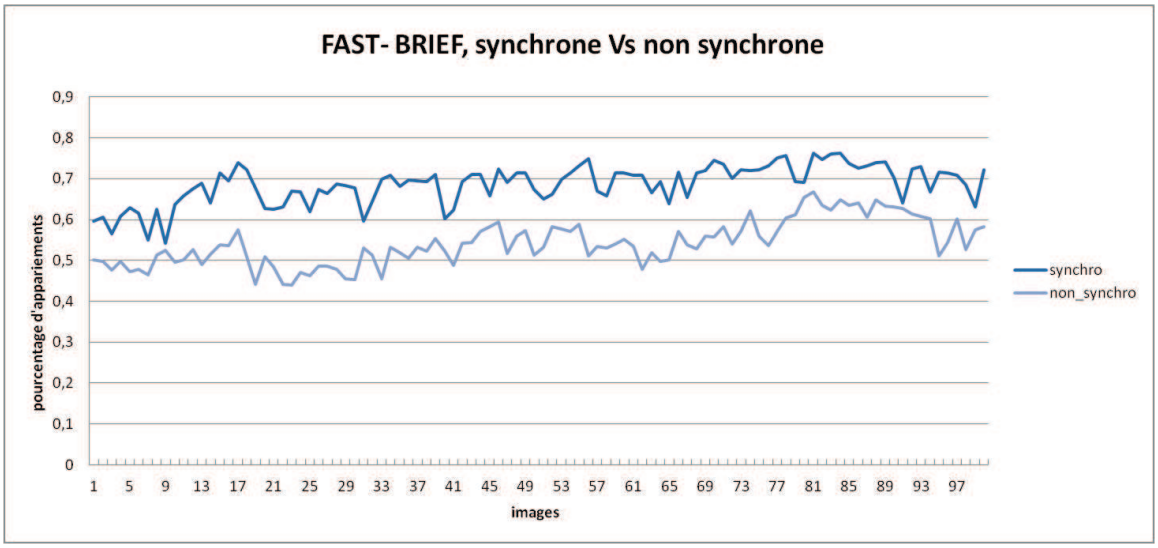


FIGURE 2.9: FAST - BRIEF appliqués à 100 images de la base KITTI, dans le cas synchrone et asynchrone : pourcentage d'appariement

2.2.2 Choix du détecteur et du descripteur

Dans l'étude de [67], Tuytelaars et Mikolajczyk donnent quelques précisions sur la sélection d'un détecteur approprié pour une application donnée. Leur indications ne constituent pas une réponse précise et définitive mais indique quelques points à prendre en compte lors de la recherche d'un détecteur approprié.

En se basant sur les résultats de la section précédente, nous avons choisi le détecteur FAST pour l'extraction des points clés et le descripteur BRIEF pour la description des points clés. Ces algorithmes permettent d'obtenir le nombre de points clés le plus élevé

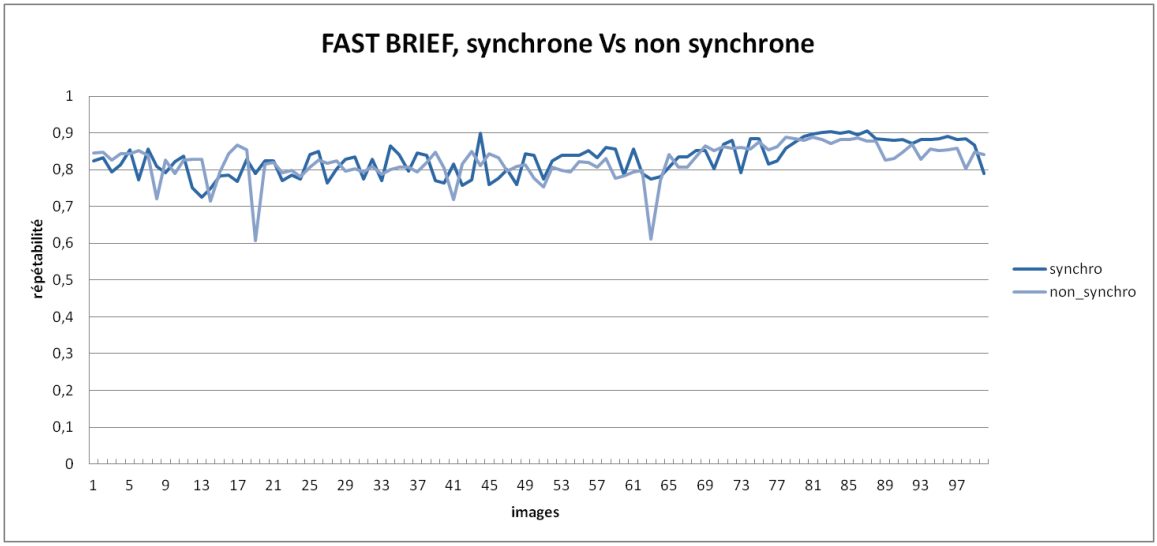
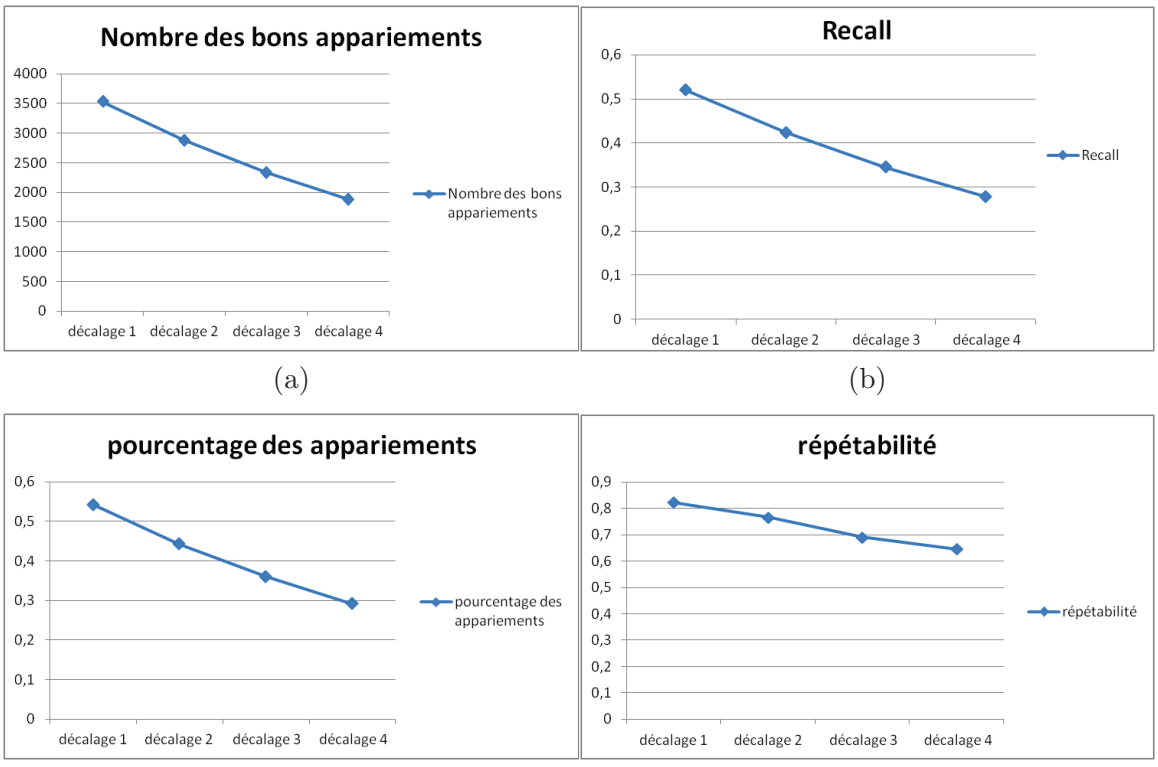


FIGURE 2.10: FAST - BRIEF appliqués à 100 images de la base KITTI, dans le cas synchrone et asynchrone : répétabilité



(a) (b)

(c) (d)

FIGURE 2.11: Évaluation de FAST - BRIEF en fonction du décalage temporel entre les images : décalage 1 désigne un décalage $t+1$, décalage 2 pour $t+2$, etc.

et le pourcentage des bons appariements le plus élevé pour une meilleure estimation de la relation géométrique entre deux images asynchrones.

Conclusion

Dans ce chapitre, nous avons présenté une étude bibliographique sur les détecteurs et les descripteurs locaux de la littérature les plus utilisés en odométrie visuelle et sur les évaluations réalisées sur ces algorithmes.

Pour choisir les algorithmes les plus appropriés pour des images de scènes routières issues d'un système non synchronisé, nous avons comparé les performances des détecteurs et des descripteurs sur des images synchrones et des images asynchrones. Le choix de ces algorithmes a été basé sur les évaluations les plus récentes dans la littérature. Pour notre application, nous avons sélectionné le détecteur et le descripteur qui ont les meilleures performances sur les images non synchronisées : le détecteur FAST pour l'extraction des points clés et le descripteur BRIEF pour la description des points clés.

Chapitre 3

Estimation du mouvement à l'échelle : la Méthode des triangles

Introduction

Ce chapitre présente une méthode d'estimation du mouvement à l'échelle à partir d'un réseau de caméras non synchronisées.

Il est important de rappeler que l'estimation des poses du système consiste à déterminer la rotation, la translation et le facteur d'échelle entre deux positions successives. Pour avoir le mouvement et la structure à l'échelle, le facteur d'échelle est généralement initialisé et maintenu ou déduit à partir d'une information supplémentaire sur la scène (avoir un capteur additif synchronisé avec la caméra au moment de l'acquisition pour faire correspondre les données extraites).

Pour s'affranchir du caractère asynchrone, nous proposons une nouvelle méthode d'estimation de mouvement qui n'impose aucune contrainte sur la synchronisation des caméras ni sur leur position relative. Nous appelons la méthode proposée la méthode des triangles. L'approche proposée se base sur l'estimation de pose entre un triplet d'images pour en déduire les facteurs d'échelle associés à chaque transformation entre deux images.

La première partie de ce chapitre présente le principe d'acquisition à partir d'un système asynchrone. La deuxième partie introduit le principe de la méthode proposée. Nous vérifions la validité de notre méthode sur des images simulées et des images réelles. Les résultats sont présentés dans la dernière partie de ce chapitre.

3.1 Le principe d'un système asynchrone

Un système de caméras asynchrone est un système pour lequel les caméras opèrent indépendamment les unes des autres. C'est-à-dire, chaque caméra acquiert une image indépendamment des autres. Dans ce cas, les caméras peuvent avoir des fréquences d'acquisition différentes et le système ne dépend pas de la caméra la plus lente. De plus, pour un tel système, les caméras peuvent être différentes dans le sens où on peut avoir un système de caméras hybrides : des caméras omnidirectionnelles et des caméras perspectives par exemple. Pour toutes ces raisons, un système asynchrone est un système très flexible et facilement modifiable. En effet, on peut ajouter ou supprimer des caméras au système sans avoir à changer le fonctionnement du système. En pratique, un système sans un circuit de synchronisation est un système asynchrone même s'il est composé par des caméras identiques et avec la même fréquence d'acquisition. En effet, le décalage du temps de traitement entre les images ou le décalage du signal d'horloge du processus d'acquisition des caméras rend les caméras non synchronisées si elle ne sont pas "forcées" à acquérir une image à un instant donné par un circuit de synchronisation externe.

Pour un système composé de N caméras, la figure 3.1 montre une distribution simple d'un système synchrone (où toutes les caméras acquièrent des images en même temps) et d'un système asynchrone (à chaque instant, uniquement une seule caméra peut acquérir une image). La représentation montre que les caméras ne sont pas alignées, géométriquement, à un instant t. Nous avons choisit de présenter le système de cette façon car en général pour un système embarqué sur un véhicule les caméras ne sont pas toujours alignées. La distribution du cas asynchrone est présentée d'une façon régulière (caméra 1, caméra 2, caméra 3 etc.) mais en pratique les images sont généralement obtenues d'une façon aléatoire.

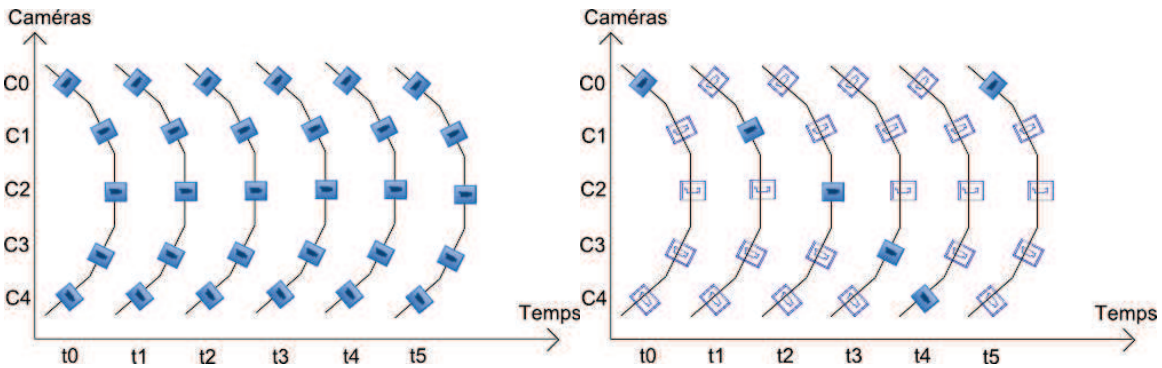


FIGURE 3.1: (a) Distribution d'images provenant d'un réseau de N caméras synchronisées (b) Distribution d'images provenant d'un réseau de N caméras non synchronisées.

Pour simplifier l'estimation de mouvement, nous proposons une méthode pour deux caméras. La méthode peut être généralisée (étendue) à N caméras. Ceci sera expliqué dans le chapitre 5. La figure 3.2 illustre la distribution d'images provenant d'un réseau de 2 caméras non synchronisées pour 3 instants t (le cas régulier). Il s'agit d'un exemple de distributions possibles. D'autres cas sont discutés plus loin.

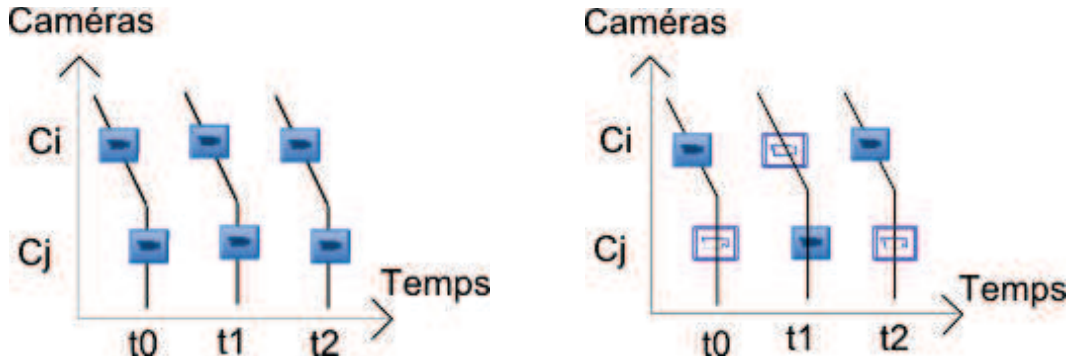


FIGURE 3.2: (a) Distribution d'images provenant d'un réseau de 2 caméras synchronisées (b) Distribution d'images provenant d'un réseau de 2 caméras non synchronisées.

3.2 La méthode des triangles

Pour pouvoir réaliser la reconstruction 3D d'un environnement dynamique autour d'un véhicule en mouvement avec un réseau de caméras non synchronisées, il faut avant toutes choses estimer le déplacement du véhicule donc estimer le déplacement des caméras les unes entre les autres au cours du temps. Estimer le mouvement d'un véhicule avec un réseau multi-caméras offre non seulement une précision sur l'estimation mais permet aussi une estimation à l'échelle (non pas à l'échelle près).

Notre méthode est inspirée par la méthode de reconstruction 3D présentée dans [65]. Cette méthode a été décrite dans le chapitre 1 dans la section 1.4. Cette méthode "simule" 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 notre méthode nous utilisons cette hypothèse dans le calcul des facteurs d'échelle du déplacement.

Notre méthode est basée sur trois hypothèses principales :

— Les caméras doivent être montées sur un système rigide et avoir des champs de vues communs. Le fait que les caméras sont fixées sur un système rigide permet

de connaitre la transformation rigide entre chaque couple de caméras (par le calibrage extrinsèque). Le chevauchement des champs de vue des caméras (deux à deux) permet d'effectuer des mises en correspondance entre les images.

— Les caméras doivent être calibrées hors lignes afin de connaître leurs paramètres intrinsèques et extrinsèques. Le calibrage extrinsèque sera utilisé pour trouver les facteurs d'échelle absolus et le calibrage intrinsèque sera utilisé dans le processus de l'estimation de pose et dans la triangulation.

— le mouvement entre deux vues consécutives est approximé par un segment de droite. Nous supposons que le mouvement entre deux vues consécutives est rectiligne (linéaire ou lisse). Cette approximation permet de considérer que les vecteurs de translation entre les trois positions d'une même caméras sont colinéaires. En effet, vu que les fréquences d'acquisition des caméras de nos jours sont de plus en plus élevées (20, 30, 40 jusqu'à 60 images par seconde), le décalage temporel entre 3 images (positions) n'est pas très grand. Par exemple, pour un véhicule équipé d'un système de caméras de 30 images par seconde et qui roule à 50 kilomètres par heure (à peu près 13 mètres par seconde), le décalage entre 3 images est de $1m30$ (a peu près une image tous les 45 centimètres).

Nous séparons la méthode en deux parties : l'estimation de poses relatives et l'estimation des facteurs d'échelle absolus. Les poses relatives des caméras sont estimées via SFM. Ensuite, les facteurs d'échelles absolus sont calculés en utilisant le calibrage extrinsèque et l'hypothèse de linéarité du déplacement.

Pour simplifier le problème, nous considérons un système avec seulement deux caméras qui acquièrent des images en trois instants. La première et la deuxième caméra, C_i et C_j, acquièrent respectivement des images I_i et I_j. La notation de temps est placée comme un indice fixé à la notation de la caméra. Par exemple, la position de la caméra C_i à l'instant t_0 est désignée par C_{i0}. Les transformations entre les poses des caméras sont désignées par T. La transformation euclidienne T représente la position d'une caméra dans le système de coordonnées de l'autre caméra. Par exemple, la transformation de la position de la caméra C_{i0} à la position de la caméra à C_{j1} est $\mathbf{T}_{i0}^{j1}$. De la même façon, la matrice de rotation et le vecteur unitaire de la translation de la position de la caméra C_{i0} exprimés dans le système de coordonnées de la caméra C_{j1} sont désignés par $\mathbf{R}_{i0}^{j1}$ et $\mathbf{t}_{i0}^{j1}$.

3.2.1 Estimation de Pose

Deux poses relatives d'une caméra peuvent être décrites par un vecteur unitaire de translation $\mathbf{t}$, une matrice de rotation $\mathbf{R}$ et un facteur d'échelle λ. La transformation

entre les deux positions peut être exprimée comme suit :

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \lambda\mathbf{t} \\ 0^\intercal & 1 \end{bmatrix} \tag{3.1}$$

Nous obtenons la matrice de rotation $\mathbf{R}$ et le vecteur de translation $\mathbf{t}$ entre deux images à partir de la matrice essentielle définissant la géométrie entre les deux poses comme il a été expliqué dans le chapitre 1.

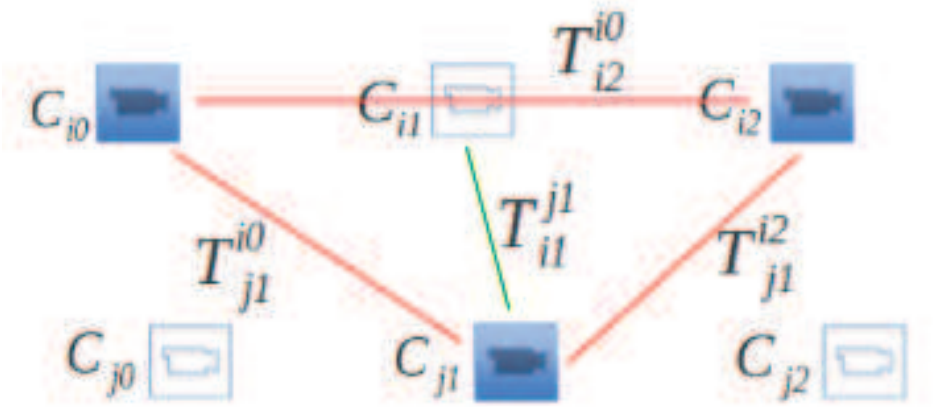


FIGURE 3.3: La méthode des triangles pour deux caméras non synchronisées : les lignes rouges désignent les transformation obtenues par SFM et les lignes vertes désignent la transformation statique issue de la calibration extrinsèque

La méthode des triangles repose sur trois matrices essentielles calculées entre trois images à trois instants : deux images provenant de la même caméra (C_i par exemple) et une image de l'autre caméra (C_j dans ce cas). Le triangle peut être modélisé comme dans la figure 3.3 qui présente toutes les transformations possibles entre les trois images formant le " triangle". Les transformations entre les trois images sont déterminées par l'algorithme des 5-points : $\mathbf{T}^{i0}_{i2}$, $\mathbf{T}^{i0}_{j1}$ et $\mathbf{T}^{i2}_{j1}$. La transformation statique $\mathbf{T}^{j1}_{i1}$ est obtenue à partir du processus de calibrage extrinsèque hors ligne.

Nous commençons notre algorithme par l'extraction et la mise en correspondance des points d'intérêt détectés dans les trois images du triangle. Nous effectuons des appariements entre les trois paires d'images. Dans notre implémentation, nous utilisons le détecteur FAST et le descripteur BRIEF. Ces étapes sont appliquées entre les images I_{i0}, I_{i2} et I_{j1}.

La caméra C_i passe par une position intermédiaire où elle n'acquiert pas d'image en raison du fait de la non synchronisation. Il s'agit de la position C_{i1} qui peut être estimée en utilisant la transformation $\mathbf{T}^{j1}_{i1}$. En résumé, nous utilisons quatre transformations :

trois calculées via SFM ($\mathbf{T}_{i2}^{i0}$, $\mathbf{T}_{j1}^{i0}$ et $\mathbf{T}_{j1}^{i2}$) et une transformation statique ($\mathbf{T}_{i1}^{j1}$) issue de l'étalonnage extrinsèque.

3.2.2 Estimation des facteurs d'échelle : méthode des triangles

Jusque là, les facteurs d'échelle absolus sont inconnus entre les poses de la caméra. Les hypothèses initialement posées permettent de moduler le système par quatre principales transformations comme il a été montré dans la figure 3.3. Les trois images (I_{i0}, I_{i2} et I_{j1}) forment un "grand triangle" entre les positions de C_{i0}, C_{i2} et C_{j1}. La pose virtuelle de la caméra C_{i1} peut être considérée comme une position intermédiaire dans le grand triangle. Cette position donne deux sous-triangles : le premier sous-triangle est formé par C_{i0}, C_{i1} et C_{j1} et le second est formé par C_{i1}, C_{i2} et C_{j1}.

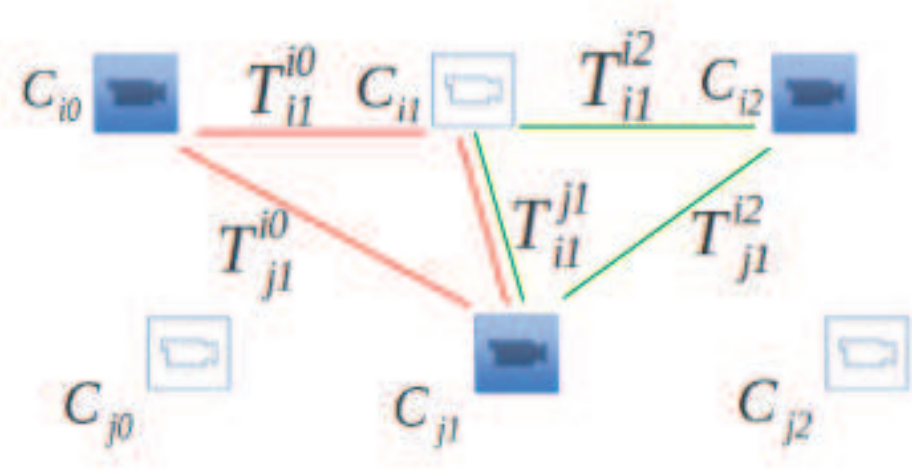


FIGURE 3.4: Le premier sous-triangle entre les caméras (C_{i0}, C_{i1} et C_{j1}) et le deuxième sous-triangle entre les caméras (C_{i1}, C_{i2} et C_{j1})

Dans le premier sous-triangle, figure 3.4, la transformation de la position C_{i1} vers le repère de la position C_{i0}, $\mathbf{T}_{i1}^{i0}$ est égale à la transformation de la position de la caméra C_{j1} vers le repère de la position C_{i0}, $\mathbf{T}_{j1}^{i0}$, multipliée par la transformation de la position de la caméra C_{i1} pour le repère de la position C_{j1}, $\mathbf{T}_{i1}^{j1}$, l'équation 3.2.

$$\mathbf{T_{i1}^{i0}} = \mathbf{T_{j1}^{i0}}\ \mathbf{T_{i1}^{j1}} \tag{3.2}$$

Les transformations euclidiennes sont exprimées dans le système de coordonnées homogènes, comme indiqué dans l'équation A.1. La transformation statique $\mathbf{T}_{i1}^{j1}$ obtenue par le calibrage extrinsèque est une transformation à l'échelle, le facteur d'échelle de cette transformation est connu. Développons l'équation 3.2, nous introduisons les facteurs d'échelles inconnues (λ_1, α) de chaque transformation dans l'équation 3.3. 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 $\mathbf{T}_{i0}^{j1}$.

$$\begin{bmatrix} \mathbf{R}_{i1}^{i0} & \lambda_1 \mathbf{t}_{i1}^{i0} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{j1}^{i0} & \alpha \mathbf{t}_{j1}^{i0} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{i1}^{j1} & \mathbf{t}_{i1}^{j1} \\ 0 & 1 \end{bmatrix} \tag{3.3}$$

En développant l'équation 3.3, nous obtenons :

$$\mathbf{R}_{i1}^{i0} = \mathbf{R}_{j1}^{i0} \mathbf{R}_{i1}^{j1} \tag{3.4}$$

$$\lambda_1 \mathbf{t}_{i1}^{i0} - \alpha \mathbf{t}_{j1}^{i0} = \mathbf{R}_{j1}^{i0} \mathbf{t}_{i1}^{j1} \tag{3.5}$$

Par ailleurs, l'équation 3.5 peut être écrite :

$$\begin{bmatrix} \mathbf{t}_{i1}^{i0} & -\mathbf{t}_{j1}^{i0} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \alpha \end{bmatrix} = \mathbf{R}_{j1}^{i0} \mathbf{t}_{i1}^{j1} \tag{3.6}$$

Dans le deuxième sous triangle, figure 3.4, la transformation de la position C_{i1} vers le repère de la position C_{i2}, $\mathbf{T}_{i1}^{i2}$ est égale à la transformation de la position de la caméra C_{j1} vers le repère de la position C_{i2} ($\mathbf{T}_{j1}^{i2}$) multipliée par la transformation statique de la position de la caméra C_{i1} pour le repère de la position C_{j1}, $\mathbf{T}_{i1}^{j1}$, l'équation 3.7.

$$\mathbf{T}_{i1}^{i2} = \mathbf{T}_{j1}^{i2} \mathbf{T}_{i1}^{j1} \tag{3.7}$$

De la même façon que le premier sous-triangle, l'équation 3.7 est développée pour obtenir :

$$\begin{bmatrix} \mathbf{R}_{i1}^{i2} & \lambda_2 \mathbf{t}_{i1}^{i2} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{j1}^{i2} & \beta \mathbf{t}_{j1}^{i2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{i1}^{j1} & \mathbf{t}_{i1}^{j1} \\ 0 & 1 \end{bmatrix} \tag{3.8}$$

λ_2 représente le facteur d'échelle associé à la transformation T_{i1}^{i2}. β représente le facteur d'échelle associé à la transformation T_{i2}^{j1}. Séparons dans l'équation 3.7 les termes en rotation et en translation, nous obtenons :

$$\mathbf{R}_{i1}^{i2} = \mathbf{R}_{j1}^{i2} \mathbf{R}_{i1}^{j1} \tag{3.9}$$

$$\lambda_2 \mathbf{t}_{i1}^{i2} - \beta t_{j1}^{i2} = \mathbf{R}_{j1}^{i2} \mathbf{t}_{i1}^{j1} \tag{3.10}$$

Dans le "grand" triangle entre les positions C_{i0}, C_{i2} et C_{j1}, figure 3.3, les transformations euclidienne entre les poses peuvent être exprimées comme dans l'équation 3.11. La transformation de la position C_{i2} exprimée dans le repère de la caméra C_{i0} ($\mathbf{T}_{i2}^{i0}$) est égale à la transformation de C_{j1} exprimée dans le repère de la caméra C_{i0} ($\mathbf{T}_{j1}^{i0}$) multipliée par la transformation de C_{i2} exprimée dans le repère de la caméra C_{j1} ($\mathbf{T}_{i2}^{j1}$).

$$\mathbf{T}_{i2}^{i0} = \mathbf{T}_{j1}^{i0}\mathbf{T}_{i2}^{j1} \tag{3.11}$$

Comme pour les sous-triangles, l'équation 3.11 devient :

$$\begin{bmatrix} \mathbf{R}_{i2}^{i}{}^{0} & (\lambda_1+\lambda_2)\mathbf{t}_{i2}^{i}{}^{0} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{j1}^{i}{}^{0} & \alpha\mathbf{t}_{j1}^{i}{}^{0} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{i2}^{j1} & \beta\mathbf{t}_{i2}^{j1} \\ 0 & 1 \end{bmatrix} \tag{3.12}$$

Après le développement de l'équation 3.12, nous séparons les termes en rotation et en translation. La translation conduit à l'équation 3.13.

$$\begin{bmatrix} \mathbf{t}_{i2}^{i0} & \mathbf{t}_{i2}^{i0} & -\mathbf{t}_{j1}^{i0} & -\mathbf{R}_{j1}^{i0}\mathbf{t}_{i2}^{j1} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \alpha \\ \beta \end{bmatrix} = 0 \tag{3.13}$$

En résumé, les équations en terme de translation pour les trois triangles étudiés sont exprimées par le système suivant :

$$\begin{cases} \lambda_1 \mathbf{t}_{i1}^{i}{}^{0} - \alpha \mathbf{t}_{j1}^{i}{}^{0} = \mathbf{R}_{j1}^{i}{}^{0}\mathbf{t}_{i1}^{j1} \\ \lambda_2 \mathbf{t}_{i1}^{i}{}^{2} - \beta \mathbf{t}_{j1}^{i}{}^{2} = \mathbf{R}_{j1}^{i}{}^{2}\mathbf{t}_{i1}^{j1} \\ \lambda_1 \mathbf{t}_{i2}^{i0} + \lambda_2 \mathbf{t}_{i2}^{i0} - \beta \mathbf{R}_{j1}^{i}{}^{0}\mathbf{t}_{i2}^{j1} - \alpha \mathbf{t}_{j1}^{i}{}^{0} = 0 \end{cases} \tag{3.14}$$

Pour résoudre ces équations et obtenir les facteurs d'échelle, nous pouvons écrire le système d'équations, l'équation 3.14, comme suit :

$$\begin{bmatrix} \mathbf{t}_{i1}^{i0} & 0 & -\mathbf{t}_{j1}^{i0} & 0 \\ 0 & \mathbf{t}_{i1}^{i2} & 0 & -\mathbf{t}_{j1}^{i2} \\ \mathbf{t}_{i2}^{i0} & \mathbf{t}_{i2}^{i0} & -\mathbf{t}_{j1}^{i0} & -\mathbf{R}_{j1}^{i0}\mathbf{t}_{i2}^{j1} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{j1}^{i0}\mathbf{t}_{i1}^{j1} \\ \mathbf{R}_{j1}^{i2}\mathbf{t}_{i1}^{j1} \\ 0 \end{bmatrix} \tag{3.15}$$

Les facteurs d'échelle absolus peuvent être retrouvés par la résolution linéaire de l'équation 3.15 par la méthode des moindres carrés. L'équation 3.15 peut être écrite comme un système linéaire de la forme :

$$\mathbf{AX} = \mathbf{B} \tag{3.16}$$

avec $\mathbf{X}$ le vecteur des facteurs d'échelle λ_1, λ_2, α et β. La transformation de C_{i0} dans le repère de C_{i1} et la transformation de C_{i2} vers C_{i1} peuvent être obtenues grâce à

l'hypothèse de linéarité entre les trois positions C_{i0}, C_{i1} et C_{i2} de la caméra C_i. Les translations entre les trois positions C_{i0}, C_{i1} et C_{i2} peuvent être exprimées en termes de vecteurs unitaires : $\mathbf{t}_{i1}^{i0}$, $\mathbf{t}_{i2}^{i1}$ et $\mathbf{t}_{i2}^{i0}$. L'équation suivante représente la relation entre ces vecteurs.

$$(\lambda_1 + \lambda_2)\mathbf{t}_{i2}^{i0} = \lambda_1 \mathbf{t}_{i1}^{i0} + \lambda_2 \mathbf{t}_{i2}^{i1} \Leftrightarrow \mathbf{t}_{i2}^{i0} = \mathbf{t}_{i1}^{i0} = \mathbf{t}_{i2}^{i1} \tag{3.17}$$

Toutes les transformations sont calculées soit par le SFM ou à partir de la calibration et l'hypothèse de linéarité. Les facteurs d'échelle peuvent être obtenus par une simple résolution de l'équation 3.16. Ainsi, nous pouvons obtenir les poses relatives et les facteurs d'échelle absolus pour tous les triangles d'une séquence.

3.3 Discussion

La méthode des triangles n'est applicable que dans le cas où deux caméras partageant un champ de vue commun acquièrent trois images : deux provenant d'une caméra et une provenant de l'autre caméra. L'ordre d'arrivée des images n'influence pas le calcul que nous avons présenté dans la section précédente. En effet, les équations 3.2, 3.7 et 3.11 sont valides quelque soit l'emplacement des caméras.

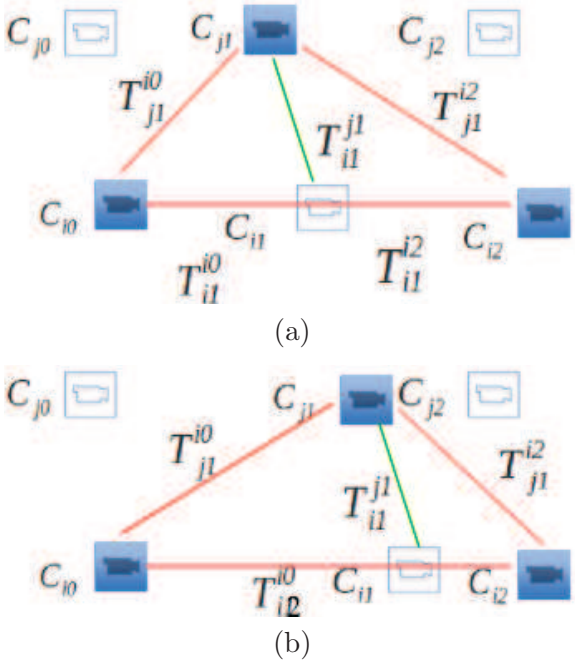


FIGURE 3.5: Différentes formes de triangles entre les caméras C_{i0}, C_{i2} et C_{j1}

La figure 3.5 présente différentes formes de triangles entre les caméras C_{i0}, C_{i2} et C_{j1}. Même pour le cas de la figure 3.5 (a) où le triangle défini dans la figure 3.3 paraît

inversé, les équations de la méthode des triangles restent valides (il suffit d'inverser les appellations caméras : C_i et C_j).

La méthode des triangles ne dépend pas de la distance du déplacement entre C_{i0}, C_{i1} et C_{i2} i.e. il n'y a aucune contrainte sur la distance entre les poses de la caméras C_i. (3.5(b)). La seule contrainte est qu'il existe suffisamment de points communs entre les images pour que les transformations soient estimées par SFM. Cela dépend de la scène et de la vitesse de la voiture. Ce point sera discuté dans l'analyse des résultats.

3.4 Résultats expérimentaux

3.4.1 Simulation

Nous appliquons l'algorithme précédemment décrit sur des données simulées. Un nuage de 100 points 3D est aléatoirement généré constituant une scène observée par deux caméras. Les points 3D sont projetés dans les deux caméras en utilisant le modèle de caméras perspectives. Nous simulons trois poses pour chacune des caméras. Les caméras virtuelles sont liées par une transformation statique afin de satisfaire nos hypothèses. Ainsi, le système virtuel se déplace linéairement. Trois images (I_{i0}, I_{j1} et I_{i2}) sont obtenues sans bruit puis avec un bruit additif de σ=0.2. Les points 3D sont devant les caméras. La figure 3.6 illustre la scène simulée.

Nous considérons que tous les points 3D sont devant les caméras et donc vus par les deux caméras (100 points 2D). Les points d'intérêt sont extraits, décrits et appariés entre les trois images formant le triangle. Les matrices essentielles sont calculées puis décomposées en rotations et en translations. En utilisant la transformation $\mathbf{T}_{i1}^{j1}$, les facteurs d'échelle sont estimés par la résolution du système d'équations obtenu par la méthode des moindres carrées.

Table 3.1: Résultats de la simulation : les facteurs d'échelle obtenus par la méthode des triangles TM et à partir de la vérité terrain GT

	λ_1	λ_2	α	β
GT	0.7291	1.4810	0.4379	2.3887
TM sans bruit	0.7291	1.4810	0.4379	2.3887
TM avec bruit	0.8313	1.8344	0.3986	2.7781
pourcentage d'erreur (avec bruit)	14.01%	23.86%	8.97%	16.30%

Dans le cas non bruité, les facteurs d'échelle estimés dévoilent une très haute précision par rapport aux facteurs d'échelle métrique obtenus de la vérité terrain. Dans le cas

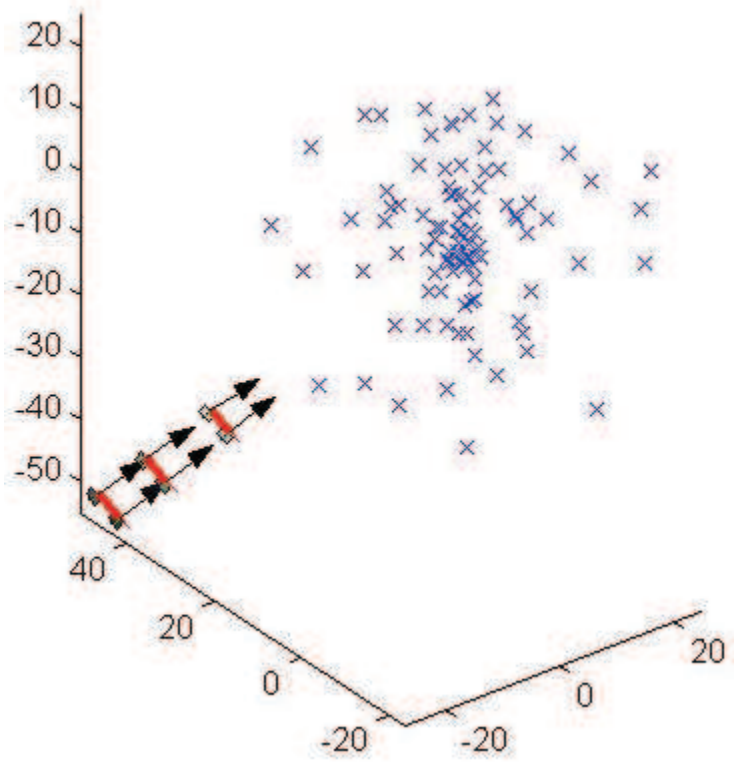


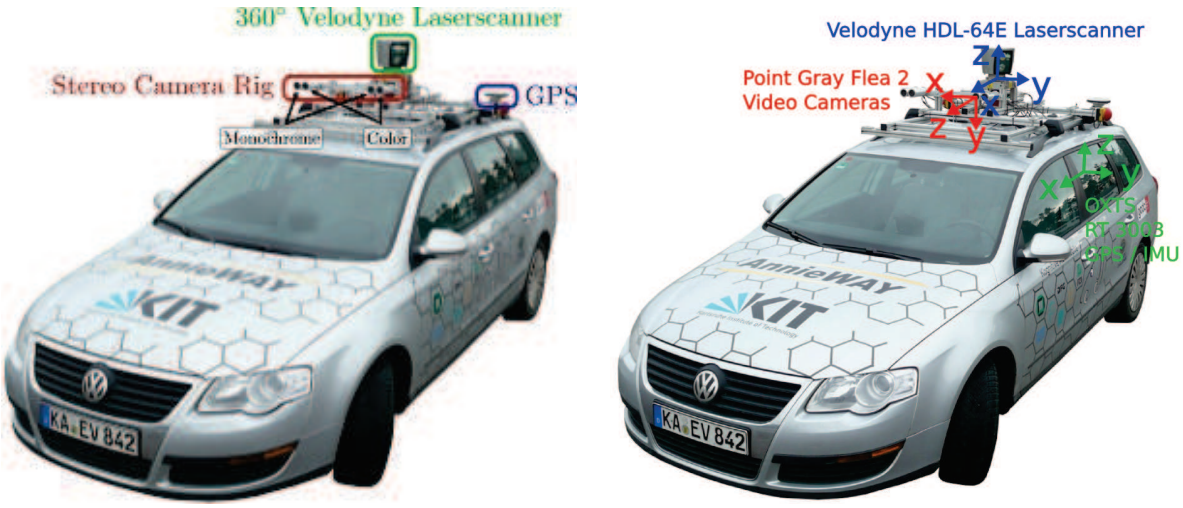
FIGURE 3.6: Scène simulée : un nuage de point 3D et deux caméras suivant une trajectoire rectiligne

bruité, les résultats présentent une erreur due au bruit additif, tableau 3.1. La simulation n'a pour objectif que de valider nos hypothèses pour un cas simple (valider l'aspect théorique de la méthode proposée) avec des données parfaites pour un seul triangle. Un déplacement d'un véhicule selon une trajectoire avec des courbures a été étudié dans le cas réel.

3.4.2 Données Réelles

Afin de valider la méthode des triangles, nous vérifions la validité de nos hypothèses pour une séquence du monde réel. Nous utilisons la banque d'images réelles de la base KITTI [9][10]. Nous comparons les résultats de notre méthode pour une séquence issue de deux caméras parfaitement synchronisées par rapport aux positions de la vérité de terrain issue des mesures d'un GPS/INS. La banque d'image KITTI est une base de données publique qui a été acquise à partir d'un véhicule instrumenté. Nous utilisons les séquences stéréo en prenant une seule image à chaque instant afin de simuler la situation asynchrone. En effet, notre méthode utilise une image sur deux pour avoir l'aspect asynchrone. Les poses relatives et les facteurs d'échelle sont calculés pour chaque ensemble de trois images formant un grand triangle.

Les capteurs utilisés dans notre évaluation sont des caméras monochromes PointGray Flea2, 1.4 Megapixels et un système de navigation inertielle GPS/RTK. Les ensembles de données sont enregistrés avec des capteurs montés sur le dessus d'un véhicule conduit sur des routes structurées, figure 3.7.



(a) (b)

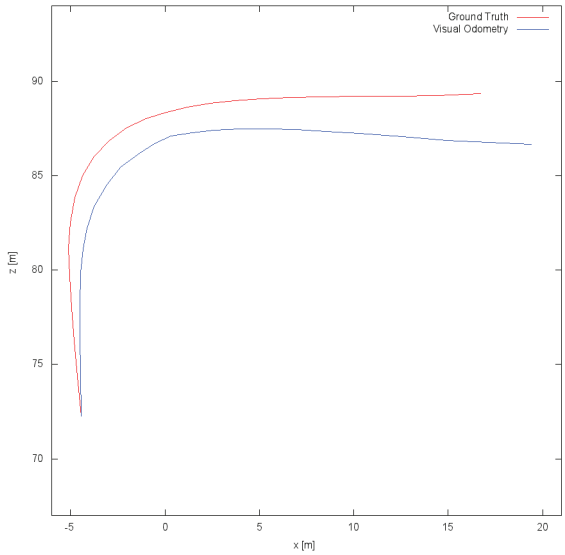
FIGURE 3.7: Le véhicule utilisé pour la collecte des séquences de la base de données KITTI [9][10].

Pour chaque série de trois images (trois instants), les points d'intérêt sont extraits en utilisant le détecteur FAST et décrits avec le descripteur BRIEF. Les appariements entre deux images permettent d'estimer une matrice essentielle grâce à l'algorithme 5-points. Les rotations et les translations relatives sont obtenues à partir des matrices essentielles estimées. Les points d'intérêt retenus comme « inliers » sont ensuite triangulés afin d'estimer les positions des points 3D. Le système d'équation 3.15 est résolu pour calculer les facteurs d'échelle absolue λ_1, λ_2, α et β.

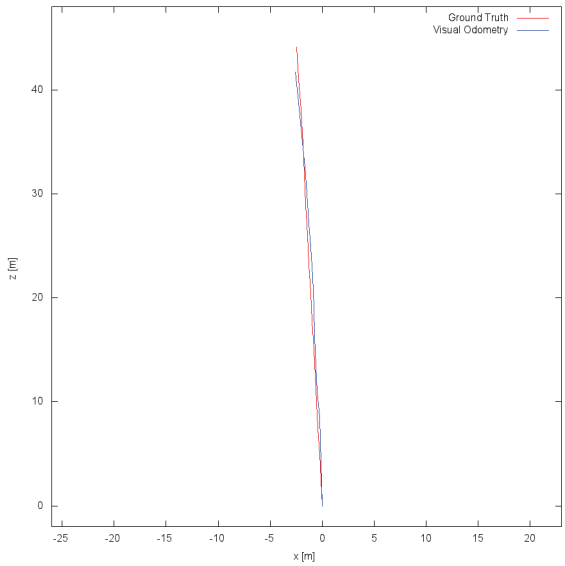
Pour évaluer notre méthode, nous comparons les résultats de l'estimation par rapport à la vérité terrain obtenue par le GPS. Dans cette thèse, nous ne comparons pas nos résultats par rapport aux autres méthodes de l'état de l'art car nous ne sommes pas dans le même cadre que les autres méthodes. En effet, notre approche se base sur des images non synchronisées. Nous avons donc moins d'informations que le cas synchrone. Nous ne comparons donc pas nos résultats aux méthodes synchrones car nous n'avons pas les mêmes informations en entrée. Nous ne comparons également pas notre méthode aux méthodes monoculaires car nous ne sommes pas dans la même configuration. Dans le cas monoculaire, le déplacement d'une caméra (entre deux instants consécutifs) présente en général de faibles changements de champs de vue même à grande vitesse. Cependant, l'estimation à partir de plusieurs caméras non synchronisées subit des changements de champs de vue plus important.

Nous évaluons notre méthode pour le cas des trajectoires rectilignes et les virages puis nous l'évaluons qualitativement et quantitativement et nous nous positionnons par rapport à quelques autres méthodes d'odométrie visuelle en termes d'erreurs de l'estimation.

3.4.2.1 les trajectoires rectilignes et les virages



(a) la séquence 0 : trajectoires de l'instant 0 à l'instant 50



(b) la séquence 0 : trajectoires de l'instant 80 à l'instant 150

FIGURE 3.8: Les trajectoires rectilignes et les virages. La comparaison des trajectoires du véhicule estimées par la méthode des triangles (les trajectoires en bleu) par rapport à la vérité terrain issus du GPS (les trajectoires en rouge)

Comme l'a montré la simulation, pour les trajectoires rectilignes, nous obtenons une estimation très proche de la vérité terrain : la figure 3.8 (a). La figure 3.8 (b) illustre

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, en général, le véhicule ne roule pas très rapidement dans les virages. De plus, les caméras de nos jours ont des fréquences d'acquisitions suffisamment élevées. Tenant compte de ces deux points, l'approximation du déplacement entre deux images d'une même caméra à un segment de droite reste valide. La trajectoire dans un virage peut être donc interprétée comme une succession de trajectoires rectilignes entre deux images d'une caméra.

3.4.2.2 L'évaluation qualitative de la méthode des triangles

Les figures 3.9, 3.10 et 3.11 illustrent les trajectoires réelles et estimées. Les trajectoires en bleu représentent les positions du GPS (la vérité terrain) et celles en rouge représentent les trajectoires estimées par la méthode des triangles. Les longues séquences comme la séquence 0 (4540 images) et la séquence 2 (4660 images) présentent des dérives importantes dues aux accumulations des erreurs. Les séquences plus courtes comme la séquence 3 (800 images), la séquence 4 (270 images) et la séquence 5 (2760 images) présentent des trajectoires proches de la vérité terrain.

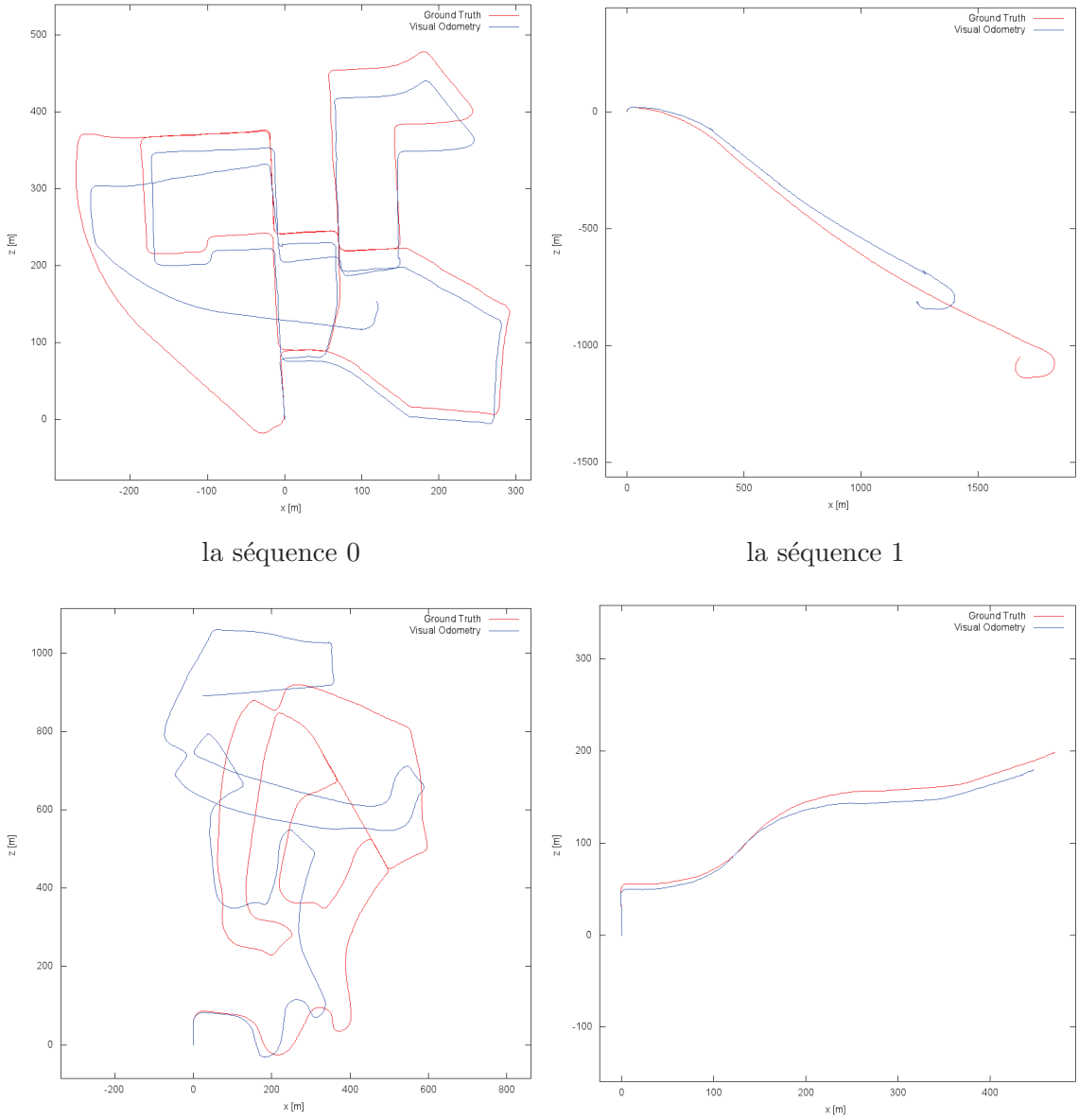
3.4.2.3 L'évaluation quantitative de la méthode des triangles

TABLE 3.2: Les valeurs moyennes de λ_1 et λ_2 (TM et GT) pour 223 triangles

	TM	GT	TM/GT
λ_1	0.805	0.706	1.14022
λ_2	0.864	0.705	1.22553

Le tableau 3.2 présente une comparaison quantitative des valeurs moyennes des facteurs d'échelle λ_1 et λ_2 pour 223 triangles de la séquence 0. La figure 3.12 montre la distribution des valeurs de $\lambda_1 + \lambda_2$ estimées par la méthode des triangle TM et obtenus de la vérité terrain GT pour la même séquence. Les pics proviennent des nombres insuffisants de points d'intérêt. Cette courbe présente les données brutes sans aucun filtrage d'où l'apparition d'erreurs inhérent aux techniques utilisées. Les erreurs apparaissent sur une image (un triangle) mais n'a pas une conséquence particulière sur la courbe.

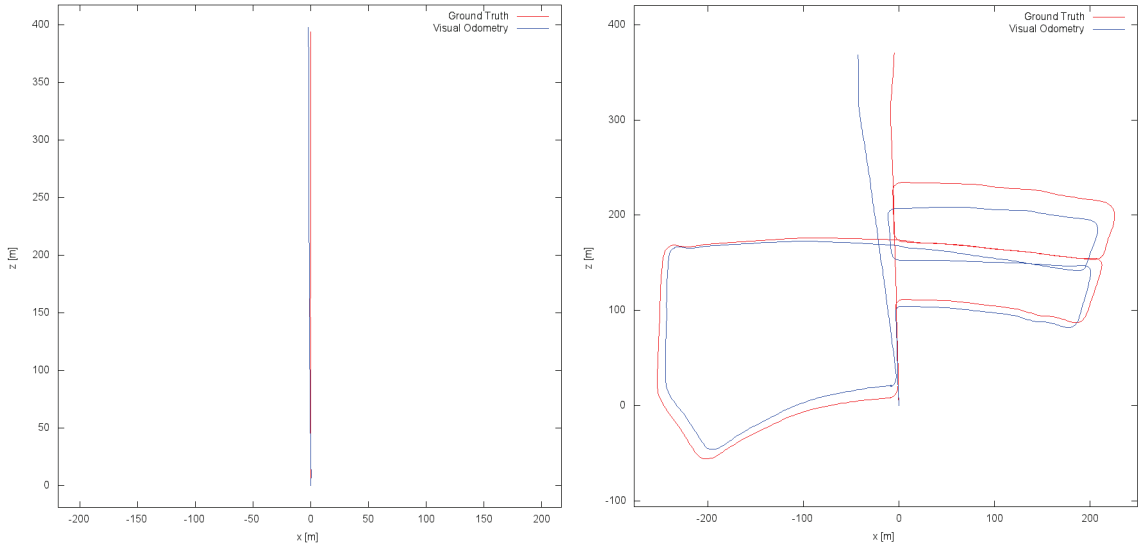
Nous évaluons notre méthode quantitativement par les erreurs de rotations et de translations. Nous utilisons l'évaluation proposée par la base KITTI [10]. En effet, les évaluations basées sur l'erreur de la trajectoire au point final peuvent être trompeuses car la mesure dépend fortement du moment où elle a été prise. Geiger et al. [10] proposent une



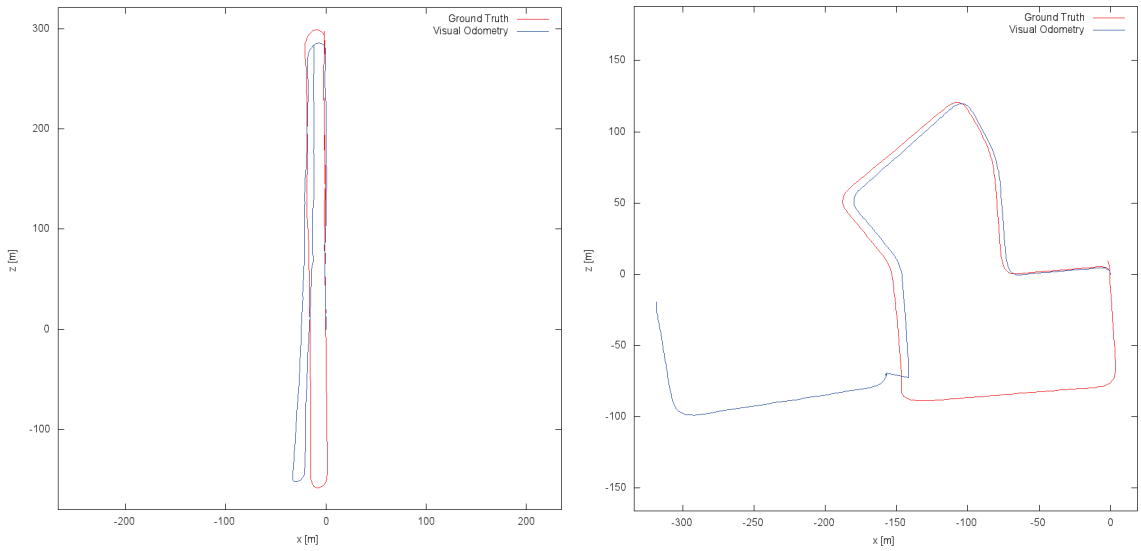
la séquence 0 la séquence 1

la séquence 2 la séquence 3

FIGURE 3.9: Les trajectoires des séquences 0 à 3 de la base KITTI. La comparaison des trajectoires du véhicule estimées par la méthode des triangles (les trajectoires en bleu) par rapport à la vérité terrain issus du GPS (les trajectoires en rouge).

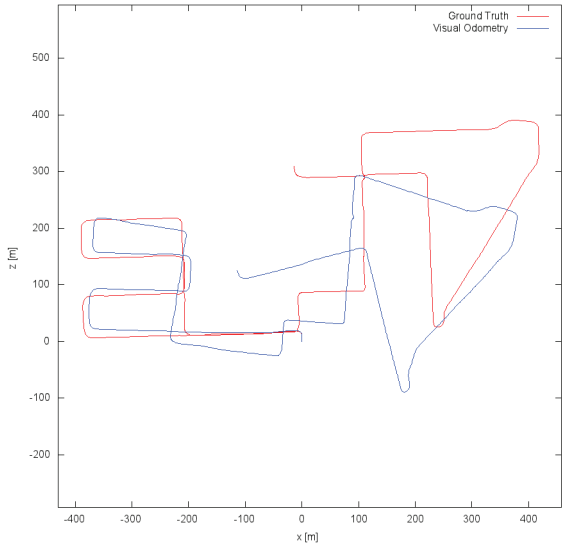


la séquence 4 la séquence 5

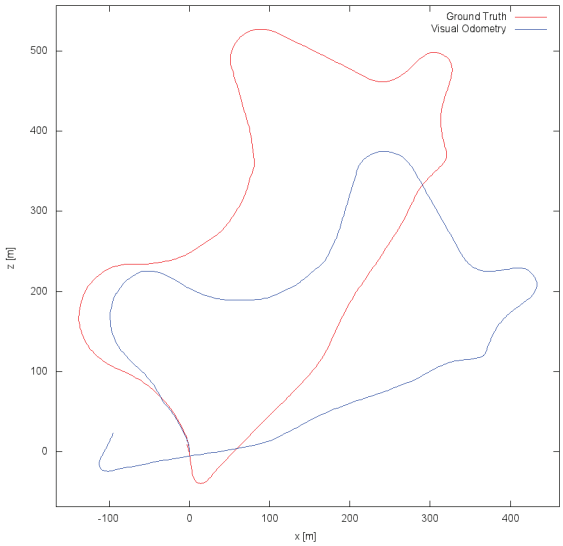


la séquence 6 la séquence 7

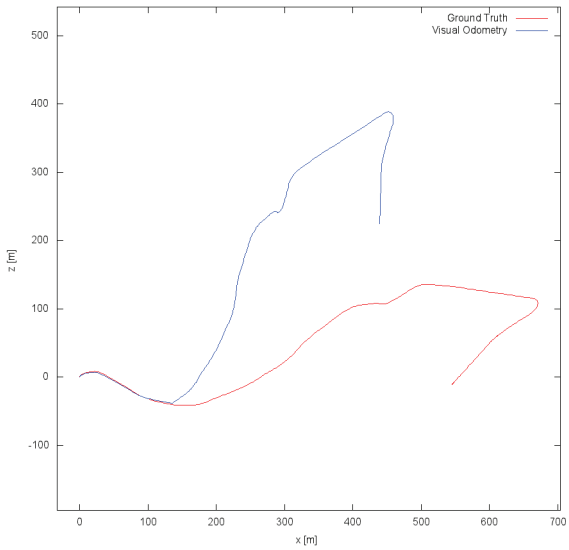
FIGURE 3.10: Les trajectoires des séquences 4 à 7 de la base KITTI. La comparaison des trajectoires du véhicule estimées par la méthode des triangles (les trajectoires en bleu) par rapport à la vérité terrain issus du GPS (les trajectoires en rouge)



la séquence 8



la séquence 9



la séquence 10

(a)

FIGURE 3.11: Les trajectoires des séquences 8 à 10 de la base KITTI. La comparaison des trajectoires du véhicule estimées par la méthode des triangles (les trajectoires en bleu) par rapport à la vérité terrain issus du GPS (les trajectoires en rouge)

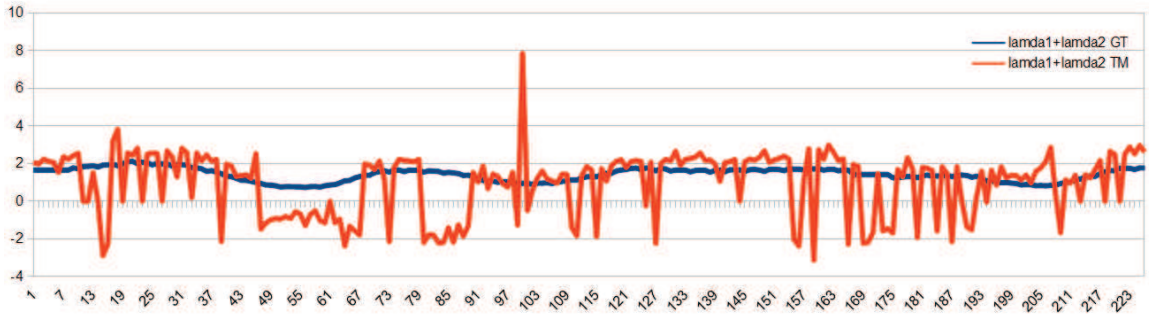


FIGURE 3.12: La distribution brute de $\lambda_1+\lambda_2$ (TM et GT) pour 223 triangles

évaluation des rotations et des translations séparément en fonction de la longueur des trajectoires et la vitesse de conduite.

Les erreurs sont mesurées en pourcentage par mètre pour la translation et en degrés par mètre pour la rotation. Ces erreurs sont mesurées pour toutes les sous-séquences possibles : les sous-séquences de longueur (100, ..., 800) mètres. Les sous-séquences tiennent en compte des séquences les plus longues et offrent une indication de la performance réelle de l'estimation.

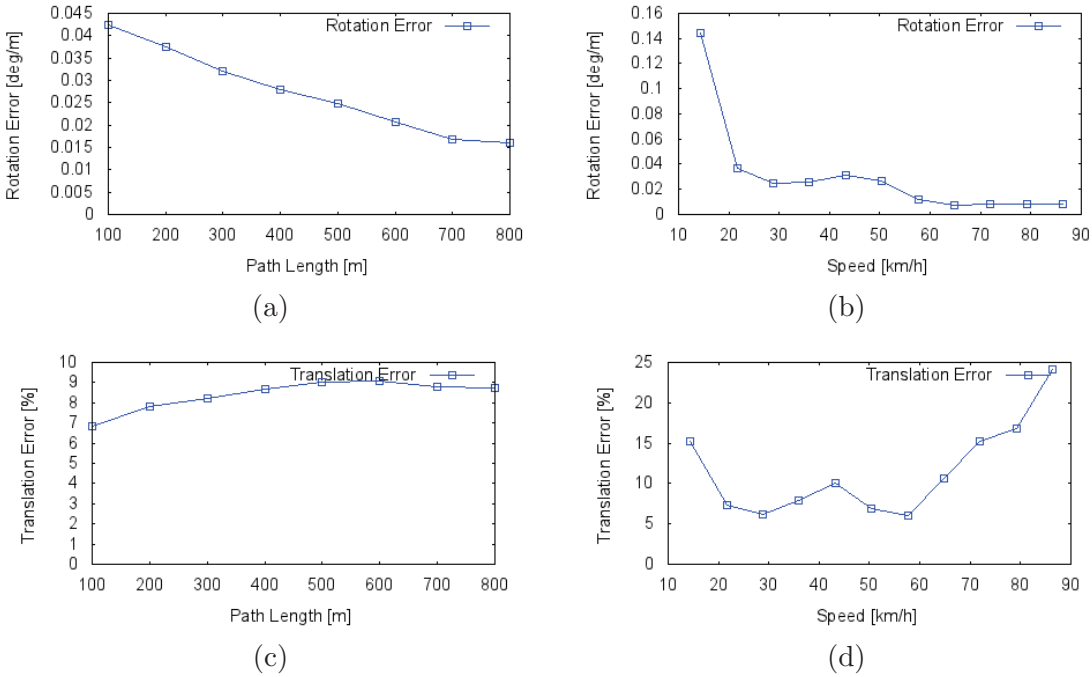


FIGURE 3.13: Évaluation des 10 premières séquences de la base KITTI (la moyenne)

La figure 3.13 montre l'évaluation moyenne des dix séquences en termes des erreurs de translation et de rotation. Nous avons obtenu des erreurs de rotations entre 0.041 et 0.015 degrés par mètres pour des sous-séquences entre 100 et 800 mètres (figure 3.13(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 sous-séquences de 100 et 800 mètres et entre 5 et 22% pour des vitesses entre 5 et 90 kilomètres par heur.

Nous remarquons que les erreurs de rotations sont plus élevées pour les sous-séquences courtes et pour les vitesses lentes. Les allures décroissantes pour les deux courbes des erreurs de rotations sont retrouvées par les courbes obtenues pour d'autres méthodes évaluées sur les 10 dernières séquences de la base de données KITTI, même la meilleure méthode d'odométrie visuelle (une méthode basée sur la stéréo vision) [83]. Il s'agit des 10 séquences fournies sans la vérité terrain. Ces 10 séquences présentent des scènes très similaires aux scènes des 10 premières.

Les erreurs de translations croissent d'à peu près 2% pour les sous-séquences 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 sous séquences. Donc, les 2% d'erreurs en plus représentent les erreurs accumulées en translation. Nous expliquons ces erreurs par les erreurs engendrées par le calcul des facteurs d'échelle. De faibles erreurs dues à l'approximation de la trajectoire linéaires s'accumulent pour les sous-séquences les plus longues.

Les méthodes basées sur les données de la base KITTI (utilisant les images) sont évaluées et classées par les erreurs de rotations et translations. Nous utilisons moins de données que les autres méthodes et nous arrivons à obtenir des résultats meilleures que des méthodes utilisant le réseau de caméras en synchrone (la dernière méthode du classement de l'évaluation sur la base KITTI est à $22,50\%$ d'erreurs de translations).

Les figures 3.14, 3.15, 3.16, 3.17, 3.18, 3.19, 3.20, 3.21, 3.22, 3.23 et 3.24 montrent les évaluations des erreurs de translations et de rotations pour chaque séquence : respectivement de la séquence 0 à la séquence 10.

3.4.2.4 Évaluation de la méthode pour chaque séquence

La séquence 0 est une séquence de 4540 images de scènes urbaines. Les scènes comportent essentiellement des bâtiments, des arbres et des voitures. La distance parcourue dans cette séquence est de 2.232 kilomètres, la distance moyenne entre les prises de vue est de 0.49 mètres et la fréquence d'acquisition des images est de 10 images par seconde. L'erreur moyenne de translation en fonction des sous-séquences est de 5.1%.

Malgré la longueur de la séquence et le fait qu'elle comporte plusieurs virages, notre méthode donne de bons résultats mais la séquence présente une dérive vers la fin. La figure 3.25 présente une portion de 4000 images. La figure montre que la trajectoire

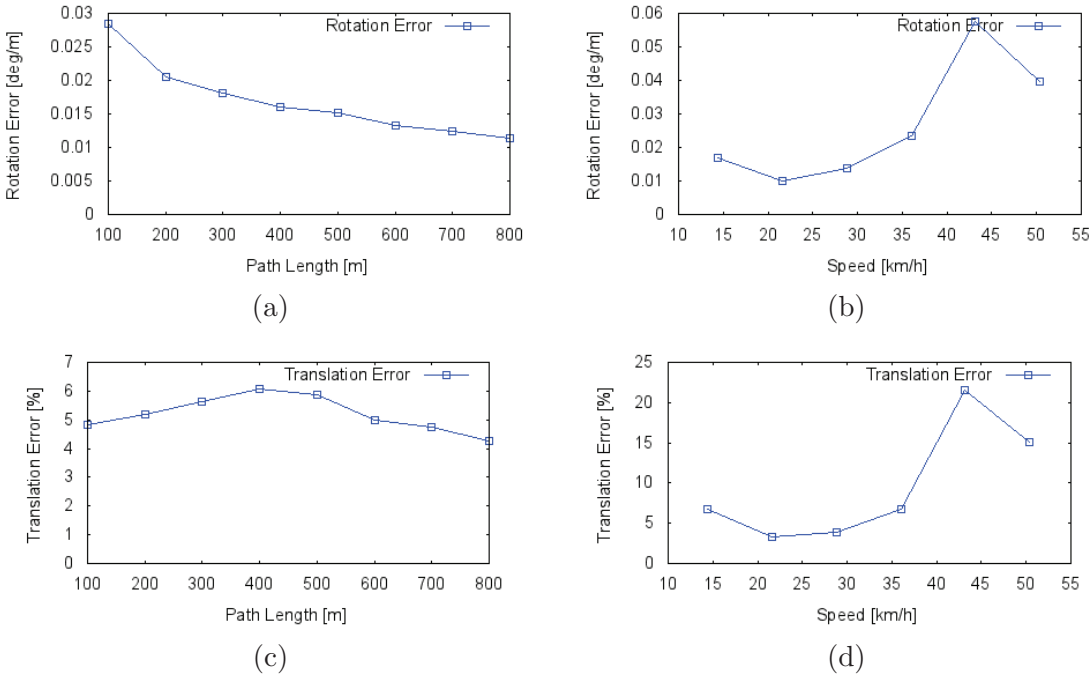


FIGURE 3.14: évaluation de la séquence 0 de la base KITTI

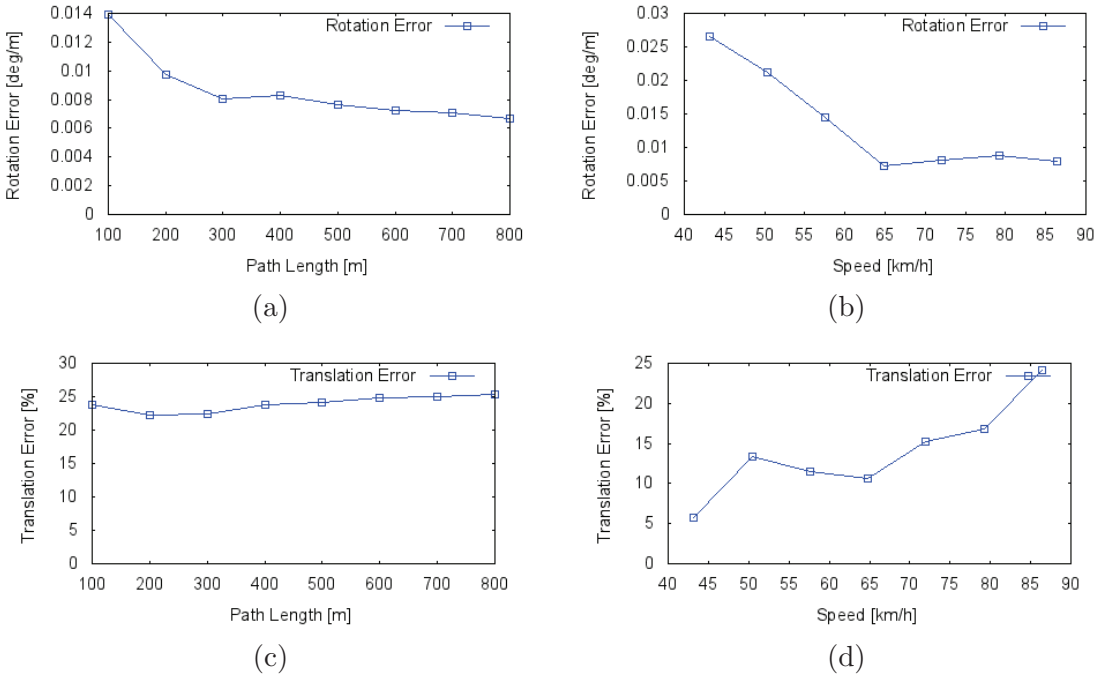


FIGURE 3.15: évaluation de la séquence 1 de la base KITTI

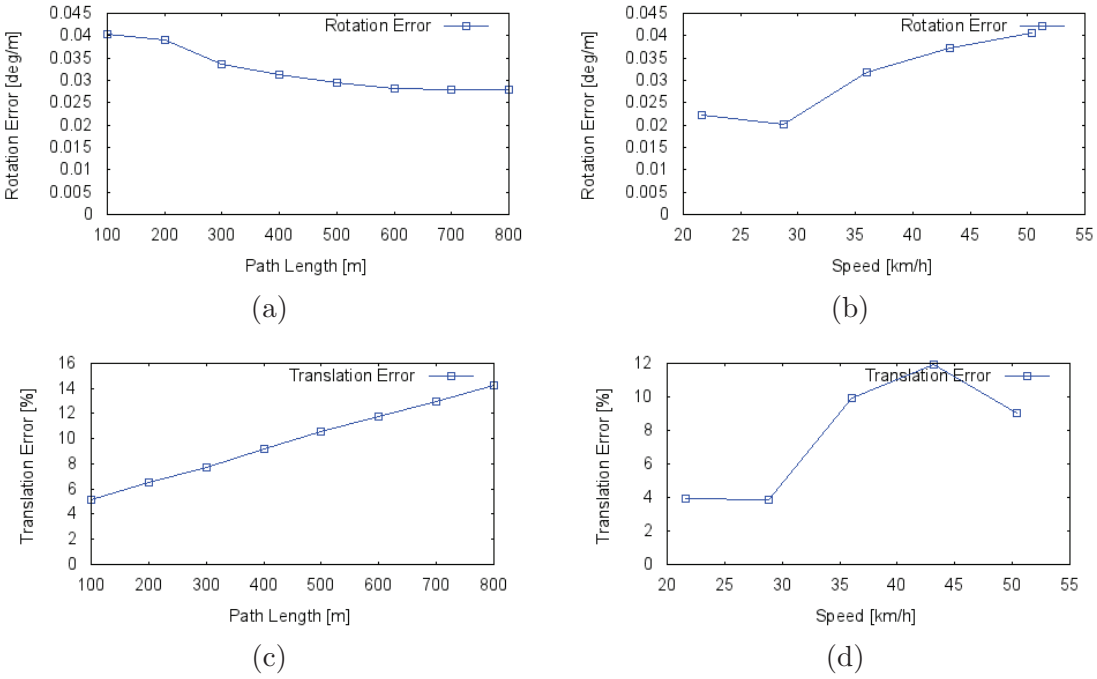


FIGURE 3.16: Évaluation de la séquence 2 de la base KITTI

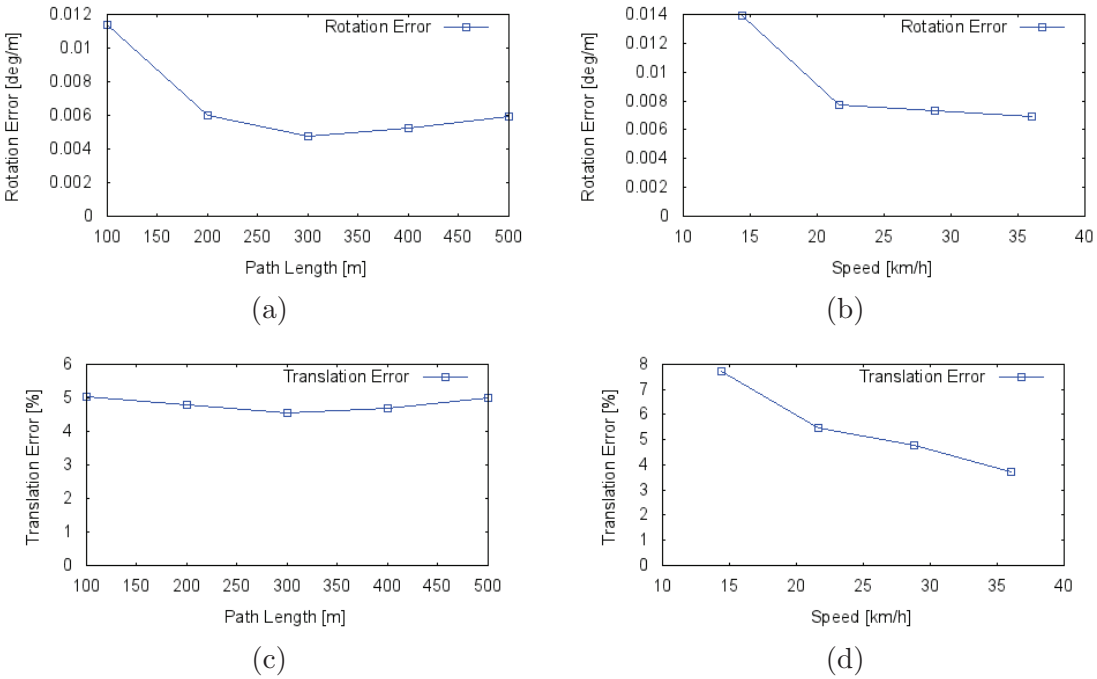


FIGURE 3.17: Évaluation de la séquence 3 de la base KITTI

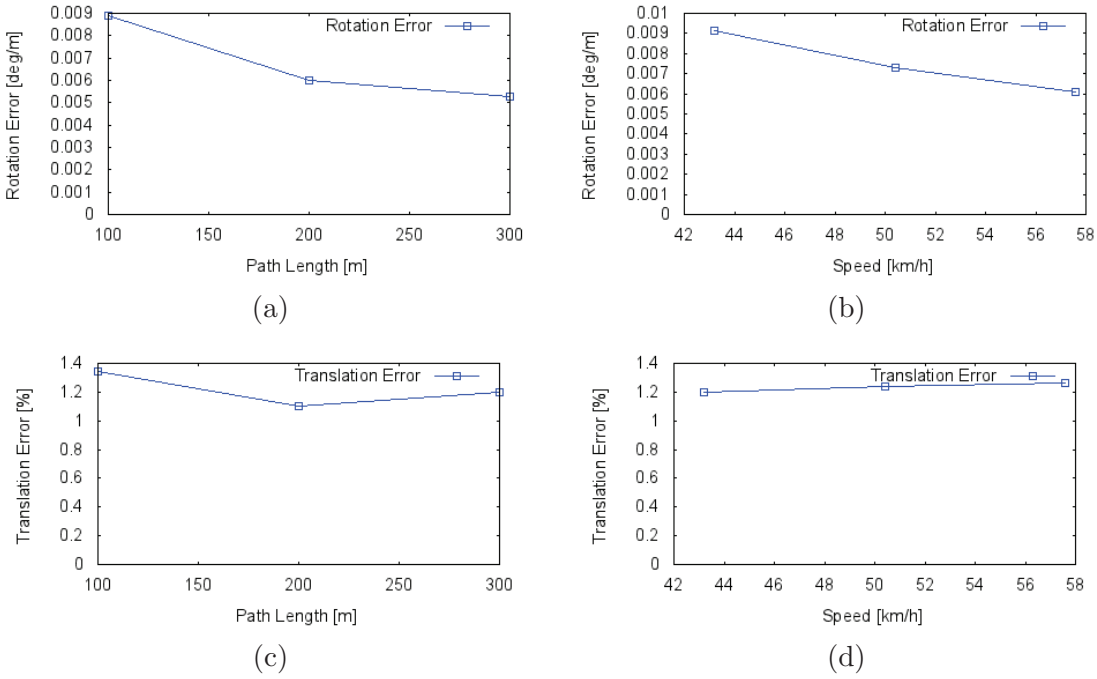


FIGURE 3.18: Évaluation de la séquence 4 de la base KITTI

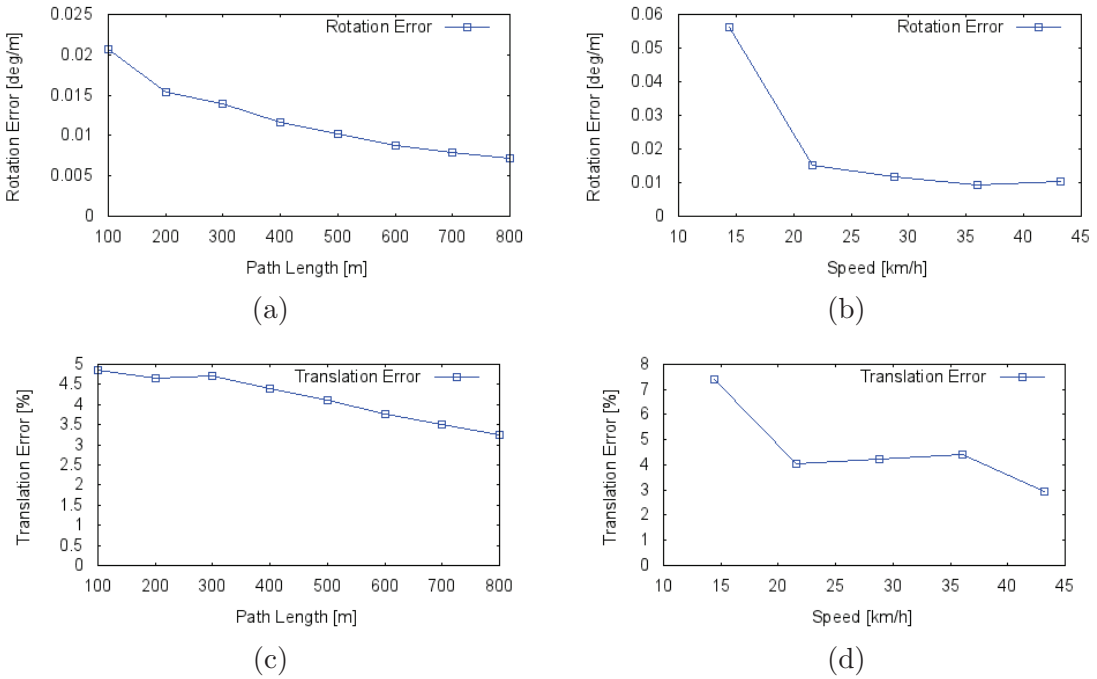


FIGURE 3.19: Évaluation de la séquence 5 de la base KITTI

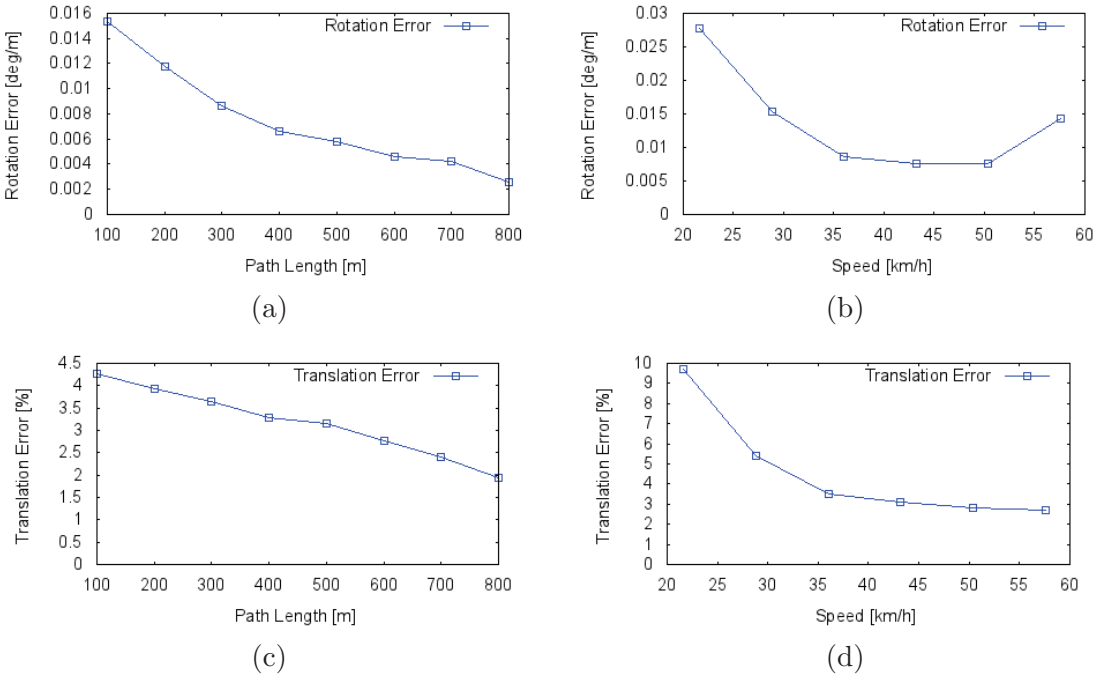


FIGURE 3.20: Évaluation de la séquence 06 de la base KITTI

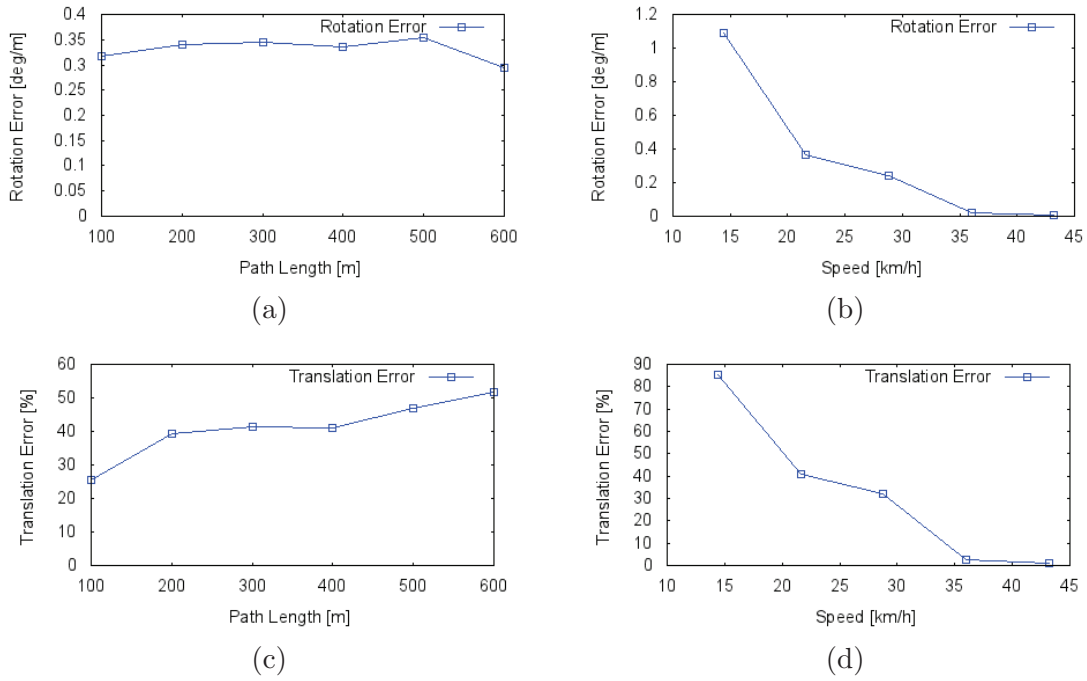


FIGURE 3.21: Évaluation de la séquence 7 de la base KITTI

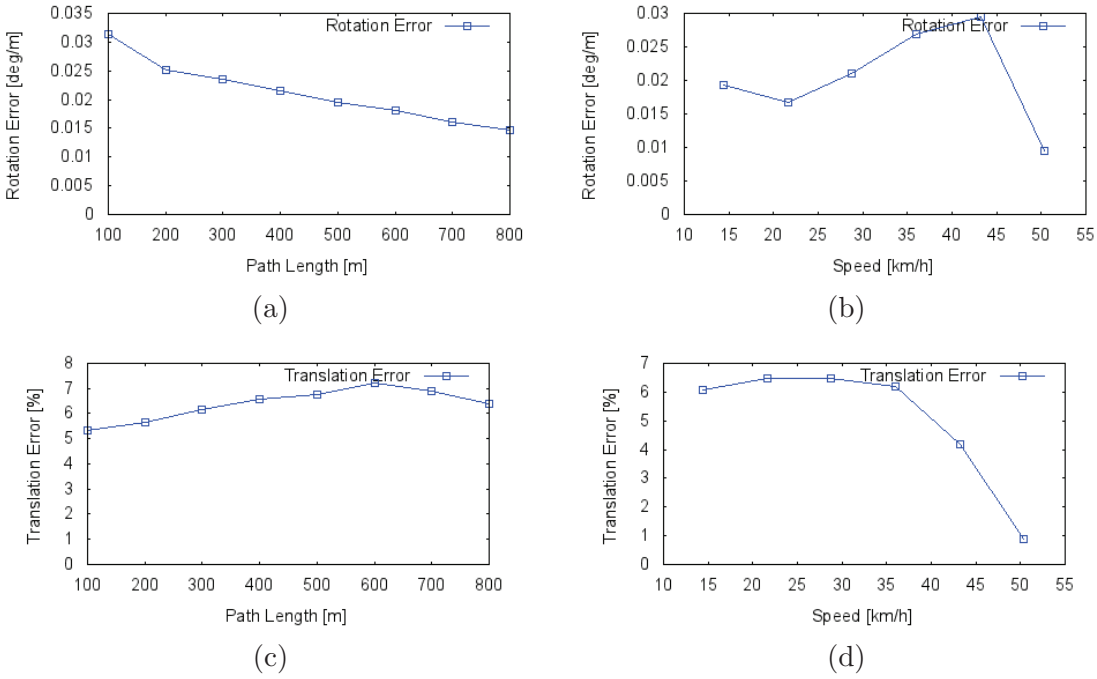


FIGURE 3.22: Évaluation de la séquence 08 de la base KITTI

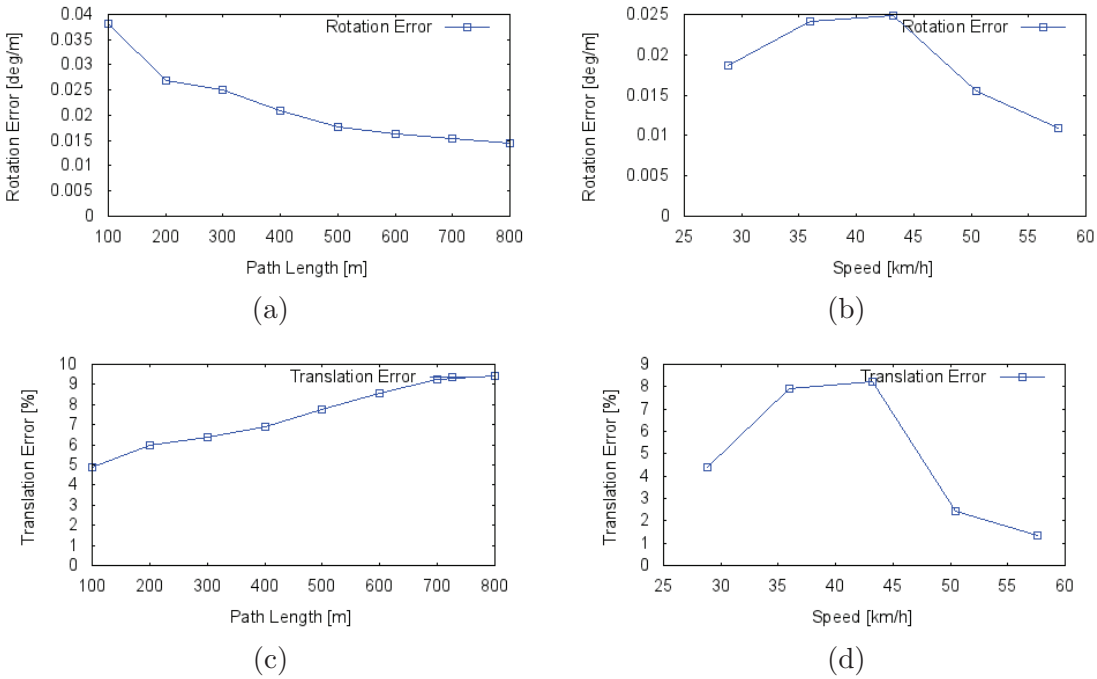


FIGURE 3.23: Évaluation de la séquence 9 de la base KITTI

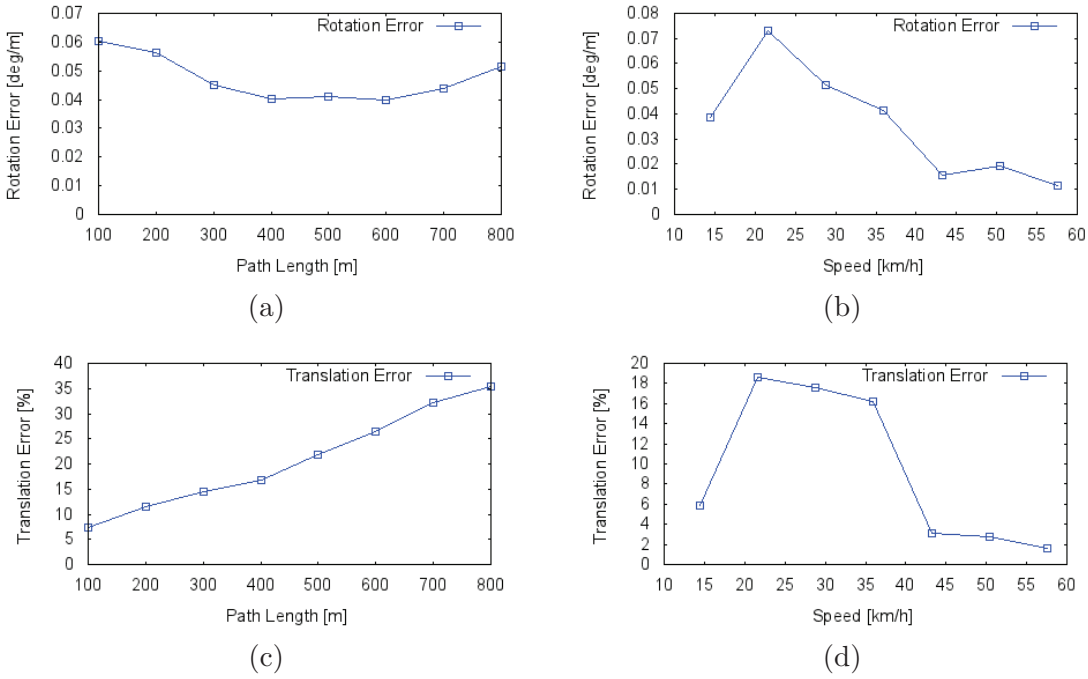


FIGURE 3.24: Évaluation de la séquence 10 de la base KITTI

estimée est proche de la trajectoire issue des données GPS. La dérive est donc issue principalement de l'accumulation des erreurs.

La séquence 1 est une séquence de 1100 images acquises sur une voie rapide de distance de 2.316 kilomètres. Les scènes comportent essentiellement des marquages au sol, de la végétation et quelques voitures. Les scènes sont donc pauvres en textures et par la suite, les points d'intérêt sont moins nombreux que dans la séquence 0 ce qui affecte l'estimation de la trajectoire. Dans quelques portions de la séquence, il y a des voitures sur la voie opposée. La détection de points d'intérêt sur ces véhicules affecte l'estimation.

Malgré une vitesse élevée, notre méthode donne une bonne estimation au début de la trajectoire. Cependant, dés qu'il y a des voitures qui roulent dans le sens inverse l'estimation du vecteur unitaire de translation devient fausse. La figure 3.26(a) présente la première portion de la séquence pour laquelle la trajectoire estimée est proche de la trajectoire réelle. Les figures 3.26(b), (c) et (d) présentent les portions de la trajectoire où il y a des voitures qui roulent dans le sens inverse. Les erreurs des mauvaises estimations sont accumulées.

La séquence 2 est une séquence de 4660 images acquises sur une distance de 3,840 kilomètres. La séquence présente essentiellement des scènes avec de la végétation aux bords des routes. Il y très peu de bâtiments et un faible marquage routier. Quelques voitures sont garées sur les bords et quelques voitures roulent dans le sens inverse. Les scènes sont donc difficiles à traiter par un algorithme qui se base sur les points d'intérêt.

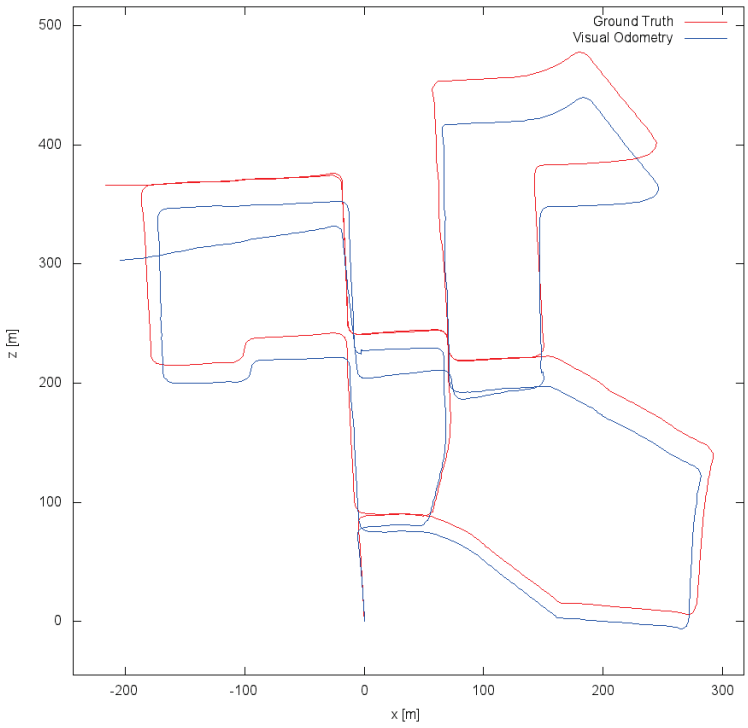


FIGURE 3.25: La trajectoire de la séquence 00 de la base KITTI (les première 3879 images).

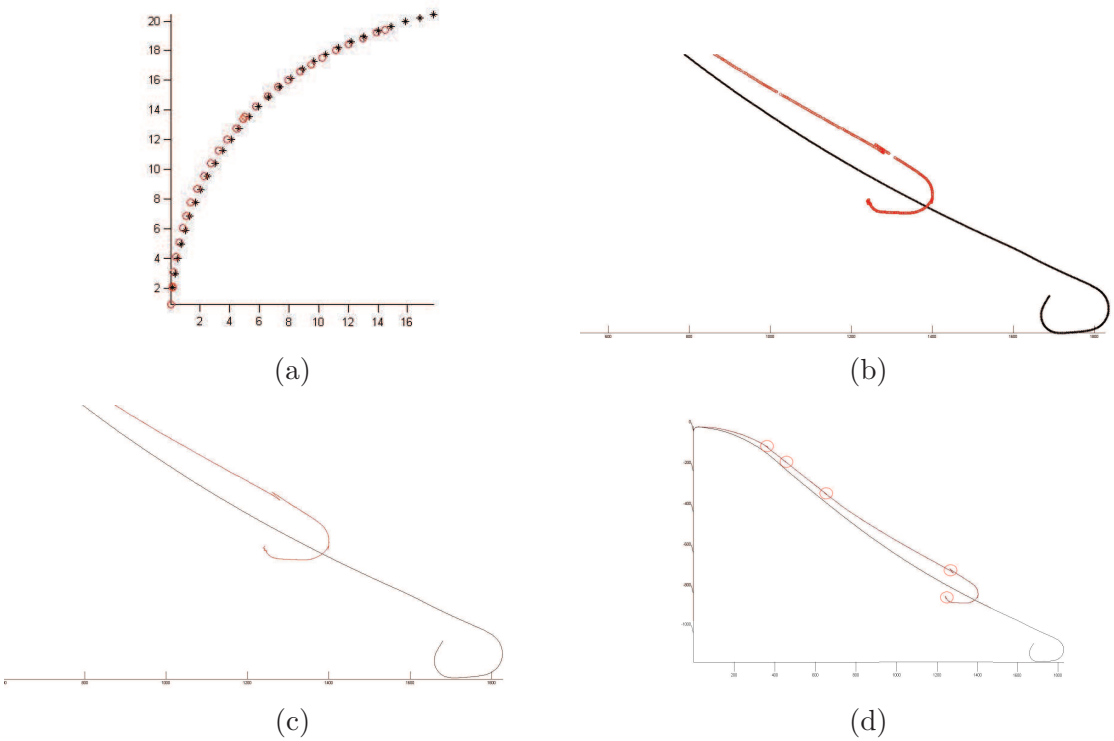


FIGURE 3.26: La trajectoire de la séquences 1 : plus de détails

Le début de la trajectoire présente de bons résultats (figure 3.27) mais les erreurs dues aux mauvaises estimations s'accumulent ce qui explique la trajectoire résultante.

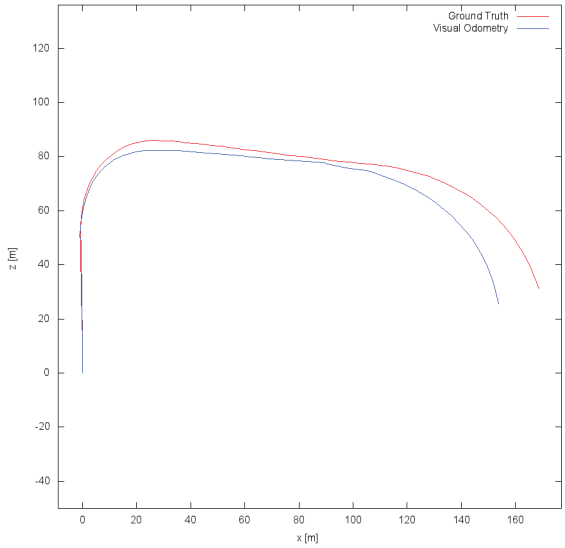


FIGURE 3.27: La trajectoire de la séquence 2 de la base KITTI (les premières 226 images).

La séquence 3 est une séquence de 800 images acquises sur une distance d'environ 200 mètres. La séquence présente des scènes avec beaucoup de végétation et quelques bâtiments. La trajectoire obtenue est proche de la vérité terrain avec des erreurs moyennes de 5% pour les translations et de 0.006 degrés par mètre pour les rotations pour des sous-séquences de 100 à 500 mètres.

La séquence 4 présente les meilleurs résultats de notre méthode. Il s'agit d'une séquence de 270 images acquises sur une distance d'environ 400 mètres. La séquence présente des scènes avec 4 voies. La route contient des marquages au sol (lignes continues, lignes discontinues et des flèches) ainsi que des feux et des carrefours. Malgré la forte végétation dans les scènes, les quelques bâtiments et des voitures qui roulent en sens inverse, les scènes restent très structurées par le marquage. La trajectoire obtenue est très proche de la vérité terrain avec des erreurs moyennes de 1.2% pour les translations et de 0.006 degrés par mètre pour les rotations pour des sous-séquences de 100 à 300 mètres.

La séquence 5 présente des scènes très similaires à la séquence 0 : des bâtiments, des voitures. Les scènes de cette trajectoire sont très riches. Il s'agit d'une séquence de 2760 images acquises sur une distance d'environ 1050 mètres. La trajectoire obtenue est proche de la vérité terrain avec des erreurs moyennes de 4% pour les translations et de 0.01 degrés par mètre pour les rotations pour des sous-séquences de 100 à 800 mètres.

La séquence 6 est une séquence de 1100 images acquises sur une distance d'environ 1.212 kilomètres. Il s'agit de scènes urbaines avec des bâtiments. La trajectoire obtenue est

proche de la vérité terrain avec des erreurs moyennes de 3% pour les translations et de 0.006 degrés par mètre pour les rotations pour des sous-séquences de 100 à 800 mètres.

La séquence 7 est une séquence de 1100 images acquises sur une distance d'environ 438 mètres. Les scènes de cette trajectoire sont urbaines et structurées. La trajectoire obtenue est proche de la vérité terrain avec des erreurs moyennes de 40% pour les translations et de 0.3 degrés par mètre pour les rotations pour des sous-séquences de 100 à 800 mètres. En effet, dans la trajectoire, le véhicule s'arrête à un stop et les voitures de la route principale roulent dans les deux sens. Le passage d'un camion sur la route principale dans le sens inverse à la trajectoire du véhicule (figure 3.28) a engendré une erreur de rotation (dans l'estimation à partir de l'algorithme du 5 point). L'accumulation des poses a propagé cette mauvaise estimation.

La séquence 8 est une séquence de 4070 images acquises sur une distance d'environ 1512 mètres. La plupart des scènes capturées sont des scènes statiques composées de bâtiments, de véhicules stationnés et de végétation. La trajectoire obtenue présente une dérive vers la fin. Les erreurs moyennes sont de 6% pour les translations et de 0.03 degrés par mètre pour les rotations pour des sous-séquences de 100 à 800 mètres. En effet, c'est une longue séquence avec quelques fois des scènes avec beaucoup de végétation. La dérive est due à l'accumulation des erreurs dans les scènes peu structurées.

La séquence 9 est une séquence de 1590 images acquises sur une distance d'environ 1217 mètres. La trajectoire obtenue présente des erreurs moyennes de 6.5% pour les translations et de 0.02 degrés par mètre pour les rotations pour des sous-séquences de 100 à 800 mètres. Le début de la séquence présente des scènes composées principalement de végétation. Les scènes sont donc très peu structurées d'où la déviation obtenue dès le début.

La séquence 10 est une séquence de 1200 images acquises sur une distance d'environ 384 mètres. La trajectoire obtenue présente des erreurs moyennes de 7.5% pour les translations et de 0.025 degrés par mètre pour les rotations pour des sous-séquences de 100 à 800 mètres.

Comme la séquence 9, la séquence 10 présente des scènes très peu structurées avec beaucoup de végétation au début de la trajectoire (figure 3.29) d'où le mauvais départ de la trajectoire estimée. Ensuite, les scènes deviennent plus structurées ce qui explique les allures similaires des trajectoires.

Il est clairement démontré que notre méthode est validée et que les hypothèses initialement posées le sont aussi. Cependant, la comparaison entre les trajectoires montre une dérive surtout dans les virages. Nous expliquons cette dérive par les erreurs d'estimation au niveau des rotations et des translations et au niveau des facteurs d'échelle. En

FIGURE 3.28: Le passage d'un camion sur la route principale dans le sens inverse à la trajectoire du véhicule (images 633, 635, 637, 641 et 644 de la séquence 7)

image 67 de la séquence 9

image 22 de la séquence 10

FIGURE 3.29: Des scènes très peu structurées avec beaucoup de végétations au début des trajectoires 9 et 10

effet, les rotations et les translations sont estimées en se basant sur les appariements 2D entre les images. Ces erreurs peuvent être dues à la distribution des appariements dans les images qui n'est peut-être pas toujours homogène, d'où les erreurs d'estimations. En outre, nous avons trouvé deux erreurs majeures dans deux situations qui sont dues à un faible nombre de caractéristiques appariées, il s'agit des occultations dans les images. Comme nous estimons la pose d'une position courante dans le repère d'une pose précédemment estimée, les erreurs d'estimations s'accumulent.

Conclusion

Dans ce chapitre, nous avons présenté une nouvelle méthode d'estimation de mouvement, appelée la méthode des triangles qui n'impose aucune contrainte sur la synchronisation. Nous supposons la trajectoire entre deux images consécutives provenant d'une même caméra est approximée à un segment linéaire et que les caméras sont calibrées. Notre méthode est basée sur l'hypothèse de linéarité de la trajectoire entre deux images consécutives d'une même caméra. L'algorithme proposé considère trois images, dont deux issues de la même caméra et la troisième d'une caméra voisine. Avec des algorithmes classiques, nous avons estimé les poses relatives entre ces images. L'ensemble des poses et de la structure ne peut être estimé qu'à un facteur d'échelle près. La connaissance de la distance entre les deux caméras (estimée par un étalonnage hors ligne) combinée avec

une hypothèse de mouvement rectiligne du système, nous a permis d'estimer les facteurs d'échelle absolus.

L'approche présente une grande précision au niveau de l'estimation du mouvement pour un déplacement rectiligne et reste acceptable pour les virages. Bien que nous utilisions une méthode robuste, certaines erreurs peuvent se propager en raison des éventuelles erreurs dans l'estimation des facteurs d'échelle. Ces erreurs peuvent être améliorées par des méthodes d'optimisation comme les méthodes d'ajustement de faisceaux.

Chapitre 4

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

Introduction

Les longues séquences d'images représentent un défi important pour l'estimation de mouvement d'un véhicule. En effet, les petites erreurs issues du processus d'estimation sont accumulées itérativement lors de la séquence, ce qui provoque une dérive dans les trajectoires estimées. Ce problème peut être traité par un ajustement de faisceaux (AF). Si il est appliqué pour une séquence entière, cela correspond à un ajustement de faisceaux global, et si il est appliqué localement pour certaines images, ce sera un ajustement de faisceaux fenêtré ou local.

L'AF a un rôle important dans les applications de vision par ordinateur basées sur la reconstruction 3D et le SFM. Il permet d'affiner les paramètres du mouvement et la structure 3D décrivant l'environnement [55]. C'est une étape d'optimisation à la fois des positions 3D et des paramètres des caméras. L'AF consiste à minimiser les erreurs de reprojection entre les points détectés et les points reprojetés dans l'image à partir de la structure 3D.

Bien que la méthode des triangles soit une méthode robuste dans le cas des trajectoires rectilignes, certaines erreurs se produisent dans les virages. Ces erreurs sont engendrées par l'hypothèse de linéarité et réduisent la précision de l'estimation de l'échelle absolue. L'imprécision de l'estimation des facteurs d'échelle entraîne de grandes erreurs sur l'estimation du mouvement et de la structure. Pour améliorer la précision de l'échelle absolue,

nous proposons une optimisation des facteurs d'échelle. Ainsi le système multi-caméras asynchrone devient plus complet et plus précis.

Les principales contributions présentées dans ce chapitre se résument dans la minimisation des erreurs imposées par l'hypothèse de mouvement rectiligne de la méthode des triangles. Pour mettre l'accent sur l'optimisation de l'échelle absolue, nous supposons que les matrices de rotation et les vecteurs de translation estimés initialement par l'algorithme de 5-points [26] sont peu erronés et nous n'optimisons que les facteurs d'échelle et la structure 3D.

La première section de ce chapitre définit le principe d'un ajustement de faisceaux classique. Un positionnement par rapport à l'état de l'art est présenté dans la deuxième section. La méthode présentée dans la section 4.3 détaille la formulation de l'AF avec la nouvelle paramétrisation proposée. Dans la section 4.4, nous présentons et discutons des résultats obtenus sur la base de données KITTI.

4.1 Ajustement de faisceaux

L'ajustement de faisceaux consiste à minimiser l'erreur entre les faisceaux de rayons 3D et 2D. La minimisation des erreurs 2D repose sur la minimisation des erreurs de reprojection mesurées dans les images. L'erreur de reprojection est mesurée entre le point observé dans l'image (initialement détecté par le détecteur de points d'intérêt) et la projection du point 3D dans l'image. La projection d'un point 3D peut être obtenue par une fonction de projection. Nous rappelons la fonction de projection définie dans la section 1.1.2 du chapitre 1 :

$$
\begin{aligned}
F : \mathbb{P}^3 &\longrightarrow \mathbb{P}^2 \\
\mathbf{X}^{monde} \longmapsto \mathbf{x}^{image} &= \mathbf{K} \begin{bmatrix} \mathbf{R} & s \times \mathbf{t} \\ 0^\intercal & 1 \end{bmatrix}^{caméra}_{monde} \mathbf{X}^{monde}
\end{aligned} \tag{4.1}
$$

Dans le cas général, l'AF sert à optimiser les paramètres intrinsèques (la matrice de calibrage K), les paramètres extrinsèques (la rotation et la translation) et les points 3D. La fonction de reprojection est donc une fonction dont le nombre d'inconnues est inférieur au nombre d'équations. Il faut donc résoudre le problème en minimisant l'erreur de reprojection comme un problème d'optimisation de type moindre carrés non linéaire, comme l'algorithme de Levenberg-Marquardt.

4.1.1 ×

Formulation du problème

Pour un ensemble de points 3D $\mathbf{X}$ et de caméras, la matrice de projection d'un point 3D $(\widehat{X}_j)$ dans une caméra i est $(\widehat{P}^i)$.

Soit (x_j^i) le point 2D image du point 3D $(\widehat{X}_j)$ dans le repère de la caméra i. Soient n poses des caméras et m points 3D, la fonction coût correspondant à l'erreur de reprojection à minimiser peut donc s'écrire :

$$\min_{\widehat{\mathbf{X}}_j, \widehat{\mathbf{P}}^i} \sum_{i=1}^{n} \sum_{j=1}^{m} d(\widehat{\mathbf{P}}^i \, \widehat{\mathbf{X}}_j, \, \mathbf{x}_j^i)^2 \tag{4.2}$$

où d correspond à la distance euclidienne définie dans le chapitre 1 (l'équation 1.26). La minimisation de la fonction coût est effectuée par l'algorithme de Levenberg-Marquardt. Cet algorithme consiste essentiellement à calculer la matrice jacobienne et à résoudre le système d'équations de manière itérative. La minimisation de l'erreur de reprojection par cet algorithme revient à résoudre l'équation normale augmentée :

$$(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})\Delta = -\mathbf{J}^T e. \tag{4.3}$$

$\mathbf{J}$ est la matrice jacobienne de la fonction de projection, λ est un réel qui varie d'itération en itération, Δ est le vecteur d'incrément des estimées contenant $6C + 3\mathbf{X}$ valeurs (pour un ensemble de points 3D $\mathbf{X}$ et de caméras C), e est le vecteur des erreurs de reprojection et $\mathbf{I}$ est la matrice identité. La matrice jacobienne est une matrice des dérivées partielles du premier ordre de la fonction de reprojection par rapport aux poses des caméras et des coordonnées des points. La jacobienne de la fonction coût est une matrice creuse, ce qui permet de réduire le temps de calcul.

4.2 Positionnement par rapport à l'état de l'art

L'AF a atteint une certaine maturation et maitrise [53] [55]. L'AF a été largement étudié pour des applications d'odométrie visuelle et des applications de SFM, [84] [85], mais il n'y a pas de travaux qui ont été consacrés aux systèmes asynchrones.

Dans la méthode présentée par Engels et al. [53], un ajustement de faisceaux "fenêtré" (windowed bundle adjustment) a été présenté pour optimiser localement les poses des caméras. Cette méthode a été présentée pour une séquence d'un objet sur un plateau tournant et consiste à optimiser tous les paramètres. Dans une plus large échelle, Mouragnon et al. [52] ont également présenté un ajustement de faisceaux local adapté pour des applications temps réel et pour de longues séquences obtenues à partir de caméras calibrées.

Pour le cas d'AF appliqué à des caméras calibrées, les paramètres intrinsèques sont connus et ne sont donc pas à optimiser. La différence entre notre méthode et les algorithmes classiques d'AF appliqués à des caméras calibrées réside dans le nombre des paramètres à optimiser. D'une part, un algorithme classique d'AF cherche à optimiser 6 paramètres par caméra (3 paramètres par rotation (représentation par les angles par la formulation de Rodrigues) et 3 paramètres par translation) et 3 paramètres par point 3D. D'autre part, notre algorithme ne cherche à optimiser que les facteurs d'échelle et les points 3D et donc 1 paramètre par caméra et 3 paramètres par point 3D.

Dans la méthode présentée par Fraundorfer et al. [60], un ajustement de faisceaux contraint a été présenté pour un problème d'odométrie visuelle à partir d'une seule caméra montée sur un véhicule. La principale différence entre cette méthode et les méthodes d'ajustement de faisceaux classiques est la séparation de l'estimation du mouvement relatif, présentée dans [28], et l'estimation de l'échelle. En effet, les auteurs mettent l'accent sur l'estimation cohérente de l'échelle en n'optimisant que les distances entre les caméras voisines (les facteurs d'échelle relatifs). Les rotations et les directions des translations initialement estimées par la méthode 1-point RANSAC sont considérées fixes et les points 3D sont calculés à chaque itération du processus de l'optimisation. Il s'agit d'un AF global qui optimise toutes les échelles d'une trajectoire.

Les différences entre cette méthode et la méthode d'AF présentée dans ce chapitre sont :

— la configuration : la méthode de Fraundorfer et al. [60] est appliquée à un système monoculaire et notre méthode est appliquée sur un réseau de caméras,
— les auteurs proposent un algorithme basé sur l'hypothèse de mouvement circulaire alors que le notre est basé sur une hypothèse de mouvement rectiligne,
— nous optimisons les facteurs d'échelles et les points 3D simultanément et la méthode proposée par Fraundorfer et al. [60] n'optimise que les facteurs d'échelle. Ils calculent les points 3D à chaque itération avec les nouveaux facteurs d'échelle.

4.3 Optimisation des facteurs d'échelle et de la structure 3D : ajustement de faisceaux local

L'estimation initiale des facteurs d'échelle n'est pas suffisamment précise à cause des contraintes posées par la méthode des triangles surtout pour les trajectoires ayant de fortes courbures et/ou parcourues à grandes vitesses. C'est pourquoi l'estimation initiale ne peut pas être utilisée directement dans les applications de navigation, de détection d'obstacles etc.

Dans cette section, nous proposons de réaliser un AF local, c'est-à-dire sur un nombre de vues limité. Dans notre cas, les caméras sont supposées calibrées, les paramètres intrinsèques du système sont donc connus et ne sont pas à affiner. Les paramètres à optimiser par cet algorithme sont les poses du système et les coordonnées des points 3D.

Pour mettre en avant l'optimisation des facteurs l'échelle estimés par la méthode des triangles, nous supposons que les matrices de rotation et les vecteurs de translation estimés initialement sont fixes et nous n'optimisons que les facteurs d'échelle et la structure 3D.

4.3.1 Formulation

Dans le modèle sténopé, la fonction de projection d'un point 3D X de la scène dans le plan de l'image (un point 2D x) peut être écrite en utilisant une transformation perspective (équation 4.2). La fonctionnelle correspondant à l'erreur de reprojection à minimiser est définie par l'équation 4.2.

La minimisation de l'erreur de reprojection est résolue en utilisant un algorithme non linéaire des moindres carrés tel que Levenberg-Marquardt [86]. Cet algorithme permet d'obtenir des approximations successives d'un vecteur de paramètres P. Dans cette section, P désigne le vecteur de paramètres et non pas la matrice de projection.

Selon l'algorithme de Levenberg-Marquardt (algorithme 3), $\triangle$ est obtenu en résolvant l'équation normale augmentée (l'équation 4.3) à chaque itération. La résolution de cette équation nécessite le calcul de la jacobienne J. Les erreurs de reprojection sont calculées et évaluées à la fois pour P_i et P_{i+1}.

4.3.2 Calcul de la jacobienne

Dans notre méthode, la matrice jacobienne J est calculée en dérivant la fonction de projection par le facteur d'échelle s et les points 3D $\mathbf{X}$ seulement. En effet, la différenciation

Algorithm 3 Pseudo-code de l'algorithme de Levenberg-Marquardt

$i \leftarrow 0$
$\lambda \leftarrow 0.001$
calcul de $\|e(P_0)\|$
while $i <$ MAX_ITERATIONS et $\|e(P_i)\| >$ seuil **do**
  résolution de l'équation normale augmentée :
  $(J^T J + \lambda I)\triangle = -J^T e$
  évaluation du vecteur de paramètres $P_{i+1} = P_i + \triangle$
  if $\|e(P_{i+1})\| \geq \|e(P_i)\|$ **then**
    $\lambda \leftarrow 10\lambda$
  else
    $\lambda \leftarrow \lambda / 10$, $P_{i+1} = P_i + \triangle$
  end if
  $i \leftarrow i + 1$
end while

symbolique de la fonction de projection F est effectuée par rapport à s et aux coordonnées du point 3D $\mathbf{X}$. En différenciant la fonction de projection par rapport à $\mathbf{X}$, nous obtenons la matrice **JX** (matrice 2x3) comme dans l'équation 4.4. Lorsque nous dérivons la fonction de projection par rapport aux facteurs d'échelle, nous obtenons la matrice **JS** (matrice 2x3).

$$\mathbf{JX} = [\frac{\partial F}{\partial \mathbf{X}}] \text{ et } \mathbf{JS} = [\frac{\partial F}{\partial s}]. \tag{4.4}$$

Pour chaque point 3D, nous calculons les matrices JX et JS pour toutes les caméras de la fenêtre glissante. La jacobienne J aura une structure éparse (figure 4.1). Si l'on considère n caméras et m points 3D, la jacobienne sera une matrice $(2 \times n \times m) \times (1 \times n + 3 \times m)$.

4.3.2.1 Calcul mathématique détaillé :

La fonction de projection F (l'équation 4.1) peut être écrite comme dans l'équation 4.5.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r11 & r12 & r13 & s \times t1 \\ r21 & r22 & r23 & s \times t2 \\ r31 & r32 & r33 & s \times t3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4.5}$$

En développant l'équation 4.5, la fonction de projection devient :

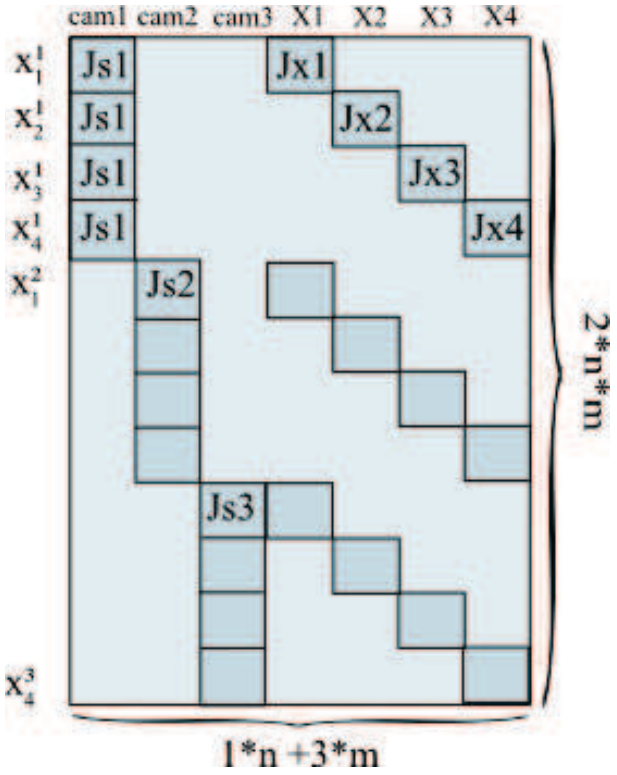


FIGURE 4.1: Structure d'une matrice jacobienne pour une fenêtre composée de trois caméras et 4 points 3D. Le bleu foncé désigne les paramètres des caméras **JS** et les paramètres des points 3D **JX** et le bleu clair représente les éléments nuls.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{X\times(fx\times r11+r31\times u0)+Y\times(fx\times r12+r32\times u0)+Z\times(fx\times r13+r33\times u0)+fx\times s\times t1+s\times t3\times u0}{X\times r31+Y\times r32+Z\times r33+s\times t3} \\ \frac{X\times(fx\times r11+r31\times u0)+Y\times(fx\times r12+r32\times u0)+Z\times(fx\times r13+r33\times u0)+fx\times s\times t1+s\times t3\times u0}{X\times r31+Y\times r32+Z\times r33+s\times t3} \\ 1 \end{bmatrix} \tag{4.6}$$

La matrice **JX** est calculée en dérivant la fonction de projection par rapport à **X**, la dérivée de l'équation 4.6 par rapport à $\mathbf{X} = (X, Y, Z)^t$ est définie par l'équation 4.7.

$$\mathbf{JX} = \begin{bmatrix} \mathbf{A} & \mathbf{B} & \mathbf{C} \\ \mathbf{D} & \mathbf{E} & \mathbf{F} \end{bmatrix} \tag{4.7}$$

Les éléments de la matrice **JX** sont définis par les équations 4.8. La matrice **JS** est obtenue en dérivant la fonction de projection par rapport au facteur d'échelle s, la dérivée de l'équation 4.6 par rapport à s est définie par l'équation 4.9.

$$
\begin{aligned}
\mathbf{A} &= \frac{(fx \times r11 + r31 \times u_0)}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} - \frac{(r31 \times (X \times (fx \times r11 + r31 \times u_0) + Y \times (fx \times r12 + r32 \times u0) + Z \times (fx \times r13 + r33 \times u_0) + fx \times s \times t1 + s \times t3 \times u0))}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} \\
\mathbf{B} &= \frac{(fx \times r12 + r32 \times u_0)}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} - \frac{(r32 \times (X \times (fx \times r11 + r31 \times u_0) + Y \times (fx \times r12 + r32 \times u0) + Z \times (fx \times r13 + r33 \times u_0) + fx \times s \times t1 + s \times t3 \times u0))}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)^2} \\
\mathbf{C} &= \frac{(fx \times r13 + r33 \times u_0)}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} - \frac{(r33 \times (X \times (fx \times r11 + r31 \times u_0) + Y \times (fx \times r12 + r32 \times u0) + Z \times (fx \times r13 + r33 \times u_0) + fx \times s \times t1 + s \times t3 \times u0))}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)^2} \\
\mathbf{D} &= \frac{(fy \times r21 + r31 \times v0)}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} - \frac{(r31 \times (X \times (fy \times r21 + r31 \times v_0) + Y \times (fy \times r22 + r32 \times v_0) + Z \times (fy \times r23 + r33 \times v_0) + fy \times s \times t2 + s \times t3 \times v_0))}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)^2} \\
\mathbf{E} &= \frac{(fy \times r22 + r32 \times v_0)}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} - \frac{(r32 \times (X \times (fy \times r21 + r31 \times v_0) + Y \times (fy \times r22 + r32 \times v_0) + Z \times (fy \times r23 + r33 \times v_0) + fy \times s \times t2 + s \times t3 \times v_0))}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)^2} \\
\mathbf{F} &= \frac{(fy \times r23 + r33 \times v_0)}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} - \frac{(r33 \times (X \times (fy \times r21 + r31 \times v0) + Y \times (fy \times r22 + r32 \times v_0) + Z \times (fy \times r23 + r33 \times v_0) + fy \times s \times t2 + s \times t3 \times v_0))}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)^2}
\end{aligned}
\tag{4.8}
$$

$$
\mathbf{JS} = \begin{bmatrix} \frac{(fx \times t1 + t3 \times u0)}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} - \frac{(t3 \times (X \times (fx \times r11 + r31 \times u0) + Y \times (fx \times r12 + r32 \times u0) + Z \times (fx \times r13 + r33 \times u0) + fx \times s \times t1 + s \times t3 \times u0))}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)^2} \\ \frac{(fy \times t2 + t3 \times v0)}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)} - \frac{(t3 \times (X \times (fy \times r21 + r31 \times v0) + Y \times (fy \times r22 + r32 \times v0) + Z \times (fy \times r23 + r33 \times v0) + fy \times s \times t2 + s \times t3 \times v0))}{(X \times r31 + Y \times r32 + Z \times r33 + s \times t3)^2} \end{bmatrix}. \tag{4.9}
$$

4.4 Résultats

Dans cette section, nous présentons les résultats de la méthode d'optimisation des facteurs d'échelle décrite dans la section précédente. 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 dans le chapitre 3. Nous avons choisi de tester l'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). Dans un deuxième temps nous appliquons l'algorithme sur l'estimation de la méthode des triangles.

4.4.1 Résultats d'un ajustement de faisceaux « optimal »

Pour l'évaluation quantitative de notre algorithme, nous appliquons l'AF en utilisant les poses de la vérité terrain (VT). Pour une fenêtre glissante de deux triangles, nous utilisons les matrices de rotation, les vecteurs unitaires de translations et les facteurs d'échelle obtenus par le GPS comme entrées de l'AF. Les points 3D sont calculés à l'échelle en utilisant les poses obtenues par le GPS puis nous appliquons l'étape de l'AF.

Ce test, que nous appelons un ajustement de faisceaux optimal, nous servira comme référence pour évaluer les facteurs d'échelle estimés après l'application de l'AF aux données estimées (dans la section suivante).

Pour évaluer les facteurs d'échelle de l'AF optimal, nous calculons le rapport du facteur d'échelle évalué par le facteur d'échelle de la VT, avant et après l'AF (les ratios avant et après l'AF) comme dans les équations 4.10 et 4.11. Les facteurs d'échelle sont les normes des vecteurs de translations. Nous obtenons quasiment les mêmes données de départ avec de très faibles erreurs que nous jugeons acceptables (des rapports quasiment égaux à 1). En effet, l'AF est appliqué sur des poses parfaites et des points 3D estimés. Comme les points 3D sont calculés par triangulation des points 2D mis en correspondance, les points 3D sont susceptibles d'être affectés par des erreurs dues aux imprécisions des processus de l'extraction, de l'appariement et de triangulation.

$$\text{ratios avant} = \frac{\text{facteur d'échelle évalué avant AF}}{\text{facteur d'échelle de la VT}} \tag{4.10}$$

$$\text{ratios après } = \frac{\text{facteur d'échelle évalué après AF}}{\text{facteur d'échelle après l'AF optimale}} \tag{4.11}$$

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 le tout soit optimisé. Les facteurs d'échelle évalués avant l'AF sont donc différents des facteurs d'échelle de la VT.

Les résultats obtenus sont très satisfaisants. La trajectoire obtenue après l'AF est presque celle du départ (la trajectoire obtenue par le GPS). Les résultats des ratios avant et après l'AF sont présentés dans le tableau 4.1. 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, de même pour les échelles 3 et 4. Les ratios calculés sont proches de 1. Nous jugeons donc que notre algorithme donne des résultats très précis.

TABLE 4.1: Ratios avant et après l'AF pour la VT bruitée par un bruit gaussian ($\sigma = 0.01$)

	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

En résumé, les erreurs obtenues sont très faibles et dues à plusieurs raisons : l'imprécision des détecteurs et des descripteurs des points d'intérêt (surtout que les détecteurs, les descripteurs ne donnent pas d'informations géométriques), 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 avec des poses bruitées, 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 (VT), quelques erreurs restent encore présentes sur la structure, c'est pourquoi nous comparons les résultats obtenus par la méthode des triangles par rapport à un AF « optimal » afin de quantifier les résultats sans prendre en compte les erreurs dues aux processus de triangulation et d'extraction des points d'intérêt.

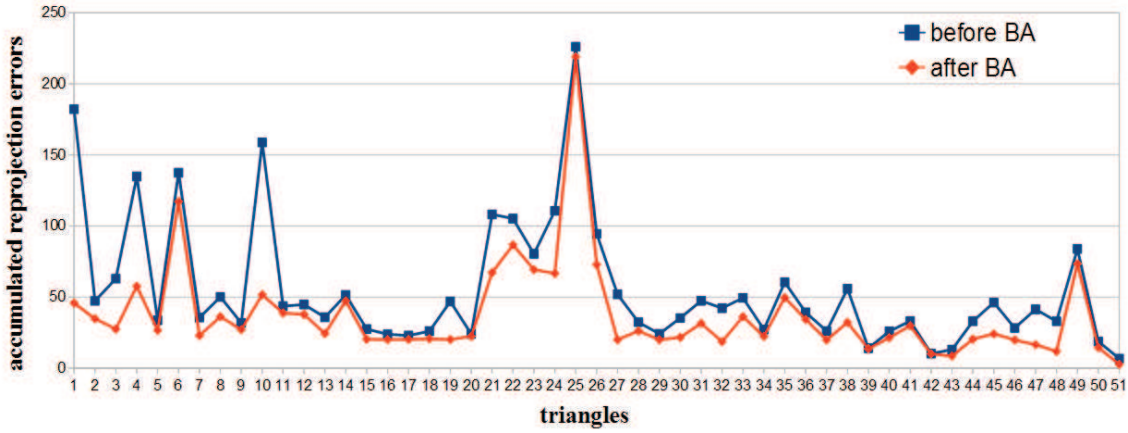


FIGURE 4.2: Les erreurs de reprojection pour 52 triangles avant et après un AF appliqué sur les données estimées par la méthode des triangles

4.4.2 Résultats de la méthode des triangles

Dans cette section, nous présentons les résultats de l'AF sur l'estimation de la méthode des triangles.

La figure 4.2 montre la distribution des erreurs de reprojection accumulées avant et après l'AF pour une séquence de 200 images. Les erreurs de reprojection ont considérablement baissé après l'AF. La figure 4.3 illustre les erreurs de reprojection des points 3D dans cinq caméras d'une fenêtre glissante. Les erreurs de reprojection sont présentées dans une image de la fenêtre par des couleurs différentes pour chaque caméra.

Nous calculons les ratios de la même façon que les équations 4.10 et 4.11 avec les facteurs d'échelle estimés par la méthode des triangles avant et après l'AF. Les résultats sont résumés dans le tableau 4.2. Les trajectoires obtenues sont présentées dans la figure 4.5. La figure montre que la trajectoire optimisée par la méthode proposée est plus proche de la VT que la trajectoire initialement estimée par la méthode des triangles (avant l'AF).

TABLE 4.2: Ratios 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

(a) erreurs de reprojection des points 3D avant AF

(b) erreurs de reprojection des points 3D après AF

FIGURE 4.3: Exemple d'erreurs de reprojection des points 3D avant et après AF, chaque couleur fait référence aux erreurs de reprojection dans une caméra de la fenêtre glissante (5 caméras)

Conclusion

Dans ce chapitre, nous avons présenté une étape d'optimisation par ajustement de faisceaux. L'approche présentée dans le chapitre précédent suppose que la trajectoire entre deux images consécutives issues d'une même caméra est rectiligne et nécessite une connaissance de la calibration géométrique effectuée hors-ligne. L'hypothèse de linéarité provoque de petites erreurs en particulier pour les virages. Nous avons appliqué un AF sur les facteurs d'échelle absolue et sur la structure 3D pour affiner l'estimation initiale. Les résultats de l'approche présentée améliorent la précision et la robustesse de l'estimation des facteurs d'échelle absolue. Cela permet de reconstruire le déplacement d'un véhicule à l'échelle réelle.

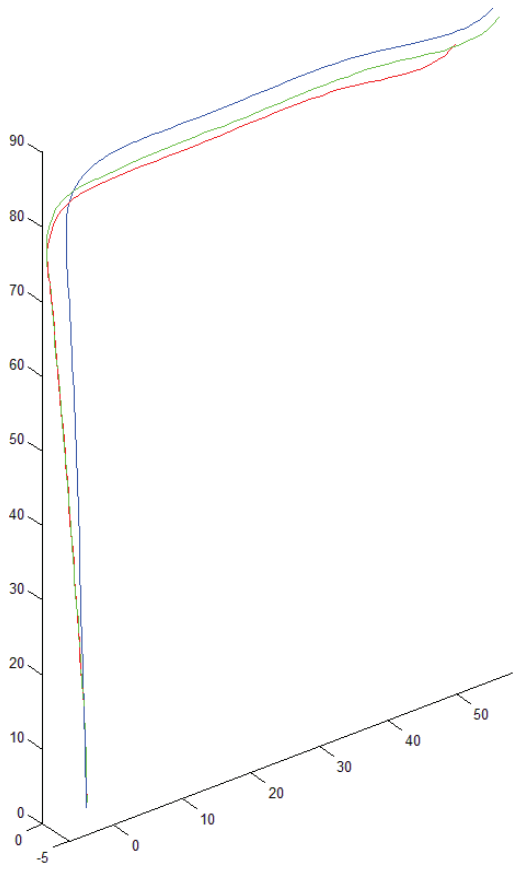


FIGURE 4.4: Trajectoires obtenues à partir de 200 images : l'estimation en rouge, la VT en bleu, l'AF en vert

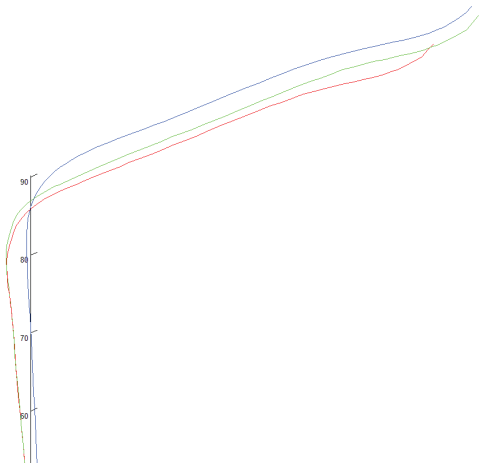


FIGURE 4.5: Zoom sur les trajectoires

Chapitre 5

Système multi-caméras non synchronisées : estimation de mouvement et application

Introduction

Dans les chapitres 3 et 4 nous avons présenté une méthode d'estimation du mouvement et de la structure à partir de deux caméras non synchronisées. Dans ce chapitre, nous introduisons une formulation à base de graphe afin de généraliser la méthode des triangles à N caméras et une application de détection d'obstacles à partir d'images non synchronisées.

La méthode de triangles peut être appliquée à N caméras par le biais de la modélisation du réseau de caméras par un graphe. Les triangles sont identifiés dans la représentation à base de graphe et les positions des caméras sont estimées en appliquant la méthode des triangles pour chaque triangle.

La première section de ce chapitre décrit la formulation à base de graphe pour la généralisation de la méthode des triangles à N caméras. La première partie de cette section présente un état de l'art des travaux antérieurs qui portent sur l'estimation de mouvement basée sur les graphes, puis la seconde partie présente la méthode et la troisième partie présente les résultats expérimentaux.

La deuxième section de ce chapitre introduit une application réalisée à partir de caméras non synchronisées : la détection de route et d'obstacles. La première partie de cette section présente un état de l'art des travaux antérieurs qui portent sur la détection

d'obstacles. La seconde partie présente la méthode et la troisième partie présente les résultats expérimentaux.

5.1 Estimation du mouvement à partir d'un réseau de caméras non synchronisées

5.1.1 Travaux antérieurs

En robotique, les algorithmes de SLAM qui reposent sur les représentations à base de graphe sont des algorithmes d'unification des problèmes de SLAM hors ligne. Ces algorithmes se basent sur les contraintes de fermetures de boucles pour avoir une meilleure carte de navigation. Ces techniques sont connues sous le nom du SLAM back-ends [3] [87]. En utilisant une telle formulation, les problèmes du SLAM sont modélisés par un graphe dont les nœuds représentent les poses du véhicule et dont les arcs représentent les contraintes entre ces poses.

Le premier algorithme de SLAM basé sur les graphes a été introduit par Lu et Milios en 1997 [46]. Récemment, le SLAM à base de graphe est devenu populaire dans la communauté robotique. La formulation à base de graphe est robuste pour les problèmes du SLAM hors ligne (back-ends) et du SLAM en ligne (front-ends).

Un problème d'optimisation de graphe de poses peut être formulé par des réseaux bayésiens dynamiques, des graphes à facteurs (factor graphs) et de champs de Markov aléatoires [3]. L'objectif est d'estimer la trajectoire d'un robot à partir des poses relatives qui peuvent être obtenues à partir de l'odométrie des roues, de l'odométrie visuelle ou de l'appariement des primitives.

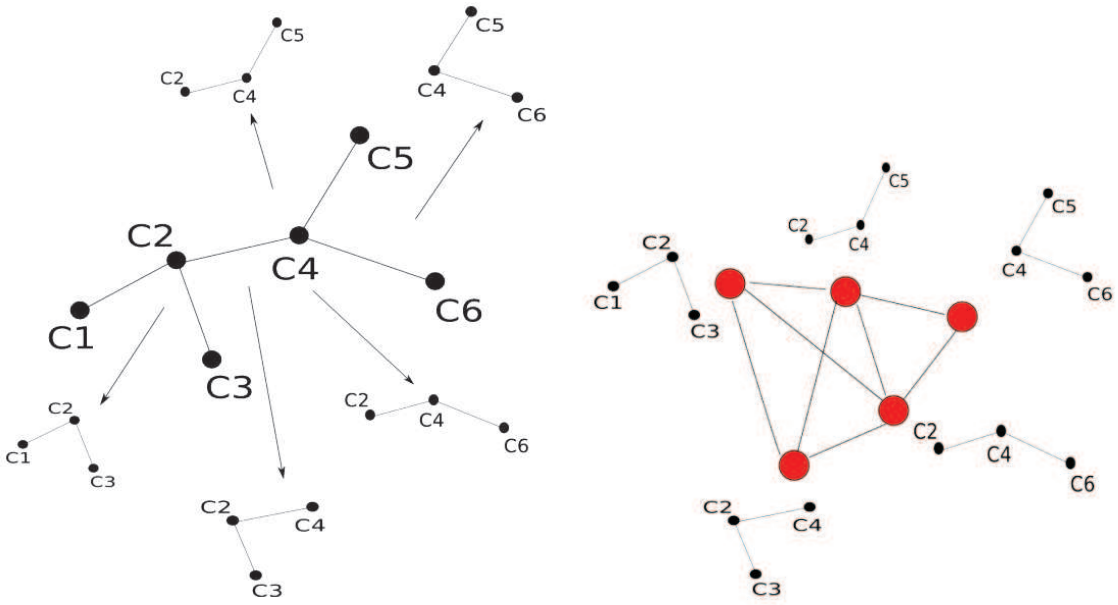
Les approches basées sur les graphes de poses nécessitent des contraintes de fermeture de boucles et des contraintes de localisations par rapport à des amers. Cependant, pour les applications d'estimation du mouvement d'un véhicule dans un environnement inconnu, la contrainte de fermeture de boucle n'est pas toujours valide.

Ces approches ont été également étudiées pour les applications d'odométrie visuelle et de SFM. Plusieurs techniques basées sur des triplets d'images en SfM ont été abordées pour la reconstruction de la structure à partir de séquences d'images [88] et pour des problématiques d'agrégation d'images non ordonnées (la collection d'images non ordonnées) [89][90].

Une approche basée sur des triplets d'images dont les problématiques se rapprochent des nôtres a été présentée par Klopschitz et al. [11]. Klopschitz et al. utilise une formulation à

base de graphe pour résoudre des problèmatiques de SFM de façon incrémentale à partir d'ensembles non ordonnées d'images. Les auteurs considèrent des triplets de prises de vues (d'images) comme l'élément de base du SfM en se basant sur le tenseur trifocal.

La première étape de cette approche [11] consiste à vérifier les appariements par la géométrie épipolaire pour définir la relation géométrique entre les images comme des arcs du graphe (les images représentent les nœuds du graphe). La deuxième étape consiste à transformer le graphe obtenu de l'étape une en un graphe basé sur des triplets, des reconstructions tri-focales. La suppression des valeurs aberrantes (outliers) est faite au moment de la vérification des triplets par des fermetures de boucle implicites. Chaque nœud de ce nouveau graphe représente un triplet et les arcs représentent les correspondances entre eux. En effet, lorsque les triplets partagent au moins une prise de vue et réussissent un test de compatibilité de points 3D, des arcs sont ajoutés au graphe de triplets. La troisième étape consiste à fusionner les arcs du graphe des triplets de manière incrémentale dans la reconstruction.



(a) énumération des triplets (b) graphe de triplets

FIGURE 5.1: Modélisation sous forme de graphe basée sur des triplets d'images. Les illustrations proviennent de [11]

Cette approche est destinée aux problèmes d'agrégation d'images non ordonnées résolues par SFM à l'échelle près. Certes, les applications d'agrégation d'images non ordonnées se rapprochent beaucoup des problèmes asynchrones car la collecte d'images provient de caméras différentes. Mais, cette approche ne convient pas à notre application. La différence entre cette méthode est la nôtre réside essentiellement dans les objectifs à atteindre. En effet, nous cherchons à avoir une reconstruction à l'échelle absolue pour des applications d'aide à la conduite. Cependant, l'approche proposée par [11] effectue une

reconstruction à l'échelle près. De plus, le choix du triplet d'images est différent entre notre méthode et celle proposée par Klopschitz et al. Leur méthode estime la reconstruction à partir de chaque triplet pertinent (une reconstruction tri-focale) à l'échelle prés alors que notre méthode estime le mouvement et la structure à l'échelle à partir de 3 images qui satisfont certaines hypothèses (la linéarité du mouvement et la rigidité entre les caméras) pour en tirer les facteurs d'échelles.

Les techniques basées sur les graphes ont été étudiées essentiellement pour des applications de SLAM et un peu moins pour l'odométrie visuelle et le SFM. Ces techniques ne sont pas adaptées pour le cas de l'odométrie visuelle non synchronisée. En effet, ces travaux se basent sur les contraintes de positionnement par rapport à des amers et de fermeture de boucles. En outre, la plupart des techniques basées sur les graphes sont effectuées hors-ligne ce qui ne convient pas au type d'applications présenté dans cette thèse.

5.1.2 Modélisation du réseau multi-caméras

Inspiré par le système multi-caméras proposé par Meilland dans [7], la configuration du réseau multi-caméras peut être réalisée sur une plate-forme mobile de façon que la scène soit vue par deux ou plusieurs caméras. Les caméras voisines partagent un champ de vision commun afin d'avoir une vue complète tout autour d'un véhicule. Dans la configuration de notre prototype, nous avons choisit de disposer les caméras de façon à recouvrir le champ de vue frontal ou devant les caméras.

Le modèle proposé est flexible. Le nombre des caméras peut être choisi facilement sans dépendre de la méthode. Avec de caméras grand angle, le système peut être embarqué sur une voiture tout en tenant compte de l'hypothèse de chevauchement entre les caméras voisines. La figure 5.2 illustre un exemple d'une configuration multi-caméras qui permet de couvrir le champ de vue frontal du véhicule. Les caméras sont montées sur un système rigide ayant des champs de vue qui s'intersectent deux à deux.

En utilisant cette configuration, la distribution d'images à partir d'un réseau asynchrone est présentée par la figure 5.3. La figure modélise des images non synchronisées à partir de cinq caméras rigidement liées par un support comme le montre la figure 5.2 (l'image de droite). Les caméras en bleu représentent les caméras qui acquièrent des images et les caméras en blanc représentent les poses virtuelles. Nous avons présenté un système dont l'acquisition est régulière (ordonnée) car c'est la représentation la plus simple et admissible pour la modélisation de la méthode. En pratique, les images peuvent être acquises de façon aléatoire (l'ordre d'acquisition d'images au court du temps).

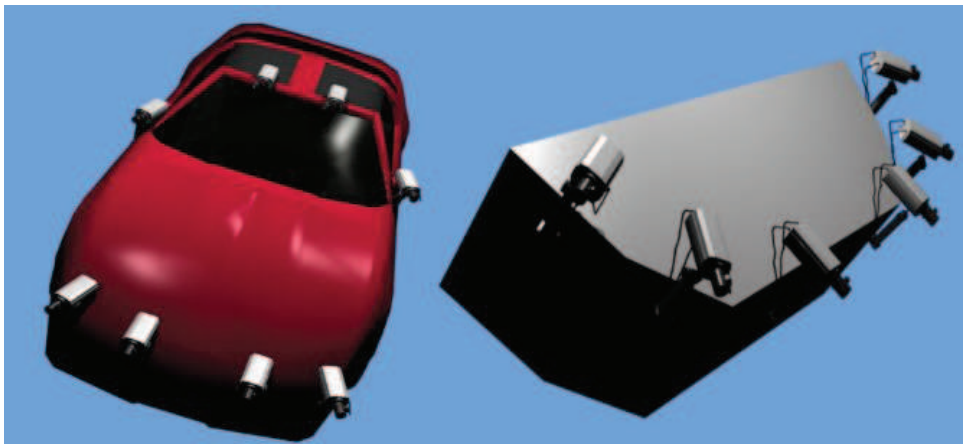


FIGURE 5.2: Exemples de bancs à caméras multiple : la configuration multi-cameras modélisée pour un prototype de laboratoire (à droite) et embarquée sur un véhicule (à gauche)

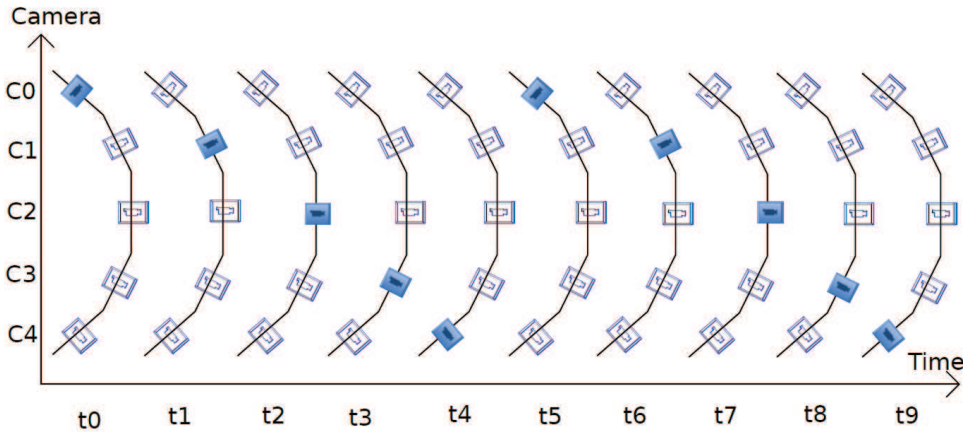


FIGURE 5.3: La distribution d'images non synchronisées à partir de cinq caméras rigidement liées par support : un cas régulier

5.1.3 Estimation du mouvement : Modélisation à base de graphe d'un réseau de caméras non synchronisées

Inspiré des formulations à base de graphes présentées par [88] et [90], nous introduisons dans cette section un nouvel algorithme basé sur les graphes pour estimer le mouvement à l'échelle à partir d'un réseau multi-caméras non synchronisées. Les relations géométriques (les transformations relatives) entre les caméras sont représentées par des arcs dans un graphe non orienté. Chaque nœud du graphe correspond à une pose d'une caméra active (qui acquiert une image). Le graphe est construit au fur et à mesure que le véhicule se déplace à travers l'environnement.

La formulation de notre algorithme comporte trois niveaux du graphe :

i) Le premier niveau du graphe (figure 5.4 (a)) consiste à ajouter toutes les transformations possibles entre les images (à l'échelle près). Une étape de vérification des

transformations permet de valider les transformations et de les ajouter comme de nouveaux arcs du graphe.

ii) Le deuxième niveau du graphe (figure 5.4 (b)) consiste à identifier les triangles. La méthode des triangles est appliquée pour chaque triangle afin de calculer les transformations à l'échelle absolue. Seules les transformations à l'échelle absolue sont gardées dans ce niveau du graphe.

iii) Le troisième niveau du graphe (figure 5.4 (c)) consiste à représenter les poses de chaque caméra dans le repère de la caméra centrale afin d'estimer la trajectoire du système.

5.1.3.1 premier niveau du graphe : Vérification des transformations

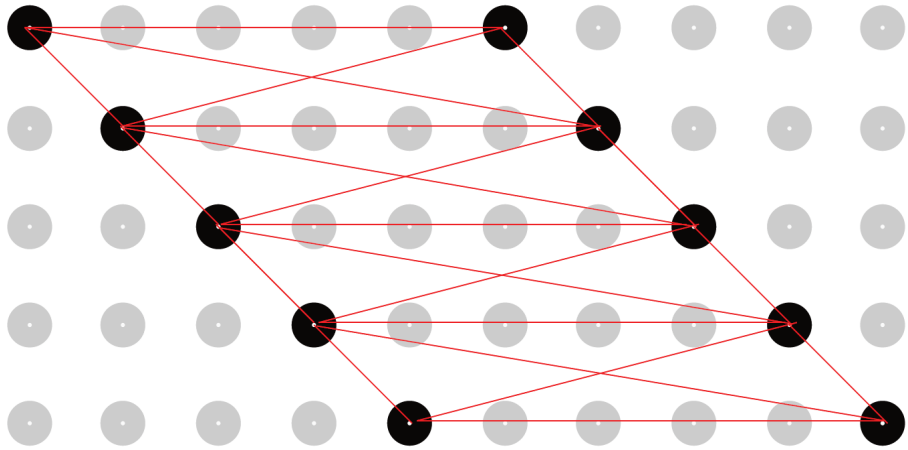
Pour construire le graphe, nous commençons par la détection des transformations possibles. Nous vérifions les transformations entre une image courante et les précédentes prises de vues acquises par les caméras voisines (qui partagent des champs de vue communs avec la caméra courante) et la transformation entre l'image courante et la dernière prise de vue acquise par la même caméra. Nous appelons cette étape l'étape de vérification des transformations (figure 5.5).

La recherche des transformations possibles entre les images se basent sur la recherche des correspondances entre les primitives des images. Le nombre de points mis en correspondance est considéré suffisant quand il dépasse un certain seuil. En effet, nous comparons le nombre des paires de points considérées comme bons appariements à la sortie de l'algorithme du 5 point avec une valeur de seuil. La transformation relative estimée à l'échelle prés entre deux images est considérée comme un arc du graphe que nous appelions le premier niveau du graphe. La figure 5.4 (a) illustre les arcs possibles pour 10 images d'une séquence acquises à partir de 5 caméras. La modélisation présentée correspond à un exemple simple d'acquisition d'images dans un ordre régulier au fil du temps. Nous avons choisi cet exemple pour la facilité de la représentation de l'approche.

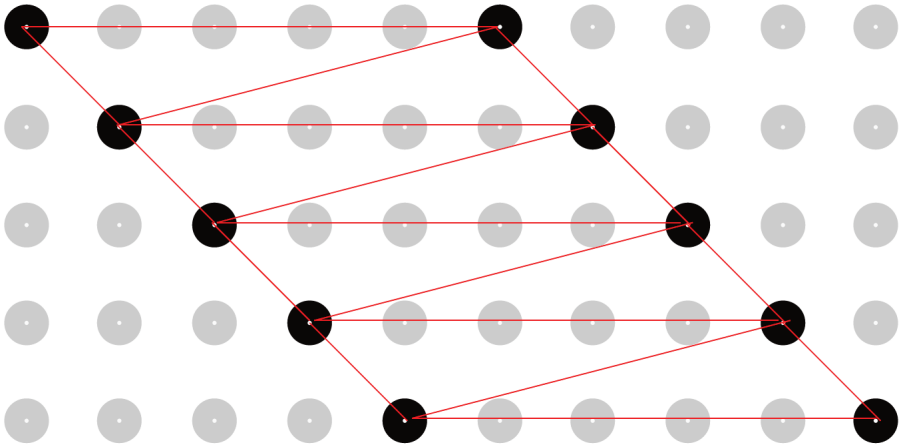
5.1.3.2 deuxième niveau du graphe : Extraction des triangles

Le deuxième niveau du graphe consiste à identifier les triangles formés par les transformations relatives validées dans le premier niveau du graphe. Les triangles doivent convenir aux contraintes posées pour la méthode des triangles : deux images provenant de la même caméra et une images provenant d'une caméra voisine.

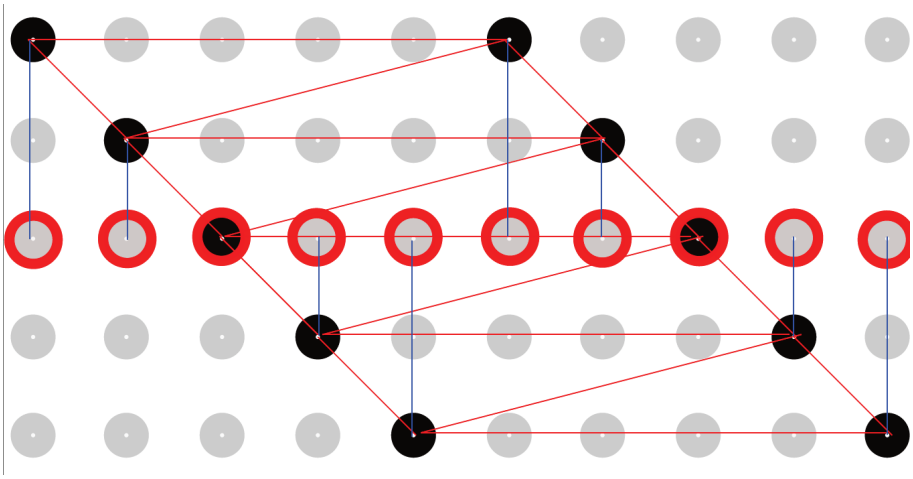
La méthode des triangles, présentée dans le chapitre 3, est appliquée pour chaque forme de triangle afin d'estimer les facteurs d'échelle absolue. Une optimisation des facteurs



(a) Premier niveau du graphe



(b) Deuxième niveau du graphe



(c) Troisième niveau du graphe

FIGURE 5.4: Modélisation à base de graphe d'un réseau de caméras non synchronisées (5 caméras) : un cas régulier

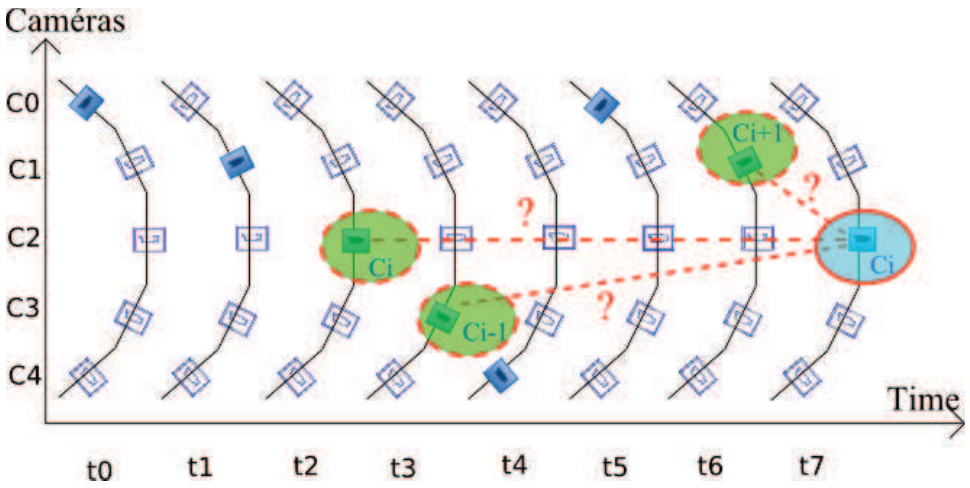


FIGURE 5.5: Premier niveau du graphe : l'étape de vérification des transformations. Pour chaque image provenant d'une caméra (C_i), nous vérifions s'il existe des transformations possibles avec la dernière image de la même caméra (C_i) et avec les dernières images des caméras voisines (C_{i-1} et C_{i+1}).

d'échelle calculés est ensuite effectuée par l'application d'un AF local au niveau du triangle pour améliorer la précision des poses et de la structure obtenue. Enfin, les transformations à l'échelle sont ajoutées au deuxième niveau du graphe comme des arcs.

À ce stade, nous obtenons un graphe connexe dont les nœuds sont les poses et les arcs sont les transformations à l'échelle constituants les triangles. La figure 5.4 (b) illustre les triangles possibles pour une séquences de 10 images provenant de 5 caméras.

5.1.3.3 troisième niveau du graphe : "ego-motion" à l'échelle

Après avoir obtenu les transformations à l'échelle, nous pouvons obtenir la trajectoire du système. Nous avons choisi de représenter les poses de chaque caméra dans le repère de la caméra centrale pour obtenir la trajectoire du véhicule à l'échelle. En effet, nous considérons la caméra centrale en tant que caméra représentant les poses du système. Toutes les poses des caméras (i.e. les nœuds du graphe) sont exprimées dans le repère de la caméra centrale puis dans le repère de la première caméra de la trajectoire.

Comme les caméras sont rigidement liées, chaque nœud du graphe est exprimé dans le repère virtuel de la caméra centrale en passant par les paramètres extrinsèques entre un nœud (la caméra courante) et la caméra centrale. Les poses de la caméra centrale sont ensuite accumulées pour obtenir la trajectoire du système (figure 5.4 (c)).

5.1.3.4 discussion des cas particuliers

Quand la pose d'une caméra est calculée à partir de deux ou plusieurs triangles, il faut choisir la meilleure estimation de la pose. Pour cela, nous attribuons un score de confiance pour chaque triangle. Ce score est obtenu à partir des erreurs de reprojection de la structure 3D. La figure 5.6 illustre un cas où la pose de la caméra C_i peut être obtenue à partir de deux triangles. Un score est attribué à chaque triangle pour l'ajout de la pose de la caméra C_i au deuxième niveau du graphe.

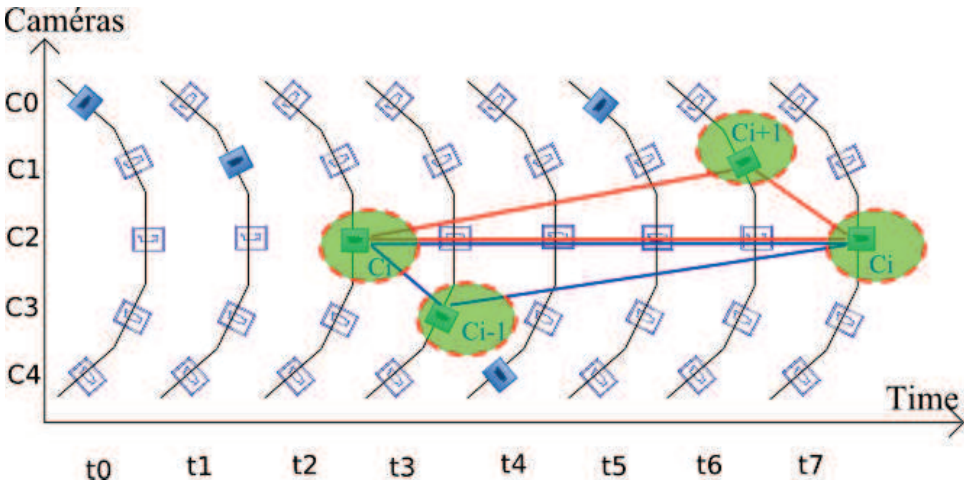


FIGURE 5.6: Deux triangles connexes. La pose de la caméra C_i peut être obtenue à partir de deux triangles (le triangle en rouge et le triangle en bleu).

Si à un instant t une caméra n'est pas reliée au graphe, la pose relative de la caméra centrale ne peut pas être estimée. Ceci peut arriver en raison d'un manque de points d'intérêt (l'image n'a donné lieu à aucune transformation possible dans le premier niveau du graphe). Dans ce cas, la pose suivante de la caméra centrale à l'instant $t+1$ (virtuelle ou estimée) est exprimée par rapport au repère de référence (la première pose de la trajectoire) en passant par la dernière pose de la caméra centrale (la pose de l'instant $t-1$). La figure 5.7 illustre un cas où la caméras C_0 n'est pas reliée au graphe.

Il est important de noter que les triangles doivent être connectés, i.e. partagent au moins une pose (un nœud) avec d'autres triangles. Dans le cas contraire, le cas d'un triangle non connecté au reste du graphe, les poses ne sont pas prises en compte dans la trajectoire. La figure 5.8 illustre le cas d'un triangle non connecté au graphe.

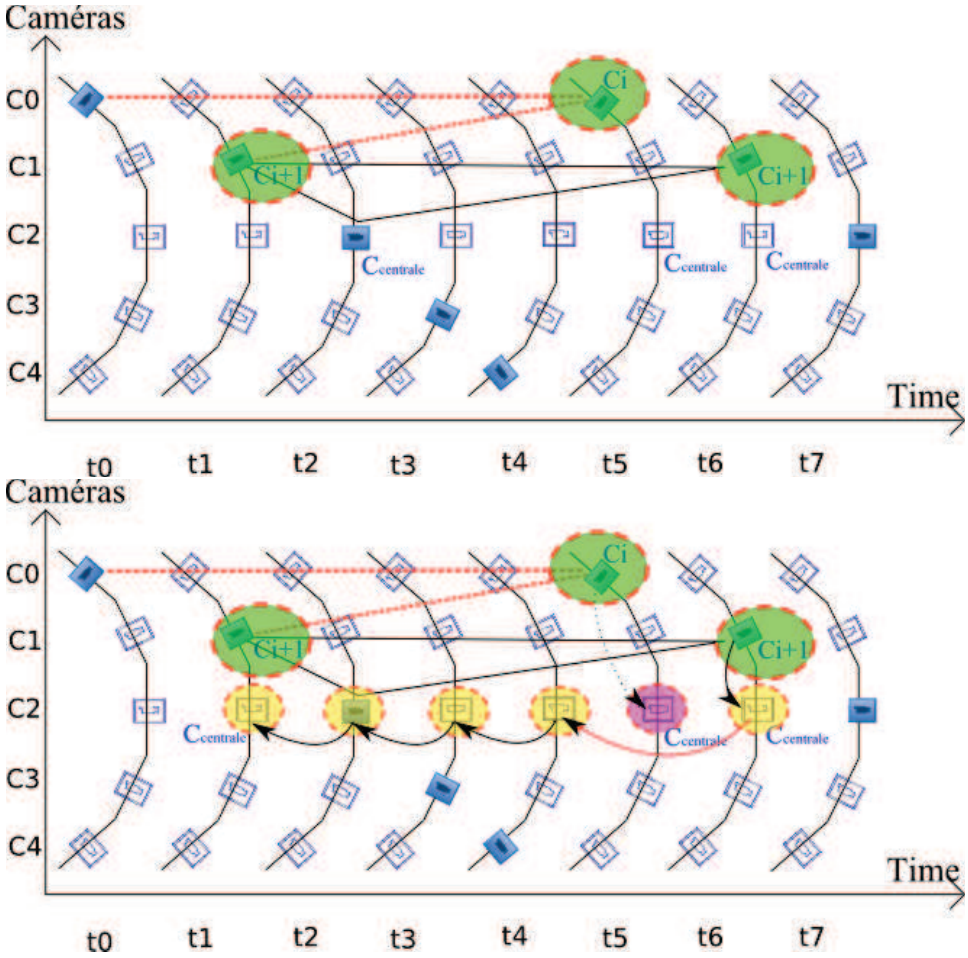


FIGURE 5.7: à un instant $t = 5$, la caméra C_0 (C_i) n'est pas reliée au graphe. La pose de la caméra suivante C_1 à l'instant $t = 6$ $(t + 1)$ est exprimée par rapport au repère de référence (la première pose de la trajectoire) en passant par la dernière pose de la caméra centrale correspondante à l'instant $t = 4$ $(t - 1)$.

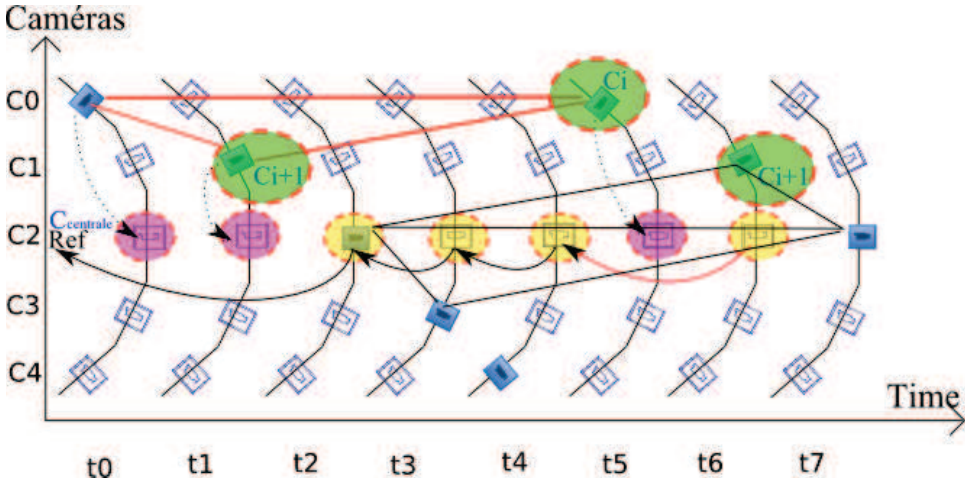


FIGURE 5.8: Cas d'un triangle non connecté au graphe (le triangle en rouge)

5.1.4 Résultats Expérimentaux

5.1.4.1 Présentation du système

La modélisation du réseau présentée par la figure 5.2 est choisie en coordination avec le système de la figure 5.9. Le prototype conçu pour valider notre méthode est présenté par la figure 5.9. Il s'agit d'un système composé de 5 caméras rigidement liées par un support qui peut être mis sur le toit d'une voiture pour les expérimentations.

FIGURE 5.9: Un réseau de 5 caméras montées sur une plateforme expérimentale

Les caméras que nous utilisons pour réaliser nos expérimentations sont des caméras Basler Ace 1600, GigE, CCD 1/1.8", 1624x1234. Ces caméras Ethernet sont montées avec une optique de 6mm ce qui nous permet d'avoir un angle de 60 degrés. Pour couvrir un champ de vue de 180 degrés devant le véhicule, nous réalisons un recouvrement de moitié entre les caméras voisines ce qui explique l'usage de cinq caméras, figure 5.10.

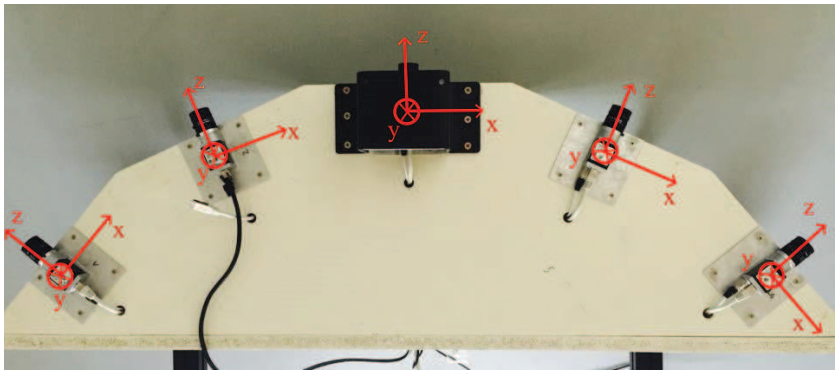


FIGURE 5.10: La plateforme expérimentale : 5 caméras rigidement liées par un support, avec des champs de vue communs (deux à deux).

5.1.4.2 Étalonnage

Le calibrage des caméras de notre système est une étape importante qui nous permet d'estimer les paramètres intrinsèques et les paramètres extrinsèques. Dans la littérature, il existe de nombreuses méthodes de calibrage qui peuvent être réparties en deux catégories. Les méthodes de la première catégorie nécessitent un protocole d'expérimentation et des outils en utilisant un pointeur laser, des mires 3D ou une mire plane pour pouvoir réaliser la calibration. Les méthodes de la deuxième catégorie sont des méthodes qui s'appuient sur les informations métriques de la scène (les méthodes d'auto-calibrage). Les méthodes de la première catégorie sont plus contraignantes mais plus robustes et précises.

FIGURE 5.11: Illustration des expérimentations lors de la calibration

Nous avons utilisé une calibration avec une mire plane (de type damier) car cette méthode donne un étalonnage précis. En connaissant la distance entre deux carreaux de la mire, nous avons effectué le calibrage des caméras deux à deux afin d'estimer les paramètres extrinsèques entre les caméras. Les paramètres intrinsèques sont estimés simultanément.

Le processus consiste à déplacer une mire successivement devant les caméras, en veillant que tous les carreaux de la mire soient vue par la caméra. Nous avons utilisé l'algorithme de calibrage proposé par Zang [91] car c'est une méthode simple et précise.

La distorsion des images est corrigée en utilisant les paramètres de distorsion de chaque caméra avant que les images ne soient utilisées pour l'estimation du mouvement (figure 5.13).

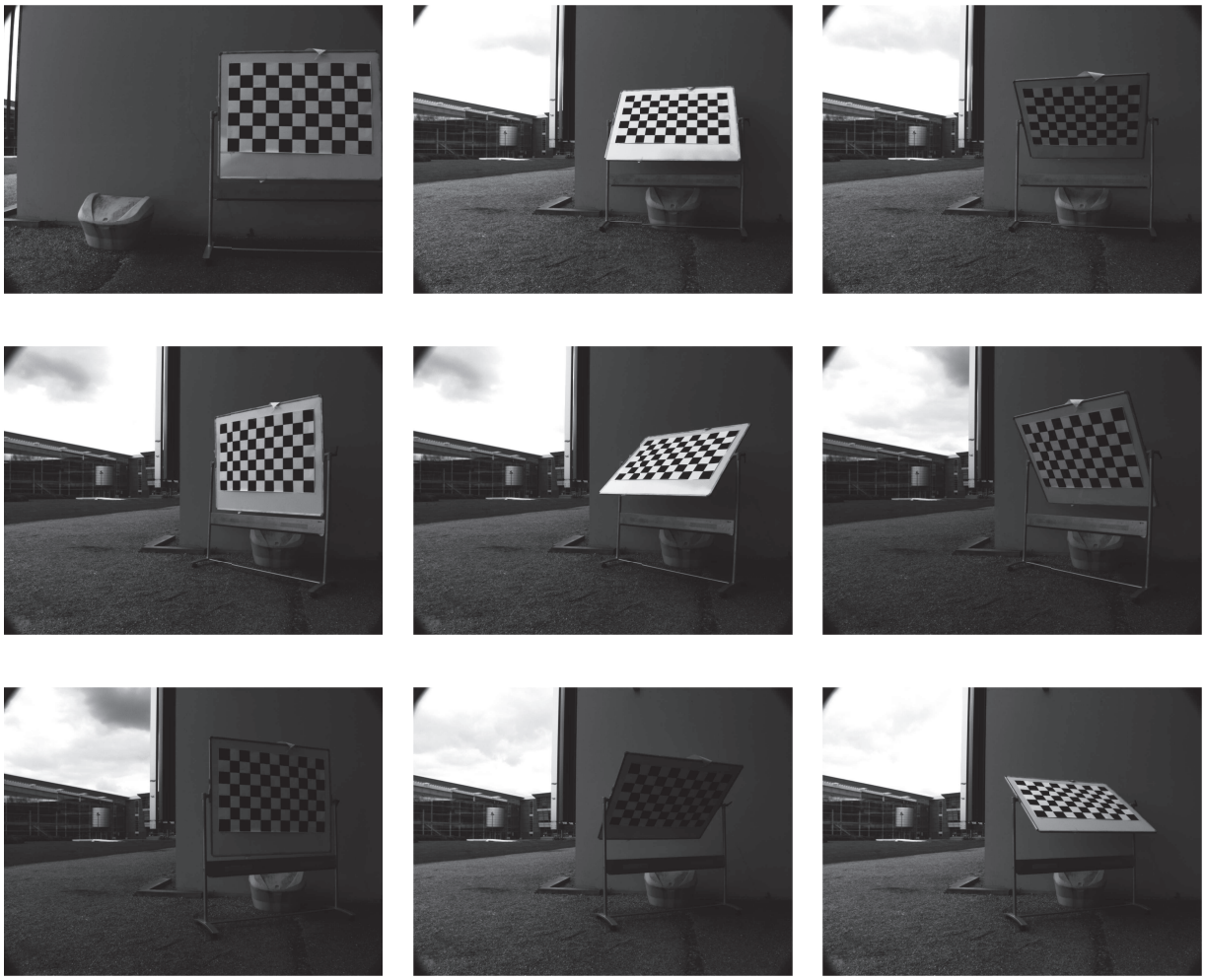


FIGURE 5.12: Exemples de 9 images prises lors de la calibration du système

(a) (b)

FIGURE 5.13: Exemples d'images où la distorsion est corrigée : (a) l'image avant la correction (b) l'image après la correction

FIGURE 5.14: Images à partir de 5 caméras non synchronisées aux instants de 126 à 130, respectivement, de la gauche vers la droite

5.1.4.3 Résultats de l'estimation de mouvement :

La plateforme expérimentale a été montée sur un véhicule pour collecter l'ensemble des données pour tester notre algorithme. L'ensemble de données a été recueilli lors de la conduite de la voiture dans une boucle d'environ 700m autour du campus universitaire de l'INSA de Rouen. La fermeture de boucle permet d'avoir une information pertinente d'évaluation de la trajectoire, nous en servons comme des informations de vérité terrain.

L'ensemble des séquences se compose principalement des scènes statiques avec quelques fois des voitures en mouvement et des piétons. Les données GPS ont été enregistrées au cours de l'expérimentation à partir d'un téléphone mobile (de type iphone6) pour la représentation cartographique, figure 5.15. Un total de 5 x 3600 images sont utilisées dans le test de notre algorithme. Un exemple d'images non synchronisées est représenté dans la figure 5.14.

La figure 5.16 montre quelques images capturées lors des expériences au campus de l'INSA de Rouen.

Pour chaque nouvelle image, nous commençons par la vérification des transformations possibles. Les primitives ont été extraites par FAST et décrites par BRIEF et les transformations relatives ont été estimées en utilisant l'algorithme de 5 points. Une transformation relative est valide quand le nombre d'appariements obtenus est plus grand qu'un certain seuil. Pour les résultats présentés nous prenons un seuil de 50 points.

Quand une transformation est considérée valide, les rotations et les translations relatives sont estimées à partir des appariements. Ensuite, pour chaque pose, les triangles possibles sont vérifiés.

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. Les valeurs moyennes des facteurs d'échelle obtenues pour la trajectoire effectuée sont présentées dans le tableau 5.1.

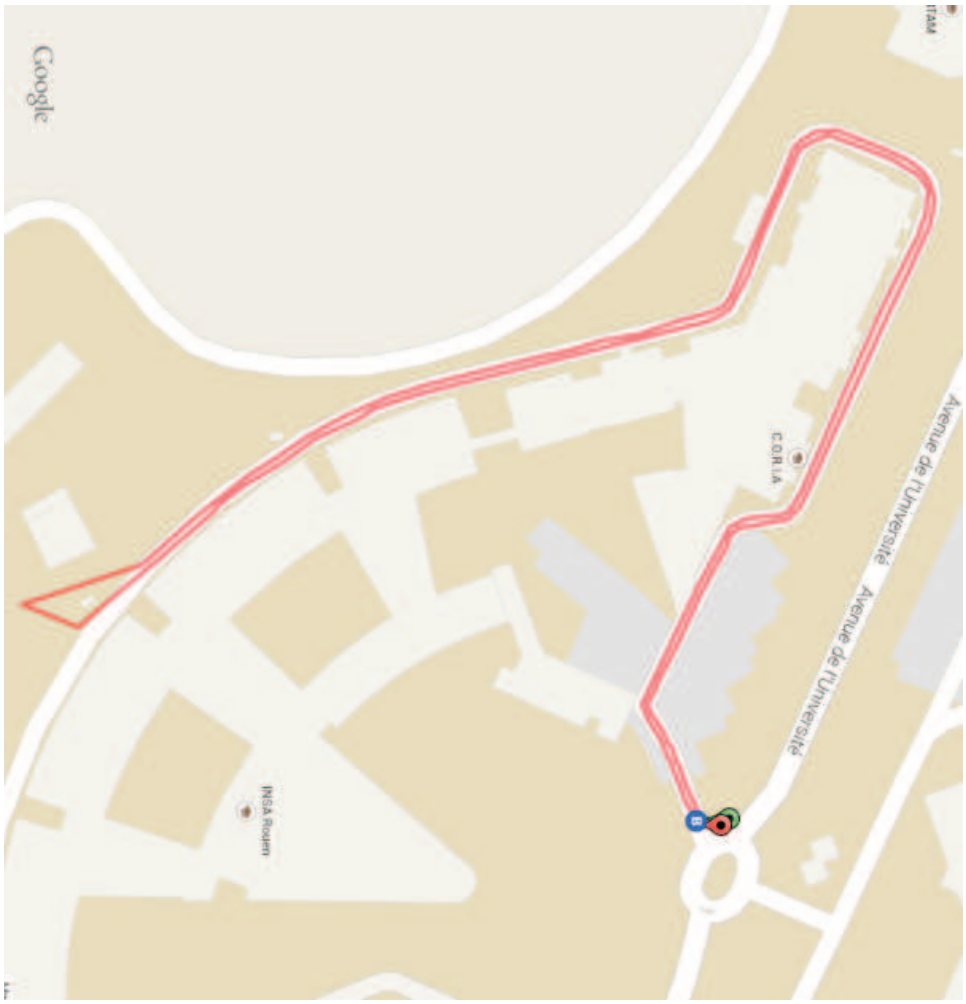


FIGURE 5.15: La trajectoire enregistrée par un téléphone mobile durant les acquisitions

TABLE 5.1: Les facteurs d'échelle absolue

λ_1	λ_2	α	β
0.57123 ± 0.014	0.26027 ± 0.003	0.657 ± 0.013	0.345 ± 0.002

Les scores sont attribués pour chaque triangle en se basant sur les erreurs de reprojection accumulées après l'étape de l'AF. Un score est attribué pour chaque transformation du triangle. Quand une pose d'une caméra est issue de plus d'un triangle, les scores sont comparés pour choisir la "meilleure" pose de la caméra.

Nous exprimons chaque nœud du graphe dans la caméra centrale afin d'avoir la trajectoire du véhicule. Une portion de la trajectoire obtenue est représentée par la figure 5.17.

(a)

(b)

(c)

FIGURE 5.16: la plateforme expérimentale montée sur le véhicule lors des expérimentations

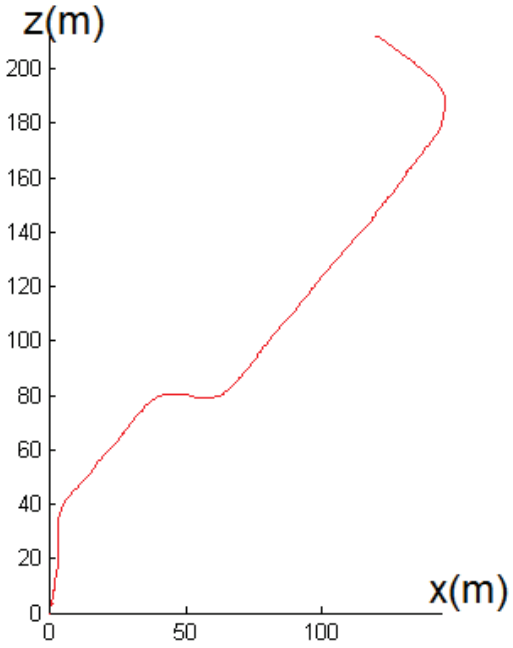


FIGURE 5.17: la trajectoire estimée pour la première partie de l'ensemble de données en utilisant le réseau de caméras non synchronisées (1400 images)

5.2 Détection d'obstacle à partir d'un réseau de caméras non synchronisées

Dans cette section, nous introduisons une nouvelle approche d'extraction du plan de la route et de détection d'obstacles en se basant sur nos travaux précédents.

La détection de la route et d'obstacles est effectuée à partir de cartes de UV-disparités éparses obtenues à partir d'images non synchronisées. Nous calculons une carte de disparités éparse à partir de la structure 3D à l'échelle estimée avec la méthode des triangles. Ensuite, les cartes U-disparités et les cartes V-disparités éparses sont calculées à partir des cartes de disparités éparses. Une transformation de Hough est ensuite appliquée aux cartes de V-disparités afin d'en extraire le plan de la route et aux cartes de V-disparités et de U-disparités afin d'en détecter les obstacles présents dans la scène.

Nous utilisons un filtre de Kalman pour prédire la nouvelle position de la route afin de gagner du temps en minimisant la région de recherche.

5.2.1 Travaux antérieurs

Les applications de détection d'obstacles ont été largement étudiées dans les domaines robotiques et véhicules intelligents. De nombreuses études de détection d'obstacle sont basées sur une carte de disparités dense, [92] [93] [94][95][96]. Lorsque la détection est

effectuée par des systèmes multi-caméras, une carte de disparités dense est calculée puis est analysée pour extraire les obstacles. Cependant, en utilisant des caméras non synchronisées, une carte de disparités dense ne peut pas être calculée à partir d'images non synchronisées qui sont décalées dans le temps et dans le lieu.

L'approche de détection de la route et des obstacles à partir des cartes de UV-disparités a été étudiée et utilisée par plusieurs travaux dans la littérature [97][98][99][100] [101][102].

Une reconstruction dense peut être plus efficace, mais a un coût plus élevé car elle nécessite la synchronisation pour rectifier les images. La plupart des algorithmes qui se basent sur la reconstruction 3D ont été conçus en tenant compte de la synchronisation et de la contrainte de la géométrie épipolaire. Une carte de disparités éparse comporte moins d'informations qu'une carte de disparités dense. Toutefois, une carte de disparités éparse offre plus de souplesse et peut être adaptée à de nombreuses applications car elle peut être obtenue en relâchant la contrainte de la synchronisation.

5.2.2 Extraction de la route

5.2.2.1 carte de disparités éparse

Une carte de disparités éparse peut être obtenue à l'aide d'un algorithme de SFM classique basé sur des approches 2D-2D. Après le calcul de la structure 3D à l'échelle, nous pouvons calculer la disparité de chaque point 3D.

La performance d'une carte de disparités dépend fortement du détecteur de point d'intérêt utilisé. En effet, une carte de disparités éparse contient peu d'informations par rapport à une carte dense. Pour avoir une carte éparse riche en informations, il faut qu'elle contienne le plus de points possible. Il est donc important d'utiliser un détecteur de point d'intérêt qui fournit assez de points ou qui a une répétabilité élevée pour que le nombre de points (considérés comme bons appariements) contribuant au calcul des disparités soit élevé.

La disparité est définie comme étant l'écart de position horizontale des pixels mis en correspondance. En stéréovision, pour deux points $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ images d'un point 3D $\mathbf{X}$ de coordonnées (X, Y, Z), la disparité $d_m = \mathbf{x}_1 - \mathbf{x}_2$ est définie par l'équation 5.1.

$$\begin{aligned} \frac{\mathbf{x}_1 - \mathbf{x}_2}{b} &= \frac{f}{Z} \\ \Longleftrightarrow d_m &= \frac{f * b}{Z} \end{aligned} \tag{5.1}$$

où f est la distance focale des caméras et b est la distance entre les centres optiques des deux caméras. La disparité est donc inversement proportionnelle à la profondeur (Z). La grandeur d_m est obtenue en mètre. La disparité d_p exprimée en pixel est donnée par l'équation 5.2.

$$\alpha = k * f \tag{5.2}$$
$$\Longleftrightarrow f = \frac{\alpha}{k} \tag{5.3}$$
$$\Longleftrightarrow d_p = \frac{\alpha * b}{k * Z} \tag{5.4}$$

5.2.2.2 carte de V-disparité

Initialement introduite par Labrayde dans [93], l'image V-disparité représente les caractéristiques géométriques d'une scène. Il s'agit d'une représentation de la carte de disparité dans le v-espace. Une carte de disparité est définie comme étant une image dont le repère est situé en haut à gauche et dont l'axe des abscisses correspond à la disparité et l'axe des ordonnées correspond au numéro de ligne image. Le nombre de colonnes de la carte de V-disparités est égal aux disparités maximales.

La carte de V-disparité permet d'extraire le plan de la route à partir de la carte de disparité. La route est considérée comme une succession de portions de plans horizontaux. Il peut être obtenu directement à partir d'une carte de disparité dense par l'accumulation des pixels ayant la même disparité d dans chaque ligne.

L'algorithme permettant de calculer la V disparité à partir d'une carte de disparité dense est présenté par 4.

Algorithm 4 Le calcul de la carte V-disparité (dense)

```
for chaque ligne i de la carte de disparité dense do
    for chaque colonne j de la carte de disparité dense do
        if (disp(i,j)>0) then
            Vdisp ((disp (i,j),j))++
        end if
    end for
end for
```

Nous avons modifié l'algorithme de calcul de la V-disparité de telle sorte qu'elle soit calculée à partir d'une la carte de disparité éparse. L'algorithme est présenté par l'algorithmes 5.

Algorithm 5 Le calcul modifié de la carte V-disparité (éparse)

```
for chaque ligne i de la carte de disparité éparse do
    for chaque colonne j de la carte de disparité éparse do
        if (disp(i,j)>0) then
            Vdisp ((disp (i,j),j))=255
        end if
    end for
end for
```

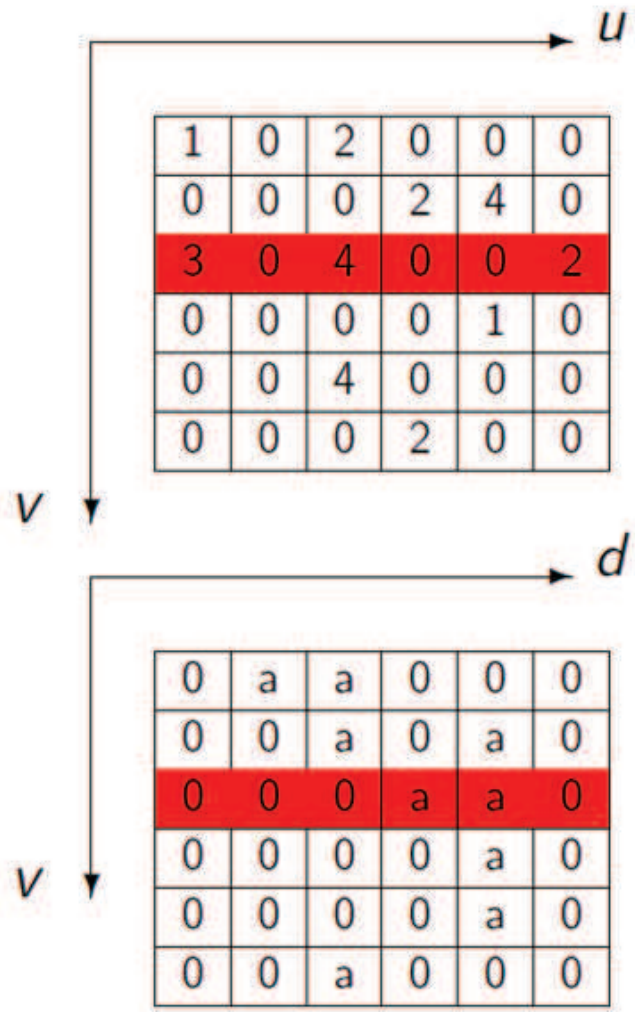


FIGURE 5.18: La différence entre le calcul de la V-disparité dense et éparse.

La figure 5.18 illustre la différence entre les cartes de V-disparité obtenues à partir de cartes de disparités dense et éparse. La figure en haut présente une modélisation simple pour une carte V-disparité calculée à partir d'une carte de disparité dense et la figure en bas présente une modélisation pour une carte obtenue à partir d'une carte de disparité éparse par l'algorithme 5 (a est une valeur fixe, $a = 255$).

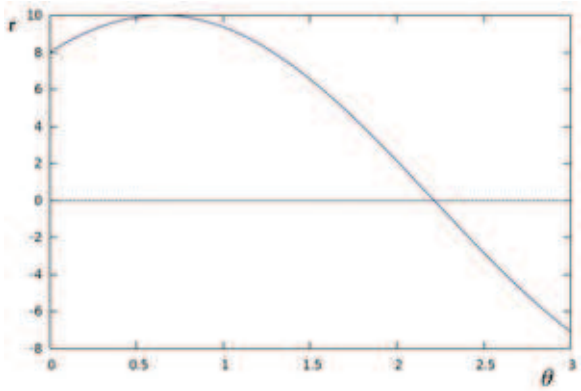


FIGURE 5.19: Un exemple de la sinusoïde représentant toutes les droites (ρ, Θ) passant par un point de coordonnées $x = 8$ et $y = 6$) (image provenant de la documentation de la bibliothèque opencv)

5.2.2.3 Extraction du plan de la route

La route est modélisée comme une succession de parties de plan. Un plan dans la carte de disparité éparse devient une ligne droite dans la carte V-disparité. La route est donc représentée par une droite oblique et les obstacles sont représentés par des droites verticales. Pour chercher le plan de la route, il faut chercher la droite qui lui correspond à partir de la carte de V-disparité. Pour pouvoir l'extraire, nous utilisons la transformée de Hough.

La transformée de Hough a été proposée par Paul Hough en 1962 [103]. C'est une technique efficace et simple qui permet de détecter les lignes droites [104]. Nous l'avons appliqué sur l'image de V-disparité pour extraire le plan de la route (représenté par une droite oblique).

La transformée de Hough permet de parcourir tous les points pour trouver la droite contenant le maximum de points. Une droite peut être définie par un vecteur de coordonnées (ρ, Θ) dans le repère complexe. Pour un point de coordonnées (x, y), il existe une infinité de droites (définies par (ρ, Θ)) ayant comme équation $\rho = xcos\Theta + ysin\Theta$. En traçant toutes les combinaisons (ρ, Θ) représentant les droites passant par un point, une sinusoïde est obtenue (figure 5.19). Répétant ce processus pour tous les points de la carte de V-disparité, on obtient une représentation de plusieurs sinisoïdes dans l'espace de Hough. Les lignes peuvent être détectées par la recherche des intersections entre les courbes. La ligne représentée par l'intersection issue du maximum de courbes est la ligne ayant le plus de points. La droite obtenue a donc comme coordonnées les valeurs (ρ, Θ) de l'intersection. En général, nous pouvons définir un seuil du nombre minimal d'intersections nécessaires pour détecter une ligne.

Pour éviter l'extraction des lignes supplémentaires qui ne correspondent pas à la droite représentant le plan de la route, le seuil du nombre minimal de droites est choisi automatiquement par rapport à des critères donnés. En effet, la transformée de Hough doit extraire une seule ligne de façon à ce que le nombre de points qui appartient à la droite soit maximum. Le processus de détection de droite est répété jusqu'à l'obtention d'une seule droite. Le seuil est donc choisi automatiquement de manière adaptée pour chaque paire d'images.

L'algorithme du choix du seuil est présenté par l'algorithme 6. Nous avons fait des essais sur plusieurs images pour initialiser la valeur de seuil (150). Nous avons aussi remarqué que la valeur de Θ (l'inclinaison de la droite est toujours au tour d'une même valeur dans les images (sauf dans les virages). Pour cela, nous ne prenons en compte que les droites d'inclinaison autour de cette valeur fixée (par exemple pour une inclinaison égale à 2).

Algorithm 6 Le calcul du seuil adapté pour chaque image

```
Initialiser le seuil (s⟵150)
while le nombre de droites retenues est différent de 1 do
    Appliquer la transformée de Hough avec le seuil s
    for chaque ligne détectée (lines(i)) do
        (ρ,Θ)⟵lines(i)
        if la partie entière de Θ est autour de la valeur de l'inclinaison then
            la droite lines(i) est retenue
        end if
    end for
    s−−
end while
```

Pour certaines situations, l'inclinaison de la droite représentant la route change brusquement (lors d'une montée par exemple). Pour cela, nous utilisons un filtre de Kalman pour estimer l'inclinaison.

5.2.3 Prédiction du plan de la route : filtre de Kalman

Pour appliquer l'approche décrite précédemment sur de longues séquences, nous appliquons une étape de prédiction du plan de la route en utilisant le filtre de Kalman. En effet, pour chercher le plan de la route dans une carte V-disparité courante, nous ne gardons qu'une zone d'intérêt autour d'une droite prédite par le filtre de Kalman (représentant le plan de la route prédit) afin de rétrécir la zone de recherche. Cette technique nous permet de simplifier et d'accélérer les traitements.

La zone d'intérêt dans la V-disparité est définie autour de la droite estimée par le filtre de Kalman représentant le plan de la route. Le filtre de Kalman estime les paramètres (ρ, Θ)

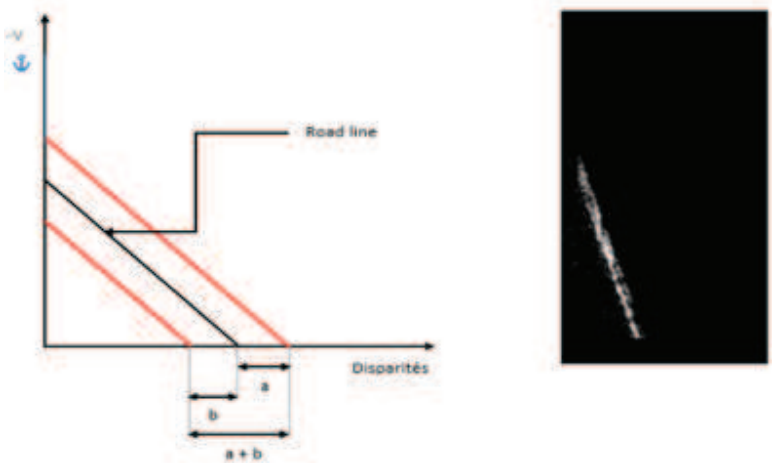


FIGURE 5.20: La définition de la région d'intérêt dans la V disparité

de la droite (la prédiction du vecteur d'état $(\rho, \Theta)^t$). Nous considérons que la zone de la V-disparité limités par une droite légèrement supérieures et légèrement inférieure de la ligne estimée (figure 5.20). Le reste de la carte V-disparité est éliminé avant l'application de la transformée de Hough.

Pour définir la zone de recherche dans la carte V-disparité, nous fixons deux bornes a et b, figure 5.20 : a est la distance au-dessus de la ligne de route donnée par Kalman et b est la distance en dessous. La région de recherche est donc choisie d'une façon adaptée à l'image étudiée.

5.2.4 Détection des obstacles

5.2.4.1 Carte de U-disparité

De même, comme pour la carte de V-disparité, la carte U-disparité peut être calculée par l'accumulation des pixels ayant la même valeur dans chaque colonne de la carte de disparité. L'algorithme est décrit par l'algorithme 7.

Algorithm 7 Le calcul modifié de la carte U-disparité (éparse)

```
for chaque ligne i de la carte de disparité éparse do
    for chaque colonne j de la carte de disparité éparse do
        if (disp(i,j)>0) then
            Udisp(j, (disp (i,j))=255
        end if
    end for
end for
```

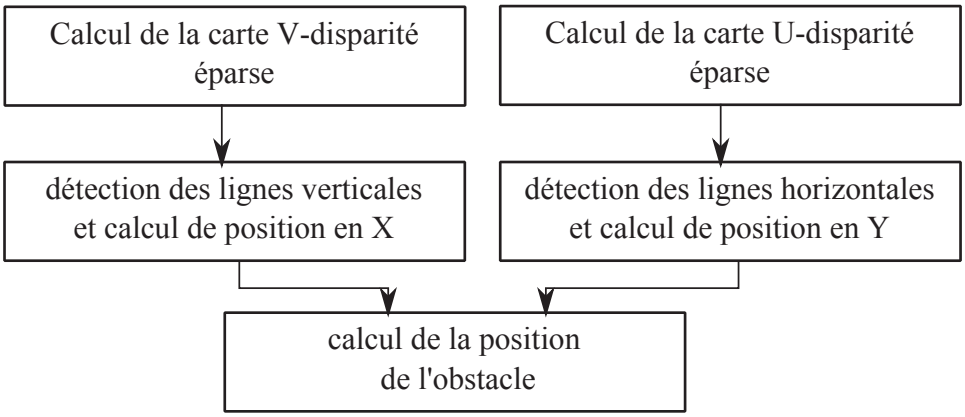


FIGURE 5.21: La détection d'obstacles à partir des cartes UV-disparité éparses

5.2.4.2 Détection des obstacles

Les obstacles sont représentés par des lignes verticales dans les cartes V-disparité et avec des lignes horizontales dans les cartes U-disparité. En appliquant la transformée de Hough sur ces deux dernières, on peut facilement extraire les hauteurs des obstacles à partir de les longueurs des lignes verticales dans les cartes V-disparité et les largeurs des obstacles à partir des longueurs des lignes horizontales dans les cartes U-disparité. De la même façon que dans le cas de la détection d'obstacles à partir des cartes UV-disparité dense, la figure 5.21 illustre le processus de détection des obstacles à partir des cartes UV-disparités éparses.

5.2.5 Résultats

Nous avons appliqué l'algorithme décrit précédemment sur une séquence de la banque d'images réelles de la base KITTI [9][10]. Nous simulons l'aspect asynchrone de la même manière que pour les expérimentations décrites dans les chapitres 3 et 4. Nous appliquons la méthode des triangles afin d'estimer le mouvement et la structure à l'échelle, puis nous calculons la carte de disparité éparse pour chaque paire d'images.

Les résultats obtenus sont regroupés dans la figure 5.22. La figure 5.22 (a) est un exemple d'une carte de disparité éparse calculée à partir de deux images non synchronisées. Les points sont présentés par différents niveaux de rouge pour la clarté de l'affichage.

Nous générons ensuite les cartes de UV-disparité pour extraire le plan de la route et les obstacles. Les résultats de l'application de la transformée de Hough sont illustrés par la figure 5.22 (b, c, d). Nous présentons les lignes verticales qui représentent les obstacles en rouge et la ligne oblique qui représente le plan de la route en vert.

Les résultats de détection de route sont présentés par la figure 5.22 (e) : les cercles bleus représentent les points considérés comme des points de la route. Les premiers résultats sont satisfaisants mais quelques faux positifs apparaissent. Les fausses détections peuvent être éliminées par des approches de filtrage.

La figure 5.22 (f) montre un exemple d'une carte U-disparité et la figure 5.23 présente un exemple de détection d'obstacles à partir des cartes UV-disparité. La largeur et la hauteur de chaque obstacle sont déduites de la largeur et de la hauteur des segments de lignes détectées dans les cartes de UV-disparité éparses. La figure 5.23 montre les boîtes qui englobent les obstacles (les bounding box). Cette figure illustre l'estimation des largeurs et des hauteurs à partir des cartes UV-disparité éparses.

Nous jugeons que les résultats obtenus sont acceptables. La plupart des objets de la scène sont extraits, mais la détection doit être réétudier pour améliorer la précision du processus. En effet, l'application d'opérateurs morphologiques sur les cartes UV-disparité peut améliorer la dimension des obstacles. Cette partie devrait être étudiée attentivement.

Conclusion

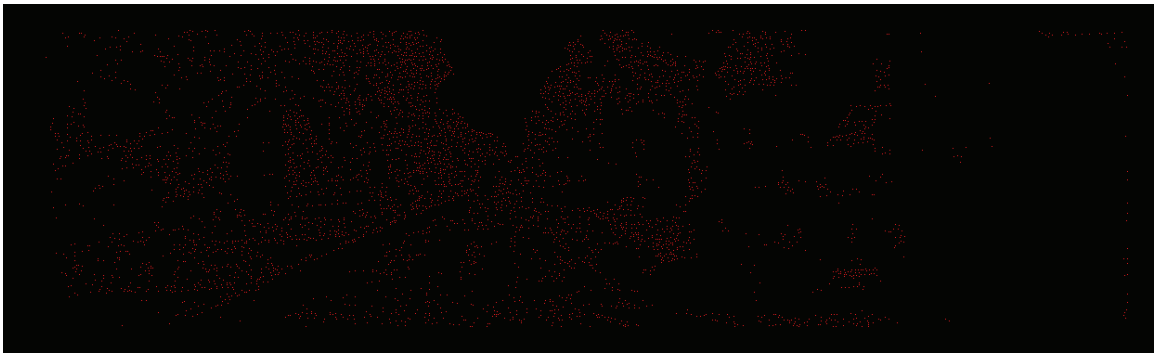
Dans la première partie de ce chapitre, nous avons présenté la méthode d'estimation du mouvement et de la structure étendue à N caméras ($N > 2$). Nous avons présenté un réseau multi caméras par un graphe dont les nœuds représentent les poses des caméras et les arcs représentent les transformations entre les caméras. Le premier niveau du graphe permet des vérifier les transformations possibles entre les caméras. Ces dernières sont estimées à l'échelle prés. Les triangles sont ensuite identifiés dans le deuxième niveau du graphe. A ce stade, seules les transformations calculées à l'échelle (par la méthode des triangles) sont conservées. Enfin, le troisième niveau du graphe permet de représenter les poses de chaque caméra dans le repère de la caméra centrale afin d'avoir la trajectoire.

Nous avons évalué la méthode proposée sur les images acquises à partir d'une plate-forme expérimentale comprenant cinq caméras asynchrones. Les résultats obtenus sont cohérents et montrent que la méthode des triangles peut être généralisée à N caméras.

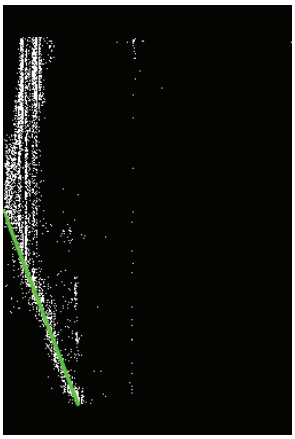
Dans la deuxième partie de ce chapitre, nous avons présenté une méthode de détection du plan de la route et d'obstacles à partir d'images non synchronisées. La méthode exploite les informations d'une carte de disparité éparse pour en générer une carte de V-disparité et une carte de U-disparité. Le plan de la route est représenté par la droite oblique de la carte de V-disparité. Cette droite est extraite à l'aide de la transformée de Hough.

Les obstacles sont représentés par des segments de droites verticaux de la carte de V-disparité et des segments de droites horizontaux de la carte de U-disparité. Les longueurs des segments verticaux de la carte de V-disparité représentent les hauteurs des obstacles et les longueurs des segments horizontaux de la carte de U-disparité représentent les largeurs des obstacles.

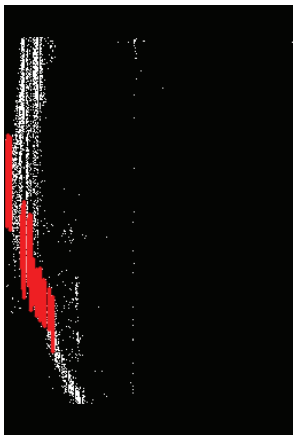
Nous avons évalué cette méthode sur les images de la base de données KITTI. Les résultats obtenus sont satisfaisants mais la détection doit être réétudiée pour améliorer la précision du processus.



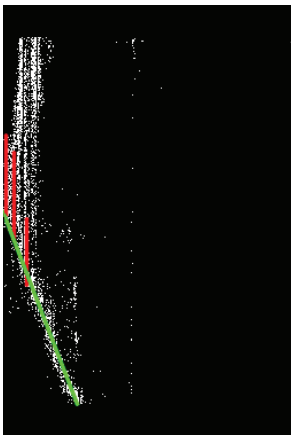
(a) Carte de disparité éparse



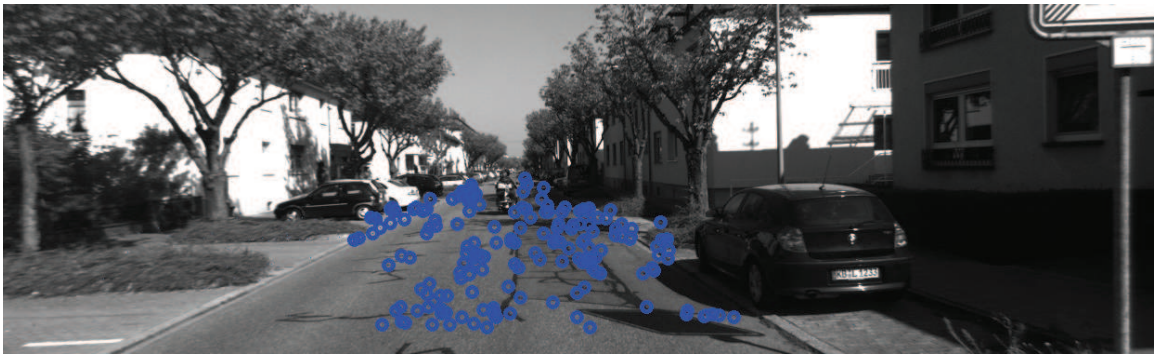
(b) V-disparité : la route



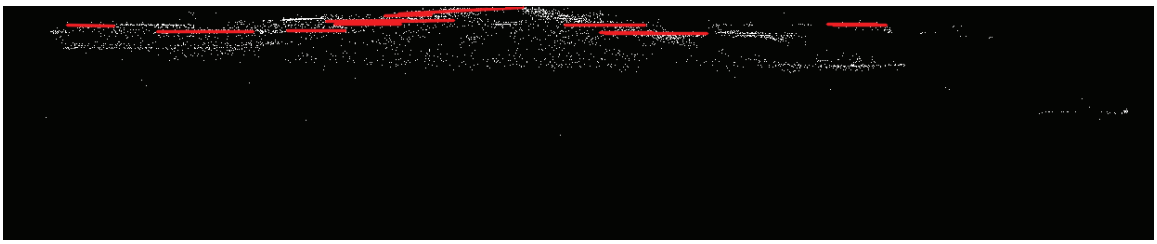
(c) V-disparité : les obstacles



(d) V-disparité résultantes



(e) Les points de l'image représentant le plan de la route



(f) U-disparité : détection des obstacles présentés par les lignes horizontaux

FIGURE 5.22: Les résultats de détection du plan de la route et des obstacles. Les lignes rouges correspondent aux lignes représentant les obstacles dans les cartes UV-disparité. Les lignes vertes sont les lignes représentant le plan de la route dans la carte V-disparité. Les cercles bleus dans (e) représentent les points appartenant à la ligne verte dans la V-disparité

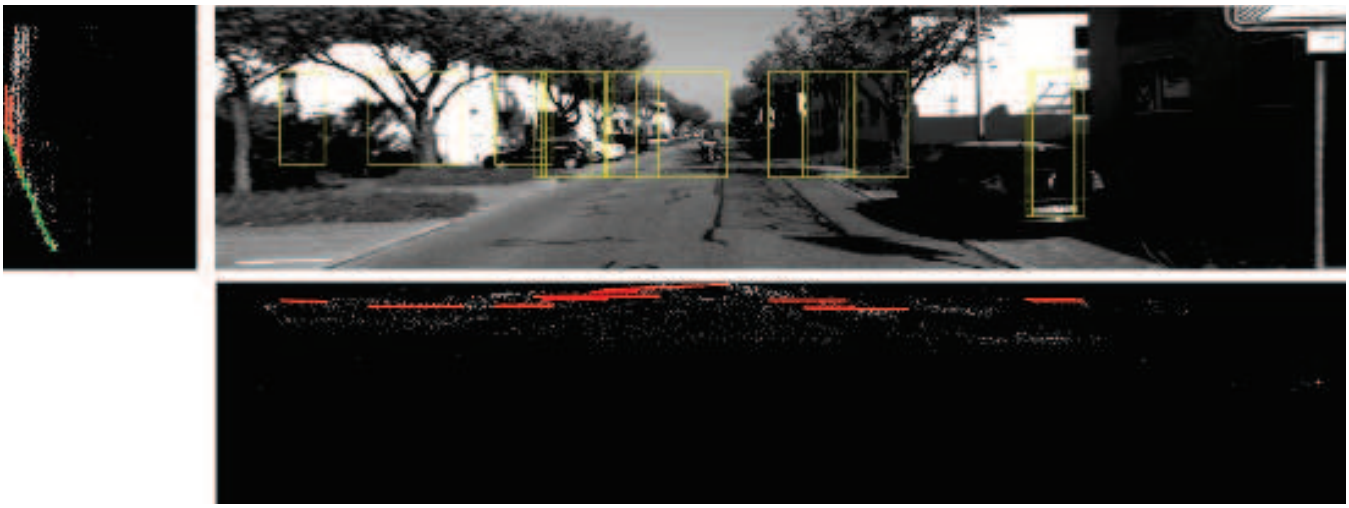


FIGURE 5.23: Un exemple de détection d'obstacles à partir des cartes UV-disparité

Chapitre 6

Conclusion et perspectives

Conclusion

Les travaux présentés dans ce manuscrit portent sur l'estimation du mouvement et de la structure 3D de l'environnement autour d'un véhicule. Il s'agit d'une étape primordiale pour plusieurs applications dans le domaine robotique et le domaine de l'assistance à la conduite. Il est bien établi que les caméras sont très utiles pour avoir une reconstruction 3D et un positionnement précis. En effet, un système multi-caméras permet d'augmenter le champ de vision et d'avoir une vue complète autour du véhicule.

Néanmoins, l'estimation du mouvement et de la structure à l'échelle à partir d'un système multi-caméras n'est possible que si les caméras sont synchronisées et si la scène est prise sous différents angles de vue au même instant. La synchronisation devient un inconvénient majeur car elle impose l'ajout d'un circuit électronique et d'un câblage supplémentaire. Pour s'affranchir des contraintes d'un système synchrone, un réseau de caméras asynchrone peut être employé et présente de nombreux intérêts. Un système asynchrone peut être développé avec des dispositifs à faible coût, ce qui est souvent souhaitable pour les applications grand public. Un autre avantage est qu'avec un tel système, l'acquisition ne dépend plus de la caméra la plus lente ou du dispositif de synchronisation lui-même et les images peuvent être acquises en continu à partir de chaque caméra séparément.

L'estimation du mouvement et de la structure (Structure From Motion, SFM) à partir d'un système asynchrone évoque plusieurs questions fondamentales qui doivent être résolues. Pour répondre à ces questions, nous avons proposé une méthode qui permet d'obtenir le mouvement et la structure à l'échelle absolue en prenant soin d'estimer les facteurs d'échelle. L'échelle est importante car elle permet de pouvoir dimensionner

l'environnement pour les applications d'aide à la conduite et de navigation autonome, comme la détection d'obstacle par exemple.

Une étude bibliographique a été établie autours des problème de SFM et de l'odométrie visuelle et les configurations possibles pour l'obtention d'une vue complète autour d'un véhicule en circulation a été discuté. Après une étude des différentes approches, nous avons choisi d'utiliser les techniques 2D-2D pour le processus d'odométrie visuelle : le mouvement est estimé à partir des primitives 2D mises en correspondance entre les images. Ce choix est pertinent puisque nous n'employons aucune connaissance a priori sur la scène 3D et ne cherchons pas à synchroniser les flux d'images.

Une évaluation des méthodes de détection et de description des primitives 2D a été présentée pour les détecteurs et les descripteurs locaux les plus utilisés en odométrie visuelle afin d'étudier la robustesse de ces algorithmes. Les résultats expérimentaux de cette étude ont été évalués sur des images d'une base de données réelles : la banque d'images KITTI. L'aspect asynchrone est obtenu en prenant une seule image à chaque instant t. À cause des objets dynamiques dans la scène, les performances des détecteurs et des descripteurs sur des images synchrones sont meilleures que sur des images asynchrones, ce qui rend l'estimation du mouvement à partir de caméras non synchronisées un véritable défi scientifique. Notre choix a été porté sur le détecteur FAST pour l'extraction des points-clés et le descripteur BRIEF pour la description des points-clés. Ces deux derniers ont les meilleures performances sur les images non synchronisées.

Nous avons présenté une nouvelle méthode d'estimation de mouvement pour un réseau multi-caméras non synchronisées que nous avons appelée la méthode des triangles. Notre méthode repose sur l'utilisation de trois images formant un triangle : deux images provenant de la même caméra et une image provenant d'une caméra voisine. Nous supposons que la trajectoire entre deux images consécutives provenant d'une même caméra est approximée par un segment linéaire et que les caméras sont calibrées. La connaissance de la distance entre les deux caméras (estimée par un étalonnage hors ligne) combinée avec l'hypothèse de mouvement rectiligne du système, nous a permis d'estimer les facteurs d'échelle absolue.

Bien que notre approche présente de bons résultats pour les trajectoires rectilignes, certaines erreurs peuvent se propager en raison des éventuelles imprécisions dans l'estimation des facteurs d'échelle. Ces erreurs sont améliorées par une méthode d'optimisation : un ajustement de faisceaux local (AF). Nous avons appliqué un AF sur les facteurs d'échelle absolue et sur les points 3D pour affiner l'estimation initiale.

Dans la dernière partie de la thèse, nous avons présenté la méthode d'estimation du mouvement et de la structure étendue à N caméras ($N > 2$). La formulation est basée

sur un graphe de poses dont les nœuds représentent les poses des caméras et les arcs représentent les transformations entre elles. Le premier niveau du graphe permet de vérifier les transformations possibles entre les caméras. Ces dernières sont estimées à l'échelle près. Les triangles sont ensuite identifiés dans le deuxième niveau du graphe. Enfin, le troisième niveau du graphe permet de représenter les poses de chaque caméra dans le repère de la caméra centrale afin d'obtenir la trajectoire. Nous avons évalué la méthode proposée sur les images acquises à partir d'une plateforme expérimentale comprenant cinq caméras non synchronisées.

Enfin, nous avons présenté une application fondamentale dans les domaines d'aide à la conduite et de la conduite automatisée. Il s'agit d'une méthode de détection de la route et des obstacles à partir d'images non synchronisées. La méthode se base sur une carte de disparité éparse pour en calculer une carte de V-disparité et une carte de U-disparité. Le plan de la route et les obstacles sont représentés par des segments de droites dans les cartes de V-disparité et de U-disparité. Ces segments sont extraits à l'aide de la transformée de Hough. Les dimensions des obstacles sont obtenues par les longueurs des segments verticaux de la carte de V-disparité et des segments horizontaux de la carte de U-disparité. Le plan de la route représenté par une droite oblique dans la carte de V-disparité est suivi et prédit par un filtre de Kalman afin d'obtenir les meilleurs résultats et limiter la zone de recherche. Les résultats obtenus sur les images de la base de données KITTI sont satisfaisants mais la détection doit être améliorée en terme de précision.

Tous les algorithmes développés dans cette thèse ont été évalués expérimentalement sur des images de scènes routière réelles. Les résultats montrent que l'utilisation des réseaux asynchrones est possible pour estimer le mouvement d'un véhicule. La méthode des triangles testée sur des séquences non synchronisées présente des erreurs de translations entre 7 et 9% et des erreurs de rotations entre 0.041 et 0.015 degrés par mètres pour des sous-séquences de 100 et 800 mètres. L'ajustement de faisceaux appliqué uniquement sur les facteurs d'échelle et la structure 3D permet d'améliorer les résultats. Notre méthode peut être appliquée à un réseau multi caméras et peut également être utilisée pour les applications de véhicules intelligents.

Perspectives

Bien que le SFM et l'odométrie visuelle aient atteint une certaine maturité, ces thèmes de recherches restent encore à explorer puisque les méthodes existantes sont trop coûteuses et/ou ne sont pas suffisamment robustes. Les applications réelles basées sur la vision nécessitent des systèmes bas coûts facilement embarquables sur un véhicule.

Les réseaux composés de caméras asynchrones ne rehaussent non seulement les contraintes en stéréo vision, mais aussi les problèmes d'estimation du mouvement. Il est très intéressant d'utiliser un système multi-caméras asynchrone en raison de sa flexibilité. Notre méthode convient à de nombreux domaines tels que la construction automobile, les véhicules autonomes et les applications robotiques qui préfèrent mettre en œuvre des systèmes asynchrones.

Les problématiques initialement posées concernant l'estimation du mouvement et de la structure de l'environnement proche à l'échelle absolue ont été localement résolues. Néanmoins, les longues séquences et certaines situations posent encore quelques problèmes pour l'odométrie visuelle. Pour résoudre ces problèmes, l'ajout d'une connaissance du modèle 3D pour faire une localisation absolue peut être une solution envisageable, comme les applications de localisation par mémoire visuelle par exemple.

Pour résoudre le problème de dérives, une autre solution est d'envisager une démarche basée sur une caméra virtuelle représentant la scène 3D autour du véhicule. Cette caméra possède une vision panoramique de la scène, son champ correspond au champ du réseau de caméra. Le graphe de poses peut être localement lié aux positions de cette caméra virtuelle. L'optimisation peut être effectuée localement entre deux positions de la caméra virtuelle et les points 3D qui y appartiennent. Ensuite, une optimisation globale peut être effectuée à intervalles réguliers pour réduire les problème de dérives.

Les thématiques de SFM et de l'odomérie visuelle restent sans doute des sujets de recherche à explorer dans les domaines de la vision par ordinateur et de la robotique mobile car il existe encore de nombreux problèmes pratiques à résoudre.

La détection d'obstacles autour du véhicule est une application fondamentale. L'approche proposée dans cette thèse peut être améliorée, par exemple en densifiant la carte de disparités éparse. En effet, la carte représentant les objets mobiles peut être enrichie en étudiant le processus de mise en correspondance. En décomposant l'image en sous-régions, les appariements peuvent être effectués entre les motifs afin d'obtenir plus de points mis en correspondance et par suite plus de disparités. Ensuite, la carte peut être densifiée en propageant les régions autour des points d'intérêt pour se rapprocher au mieux d'une carte dense. Pour valider ou améliorer les détections, les obstacles détectés peuvent être suivis dans les autres images du réseau, en particulier les objets mobiles.

Annexe A

Les détecteurs et les descripteurs de points d'intérêt

Certains détecteurs et descripteurs de points d'intérêt sont introduits dans cette section. Il s'agit des algorithmes les plus connus et les plus utilisés en odométrie visuelle.

A.1 HARRIS et ses variantes

Harris [105] a été proposé par Harris et Stephens en 1988. Il s'agit d'un détecteur de coins basé sur la matrice d'auto-corrélation qui décrit la distribution du gradient du voisinage d'un point afin de chercher les maximums locaux supérieurs à un certain seuil. L'évaluation de Schmid et al. a montré que Harris est le détecteur le plus robuste et le plus informatif [81].

Harris est un détecteur invariant aux transformations de translation et de rotation. Il est également stable aux variations d'illuminations. Cependant, ce détecteur est sensible aux changements d'échelles et aux transformations affines.

Il existe plusieurs extension de Harris : Harris Laplace, Harris Affine [106]. Harris Laplace est basé sur le détecteur de Harris multi-échelles pour localiser les points d'intérêts aux différents niveaux d'échelles. L'opérateur Laplace est utilisé pour la sélection de l'échelle. Harris Affine, quant à lui, détecte la région initiale avec le détecteur de Harris Laplace puis estime une région de forme elliptique avec la matrice d'auto-corrélation puis une normalisation de cette région permet d'obtenir une région de forme circulaire. Ensuite de nouvelles régions et échelles des points d'intérêt sont re-détectées. Ce processus est répété jusqu'à ce que les valeurs propres de la matrice d'auto-corrélation soient égales.

A.2 MSER

MSER (Maximally stable extremal regions) est un détecteur de région proposé en 2004 par Matas et al. [78]. Le mot "Extremal" désigne que tous les pixels à l'intérieur d'une région identifiée par MSER ont soit une intensité supérieure ou inférieure par rapport à tous les pixels sur son contour. Le "maximally stable" dans MSER décrit la propriété d'optimisation lors de la sélection des seuils. Cet algorithme procède comme un algorithme de segmentation. La méthode extrait les régions d'intensité homogènes qui sont stables sur une large plage de seuils. Les régions sont définies par une fonction d'intensité dans la région et son contour, ce qui conduit à extraire plusieurs points clés des régions. La stabilité de MSER est basée sur le changement de surface relative ce qui rend ce détecteur invariant aux transformations affines (photométrique et géométrique).

A.3 STAR

STAR est un détecteur de point-clé qui a été implémenté par la bibliothèque OpenCV. Il est dérivé du détecteur CenSurE (Center Surround Extrema) [107]. Les auteurs visent à la formation d'un détecteur multi-échelles en pleine résolution spatiale. Comme défini dans [107], le sous-échantillonnage effectué par SIFT et SURF affecte la précision de la localisation. Le détecteur utilise une approximation bi-niveau du filtre Laplacien des gaussiennes (LOG). La forme circulaire du masque est remplacée par une approximation qui préserve l'invariance de rotation et permet l'utilisation de l'image intégrale pour un calcul efficace. L'espace d'échelles a été créé sans interpolation, par l'application de masques de tailles différentes.

A.4 FAST

Le détecteur FAST (Features from Accelerated Segment Test) est un détecteur de coins qui a été introduit en 2006 par Rosten et Drummond dans [108] et [12]. Ce détecteur est basé sur le détecteur SUSAN [109]. SUSAN calcule la fraction des pixels dans une région (le voisinage d'un pixel) ayant une intensité similaire à celle du pixel central. Cette idée est reprise par Fast qui compare les intensités des pixels qui appartiennent à une zone sous forme d'un cercle de rayon fixé autour d'un point central (le point d'intérêt candidat).

Dans un premier temps, le test des segments est appliqué pour chaque coin candidat P. Pour un pixel P d'intensité I_p, si N pixels appartenant au cercle ont comme intensité I_{p+t}

ou I_{p-t}, t est un seuil configuré, le candidat P est considéré comme un point d'intérêt. La figure A.3 illustre la détection dans un motif d'image. Il est démontré que les meilleurs résultats sont obtenus lorsque N = 9, l'algorithme correspondant est appelé FAST-9 [12].

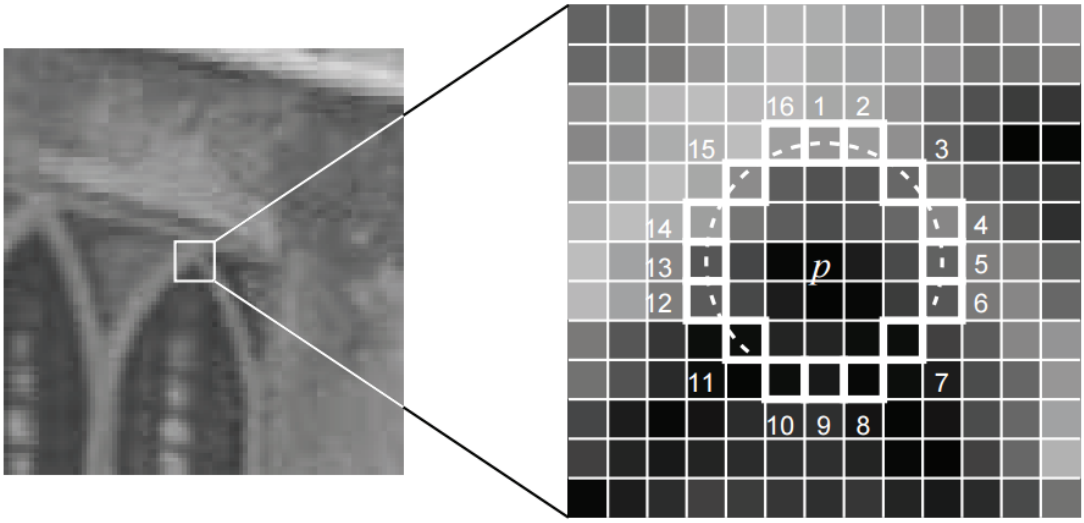


FIGURE A.1: Détéction de point d'interet avec le détecteur FAST - Test sur de 12 points dans un motif d'une image [12].

Pour choisir quel pixel dans le cercle doit être considéré comme candidat potentiel, l'algorithme ID3 [110] est appliqué. Ceci est mesuré par l'entropie des réponses de classification d'angle positives et négatives basées sur ce pixel. Ce test produit de nombreuses réponses adjacentes autour du point d'intérêt. Pour cela, une suppression non maximale est appliquée en utilisant une fonction coût (V) afin de supprimer les candidats ayant un V élevé.

Il a été prouvé que Fast est un détecteur très fiable grâce à sa répétabilité élevée. Il est devenu l'un des détecteurs les plus utilisés pour les applications temps-réel telles que la méthode PTAM d'estimation de pose des caméras présentée par Klein et al dans [111] et la mise en correspondance robuste en 2.3 µs présentée par Taylor et al. dans [112].

A.5 AGAST

Le détecteur AGAST (Adaptive and Generic Corner Detection Based on the Accelerated Segment Test) est un détecteur de coin qui a été proposé en 2010 par Mair et al. [113]. Semblable à FAST, AGAST est également basé sur le test des segments accélérés. AGAST utilise des arbres de décision binaires pour terminer le test des segments accéléré. Deux arbres sont construits, le premier pour les régions homogènes et le second pour les régions structurées. En combinant les deux arbres de décision, AGAST

s'adapte automatiquement pour fournir l'arbre de décision le plus efficace pour la région de l'image étudiée. AGAST n'a pas besoin d'une phase d'apprentissage et il préserve la même détection et répétabilité que le détecteur FAST.

A.6 SIFT

SIFT (Scale Invariant Feature Transform) a été proposé par David Lowe en 1999 [114] et amélioré en 2004 [13]. SIFT est un algorithme de détection et de description de points d'intérêt. La première étape est la détection des points. Les points sont définis par leurs coordonnées images et par leurs facteurs d'échelle caractéristiques. Les zones d'intérêt sont définies comme des zones circulaires de rayons proportionnels au facteur d'échelle. Après la détection, le filtrage et l'amélioration de la précision des points d'intérêt, la deuxième étape de l'algorithme consiste à calculer les descripteurs.

L'algorithme SIFT se compose en quatre étapes : la détection d'extrema dans l'espace des échelles, la localisation du point clé, l'assignation d'orientation et le calcul du descripteur du point clé.

Les points clés sont extraits d'un espace discret qui comporte trois dimensions : les coordonnées 2D et le facteur d'échelle. Ensuite, l'image subit un lissage en calculant le gradient du facteur d'échelle (la convolution de l'image par un filtre gaussien dont le paramètre est le facteur d'échelle). Cette étape permet d'estomper les petits détails de l'image. La détection des points se fait en étudiant la différence de gaussiennes (DoG) obtenue par :

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{A.1}$$

où L est gradient de facteur d'échelle σ, x,y sont les coordonnées du point et k est un paramètre fixe de l'algorithme.

L'image obtenue ne contient que les points observables dans des facteurs d'échelle entre σ et $k\sigma$. Un point candidat est alors un extremum du DoG par rapport à son voisinage. L'algorithme utilise une pyramide d'échelles pour avoir des échelles différentes.

Cette étape de détection d'extremums produit plusieurs points clés candidats. Ensuite, les points instables sont éliminés en supprimant les points de faible contraste et les points situés sur les contours. La précision de la localisation est améliorée par interpolation des coordonnées.

La dernière étape de la détection est l'assignation d'orientation : à chaque point détecté une ou plusieurs orientations sont attribuées en se basant sur la direction des gradients dans le voisinage du point. Le descripteur est ensuite calculé relativement à ces orientations d'où l'invariance du descripteur SIFT à la rotation. Le calcul des vecteurs descripteurs se fait sur une région de 16 par 16 pixels, subdivisée en 4x4 zones de 4x4 pixels chacune. Un histogramme des orientations comportant 8 intervalles est calculé sur chaque zone. Les 16 histogrammes obtenus (comportant 8 intervalles chacun) sont concaténés et normalisés pour fournir un vecteur de dimension 128 (16x8), figure A.2.

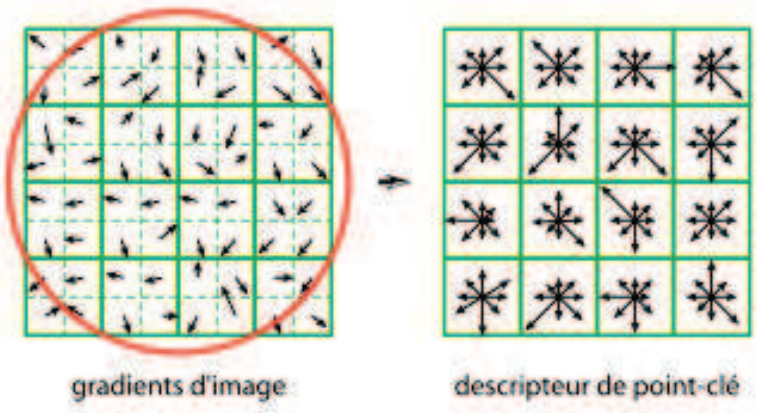


FIGURE A.2: Construction du descripteur SIFT [13].

Plusieurs études sur les performances des descripteurs locaux [69] ont conclu que SIFT donne les meilleurs résultats. Le descripteur reste invariant pour des transformations affines de moins de 50 degrés.

L'étude de [69] a prouvé que les descripteurs à base de SIFT dépassent en performance les autres types de descripteurs locaux en présence de scènes texturées.

Il existe plusieurs variantes de l'algorithme SIFT, les plus connues A-SIFT et PCA-SIFT.

A.6.1 PCA-SIFT

PCA (analyse en composantes principales) est une technique standard pour la réduction de dimension. Ke et Sukthankar ont combiné le descripteur SIFT avec l'algorithme PCA pour réduire la dimension des descripteurs vectoriels SIFT [115] [116]. Le PCA-SIFT a la même entrée que le descripteur SIFT standard : L'emplacement des sous-pixels, l'échelle et les orientations dominantes des points clés. Le vecteur du gradient de l'image est projeté dans un vecteur caractéristique compact. En comparant avec SIFT, PCA-SIFT nécessite moins d'espace et de composants ce qui le rend plus rapide.

A.6.2 ASIFT

L'idée du ASIFT [117] est de combiner la simulation et la normalisation de la méthode SIFT. Ce détecteur, SIFT, couvre 4 parmi les 6 paramètres de la transformation affine par le biais de la normalisation des rotations et des translations, et la simulation d'une pyramide d'échelle. Pour obtenir une invariance totale aux transformations affines, ASIFT couvre les 6 paramètres. Grâce à cette caractéristique, il est la seule méthode complètement invariante aux transformations affines.

ASIFT simule avec suffisamment de précision les distorsions provoquées par une variation de la direction de l'axe optique de la caméra. Ensuite, il applique la méthode SIFT pour comparer les images simulées de telle sorte que tous les six paramètres soient couverts. En d'autres termes, ASIFT simule trois paramètres : la taille, l'angle de longitude et l'angle de latitude de la caméra (ce qui est équivalent à l'inclinaison) et normalise les trois autres paramètres. Malgré ses performances en terme d'invariance aux transformations affines, ASIFT est très couteux en terme de temps.

A.7 SURF

SURF (Speeded-Up Robust Features) est une méthode proposée par Bay et al. [118]. Cette méthode peut être considérée comme une approximation de SIFT. Le détecteur SURF s'appuie sur le détecteur SIFT mais utilise une façon légèrement différente pour la détection des points clés. SIFT construit la pyramide d'échelles, applique la convolution des échelles supérieures et inférieures de l'image avec l'opérateur différence de gaussiennes (DoG) et cherche de les extremums locaux dans l'espace d'échelle. Les échelles du SURF filtrent au lieu de réduire itérativement la taille de l'image. Cela permet d'éviter le repliement mais limite l'invariance au changement d'échelle. Une deuxième différence entre SURF et SIFT est que SURF utilise un filtre de 9x9 pour approximer les dérivés gaussiennes partielles de second ordre dans SIFT.

Pour l'étape de la description, les ondelettes de Haar sont utilisées, ce qui permet de déterminer le gradient en x et y. Afin d'obtenir l'orientation dominante, les réponses d'ondelettes de Haar calculées pour tous les points appartenant à une zone de rayon 6s autour du point détecté, où s est l'échelle à laquelle ce point a été détecté. Une fois que les réponses d'ondelettes sont calculées et pondérées par une gaussienne, l'orientation dominante est estimée en additionnant toutes les réponses horizontales et verticales au sein d'une fenêtre glissante couvrant un angle de $\Pi/3$. L'orientation ayant le vecteur le plus long est l'orientation dominante du descripteur.

Une fenêtre carrée autour du point clé de taille de 20s est divisée en sous-régions 4x4 et chaque sous-région est divisée en 5x5 points d'échantillonnage régulièrement espacés. Les réponses d'ondelettes de Haar pour les directions horizontales dx et les directions verticales dy sont calculées à chaque point d'échantillonnage, puis multipliées par chaque sous-région. Le descripteur pour chaque sous-région de la réponse consiste à trouver les valeurs absolues de chacune des directions principales. Par conséquent, le vecteur de descripteur complet pour toute 4x4 sous-région a une longueur de 64. Pour son temps de traitement relativement rapide et sa robustesse aux transformations d'images typiques, SURF est devenu le descripteur standard.

A.8 BRIEF

BRIEF (Binary Robust Independent Elementary Features) est un nouveau descripteur binaire proposé en 2010 par Calonder [82]. Après l'étape de détection, la zone du voisinage du point d'intérêt est lissée pour réduire la sensibilité au bruit et augmenter la stabilité et la répétabilité du descripteur. Dans le motif lissé, 128 paires de pixels autour du point clé sont sélectionnés pour les tests binaires. Ces tests consistent à comparer les intensités de chaque paire de pixels, la valeur 1 signifie que la première valeur est plus grande que la seconde, 0 dans le cas contraire. Après les essais binaires sur ces paires de pixels, une chaîne binaire de 128 bits est construite. Cette chaine représente le descripteur BRIEF. Ce descripteur est rapide à construire et à apparier en raison de sa nature binaire, mais il n'est pas invariant à la rotation et à l'échelle.

En plus de sa rapidité, la performance de BRIEF est similaire à SIFT pour plusieurs critères, notamment la robustesse aux changements d'illumination, au flou et à la distorsion de la projection perspective. Cependant, il est très sensible à la rotation dans le plan.

A.9 ORB

ORB (Oriented FAST and Rotated BRIEF) a été développé en 2011 par Rublee et al. [119]. ORB, comme son nom l'indique, est basé sur le détecteur Fast et le descripteur BRIEF.

L'algorithme utilise FAST-9 comme détecteur de points. Pour ajouter une information sur l'orientation du pixel, Fast-9 est appliqué sur une pyramide d'échelles. Ensuite les points détectés sont filtrés en se basant sur le filtre de coin de Harris. Le filtrage est

effectué dans chaque niveau de la pyramide pour rejeter les contours et fournir un score raisonnable, seuls les N premiers points sont repris.

Pour un motif de l'image, une approche appelée le centroïde C de l'intensité est obtenue à partir des moments de la région. Un vecteur partant du centre du motif au centroïde C est construit afin d'obtenir l'orientation du motif. Cette orientation et sa matrice de rotation correspondante sont utilisées pour tourner le motif afin de couvrir l'insuffisance de l'invariance à la rotation. ORB applique ensuite le descripteur BRIEF sur le motif "tourné" pour calculer le descripteur binaire.

ORB est alors rapide vue sa nature binaire tout en gardant l'invariance au changement d'échelle et à la rotation.

A.10 BRISK

BRISK (Binary Robust Invariant Scalable Keypoints) [14] a été proposé par Leutenegger en 2011. Selon l'auteur, il est plus rapide que SURF et surpasse les descripteurs de l'état de l'art en terme de performance. Le détecteur BRISK est basé sur le détecteur FAST combiné avec un descripteur de type chaine de bits, d'où sa rapidité. Le descripteur est récupéré à partir d'une comparaison d'intensité par échantillonnage du voisinage de chaque point d'intérêt.

Une pyramide d'échelles est construite tout en prenant en compte les inter-octaves entre les couches des niveaux d'échelle. Pendant la phase de détection, les extrema locaux dans l'espace échelle basés sur le score FAST sont recherchés. Le score FAST est défini comme le seuil maximal pour un détecteur FAST-9. Pendant la phase de description, BRISK effectue un échantillonnage sur le voisinage des points clés pour obtenir des modèles circulaires, figure A.3.

Les paires de points d'échantillonnage sont divisées en deux sous-ensembles en se basant sur la distance d'échantillonnage entre deux points : les paires ayant de courtes distances et les paires ayant de longues distances. Les paires ayant de longue distance sont utilisées pour le calcul des gradients locaux pour obtenir l'orientation du descripteur BRISK et les paires ayant de courte distance sont utilisées pour la construction du descripteur BRISK passant par un test binaire.

L'utilisation de la pyramide échelle, la rotation par l'orientation du motif et la forme binaire du descripteur rendent le descripteur BRISK invariant à l'échelle, invariant à la rotation et rapide.

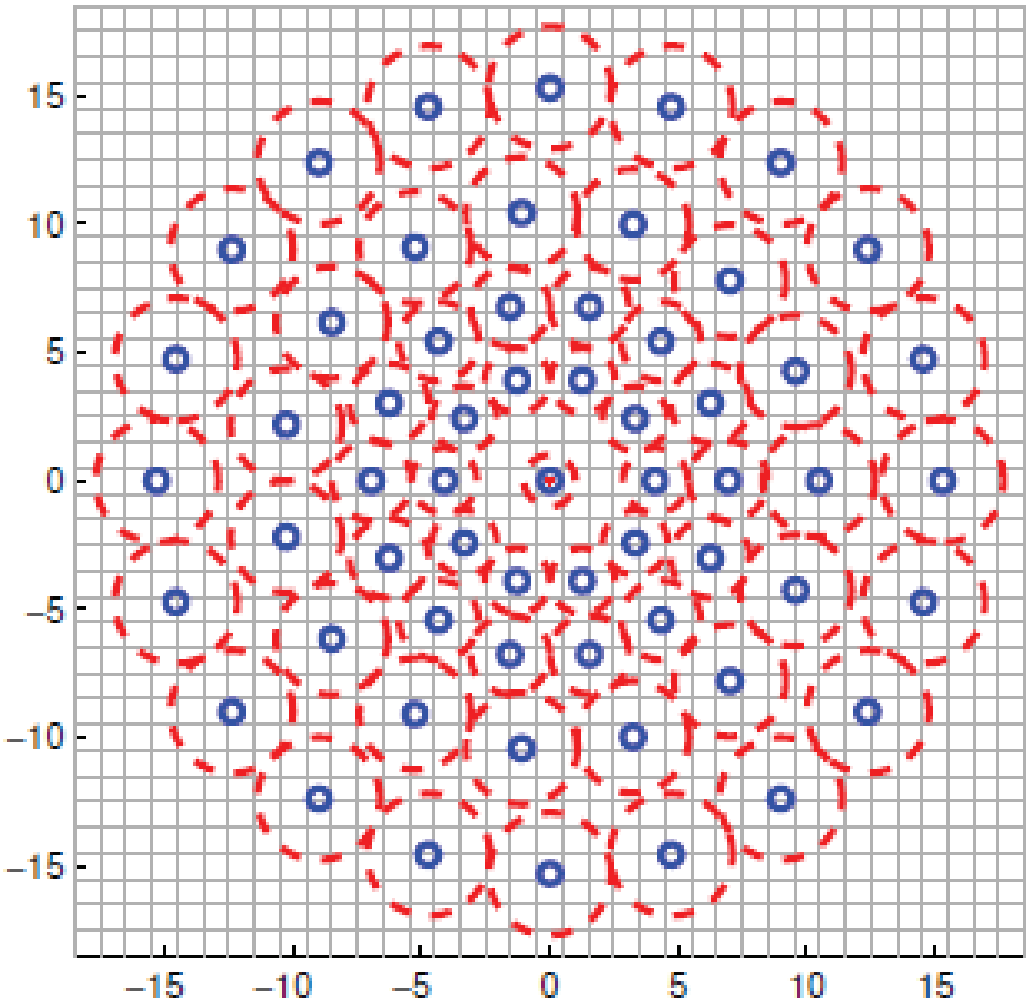


FIGURE A.3: BRISK : échantillonnage pour N = 60 points à l'échelle 1. Illustration prise de [14].

A.11 FREAK

FREAK (Fast Retina Keypoint) est un descripteur de point d'intérêt récemment proposé en 2012 par Alahi et al. [15]. FREAK est inspiré par le système visuel humain et plus précisément la rétine. Selon les auteurs, il est rapide et surpasse les descripteurs SIFT, SURF et BRISK en terme de robustesse.

Il s'agit d'un descripteur de codage binaire obtenu par une cascade de chaînes binaires calculées par simples comparaisons d'intensité basées sur le modèle de la rétine. En effet, l'analogie est obtenue par rapport à la façon dont l'homme peut capturer de grandes informations visuelles dans les régions périphériques de la rétine et de petites informations visuelles dans le centre. Pour simuler le modèle de la rétine, les pixels sont considérés comme les photorécepteurs rétiniens. Le principal intérêt provient de l'extraction rapide des extremas et de l'adaptation aux applications à grande échelle.

Pour comparer l'intensité des pixels du voisinage d'un point d'intérêt, FREAK utilise un échantillonnage basé sur le modèle de la rétine. Comme l'échantillonnage effectué par BRISK, l'échantillonnage effectué par FREAK est également circulaire, la différence est que les points proches du centre du motif ont une densité plus élevée que les points éloignés du centre, figure A.4. Le descripteur est obtenu par seuillage de la différence entre des paires de champs réceptifs et leur noyau Gaussien correspondant. La chaine de bits du descripteur est formée par une séquence de bits représentant les différences de gaussiennes (DoG).

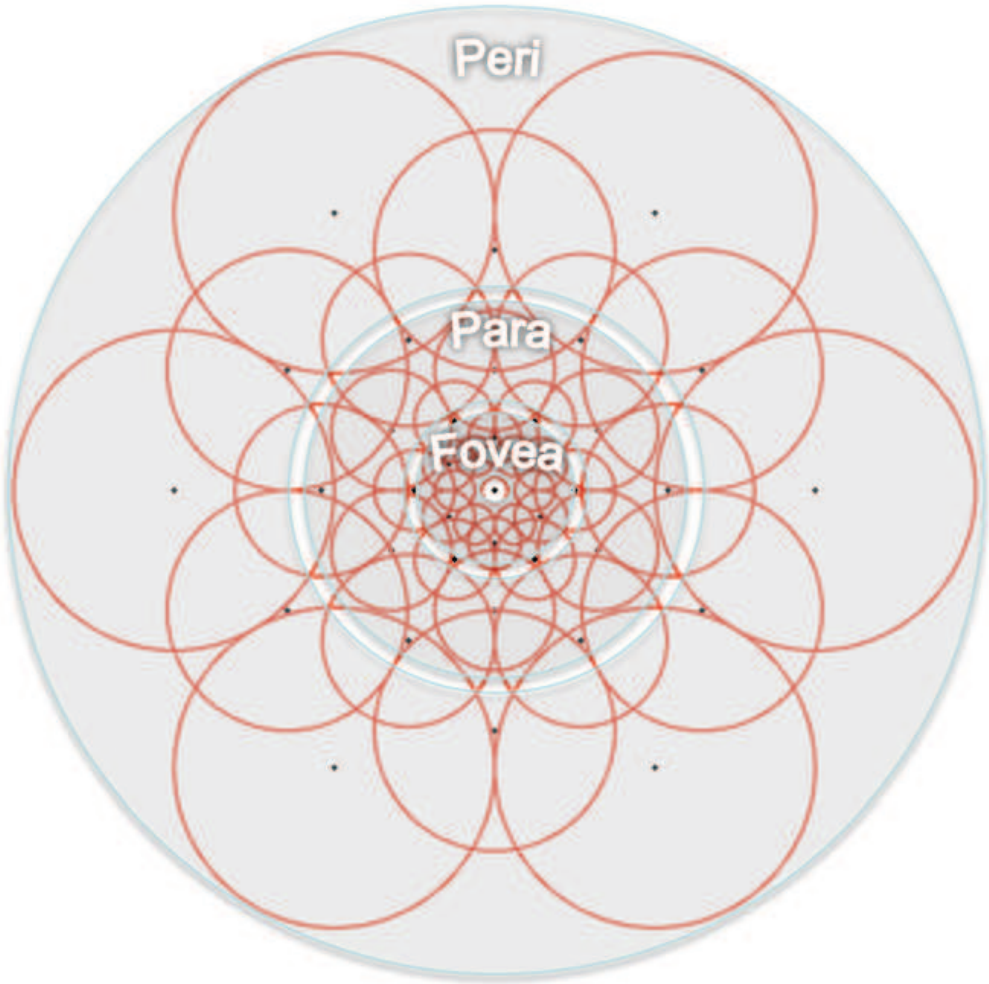


FIGURE A.4: FREAK : échantillonnage du modèle de la rétine [15].

Bibliographie

[1] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry [tutorial]. *Robotics & Automation Magazine, IEEE*, 18(4) :80–92, 2011.

[2] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[3] Giorgio Grisetti, Rainer Kümmerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based slam. *Intelligent Transportation Systems Magazine, IEEE*, 2(4) :31–43, 2010.

[4] Alexandre Eudes. *Localisation et cartographie simultanées par ajustement de faisceaux local : propagation d'erreurs et réduction de la dérive à l'aide d'un odomètre*. PhD thesis, Université Blaise Pascal-Clermont-Ferrand II, 2011.

[5] Davide Scaramuzza. 1-point-ransac structure from motion for vehicle-mounted cameras by exploiting non-holonomic constraints. *International journal of computer vision*, 95(1) :74–85, 2011.

[6] Jean-Philippe Tardif, Yanis Pavlidis, and Kostas Daniilidis. Monocular visual odometry in urban environments using an omnidirectional camera. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2531–2538. IEEE, 2008.

[7] Maxime Meilland. *Cartographie RGB-D dense pour la localisation visuelle temps-réel et la navigation autonome*. PhD thesis, Ecole Nationale Supérieure des Mines de Paris, 2012.

[8] Marcus Svedman, Luis Goncalves, Niklas Karlsson, Mario E. Munich, and Paolo Pirjanian. Structure from stereo vision using unsynchronized cameras for simultaneous localization and mapping. In *IROS*, pages 3069–3074. IEEE, 2005. ISBN 0-7803-8912-3.

[9] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics : The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[10] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving ? the kitti vision benchmark suite. In *Conference on Computer Vision and PatternRecognition (CVPR)*, 2012.

[11] Manfred Klopschitz, Arnold Irschara, Gerhard Reitmayr, and Dieter Schmalstieg. Robust incremental structure from motion. In *Proc. 3DPVT*, volume 2, 2010.

[12] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer, 2006.

[13] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2) :91–110, November 2004. ISSN 0920-5691. doi : 10.1023/B:VISI.0000029664.99615.94.

[14] Stephan Leutenegger, Margarita Chli, and Roland Siegwart. Brisk : Binary robust invariant scalable keypoints. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.

[15] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak : Fast retina keypoint. In *CVPR*, pages 510–517. IEEE, 2012. ISBN 978-1-4673-1226-4.

[16] Rawia Mhiri, Pascal Vasseur, Stephane Mousset, Remi Boutteau, and Abdelaziz Bensrhair. Visual odometry with unsynchronized multi-cameras setup for intelligent vehicle application. In *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, pages 1339–1344. IEEE, 2014.

[17] Rawia Mhiri, Pascal Vasseur, Stéphane Mousset, Rémi Boutteau, and Abdelaziz Bensrhair. Estimation à l'échelle du mouvement d'un réseau multi-caméras non synchronisées. In *Reconnaissance de Formes et Intelligence Artificielle (RFIA) 2014*, 2014.

[18] Rawia Mhiri, Pascal Vasseur, Stéphane Mousset, Rémi Boutteau, and Abdelaziz Bensrhair. Accurate scale estimation based on unsynchronized camera network. In *International Conference on Image Processing (ICIP), 2015 IEEE*. IEEE, 2015.

[19] Rawia Mhiri, Pascal Vasseur, Stéphane Mousset, Rémi Boutteau, and Abdelaziz Bensrhair. Estimation du mouvement et de la structure à l'échelle absolue à partir d'un réseau multi-caméras non synchronisées. In *Journées francophones des jeunes chercheurs en vision par ordinateur*, 2015.

[20] Rawia Mhiri, Hichem Maiza, Stéphane Mousset, Khaled Taouil, Pascal Vasseur, and Abdelaziz Bensrhair. Obstacle detection using unsynchronized multi-camera network. In *International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2015)*, 2015.

[21] M Srinivasan, Shaowu Zhang, M Lehrer, and T Collett. Honeybee navigation en route to the goal : visual flight control and odometry. *The Journal of Experimental Biology*, 199(1) :237–244, 1996.

[22] Clark F Olson, Larry H Matthies, Marcel Schoppers, and Mark W Maimone. Stereo ego-motion improvements for robust rover navigation. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1099–1104. IEEE, 2001.

[23] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652. IEEE, 2004.

[24] Gim Hee Lee, Friedrich Fraundorfer, and Marc Pollefeys. Motion estimation for a self-driving car with a generalized camera. In *CVPR*, 2013.

[25] Paul Furgale, Ulrich Schwesinger, Martin Rufli, Wojciech Derendarz, Hugo Grimmett, Peter Muhlfellner, Stefan Wonneberger, Julian Timpner, Stephan Rottmann, Bo Li, et al. Toward automated driving in cities using close-to-market sensors : An overview of the v-charge project. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 809–816. IEEE, 2013.

[26] David Nistér. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6) :756–770, 2004.

[27] Diego Ortin and JMM Montiel. Indoor robot motion based on monocular images. *Robotica*, 19(03) :331–342, 2001.

[28] Davide Scaramuzza, Friedrich Fraundorfer, and Roland Siegwart. Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 4293–4299. IEEE, 2009.

[29] Chiara Troiani, Alessio Martinelli, Christian Laugier, and Davide Scaramuzza. 1-point-based monocular motion estimation for computationally-limited micro aerial vehicles. In *Mobile Robots (ECMR), 2013 European Conference on*, pages 13–18. IEEE, 2013.

[30] Bo Li, Liang Heng, Gim Hee Lee, and Marc Pollefeys. A 4-point algorithm for relative pose estimation of a calibrated camera with a known relative rotation angle. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1595–1601. IEEE, 2013.

[31] Friedrich Fraundorfer, Petri Tanskanen, and Marc Pollefeys. A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles. In *Computer Vision–ECCV 2010*, pages 269–282. Springer, 2010.

[32] Mahzad Kalantari, Amir Hashemi, Franck Jung, and Jean-Pierre Guédon. A new solution to the relative orientation problem using only 3 points and the vertical direction. *Journal of Mathematical Imaging and Vision*, 39(3) :259–268, 2011.

[33] Oleg Naroditsky, Xun S Zhou, Jean Gallier, Stergios Roumeliotis, Kostas Daniilidis, et al. Two efficient solutions for visual odometry using directional correspondence. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(4) : 818–824, 2012.

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

[35] Quan-Tuan Luong and Olivier D Faugeras. The fundamental matrix : Theory, algorithms, and stability analysis. *International journal of computer vision*, 17 (1) :43–75, 1996.

[36] Olivier D Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig ? In *Computer Vision—ECCV'92*, pages 563–578. Springer, 1992.

[37] Richard I Hartley. Estimation of relative camera positions for uncalibrated cameras. In *Computer Vision—ECCV'92*, pages 579–587. Springer, 1992.

[38] Richard Hartley et al. In defense of the eight-point algorithm. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(6) :580–593, 1997.

[39] H Christopher Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Readings in Computer Vision : Issues, Problems, Principles, and Paradigms, MA Fischler and O. Firschein, eds*, pages 61–62, 1987.

[40] Johan Philip. A non-iterative algorithm for determining all essential matrices corresponding to five point pairs. *The Photogrammetric Record*, 15(88) :589–599, 1996.

[41] Hongdong Li and Richard I. Hartley. Five-point motion estimation made easy. In *ICPR (1)*, pages 630–633. IEEE Computer Society, 2006. ISBN 0-7695-2521-0.

[42] Richard I Hartley and Peter Sturm. Triangulation. *Computer vision and image understanding*, 68(2) :146–157, 1997.

[43] Paul A Beardsley, Andrew Zisserman, and David W Murray. Navigation using affine structure from motion. In *Computer Vision—ECCV'94*, pages 85–96. Springer, 1994.

[44] Carlo Tomasi, John Zhang, and David Redkey. Experiments with a real-time structure-from-motion system. In *Experimental Robotics IV*, pages 195–203. Springer, 1997.

[45] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1) :3–20, 2006.

[46] Feng Lu and Evangelos Milios. Globally consistent range scan alignment for environment mapping. *Autonomous robots*, 4(4) :333–349, 1997.

[47] Edwin Brock Olson, Seth Teller, and John Leonard. *Robust and efficient robotic mapping*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2008.

[48] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g 2 o : A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613. IEEE, 2011.

[49] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John Leonard, and Frank Dellaert. isam2 : Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3281–3288. IEEE, 2011.

[50] Niko Sünderhauf and Peter Protzel. Switchable constraints for robust pose graph slam. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1879–1884. IEEE, 2012.

[51] Maxime Lhuillier. Automatic scene structure and camera motion using a catadioptric system. *Computer Vision and Image Understanding*, 109(2) :186–203, 2008.

[52] Etienne Mouragnon, Maxime Lhuillier, Michel Dhome, Fabien Dekeyser, and Patrick Sayd. Real time localization and 3d reconstruction. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 363–370. IEEE, 2006.

[53] Chris Engels, Henrik Stewénius, and David Nistér. Bundle adjustment rules. In *In Photogrammetric Computer Vision*, 2006.

[54] Eshed Ohn-Bar Alfredo Ramirez and Mohan Trivedi. Panoramic stitching for driver assistance and applications to motion saliency-based risk analysis. In *Intelligent Transportation Systems (ITSC 2013)*. IEEE, 2013.

[55] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment a modern synthesis. In *Vision algorithms : theory and practice*, pages 298–372. Springer, 2000.

[56] Davide Scaramuzza and Roland Siegwart. Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. *Robotics, IEEE Transactions on*, 24(5) :1015–1026, 2008.

[57] Iwan Ulrich and Illah Nourbakhsh. Appearance-based place recognition for topological localization. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 2, pages 1023–1029. Ieee, 2000.

[58] K Jae-Hean and C Myung Jin. Slam with omnidirectional stereo vision sensor. *Proceedings of the IROS, Las Vegas (Nevada)*, 2003.

[59] Matjaž Jogan and Ales Leonardis. Robust localization using panoramic view-based recognition. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 4, pages 136–139. IEEE, 2000.

[60] Friedrich Fraundorfer, Davide Scaramuzza, and Marc Pollefeys. A constricted bundle adjustment parameterization for relative scale estimation in visual odometry. In *IEEE International Conference on Robotics and Automation*, 2010.

[61] Janosch Nikolic. Real-time 6d stereo visual odometry with non-overlapping fields of view. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 1529–1536, Washington, DC, USA, 2012. IEEE Computer Society. ISBN 978-1-4673-1226-4.

[62] DW Pooley, Michael J Brooks, AJ Van Den Hengel, and Wojciech Chojnacki. A voting scheme for estimating the synchrony of moving-camera videos. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, volume 1, pages I–413. IEEE, 2003.

[63] Lior Wolf and Assaf Zomet. Correspondence-free synchronization and reconstruction in a non-rigid scene. In *Proc. Workshop on Vision and Modelling of Dynamic Scenes, Copenhagen*, 2002.

[64] Yaron Caspi and Michal Irani. A step towards sequence-to-sequence alignment. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 682–689. IEEE, 2000.

[65] Marcus Svedman. 3-d structure from stereo vision using unsynchronized cameras. In *Masters thesis, Royal Institute of Technology (KTH*, 2005.

[66] Ahmed Elhayek, Carsten Stoll, Nils Hasler, Kwang In Kim, H-P Seidel, and Christian Theobalt. Spatio-temporal motion tracking with unsynchronized cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1870–1877. IEEE, 2012.

[67] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors : a survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3) : 177–280, 2008.

[68] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Comparing and evaluating interest points. In *Computer Vision, 1998. Sixth International Conference on*, pages 230–235. IEEE, 1998.

[69] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10) :1615–1630, 2005.

[70] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and Luc Van Gool. A comparison of affine region detectors. *International journal of computer vision*, 65(1-2) :43–72, 2005.

[71] Steffen Gauglitz, Tobias Höllerer, and Matthew Turk. Evaluation of interest point detectors and feature descriptors for visual tracking. *International journal of computer vision*, 94(3) :335–360, 2011.

[72] Jared Heinly, Enrique Dunn, and Jan-Michael Frahm. Comparative evaluation of binary features. In *Computer Vision–ECCV 2012*, pages 759–773. Springer, 2012.

[73] Ondrej Miksik and Krystian Mikolajczyk. Evaluation of local detectors and descriptors for fast feature matching. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 2681–2684. IEEE, 2012.

[74] Akash Patel, DR Kasat, Sanjeev Jain, and VM Thakare. Performance analysis of various feature detector and descriptor for real-time video based face tracking. *International Journal of Computer Applications*, 93(1), 2014.

[75] Adam Schmidt, Marek Kraft, and Andrzej Kasiński. An evaluation of image feature detectors and descriptors for robot navigation. In *Computer Vision and Graphics*, pages 251–259. Springer, 2010.

[76] Friedrich Fraundorfer and Davide Scaramuzza. Visual odometry : Part ii : Matching, robustness, optimization, and applications. *Robotics & Automation Magazine, IEEE*, 19(2) :78–90, 2012.

[77] Paul R Beaudet. Rotationally invariant image operators. In *International Joint Conference on Pattern Recognition*, volume 579, page 583, 1978.

[78] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22 (10) :761–767, 2004.

[79] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (11) :1254–1259, 1998.

[80] Pierre Moreels and Pietro Perona. Evaluation of features detectors and descriptors based on 3d objects. *International Journal of Computer Vision*, 73(3) :263–284, 2007.

[81] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. *International Journal of computer vision*, 37(2) :151–172, 2000.

[82] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief : Binary robust independent elementary features. *Computer Vision–ECCV 2010*, pages 778–792, 2010.

[83] Igor Cvišić and Ivan Petrović. Stereo odometry based on careful feature selection and tracking. In *European Conference on Mobile Robots (ECMR)*, 2015.

[84] Alexandre Eudes, Sylvie Naudet-Collette, Maxime Lhuillier, and Michel Dhome. Weighted local bundle adjustment and application to odometry and visual slam fusion. 2010.

[85] Sebastian Scherer, Andreas Zell, et al. Efficient onbard rgbd-slam for autonomous mavs. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1062–1068. IEEE, 2013.

[86] Kenneth Levenberg. A method for the solution of certain problems in least squares. *Quarterly of applied mathematics*, 2 :164–168, 1944.

[87] Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5-6) :403–429, 2006.

[88] Andrew W Fitzgibbon and Andrew Zisserman. Automatic camera recovery for closed or open image sequences. In *Computer Vision—ECCV'98*, pages 311–326. Springer, 1998.

[89] Christopher Zach, Arnold Irschara, and Horst Bischof. What can missing correspondences tell us about 3d structure and motion ? In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[90] Olivier D Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig ? In *Computer Vision—ECCV'92*, pages 563–578. Springer, 1992.

[91] Zhengyou Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11) :1330–1334, 2000.

[92] John Franklin Reid. Obstacle detection using stereo vision, July 24 2007. US Patent 7,248,968.

[93] Raphael Labayrade, Didier Aubert, and Jean-Philippe Tarel. Real time obstacle detection in stereovision on non flat road geometry through" v-disparity" representation. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 646–651. IEEE, 2002.

[94] Zohir Irki, Hamza Bendaoudi, Michel Devy, and Abdelhakim Khouas. Fpga implementation of the v-disparity based obstacles detection approach. In *Control & Automation (MED), 2013 21st Mediterranean Conference on*, pages 1104–1111. IEEE, 2013.

[95] Alberto Broggi, Claudio Caraffi, Rean Isabella Fedriga, and Paolo Grisleri. Obstacle detection with stereo vision for off-road vehicle navigation. In *Computer Vision and Pattern Recognition-Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, pages 65–65. IEEE, 2005.

[96] Zehang Sun, George Bebis, and Ronald Miller. On-road vehicle detection : A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(5) : 694–711, 2006.

[97] Nicolas Soquet, Didier Aubert, and Nicolas Hautiere. Road segmentation supervised by an extended v-disparity algorithm for autonomous navigation. In *Intelligent Vehicles Symposium, 2007 IEEE*, pages 160–165. IEEE, 2007.

[98] Alberto Broggi, Claudio Caraffi, Pier Paolo Porta, and Paolo Zani. The single frame stereo vision system for reliable obstacle detection used during the 2005

darpa grand challenge on terramax. In *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*, pages 745–752. IEEE, 2006.

[99] Min Zhang, Peizhi Liu, Xiaochuan Zhao, Xinxin Zhao, and Yuan Zhang. An obstacle detection algorithm based on uv disparity map analysis. In *Information Theory and Information Security (ICITIS), 2010 IEEE International Conference on*, pages 763–766. IEEE, 2010.

[100] Angel D Sappa, Rosa Herrero, Fadi Dornaika, David Gerónimo, and Antonio López. Road approximation in euclidean and v-disparity space : a comparative study. In *Computer Aided Systems Theory–EUROCAST 2007*, pages 1105–1112. Springer, 2007.

[101] Zhencheng Hu and Keiichi Uchimura. Uv-disparity : an efficient algorithm for stereovision based scene analysis. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 48–54. IEEE, 2005.

[102] Zhencheng Hu, Francisco Lamosa, and Keiichi Uchimura. A complete uv-disparity study for stereovision based 3d driving environment analysis. In *3-D Digital Imaging and Modeling, 2005. 3DIM 2005. Fifth International Conference on*, pages 204–211. IEEE, 2005.

[103] Hough Paul VC. Method and means for recognizing complex patterns, December 18 1962. US Patent 3,069,654.

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

[105] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[106] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International journal of computer vision*, 60(1) :63–86, 2004.

[107] Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas. Censure : Center surround extremas for realtime feature detection and matching. In *Computer Vision–ECCV 2008*, pages 102–115. Springer, 2008.

[108] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1508–1515. IEEE, 2005.

[109] Stephen M Smith and J Michael Brady. Susan—a new approach to low level image processing. *International journal of computer vision*, 23(1) :45–78, 1997.

[110] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1) :81–106, 1986.

[111] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[112] Simon Taylor, Edward Rosten, and Tom Drummond. Robust feature matching in 2.3μs. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 15–22. IEEE, 2009.

[113] Elmar Mair, Gregory D Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Computer Vision–ECCV 2010*, pages 183–196. Springer, 2010.

[114] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0164-8.

[115] Yan Ke and Rahul Sukthankar. Pca-sift : A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–506. IEEE, 2004.

[116] Yan Ke and Rahul Sukthankar. Pca-sift : A more distinctive representation for local image descriptors. In *CVPR (2)*, pages 506–513, 2004.

[117] Jean-Michel Morel and Guoshen Yu. Asift : A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2) :438–469, 2009.

[118] Sternwartstrasse 7 CH-8092 Zurich Switzerland Herbert Bay ETH Zurich, BIWI. Surf : Speeded up robust features. *Computer Vision and Image Understanding archive*, Volume 110 Issue 3, :346–359, June, 2008.

[119] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb : an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.