HAL
open science

Système de reconstruction d'environnement pour une aide au pilotage en environnement naturel

Bruno Ricaud

To cite this version:

Bruno Ricaud. Système de reconstruction d'environnement pour une aide au pilotage en environnement naturel. Robotique [cs.RO]. Université Paris sciences et lettres, 2016. Français. NNT : 2016PSLEM018 . tel-01502826

HAL Id: tel-01502826

https://pastel.hal.science/tel-01502826v1

Submitted on 6 Apr 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT

de l'Université de recherche Paris Sciences et Lettres
PSL Research University

Préparée à MINES ParisTech

SYSTÈME DE RECONSTRUCTION D'ENVIRONNEMENT POUR UNE AIDE AU PILOTAGE EN ENVIRONNEMENT NATUREL

Ecole doctorale n°432

SCIENCES DES MÉTIERS DE L'INGÉNIEUR

Spécialité INFORMATIQUE TEMPS-RÉEL, ROBOTIQUE ET MATHÉMATIQUE

COMPOSITION DU JURY :

M. Philippe BONNIFAIT
Sorbonne Universités, Université de Technologie de Compiègne, Président

M. Patrick RIVES
INRIA, Rapporteur

M. Roland CHAPUIS
UNIVERSITE BLAISE PASCAL - CLERMONT-FERRAND II, Rapporteur

M. Arnaud de LA FORTELLE
MINES ParisTech, Examinateur

M. Cyril JOLY
MINES ParisTech, Examinateur

M. Jimmy REGNIER
Nexter Systems, Examinateur

Soutenue par Bruno RICAUD
le 20 juin 2016

Dirigée par **Arnaud de LA FORTELLE**

Remerciements

Je souhaite remercier toutes les personnes qui ont contribué à la réalisation de cette thèse.

Tout d'abord, j'exprime mes plus sincères remerciements aux membres de mon jury :
P. BONNIFAIT, P. RIVES, R. CHAPUIS, A. de LA FORTELLE, C. JOLY, J. REGNIER.

Aussi, je remercie plus particulièrement A. de LA FORTELLE, Directeur du centre de robotique de MINES ParisTech. En tant que directeur de thèse, il a guidé mes réflexions et ma rédaction. Je remercie également C. JOLY, enseignant chercheur à MINES ParisTech. Son encadrement a aiguillé mon travail et son expertise a permis d'atteindre le niveau auquel il est présenté.

Ensuite, je remercie Nexter Systems en les personnes de J-M. REMBLIERE, Directeur de l'ingénierie des systèmes, J. REGNIER, Architecte Vétronique et G. BETTETO, anciennement Architecte Vétronique qui ont su croire en moi et dans ce travail de recherche.

Je remercie également tous mes collègues de Nexter Systems et mes camarades de MINES ParisTech qui ont participé par des échanges et par leur soutien à la réussite de cette thèse.

Je remercie ma famille et mes amis pour leur soutien qui m'a permis d'arriver au bout de cette thèse. Je souhaite aussi remercier particulièrement mes grands-pères, qui par leur exemple m'ont poussé à entreprendre une thèse.

Enfin, je remercie Bénédicte pour l'amour et la bienveillance dont elle fait preuve envers moi au quotidien. Elle a su me donner le courage d'entamer et de poursuivre ce travail jusqu'à son achèvement.

Table des matières

1 Introduction **1**
1.1 Contexte industriel . . . 3
1.1.1 Nexter Systems . . . 3
1.1.2 La Vétronique Nexter Systems . . . 5
1.1.3 Contraintes industrielles . . . 5
1.2 Contexte scientifique . . . 6
1.2.1 Les véhicules autonomes . . . 6
1.2.2 MINES ParisTech — Centre de Robotique (CAOR) . . . 7
1.2.3 De la navigation autonome à l'aide au pilotage . . . 7
1.3 Contributions . . . 8
1.3.1 Sécurisation du pilotage par perception d'environnement . . . 8
1.3.2 Contributions . . . 8
1.4 Organisation du manuscrit . . . 10
1.4.1 Détail de l'organisation . . . 10

I État de l'art et réponse matérielle **13**

2 État de l'art **17**
2.1 L'aide au pilotage . . . 18
2.1.1 Les systèmes civils . . . 18
2.1.2 Les systèmes militaires . . . 21
2.2 Cartographie mobile . . . 23
2.2.1 La cartographie . . . 24
2.2.2 Positionnement . . . 24
2.3 Cartographie par méthode SLAM . . . 31
2.3.1 Principe . . . 31
2.3.2 Algorithmes de filtrage associés au SLAM . . . 34
2.3.3 SLAM : correction d'erreurs et simplification . . . 39
2.3.4 Conclusion sur la cartographie . . . 41
2.4 Analyse du terrain . . . 41
2.4.1 Franchissement : une définition approfondie des obstacles . . . 42
2.4.2 Détection d'obstacles . . . 47
2.5 Conclusion sur la définition de notre système . . . 52

3 Définition des capteurs de notre système **55**
3.1 Capteurs passifs . . . 56
3.1.1 Caméras monoculaires . . . 56
3.1.2 Caméras stéréoscopiques . . . 58
3.2 Capteurs actifs . . . 63
3.2.1 Caméra 3D active . . . 63
3.2.2 Télémètres . . . 66
3.3 Conclusion . . . 70
3.4 Conclusion sur la définition de notre système . . . 74

II Système de perception d'un environnement naturel 75

4 Une plateforme mobile d'expérimentations 79
4.1 Création de la plateforme . . . 80
4.1.1 Le choix de la plateforme mobile . . . 80
4.1.2 Le système de communication ROS . . . 81
4.1.3 Le contrôle de plateforme . . . 82
4.2 Les capteurs . . . 84
4.2.1 Caméras . . . 84
4.2.2 Laser 3D à balancier . . . 84
4.2.3 Calibration . . . 88
4.3 Conclusion . . . 91

5 Reconstruction par données LIDAR et odométrie visuelle 93
5.1 Acquisition de données LIDAR et odométrie stéréoscopique . . . 94
5.1.1 Reconstruction par acquisition de données LIDAR et odométrie visuelle . . . 94
5.1.2 Résultats . . . 99
5.1.3 Conclusion . . . 109
5.2 Optimisation par ajout d'une estimation de la rotation . . . 109
5.2.1 Initialisation par estimation de la rotation . . . 109
5.2.2 Etude du déplacement d'une image entre deux instants . . . 110
5.2.3 Résultats . . . 120
5.2.4 Conclusion quant à l'estimation de la rotation . . . 125
5.3 Conclusion quant à notre algorithme . . . 127

6 Segmentation des données acquises 129
6.1 Segmentation d'obstacles dans le nuage de points . . . 129
6.1.1 Maniement du nuage de points . . . 130
6.1.2 Segmentation du sol . . . 131
6.1.3 Segmentation des obstacles . . . 131
6.1.4 Conclusion de la segmentation du nuage de points . . . 133
6.2 Segmentation de chemin . . . 133
6.2.1 Segmentation de chemin par détection du point de fuite . . . 134
6.2.2 Optimisation pour rendu temps réel . . . 138
6.2.3 Résultats . . . 140
6.3 Conclusion de la segmentation des données acquises . . . 145
6.4 Conclusion sur le système de perception d'un environnement naturel . . . 145

III Restitution à l'opérateur 147

7 Pilotage à l'aide d'un casque de réalité virtuelle 151
7.1 Moyens de restitutions . . . 151
7.1.1 La situation actuelle dans les véhicules . . . 152
7.1.2 Le casque de réalité virtuelle . . . 153
7.2 Méthodologie de l'expérience . . . 155
7.2.1 Hypothèse . . . 155
7.2.2 Variables . . . 155
7.2.3 Protocole expérimental . . . 155
7.2.4 Tâches . . . 157
7.3 Résultats . . . 160
7.3.1 Résultats et analyse . . . 160
7.4 Conclusion . . . 161

8 Conclusion et perspectives **165**
8.1 Conclusion 165
8.2 Perspectives 167

A Publications **169**

B Justification des approximations de l'estimation monoculaire de la rotation **171**
A.1 Approximations des petits angles 171
B.2 Les translations sont dépendantes de la distance au points 173
C.3 Le roulis négligé 173

Bibliographie **175**

Chapitre 1

Introduction

Sommaire

1.1 Contexte industriel **3**
1.1.1 Nexter Systems 3
1.1.2 La Vétronique Nexter Systems 5
1.1.3 Contraintes industrielles 5
1.2 Contexte scientifique **6**
1.2.1 Les véhicules autonomes 6
1.2.2 MINES ParisTech — Centre de Robotique (CAOR) 7
1.2.3 De la navigation autonome à l'aide au pilotage 7
1.3 Contributions **8**
1.3.1 Sécurisation du pilotage par perception d'environnement 8
1.3.2 Contributions 8
1.4 Organisation du manuscrit **10**
1.4.1 Détail de l'organisation 10

Qu'ils aient lieu sur une route, dans un entrepôt, ou sur un terrain d'opération extérieur, les accidents mortels ne sont pas acceptables. Depuis un grand nombre d'années, des recherches sont menées afin de limiter les risques d'accidents dans le cadre de la sécurité routière, de la sécurité du travail et de la défense. En effet, les véhicules motorisés sont source de beaucoup d'accidents dont un pourcentage non négligeable d'accidents mortels. Sur les routes de France et d'Europe, la sécurité routière est un sujet prédominant en matière de politique et de recherche, mais ce sujet ne se limite pas au cadre civil : des accidents de pilotage de véhicules militaires interviennent lors de manœuvres d'exercice et le risque est d'autant plus fort sur le terrain d'opération.

Même si dans le contexte militaire le concept de « zéro mort » [Combelles-Siegel and Géré, 2001] est considéré comme absurde, il est politiquement et médiatiquement prédominant [Dupire, 2012; Lasserre, 2012]. Ce concept est d'ailleurs très valorisé par une opinion publique requérant le retour des forces armées dès la première perte humaine. Les leçons tirées de la guerre en Afghanistan ont mis en exergue une nécessité d'augmenter le niveau de protection, et ce, même sur les véhicules les plus récents. De plus, pour l'opinion publique, les pertes accidentelles sont encore plus difficilement acceptables que des soldats morts au combat.

Cette sécurisation des véhicules, qu'ils soient civils ou militaires, se décline en plusieurs axes. Le premier est bien sûr la formation des conducteurs et des pilotes afin qu'ils maîtrisent au mieux leur véhicule. Le second concerne les « aides au pilotage » améliorant le contrôle du véhicule. Le dernier a trait aux aides au pilotage améliorant la perception de l'environnement voisin du véhicule et c'est de ce point que traite la présente thèse.

La numérisation de l'environnement devient une obligation pour les constructeurs de matériels militaires, car, depuis 2004, 78 accidents de pilotage mortels ont eu lieu dans le monde lors de manœuvres de l'OTAN [Icasualties, 2015], alors même que les véhicules sont équipés d'un grand nombre d'aides au contrôle et que les pilotes sont experts de leur matériel.

FIGURE 1.1 – Le Nexter Aravis (VBHP) ouvre l'itinéraire aux autres véhicules (VAB). Source : Nexter

Cette thèse s'inscrit dans la problématique **de sécuriser le pilotage des véhicules militaires et a comme périmètre la définition d'un système de perception d'environnement pour l'aide au pilotage en environnement naturel pour les véhicules blindés produits par Nexter.**

Dans le cas de pilotage en milieu naturel, les faibles surfaces vitrées des véhicules sont gênantes pour manœuvrer (Figure 1.1). L'opérateur peut difficilement acquérir de l'information sur le terrain proche. Son seul moyen de visualisation est un écran de petite taille lui permettant de naviguer entre les images des caméras placées autour du véhicule. En milieu structuré comme une ville ou un village, l'opérateur aura certainement une carte *a priori* de la zone qu'il doit traverser, mais dans le cas d'un milieu hostile, d'un terrain de conflit ou d'un environnement ayant subi une catastrophe naturelle, les cartes 2D ne sont pas suffisantes pour piloter de manière sécurisée. Des rues peuvent être barrées, détruites ou encombrées, l'opérateur devra alors modifier son itinéraire ou au moins l'adapter pour pouvoir évoluer dans ce terrain. Pour ce faire, il utilisera ses connaissances du gabarit du véhicule pour estimer s'il est capable de franchir un amas de gravats, d'éviter un obstacle ou d'outrepasser un cratère. Ces tâches sont déjà difficiles en situation nominale, mais dans les moments de stress intense, comme en période de contact armé, l'ensemble des procédures peut finir par surcharger cognitivement l'opérateur qui perd en efficacité et peut faire des erreurs. Malheureusement, dans ce contexte, une erreur peut être fatale pour le pilote et les passagers de son véhicule. Ainsi, aider le pilote dans certaines tâches d'analyse de l'environnement où évolue le véhicule peut permettre de sauver des vies et de mener la mission à son terme.

Sécuriser le pilotage entend dans notre cas offrir les informations nécessaires au pilote afin qu'il puisse manœuvrer en toute sécurité son véhicule. Cependant, ces informations ne doivent pas être trop nombreuses pour ne pas distraire le pilote de sa fonction première : piloter.

L'environnement opérationnel est un environnement complexe, c'est pourquoi une aide au pilotage percevant les dangers se trouvant sur le chemin du véhicule peut apporter une aide cruciale au pilote. Ces dangers peuvent être des objets, des véhicules ou des personnes se trouvant sur le chemin, mais aussi des irrégularités du sol pouvant mettre le véhicule en difficulté et ses occupants en danger. Ce problème commun aux domaines civil et militaire trouve bon nombre de solutions dans le domaine civil, mais les contraintes militaires sont telles que ces solutions ne sont pas transpo-

sables. En effet, la régularité des routes permet de mettre en jeu beaucoup d'hypothèses simplifiant la perception d'environnement. Ces hypothèses, comme le « monde plan » ou les solutions comme le « suivi de lignes » ne sont pas applicables sur un terrain chaotique où le chemin n'est pas forcément prévu pour le passage d'un véhicule.

Afin d'étudier ce sujet, une thèse CIFRE a été lancée en partenariat entre Nexter Systems et le Centre de Robotique de MINES ParisTech. La suite de ce chapitre va expliquer le contexte industriel plus en détail et les contraintes qui lui sont liées, ainsi que le contexte scientifique qui en découle. Ensuite, la problématique de la thèse sera présentée. Enfin, le plan du manuscrit sera détaillé.

1.1 Contexte industriel

Cette thèse CIFRE est un partenariat entre Nexter Systems — Architecture Vétronique et le Centre de Robotique de MINES ParisTech. Elle a débuté en décembre 2012. Le Directeur de thèse est Arnaud de La Fortelle et les maîtres de thèse sont Cyril Joly et Bogdan Stanciulescu. À Nexter Systems, l'encadrement est dispensé par Gilles Betteto et Jimmy Régnier. Étudions maintenant Nexter Systems et son engagement dans la sécurisation de ses véhicules.

1.1.1 Nexter Systems

Nexter Systems est une filiale du groupe d'armement français Nexter, lui-même descendant d'une longue lignée de manufactures d'armes, du récent GIAT aux très anciens arsenaux français (1667) [Nexter, 2013]. Son catalogue présente une dizaine de véhicules militaires, pour certains vendus et pour d'autres seulement maintenus, mais aussi un grand nombre d'équipements. [Nexter, 2015a] (Figure 1.3 page 5).

L'évolution des menaces, des mentalités et des conflits a modifié la manière de concevoir les véhicules militaires. Depuis de nombreuses années, la valeur stratégique des soldats augmente. Cette évolution est expliquée d'une part par un changement de mentalité, et d'autre part par la diminution du nombre de militaires de carrière. Cette problématique a influencé le développement des récents produits Nexter. L'Aravis présenté dans la section suivante est une démonstration de la volonté du groupe Nexter de protéger les soldats dans les situations les plus difficiles.

1.1.1.1 L'Aravis, un exemple de protection

L'**Aravis** (Figure 1.1) est un véhicule tout-terrain français de transport de troupes à quatre roues motrices conçu et fabriqué par Nexter Systems sur fonds propres. Ce véhicule sert essentiellement à l'ouverture d'itinéraire. Sa désignation dans l'armée française est VBHP (Véhicule Blindé Hautement Protégé). Comme son nom l'indique, ce VBHP Aravis est hautement protégé par un blindage correspondant au standard STANAG 4569 [NSA, 2012] niveau 4 qui reflète un très haut niveau de protection pour ses occupants. Le standard STANAG 4569 noté sur 6 niveaux, évalue la capacité de l'équipement à protéger les occupants de véhicules logistiques et de véhicules légèrement blindés vis-à-vis des attaques cinétiques (balles), d'artillerie et des explosions d'engins explosifs artisanaux (IED).

De par ses caractéristiques l'Aravis est la preuve que le groupe Nexter s'attache à développer des véhicules répondant aux menaces modernes en mettant en place des solutions innovantes. Dans son Livre Blanc, le gouvernement français s'attache aussi à répondre à cette problématique.

(a) Le Griffon. (b) Le Jaguar.

FIGURE 1.2 – Les deux futurs véhicules militaires français. Les premières versions de ces véhicules devraient arriver d'ici 2020. Source : Nexter

1.1.1.2 Les projets du Livre Blanc de la Défense

Avec le Livre Blanc de la Défense [République-Française, 2013], le gouvernement a officialisé le programme Scorpion [1] de renouvellement des blindés de l'armée de terre. Ce projet a pour but de remplacer toute l'ancienne génération de matériels, à savoir les VAB, les AMX 10RC, et les ERC 90. La première tranche de ce projet, évaluée à 5 milliards d'euros pour la période 2015-2025, envisage le développement de deux véhicules : le Griffon et le Jaguar. Ces projets seront réalisés dans le cadre d'un partenariat entre les groupes Nexter, Renault Truck Defense et Thales.

Le Griffon (Figure 1.2a), Véhicule Blindé Multi-Rôles (VBMR), sera un véhicule à six roues motrices d'un poids d'environ 24 tonnes. Il sera équipé d'armements similaires à ceux déployés sur l'Aravis et il sera décliné en six versions : transport de troupes, sanitaire, poste de commandement, observation d'artillerie, transport de mortier et reconnaissance de zones contaminées (NRBC). Il est prévu pour accueillir des fantassins équipés de l'attirail FÉLIN [Defense.gouv.fr, 2015b]. Cet attirail rend le soldat beaucoup plus performant grâce à différentes aides au combat dans son système d'arme et dans sa tenue de protection.

Le Jaguar (Figure 1.2b) devrait aussi être un véhicule à six roues motrices de 24 tonnes. Dénommé précédemment « Engin Blindé de Reconnaissance et de Combat » (EBRC), il sera équipé d'un armement plus lourd que le Griffon, avec une tourelle télé-opérée de 40 mm développée par Nexter et des missiles antichars.

La modularité est un point crucial dans la conception de ces véhicules. En outre, la Direction Générale de l'Armement (DGA) assure l'importance apportée à leur autonomie. Ces véhicules sont prévus pour pouvoir parcourir 800 kilomètres et tenir 72 heures en combat. Bien que pensés pour une utilisation française, ils seront présentés aux acheteurs étrangers.

De plus, un nouveau système de communication appelé SICS pour « Système d'Information du Combat Scorpion », devrait être intégré à ces nouveaux véhicules. Ce système d'aide au commandement est conçu pour donner des informations datant de moins de 10 secondes aux opérateurs dans les véhicules.

1. Le programme scorpion : www.defense.gouv.fr/terre/equipements/scorpion/le-programme-scorpion

FIGURE 1.3 – Des produits Nexter : Char Leclerc (au premier plan), VAB (entre les colonnes de chars) et VBL (au dernier plan). Source : Nexter

En plus de ces avancées technologiques sur le plan de l'autonomie et de l'armement, ces véhicules disposeront de tout le savoir-faire du groupe Nexter relatif à la protection des occupants. La protection n'est pas seulement fournie par la caisse du véhicule, mais aussi par l'électronique embarquée, appelée vétronique, développée par Nexter Systems.

1.1.2 La Vétronique Nexter Systems

La vétronique (vetronics en anglais) correspond à un terme générique englobant tous les systèmes électroniques d'un véhicule [Abdulmasih and Oikonomidis, 2011]. Nexter Systems travaille entre autres à la définition des logiciels de commandement, de tir, d'observation, de pilotage et de gestion des équipements des véhicules ainsi qu'à la définition de ces équipements. Toutes ces fonctions sont reliées grâce à leur produit « Integrated Modular Vetronics » (IMV). Basé sur la technologie OpenSplice DDS de Thales — PrismTech [PrismTech, 2015], ce système permet la reconfiguration de l'électronique, des logiciels et des interfaces du véhicule en fonction des équipements présents. Ainsi, l'impact de la dégradation des équipements du système est réduit et les fonctions sont assurées même avec un nombre d'équipements restreint. À l'instar de ROS (voir chapitre 4.1.2 page 81) pour la robotique, DDS utilise un système d'abonnement entre les modules leur permettant de discuter ou interagir entre eux sans pour autant être dépendant les uns aux autres. Ce système a deux objectifs :

- Analyser les événements intervenants sur le réseau.
- S'adapter à ces événements et se reconfigurer en un minimum de temps.

Bien que des liens évidents apparaissent entre les véhicules civils et militaires, l'explication précédente sur l'utilisation de ces véhicules permet d'en cerner les différences. Ces différences engendrent d'importantes contraintes dont nous allons faire l'état dans la section suivante.

1.1.3 Contraintes industrielles

Les systèmes militaires ne sont pas soumis aux mêmes contraintes que les systèmes civils. Une mise au point préalable a été établie au sein de Nexter Systems vis-à-vis de la recherche entamée :

- Le système doit être très robuste à tout type d'environnements et de situations. En effet, les systèmes militaires doivent être robustes aux conditions climatiques et à un environnement

vibratoire important causé, entre autres, par les coups de canon. Ils sont soumis à des normes similaires à la norme DO160F [135, 2011] appliquée en aéronautique. De plus, les environnements n'étant pas réguliers, le système doit pouvoir détecter les dangers sur la trajectoire du véhicule sans se baser sur des hypothèses de régularité de l'environnement.

- La sûreté de fonctionnement n'est pas la même en mode Paix et en mode Guerre. Le mode Paix, comparable à un mode de conduite civile est à l'opposé du mode Guerre où la moindre gêne peut mettre en péril la vie des occupants du véhicule. Le système ne doit donc pas contraindre le pilote en mode Guerre.
- Le système ne doit pas demander un effort de la part des opérateurs pour se l'approprier et l'utiliser. Il doit donner les bonnes informations au bon moment et ne doit pas demander de paramétrage ou de calibration particulier de la part des opérateurs.
- La place dans les véhicules militaires étant limitée, les fonctionnalités sont regroupées sur un même calculateur. Les autres systèmes (vidéo, GPS, audio, commandement) fonctionneront en même temps que les algorithmes d'aide au pilotage : ils ne doivent pas pâtir du lancement de nos traitements. De plus le stockage étant limité à quelques gigas par module, il n'est pas envisageable actuellement de stocker l'ensemble des traitements réalisés au cours d'une mission.
- La projection du système en milieu opérationnel implique que l'accent soit mis sur des solutions valorisant la discrétion des éléments externes (typiquement les capteurs).

Enfin, les militaires, qu'ils soient opérateurs ou dirigeants ne souhaitent pas léguer le contrôle de leurs véhicules à des ordinateurs [Guibert, 2011]. Cependant, les avancées réalisées pour la perception de l'environnement dans le cadre de l'automatisation de véhicules sont considérables, ainsi notre système utilisera les connaissances et les avancées apportées dans le cadre de l'automatisation de véhicules pour répondre à la problématique d'aide au pilotage, en réalisant de l'affichage au pilote.

1.2 Contexte scientifique

Le contexte industriel permet d'appréhender le besoin lié aux situations inhabituelles de notre contexte opérationnel. Ainsi, la décision de s'appuyer sur des techniques propres aux véhicules autonomes a poussé Nexter Systems à s'allier à des experts dans le domaine de l'automatisation de véhicules civils.

1.2.1 Les véhicules autonomes

Les publications scientifiques récentes nous montrent qu'un grand nombre de recherches sont réalisées dans le domaine de la perception d'environnement avec des applications sur les véhicules autonomes afin de les rendre plus « conscients » de leur environnement. En effet, rendre les véhicules « intelligents » est actuellement un sujet majeur pour la recherche mondiale de par les applications qu'elle laisse envisager, que ce soit dans l'optique de la sécurité des passagers [Parent, 2004], de la gestion du trafic urbain [Grégoire, 2014] ou de l'optimisation de la consommation des ressources [Fritz et al., 2004]. Des études ont été menées en milieu naturel, notamment dans le cadre de la robotique d'exploration planétaire [Mallet, 2001], ou pour l'envoi de robots militaires [Larson and Trivedi, 2011a], mais la plupart des recherches se concentrent sur l'analyse de zones urbaines pour l'automatisation des véhicules civils. Le « Defense Advanced Research Projects Agency (DARPA) Grand Challenge » [Broggi et al., 2010] a permis à beaucoup de laboratoires de monter en compétences dans le domaine de l'automatisation de véhicules. À la suite de ce défi, des projets ont été développés par le DARPA pour la réalisation de véhicules autonomes militaires ; le Crusher, et le Black Knight en sont des exemples [Winslow, 2007]. Ces projets de recherche ont permis de créer des systèmes réalisant des missions en environnement naturel sur de longues distances, de manière

quasi autonome, comme le véhicule développé par le laboratoire VisLab de l'université de Parme, le VIAC [Bertozzi et al., 2011]. Malgré tout, rares sont les laboratoires français s'intéressant aux recherches relatives aux véhicules militaires terrestres [Monnin, 2007].

Le Centre de Robotique de MINES ParisTech (CAOR) s'est intéressé à cette problématique par le biais de ses recherches en robotique mobile [Steux and ElHamzaoui, 2010], et s'est illustré lors de la compétition CAROTTE [Stic et al., 2011] organisée par la DGA de 2010 à 2012.

1.2.2 MINES ParisTech — Centre de Robotique (CAOR)

Le Centre de Robotique de MINES ParisTech (CAOR) travaille avec les sociétés du domaine de la défense, mais la plupart de ses recherches sont liées à des problématiques civiles. Un axe important des travaux du CAOR est la réflexion et la mise au point d'outils, d'algorithmes et de capteurs dans le cadre de la conduite automatisée. De nombreuses applications ont ainsi été développées dans le domaine des Systèmes de Transport Intelligents (ITS). De plus, des outils et algorithmes performants de reconnaissance visuelle et de catégorisation d'objets (piétons, voitures, visages, etc) ont été développés dans ce contexte [Zaklouta, 2011; Moutarde and Stanciulescu, 2008]. Au travers de cette thèse, l'expertise apportée par le Centre de Robotique dans le domaine de l'automatisation de véhicules en milieu urbain est mise à profit pour les milieux naturels et semi-structurés avec contraintes militaires. Ces contraintes militaires obligent certes à modifier les capteurs et méthodes utilisés, mais n'effacent pas l'expérience acquise dans les recherches réalisées en robotique mobile [Steux and ElHamzaoui, 2010; Wirbel et al., 2013; Espino-Corsino et al., 2011], en cartographie mobile [Poreba and Goulette, 2015; Marcotegui and Deschaud, 2012], ou en communication inter véhicules [A. de La Fortelle, 2012; Luca et al., 2012; Milanes et al., 2011]. C'est donc fort de son expertise dans le domaine que le CAOR a accepté de travailler sur le sujet de l'aide au pilotage pour les véhicules blindés en environnement naturel ou semi-structuré.

De grands progrès ont été réalisés sur la perception de l'environnement dans les recherches sur l'automatisation de véhicules et à l'inverse de la progression historique dans le domaine, nous nous sommes inspirés des méthodes appliquées sur les véhicules autonomes comme base de travail pour la réalisation d'une aide au pilotage non directive.

1.2.3 De la navigation autonome à l'aide au pilotage

Le DARPA Grand Challenge [Thrun et al., 2006] fut une source d'informations très importante. En effet, ce défi a mis en compétition un grand nombre de laboratoires afin de les faire participer à une épreuve de navigation autonome en milieu naturel, puis, plus récemment en environnement urbain. Le DARPA Grand challenge a été lancé en 2003 par le DARPA [DARPA, 2015], un grand laboratoire américain de recherche militaire, pour stimuler l'innovation dans le domaine de la navigation des véhicules autonomes. Le but de ce défi était de construire un véhicule robotisé autonome capable de traverser un terrain en milieu naturel non connu *a priori*.

À la suite de cette compétition, un grand nombre d'articles a été publié décrivant les différentes méthodes et techniques utilisées pour atteindre les objectifs demandés [Zhao et al., 2008; Lu et al., 2009; Caraffi et al., 2007; Kuhnert, 2008; Systems, 2009; Broggi et al., 2010; Hadsell et al., 2009; Milanes et al., 2011; Katramados et al., 2008].

D'autre part, ce défi a poussé des laboratoires à continuer dans des voies similaires. On peut citer comme exemple le VIAC [Luca et al., 2012; Bertozzi et al., 2011]. Ce projet, réalisé par le laboratoire VisLab du Pr. Broggi, est une expérience issue du travail réalisé sur le véhicule Terramax [Broggi et al., 2010] qui a entres autres participé au DARPA Grand Challenge 2005. Celui-ci est à l'origine d'un voyage de trois mois en véhicule autonome de 13000 km entre l'Italie et la Chine.

Comme énoncé plus tôt, malgré ces avancées considérables, les pouvoirs décisionnels militaires français ne sont pas encore prêts à donner le contrôle de leurs véhicules à des ordinateurs. C'est

pourquoi nous avons orienté notre recherche vers des apports pour l'opérateur non dirigistes. Ainsi, sur la base des méthodes étudiées pour le pilotage autonome, nous avons cherché à réaliser un système d'aide au pilotage pour les opérateurs de véhicules militaires. En effet, les méthodes réalisées pour la navigation autonome utilisent des capteurs et des traitements de données qui permettent aux véhicules de comprendre leur environnement, mais pourraient aussi aider le pilote à mieux le comprendre. De plus, ces méthodes qui partent du postulat que le destinataire des traitements n'a aucune connaissance, peuvent être modifiées et simplifiées dans leurs traitements afin de n'extraire que les informations essentielles pour un pilote humain. Par conséquent, au cours de cette thèse nous ne nous intéressons qu'aux aides au pilotage temps réel et non à la planification longue distance. Le postulat de départ est de rechercher des méthodes et capteurs permettant la création d'une aide au pilotage militaire.

Ainsi, nous détaillons notre réponse à la problématique de cette thèse dans la section suivante.

1.3 Contributions

Les précédentes sections ont permis de préciser le problème de *sécuriser le pilotage des véhicules militaires avec comme périmètre la définition d'un système de perception d'environnement*. Ainsi, nous allons, au travers des deux prochaines sections, expliquer la problématique qui en découle et la réponse que nous lui avons apportée.

1.3.1 Sécurisation du pilotage par perception d'environnement

La protection des opérateurs de véhicules militaires et l'intégrité de ces véhicules sont des besoins primordiaux pour l'armée de terre. Pour y répondre nous définissons un système technique composé de méthodes inhabituelles, d'algorithmes et de capteurs satisfaisant les contraintes industrielles.

Le but du projet d'aide au pilotage envisage d'offrir au pilote une indication sur la voie à suivre avec son véhicule. Cette aide au pilotage, bien que complexe dans son fonctionnement, se veut simple au niveau de l'affichage (Figure 1.4). Cet aide consiste à afficher au pilote quatre informations. Trois zones distinctes représentant la voie à suivre (sûre), la zone dangereuse, et la zone intermédiaire où une attention particulière est requise. La quatrième information est l'indication des obstacles détectés dans les zones dangereuses.

Pour répondre à cette problématique, nous avons procédé à l'étude au sens large de l'aide au pilotage dans le contexte militaire en environnement naturel et semi-structuré afin de mettre en exergue les moyens et les capteurs utilisables pour réaliser un système d'aide au pilotage. Ainsi, nous offrons une réponse technique pour réaliser un tel système au travers d'une étude des méthodes et algorithmes existants applicables à notre cas d'application, ainsi qu'une étude des capteurs utilisables avec de telles méthodes. De plus, le concept développé dans cette thèse a nécessité la création d'une solution algorithmique évaluée grâce à une plateforme d'expérimentation. Cette plateforme est équipée de l'ensemble de la solution définie et permet de mettre en exergue les limites de notre concept. Enfin, en fournissant une réflexion s'intéressant aux différents éléments séparant l'opérateur de l'environnement, une étude sur les moyens de restitution à l'opérateur complétera la solution présentée.

1.3.2 Contributions

Les contributions de cette thèse sont de différents ordres. Dans un premier temps, l'étude des systèmes d'aide au pilotage, des capteurs et des méthodes, nous a permis de réaliser un état de l'art de ces différents sujets. L'étude fournie permet d'appréhender le large spectre des capteurs et

FIGURE 1.4 – Rendu prévisionnel de la solution finale de notre ADAS. La zone verte est la zone sûre sur laquelle le pilote peut rouler. La zone rouge est une zone dangereuse, car des obstacles, en noir, ont été détectés. La zone jaune est une zone intermédiaire dans laquelle une attention particulière est nécessaire.

méthodes utilisés dans le cadre de l'aide au pilotage et plus particulièrement ceux viables pour une application dans le domaine militaire.

Dans un second temps, en s'appuyant sur les capteurs et méthodes issu de cet état de l'art, nous avons réalisé une plateforme robotique. Elle a été conçue dans une optique d'essai et de validation des algorithmes présentés dans ce manuscrit. Cette plateforme est équipée d'un couplage de capteurs LIDAR et de caméras stéréoscopiques. Le capteur LIDAR 2D a été motorisé afin d'acquérir des nuages de points 3D pour un prix faible.

La reconstruction de l'environnement frontal du véhicule est réalisée par la fusion des données LIDAR et images stéréoscopiques. Les données stéréoscopiques permettent de fournir la position de la plateforme tandis que les données LIDAR sont utilisées pour reconstruire l'environnement sous la forme d'un nuage de points 3D [RICAUD et al., 2015b]. Cette fusion donne des résultats cohérents sur le plan de la reconstruction, mais sa faible précision globale nous a amenés à travailler sur des améliorations.

La troisième contribution de cette thèse réside dans l'optimisation de la fusion des capteurs par l'ajout d'une odométrie monoculaire. Ainsi, nous fournissons une reconstruction en nuage de points plus précise et plus rapide en fusionnant des odométries monoculaire et stéréoscopique. Les résultats obtenus font apparaître un fort gain sur la cohérence globale. Malgré une précision locale insuffisante, les résultats laissent envisager des applications possibles sur les véhicules visés.

Nos travaux ont également porté sur l'étude de différents moyens de segmentation et d'analyse de données. D'une part, nous avons travaillé à l'amélioration d'une méthode de segmentation d'image [RICAUD et al., 2014] et d'autre part, nous avons implémenté une méthode de segmentation d'obstacles par accroissement de région. En effet, ce chapitre expose le gain d'information obtenue sur l'environnement par la reconstruction réalisée. Ainsi, nous exposons des techniques qui permettront

de transformer notre système de perception d'environnement en une aide au pilotage.

Enfin, nous avons mené une étude sur la restitution à l'opérateur. En effet, la restitution est un sujet essentiel dans l'efficacité du pilotage de véhicule sur écran. Cette étude apporte des résultats concrets permettant de prouver l'intérêt des casques de réalité virtuelle dans les applications de pilotage [RICAUD et al., 2015a].

Cette solution technique innovante par son contexte, sa problématique et la réponse proposée va être expliquée au travers des trois parties de ce manuscrit. La première partie expose l'état de l'art des méthodes applicables et la réponse matérielle qui en a été déduite. La seconde partie présente une solution innovante de cartographie d'environnement, exploitable selon des méthodes de segmentation visuelles et sur le nuage de points. Enfin, la troisième partie offre des éléments de réponse sur les moyens de restitution au pilote.

1.4 Organisation du manuscrit

Ce manuscrit est divisé en trois parties qui représentent les trois axes de travail développés dans cette thèse.

Premièrement, nous établissons un concept répondant à la problématique, définie précédemment, par l'étude de l'état de l'art des méthodes et des capteurs permettant d'établir la réponse matérielle au travers d'un concept novateur.

Deuxièmement, nous validons ce concept par la mise en place d'une plateforme d'expérimentations et des développements algorithmiques. En effet, suite à la définition de notre concept une plateforme d'expérimentations à été mise en place pour l'évaluer avec des données réelles. De même, des développements algorithmiques issus en partie de l'état de l'art ont été créés pour percevoir l'environnement avec nos capteurs par une reconstruction d'environnement. L'évaluation du concept s'est faite par la mise en place d'une vérité terrain permettant de mesurer les performances de notre concept de perception vis-à-vis de l'environnement réel. Suite aux résultats obtenus, une optimisation de la méthode originale à permis un fort gain de performance. Enfin, nous montrons que la mise en place de travaux d'analyse de la reconstruction créée est possible au travers de deux développements de segmentation visuelle et du nuage de points.

Dernièrement, nous exposons des travaux de recherche d'amélioration de notre concept. En effet, nous évaluons la possibilité de modifier le moyen de restitution de l'information à l'opérateur pour améliorer sa perception de cette information. Dans ce but, nous présentons l'intérêt du casque de réalité virtuelle comme moyen de restitution alternatif au travers d'une étude sur l'*Oculus Rift*.

1.4.1 Détail de l'organisation

La première partie aborde les recherches réalisées sur l'état de l'art et la réponse matérielle que nous pouvions apporter.

Partie I — État de l'art et réponse matérielle

La première partie de ce manuscrit concerne l'état de l'art réalisé au cours de cette thèse. Au travers de deux chapitres, nous présentons les méthodes et capteurs nous ayant permis de développer notre concept.

Chapitre 2 — État de l'art

Ce chapitre présente l'état de l'art vis-à-vis des aides au pilotage existantes dans le domaine civil et dans le domaine militaire. Nous présentons également les moyens et méthodes de perception et d'analyse du terrain en environnement naturel.

Chapitre 3 — Définition des capteurs de notre système

Ce chapitre aborde l'étude réalisée sur les capteurs optimaux répondant à nos besoins. Avec le chapitre précédent, la partie d'état de l'art nous a permis de définir notre concept de perception d'environnement.

Partie II — Système de perception d'un environnement naturel

La seconde partie de ce manuscrit a pour objet l'explication de la solution de reconstruction d'environnement développée au cours de cette thèse et des traitements qui en découlent.

Chapitre 4 — Une plateforme mobile d'expérimentations

Une des contributions de cette thèse est la réalisation d'une plateforme robotique d'expérimentations. Ce chapitre traite de la création de cette plateforme.

Chapitre 5 — Reconstruction par données LIDAR et odométrie visuelle

Dans ce chapitre la méthode de reconstruction d'environnement que nous avons développée est présentée ainsi que les résultats obtenus à partir de cette méthode. Dans un second temps, ce chapitre aborde l'optimisation apportée au système par l'ajout d'une méthode de vision monoculaire ainsi que les résultats obtenus par ce biais.

Chapitre 6 — Segmentation des données acquises

Ce chapitre traite quant à lui de l'analyse de la reconstruction créée grâce aux traitements développés lors du chapitre précédent. Dans un premier temps nous nous intéressons à une méthode de segmentation d'images. Dans un second temps, nous nous intéresserons à la segmentation du nuage de points acquis grâce lors de la reconstruction de l'environnement.

Partie III - Restitution à l'opérateur

Cette dernière partie développe l'étude réalisée sur la problématique de la restitution à l'opérateur.

Chapitre 7 — Pilotage à l'aide d'un casque de réalité virtuelle

Au travers de travaux mis en place avec l'Oculus Rift [VR, 2015] nous présentons l'intérêt de tels matériels comme moyen alternatif de restitution pour le pilotage de véhicules blindés en remplacement des écrans classiques.

Première partie

État de l'art et réponse matérielle

Introduction à la partie I Dans le chapitre 1 nous avons présenté le contexte et les contraintes de cette thèse. Le contexte militaire est à l'origine de plusieurs contraintes particulières qui ont guidé notre réponse à la problématique *sécuriser le pilotage des véhicules militaires avec un système de perception d'environnement*. L'introduction a ainsi précisé cette problématique et exposé les contributions réalisées dans cette thèse ainsi que le détail du manuscrit.

Cette première partie présente tout d'abord dans le chapitre 2 - *État de l'art* - les aides au pilotage existantes de manière générale qu'elles soient civiles ou militaires. Aucune de ces aides ne répondant complètement à notre application, nous nous intéressons ensuite à la création de notre propre système de perception d'environnement par l'acquisition et la reconstruction de cet environnement. Nous exposons l'état de l'art des méthodes existantes de perception de l'environnement. Au travers de la présentation des méthodes de cartographie mobile, de cartographie par SLAM [2], de positionnement relatif et absolu, nous expliquons notre choix de réaliser une cartographie mobile par positionnement relatif. Enfin, nous définissons les contraintes de franchissement d'obstacles relatives à notre application et expliquons le choix d'un capteur actif d'acquisition des distances pour la détection des obstacles dits négatifs.

Le chapitre 3 - *Définition des capteurs de notre système* - expose nos choix d'un point de vue matériel. En effet, compte tenu de la décision de réaliser de la cartographie mobile, nous utilisons un LIDAR pour l'acquisition de l'environnement et une caméra stéréoscopique pour réaliser une odométrie visuelle. Ce chapitre détaille le fonctionnement de ces capteurs ainsi que celui des autres capteurs de l'état de l'art et détaille cette décision.

2. SLAM : De l'anglais Simultaneous Localization And Mapping, est un principe où l'on construit une carte et on s'y localise en même temps. Ce principe est très connu en robotique mobile.

Chapitre 2

État de l'art

Sommaire

2.1 L'aide au pilotage **18**
2.1.1 Les systèmes civils 18
2.1.2 Les systèmes militaires 21
2.2 Cartographie mobile **23**
2.2.1 La cartographie 24
2.2.2 Positionnement 24
2.3 Cartographie par méthode SLAM **31**
2.3.1 Principe 31
2.3.2 Algorithmes de filtrage associés au SLAM 34
2.3.3 SLAM : correction d'erreurs et simplification 39
2.3.4 Conclusion sur la cartographie 41
2.4 Analyse du terrain **41**
2.4.1 Franchissement : une définition approfondie des obstacles 42
2.4.2 Détection d'obstacles 47
2.5 Conclusion sur la définition de notre système **52**

La première partie de ce manuscrit va présenter dans un premier temps l'état de l'art ayant permis de définir quelles méthodes, quels algorithmes et quels capteurs permettent de répondre à notre problématique évoquée dans le chapitre précédent.

L'analyse de l'état de l'art menée lors de cette étude a pour objectif de mettre en avant les aides apportées actuellement aux pilotes de véhicules blindés et les conducteurs de véhicules civils afin de visualiser les possibilités offertes pour la création de notre système. En outre, nous étudierons également les méthodes de cartographie d'environnement existantes ainsi que les moyens d'extraction d'obstacles à partir des données acquises. Enfin, la mise en perspective de nos besoins industriels et des capacités des capteurs du marché nous permettra de conclure sur les capteurs et les méthodes vers lesquels notre système s'orientera.

Ce chapitre débute par la présentation des différentes aides au pilotage civiles et militaires existantes sur le marché. Ensuite, la nécessité d'avoir une représentation locale de notre environnement nous amène à nous intéresser à la cartographie mobile qui, comme nous l'exposons, répond à notre problématique. Les principales techniques de cartographie et de positionnement seront présentées. La troisième partie traitera de la localisation et de la cartographie simultanée (SLAM) en expliquant les principes généraux et les méthodes utilisées couramment. Enfin, nous aborderons le sujet de l'analyse du terrain par la définition détaillée des obstacles, l'étude des possibilités de détection, des méthodes d'extraction de ces obstacles et des méthodes de simplification de données. La dernière partie conclura ce chapitre en décrivant la méthode choisie pour l'élaboration de notre système et l'explication de ce choix.

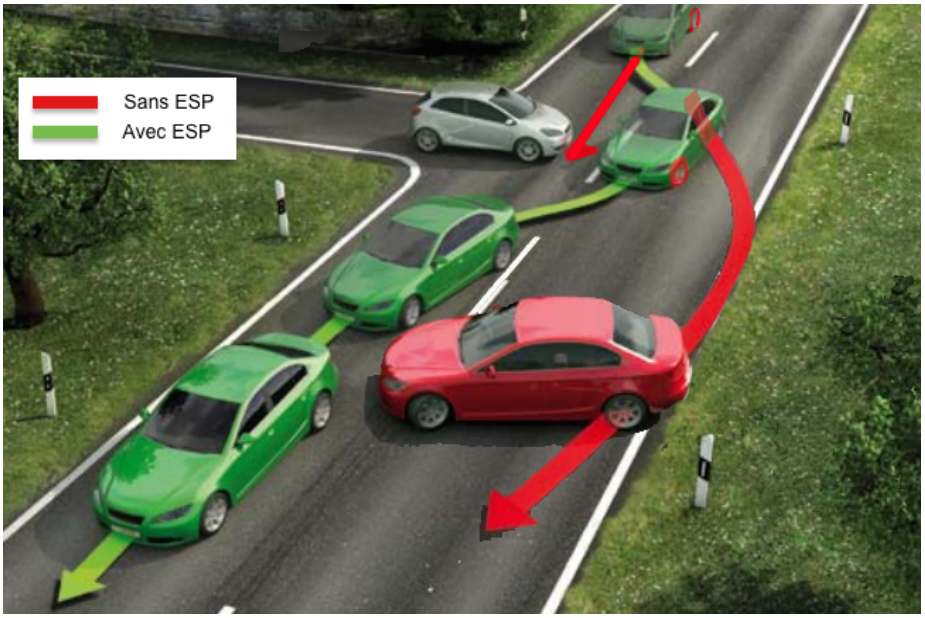


FIGURE 2.1 – Illustration du système ESP. Ce système comme l'ABS fait partie des plus anciennes aides au pilotage. Freinage arrière pour éviter le sous-virage et freinage avant pour éviter le survirage (en vert). En rouge la situation sans l'aide de l'ESP. Image créée à partir de visuels modifiés réalisés par BOSCH [Bosch, 2015b].

2.1 L'aide au pilotage

Les aides au pilotage (ADAS « Advanced Driver Assistance System ») regroupent tous les types d'aides au pilotage à la fois civiles ou militaires. Bien que nous nous intéressions particulièrement à la perception de l'environnement, il est important de voir que l'éventail d'aides est très large. Dans un premier temps, nous allons nous intéresser aux aides civiles puis dans un second temps, nous étudierons les aides proposées dans le domaine militaire.

2.1.1 Les systèmes civils

À titre d'exemple, le constructeur automobile Renault définit les ADAS comme l'ensemble « des technologies au service de la facilité de conduite et de la sécurité » [Renault, 2015]. Ces aides interviennent en donnant des indications au conducteur ou en intervenant sur la mobilité du véhicule (Figure 2.2 page 20).

Les constructeurs automobiles souhaitant rendre leurs véhicules de plus en plus « intelligents » pour des raisons de sécurité [Parent, 2004], de consommation [Fritz et al., 2004] ou de congestion du trafic [Grégoire, 2014], un grand nombre d'aides au pilotage sont disponibles sur le marché civil.

Commençons par les aides permettant à un conducteur de garder le contrôle dans des situations complexes en donnant au véhicule la possibilité de « piloter » certaines fonctions :

- L'ABS (*Antiblockiersystem* [Bosch, 2015a]) ou Anti-Blocage de Sécurité des roues empêche le blocage des roues lors d'un freinage brusque. Effectivement, le blocage des roues peut entraîner un dérapage conduisant à la perte de contrôle du véhicule. L'ABS va, grâce à des capteurs d'adhérence, fournir une puissance de freinage relative aux conditions d'adhérence.
- L'ESP (*Electronic Stability Program* [Bosch, 2015b]) ou Stabilisateur Électronique Programmable, contrôle la trajectoire du véhicule et tente de la corriger. Ce système est composé de

trois capteurs : un capteur mesurant l'angle du volant, un capteur de vitesse au niveau de chaque roue et un capteur de lacet mesurant la vitesse de rotation autour de l'axe de lacet du véhicule. Le calculateur a pour rôle de déterminer s'il doit intervenir, à quel l'instant et le mode d'intervention qu'il doit appliquer sur les freins du véhicule (Figure 2.1).

– Le DST (*Dynamic Steering Torque* [BMW, 2015]), ou détecteur des conditions d'adhérence gère dynamiquement la direction assistée. Dans les situations critiques il peut diminuer la direction assistée afin donner au conducteur de meilleures sensations et un meilleur contrôle de son véhicule.

De surcroît, certaines voitures, comme la Peugeot 5008, sont aussi équipées de systèmes anticollision (ARTIV : Aide au Respect des Temps Inter Véhicules ou *Distance Alert* de Peugeot [Peugeot, 2015]), permettant d'enclencher des avertisseurs puis un freinage d'urgence du véhicule quand un objet immobile ou avançant à une vitesse plus faible est détecté devant le véhicule. Cette détection est réalisée à l'aide de lidar infrarouge, de radars ou de sonars scannant devant le véhicule sur un plan horizontal à hauteur du parechoc avant. Au demeurant, la détection de piétons est de plus en plus fréquente en plus de la seule détection de véhicule.

D'autres systèmes prennent le contrôle lors de situations moins dangereuses, mais tout aussi complexes pour certains utilisateurs comme, les démarrages en côte (*Hill assist*) ou même le stationnement [AutonomousParking, 2014]. Néanmoins, certaines se contentent d'avertir le conducteur par des avertisseurs sonores ou visuels :

– Le radar de recul [Jung et al., 2008] (détection de distance arrière), qui est un système similaire au système anticollision, mais qui s'applique au stationnement.

– Le détecteur de fatigue et d'inattention [Houguet, 2010] analyse le comportement du conducteur pour détecter des signes de fatigue. Ces signes peuvent être : les yeux qui se ferment, la tête qui penche en avant, un changement dans les angles de direction ou même des pressions aux pédales moins importantes qu'au départ du trajet. Il en résulte un signal sonore émis par le véhicule afin d'indiquer au conducteur de faire une pause.

– L'avertisseur de déviation de trajectoire ou encore l'alerte de franchissement involontaire de ligne (AFIL [Risack et al., 2000]) sont des systèmes fonctionnant soit par l'utilisation d'analyse d'images grâce à des caméras embarquées soit par un système de diodes mesurant la réflectance. Ce système orienté vers la route et situé sur le bouclier avant du véhicule détecte le passage d'une zone peu réfléchissante (bitume) à une zone réfléchissante (bande blanche) ; si le clignotant n'est pas activé, un avertissement est envoyé au conducteur.

– On peut bien sûr ajouter toutes les indications sur l'état du véhicule, les fameux « voyants » qui indiquent par exemple au conducteur si son frein à main est enclenché, si les ceintures de sécurité ne sont pas attachées ou si le niveau d'huile est trop faible.

À cela s'ajoute des aides ayant pour but d'augmenter le confort de conduite des utilisateurs, à l'image du limiteur de vitesse permettant de respecter les limitations sans regarder le compteur ou les systèmes de navigation (GPS [Masumoto, 1993]) donnant sa position au conducteur. Certains systèmes de navigation permettent même d'optimiser le parcours en fonction du trafic, des situations de circulation ou des préférences de l'utilisateur.

En dernier lieu, il existe des aides qui consistent simplement à assister de manière physique le conducteur afin de lui faciliter la conduite, on pense tout d'abord à la direction assistée, qui réduit la force nécessaire pour tourner le volant. On peut aussi trouver les essuie-glaces automatiques avec détecteur de pluie, évitant de lâcher son volant ; le frein de stationnement électrique ou encore l'affichage tête haute qui permet de garder les yeux sur la route.

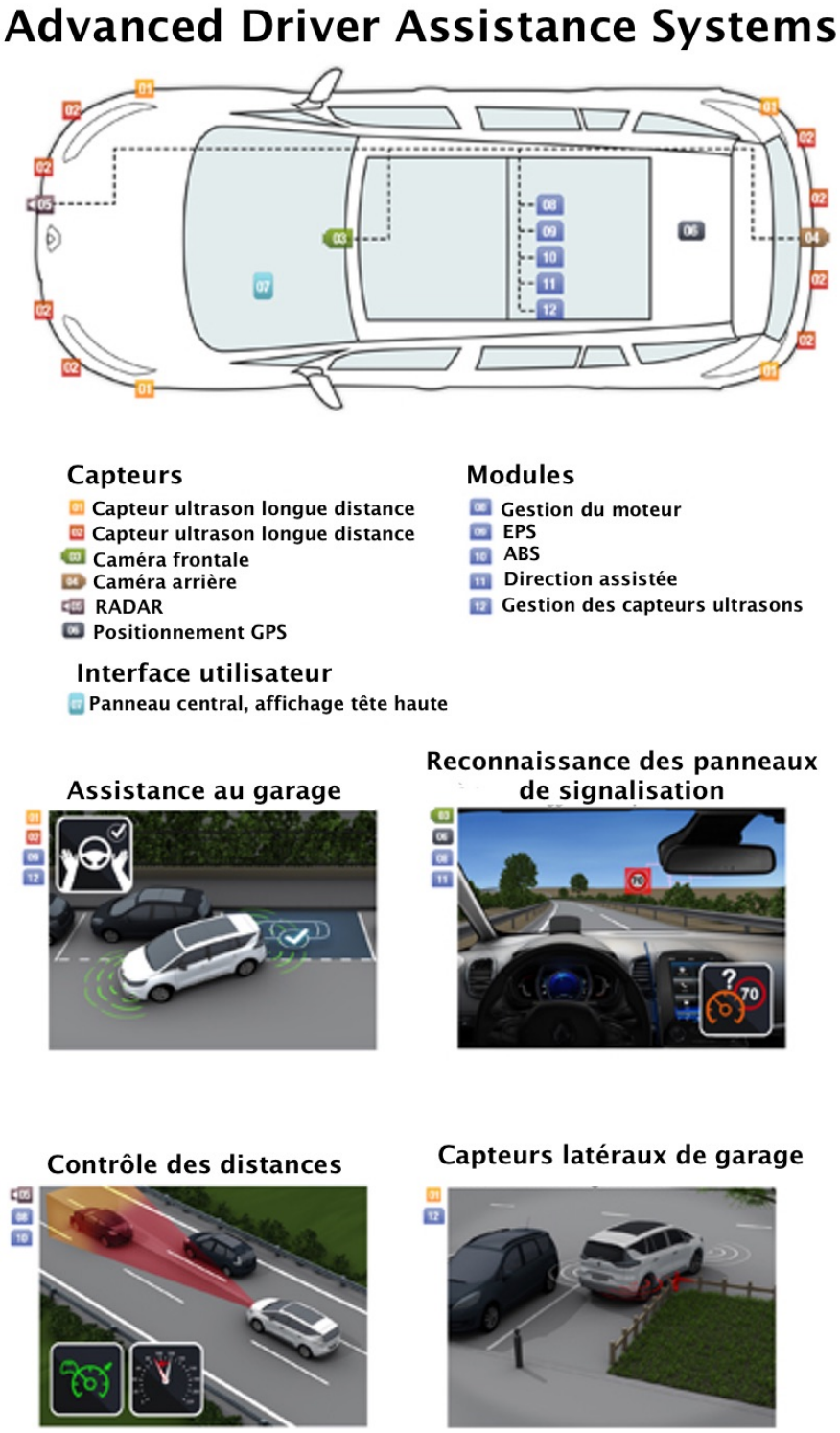


FIGURE 2.2 – Les ADAS définis par Renault. Source : Renault

Cette sous-section illustre l'orientation routière des aides au pilotage civiles. Évidemment, ce cas d'utilisation est le principal, voire le seul, pour lequel les véhicules sont conçus. Cependant, il apparaît qu'aucune de ces aides ne permet de sécuriser le pilotage de véhicule militaire par l'amélioration de la perception de tous types d'environnements.

Bien que beaucoup d'aides civiles se retrouvent sur les véhicules militaires, les *ADAS* militaires sont plus orientés vers la conduite tout-terrain et la prochaine sous-section va en faire un bilan.

2.1.2 Les systèmes militaires

La sécurité des soldats étant un sujet capital depuis plusieurs années, les véhicules militaires sont aussi équipés d'aides au pilotage. Elles sont moins nombreuses et plus spécialisées que celles disponibles pour les véhicules civils et elles se divisent en deux catégories : les aides au pilotage vétroniques [1] et les aides au pilotage orientées mobilité.

Les aides au pilotage vétroniques utilisent des capteurs extéroceptifs [2] comme les caméras ou le GPS afin d'informer le pilote sur son environnement immédiat. Les aides au pilotage orientées mobilités sont celles utilisant des capteurs proprioceptifs servant à agir sur la conduite.

Concernant la vétronique, un grand nombre de véhicules blindés sont équipés de caméras positionnées tout autour du véhicule permettant à la fois le pilotage, les manœuvres et la surveillance à 360° autour du véhicule. Ces caméras sont souvent couplées avec des systèmes de vision de nuit, qui peuvent être de plusieurs sortes :

- Des intensificateurs de lumière (figure 2.3a) augmentant la lumière réfléchie par les objets,
- Des caméras proches de l'infrarouge (NIR, 0,9-2 μm, figure 2.3b) qui filment la projection d'une lumière infrarouge sur une scène produite par la lune ou par un autre capteur tel un émetteur infrarouge.
- Des caméras thermiques permettant de détecter la chaleur des objets, qui est émise dans la longueur d'onde infrarouge (2-5 μm et 8-13 μm, figure 2.3c).

En outre, les caméras arrière servent à visualiser les débarquements de soldats et comme caméras de recul avec indication du gabarit du véhicule. Afin de visualiser les images filmées, l'affichage tête haute fait son apparition, mais les épiscopes sont encore très répandus. De surcroît, les caméras tireurs, situées sur le toit du véhicule, peuvent être affichées sur les postes de pilotage afin d'aider le pilote à se positionner au « raz de la crête », c'est à dire, à donner suffisamment de visibilité au tireur en s'exposant a minima.

Les systèmes de navigation sont aussi plus sophistiqués que les systèmes civils, ils informent le pilote, non seulement de sa position, mais aussi de la position de ses alliés, des cibles potentielles, des objectifs ou encore de la topographie du terrain. De plus, un système récent permet même, grâce à des vues aériennes, de donner des informations de visibilité aux véhicules. Ce système, appelé Carmenta, calcule toutes les lignes de vues en fonction de sa position (Figure 2.4 page 23). En effet, ce calcul offre pour une position donnée, l'ensemble des positions visibles, en fonction de la topographie du terrain et des occlusions.

À ces systèmes s'ajoute un ensemble de modules permettant de gérer l'état du véhicule, tels que les défauts moteur, d'éventuels problèmes sur les équipements internes ou les problèmes de motricité.

Concernant la mobilité, on retrouve des systèmes civils courants tels que l'ABS ou l'ESP expliqués précédemment, mais aussi d'autres, plus spécialisés :

1. Pour rappel : La vétronique (vetronics en anglais) correspond à un terme générique englobant tous les systèmes électroniques d'un véhicule [Abdulmasih and Oikonomidis, 2011].
2. Un capteur extéroceptif est un capteur qui mesure des données extérieures au véhicule.

(a) Image d'exemple transmise par un intensificateur de lumière [Intensificateur, 2015]

(b) Image provenant d'un véhicule Nexter transmise par une caméra proche infrarouge. Source : Nexter

(c) Image provenant d'un véhicule Nexter transmise par une caméra thermique. Source : Nexter

FIGURE 2.3 – Les systèmes de visions couramment utilisés à bord des systèmes militaires.

- Sur les véhicules à roues (Aravis [Defense.gouv.fr, 2015a], VBCI [Defense.gouv.fr, 2015c] ou Titus [Nexter, 2015b]), le variateur de pression de gonflage (VPG) [Teleflow, 2015] permet la gestion de la pression des pneus. Ce système (qu'on retrouve aussi sur des engins civils agricoles) est étudié pour adapter la pression en fonction de l'environnement traversé et de la nécessité d'adhérence. Sur une piste ensablée, la pression va être grandement diminuée afin de limiter l'enlisement et pour faciliter le franchissement. Au contraire sur une route en bitume, la pression va être augmentée pour limiter l'usure des pneus. La pression peut être gérée par le pilote en temps réel ou gérée automatiquement par le logiciel embarqué.
- Les véhicules à plus de 4 roues (VBCI et Titus) possèdent un système de braquage additionnel par freinage (SBAF). Ce système permet un braquage additionnel afin de tourner plus efficacement. Il se rapproche du fonctionnement d'un ESP, mais permet dans ce cas de déclencher un dérapage en forçant le survirage au contraire d'un ESP. En effet, le blocage unilatéral des roues provoque le « ripage » du véhicule et lui offre des possibilités de manœuvres plus proches des véhicules chenillés. Dans le même objectif, la motricité peut être changée en roulant afin de rendre le train arrière directeur. En effet, avec les trains avant et arrière directeurs, il est possible de réduire le rayon de braquage.
- Le dernier né des systèmes d'aide de mobilité est un variateur de garde au sol, installé sur le Titus et basé sur des suspensions à coussins pneumatiques à pression variable, il permet le franchissement d'obstacles hauts.

Un grand nombre d'aides au pilotage pour véhicules blindés existe, mais aucune ne répond à notre besoin d'anticiper les dangers encourus par le véhicule et ses occupants dans le futur.

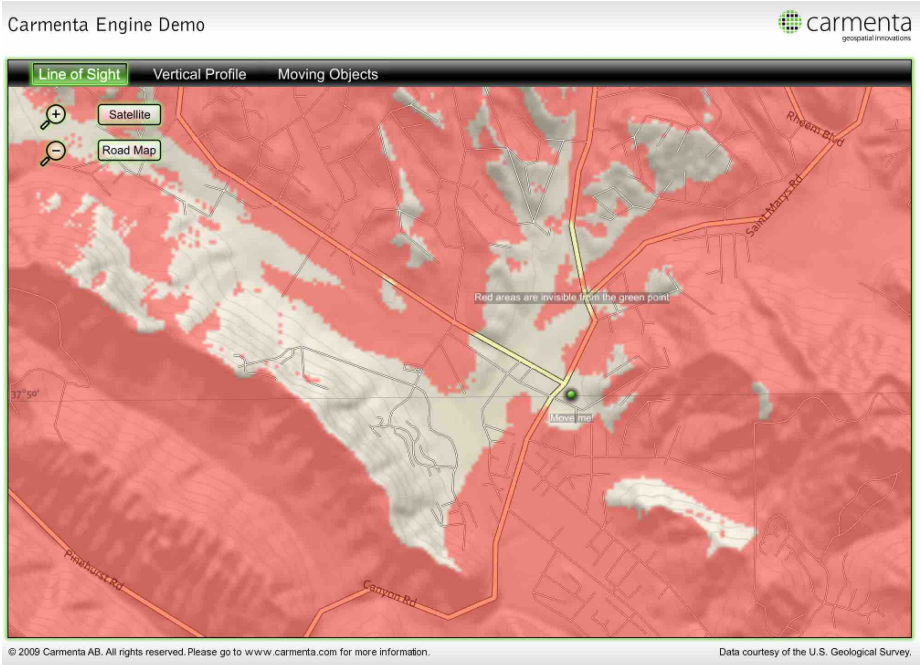


FIGURE 2.4 – Logiciel permettant le calcul des lignes de vues grâce à des images aériennes. Les lignes de vues sont les zones visibles d'un point donné en fonction de la topologie du terrain. Sur cette image, du point vert, tout ce qui n'est pas rouge est visible. Source : Carmenta

Les aides au pilotage civiles actuelles n'offrent pas non plus de tels retours, même si des radars anticollision se rapprochent de notre application, ils sont limités aux dangers se trouvant au-dessus du sol et ne prennent pas en compte les trous et les défauts de franchissement. Cependant, d'autres traitements qui ne sont pas encore utilisés au titre « d'aide au pilotage » pourraient offrir une réponse satisfaisante à notre problématique. En effet, il est nécessaire d'avoir une connaissance de l'environnement local devant le véhicule et la cartographie du terrain permettrait de percevoir les particularités et les dangers dans l'arc avant du véhicule comme nous souhaitons le faire.

Ainsi, nous nous intéressons maintenant à la cartographie mobile dont le but est de reconstruire l'environnement autour du véhicule dans un but d'analyse. De plus, la cartographie mobile possède la particularité de réaliser cette acquisition de l'environnement en mouvement comme notre application le nécessite.

2.2 Cartographie mobile

Afin de prédire les dangers encourus par notre véhicule et son équipage nous souhaitons réaliser une cartographie détaillée de l'environnement que le véhicule va traverser. La cartographie mobile regroupe les études menées dans le but de réaliser une cartographie d'un environnement avec un système en mouvement. Pour pouvoir obtenir un résultat, les méthodes font appel à deux fonctions : d'une part le système de cartographie à proprement parlé qui permet de capturer l'environnement et d'autre part le système de positionnement qui permet de replacer les données acquises dans l'espace. Cette section va traiter de ces deux sujets afin d'étudier les possibilités qu'ils offrent à la résolution de notre objectif de définition d'une aide au pilotage efficace en environnement naturel.

2.2.1 La cartographie

La cartographie mobile intéresse le domaine scientifique depuis au moins les années 80 [Vegt et al., 1987]. Ces premières expériences furent réalisées grâce à l'émergence des GPS et leur but était, et est toujours [Poreba and Goulette, 2015], d'augmenter le niveau de fiabilité de la géométrie des cartes créées de manière automatique en limitant les contrôles à réaliser au sol par des géomètres.

Afin de réaliser une cartographie mobile, deux visions sont actuellement utilisées : la cartographie temps réel et la cartographie post-traitée. Dans ce cadre, beaucoup de méthodes [Poreba, 2014; Abuhadrous et al., 2004; Hervier, 2012; Martínez et al., 2015; Marcotegui and Deschaud, 2012; Li, 1997; Diesel, 1987] utilisent le post-traitement pour obtenir des cartes très proches de la réalité. Dans notre application, la cartographie par post-traitement n'a pas d'intérêt, car les données ne seront pas gardées après qu'elles aient été dépassées par le véhicule.

Bien que nous ne souhaitions pas stocker de grande carte, le système stockera une carte dite « courante ». Cette carte sera incrémentée au fur et à mesure de l'acquisition et oubliée au fur et à mesure du passage. Ceci élimine notre problématique de stockage. En effet, le but est d'avoir une carte assez précise pour offrir des informations au pilote, mais pas trop lourde afin de respecter notre problématique d'espace de stockage et pour avoir des traitements effectués en temps réel.

Beaucoup d'applications utilisent la cartographie mobile, *Street View* en est la plus connue [Anguelov et al., 2010]. Pour réaliser *Street View*, Google a utilisé une acquisition vidéo panoramique qu'ils ont géolocalisée par GPS. De surcroît, Google travaille désormais sur des cartes plus détaillées avec des systèmes d'acquisition 3D par laser dans le but de connaître le tissu urbain de manière toujours plus fine [Anguelov et al., 2010].

Afin de réaliser une cartographie mobile, plusieurs éléments sont nécessaires. Il faut un ou plusieurs moyens d'acquisition de l'environnement et un ou plusieurs moyens de positionnement. Le positionnement permet de créer une carte au cours du temps ; sans cela, seule une carte instantanée centrée sur le véhicule est possible. Le positionnement peut être récupéré au moyen de systèmes de positionnement par balises, donnant une position absolue, mais il peut aussi être réalisé en calculant le mouvement effectué par le véhicule, donnant un positionnement relatif par rapport au point de départ ou une position connue.

2.2.2 Positionnement

Pour pouvoir réaliser des cartographies mobiles, il est nécessaire de savoir où les données sont acquises. Pour cela, deux approches principales sont utilisées [Rone and Ben-Tzvi, 2013] :

- Soit on positionne les données de manière instantanée et absolue grâce à des balises,
- Soit on retrace le déplacement réalisé par le véhicule afin de replacer les données de manière relative en fonction de son mouvement en réalisant de la fusion de données.

2.2.2.1 Positionnement absolu par balises

Dans le positionnement par balises il existe deux systèmes principaux fonctionnant sur le même schéma, le géoréférencement par satellite et le « motion capture ». Ces deux systèmes ont comme base la triangulation (ou la trilatération) d'une balise de position inconnue entre des références de positions connues.

Connaissant la distance séparant la balise de chaque référence on peut délimiter une zone dans laquelle se trouve la balise en réalisant la trilatération (ou la triangulation) de celle-ci (Figure 2.5 page suivante). Pour réaliser une trilatération, il faut au moins deux références entourant la balise. La précision de positionnement est liée à la zone de trilatération, plus il y a de références disponibles

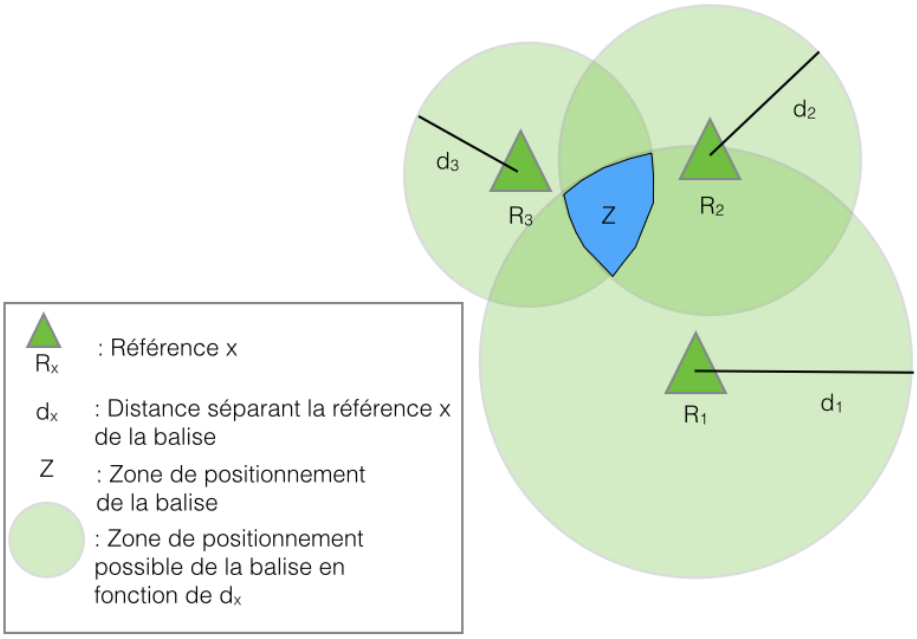


FIGURE 2.5 – Schéma du positionnement de la balise par trilatération.

plus la zone va être restreinte. De plus, la position des références doit être mesurée de manière précise pour que la zone soit précise elle aussi.

Dans le cas du GPS (*Global Positionning System* [Masumoto, 1993]), les références sont des satellites orbitant autour de la terre, leurs orbites sont connues ce qui permet de géoréférencer les balises sur terre. Le GPS est le système de positionnement américain par satellites, il possède deux homologues [Hofmann-Wellenhof, 2008], Glonass le système russe [Glonass, 2015], Galileo le système européen en cours de mise service [ASE-GNSS, 2008] et le système chinois Beidu limité à l'Asie [gov.cn, 2003].

Les systèmes GPS utilisent la trilatération (Figure 2.5), similaire à la triangulation, mais où seules les distances entre les points sont utilisées sans prendre en compte les angles. La précision standard de positionnement de ce système est de 10 mètres. Pour augmenter la précision, il est possible d'utiliser le GPS différentiel [CETMF, 2008], qui permet de réduire l'erreur jusqu'au mètre. Le principe de cette méthode est d'utiliser une station de référence terrestre fixe qui compare sa position réelle et la position indiquée par les satellites, ensuite elle va servir à calculer l'erreur de positionnement et à propager la correction aux balises. Ce système performant réduit l'erreur à un mètre, mais il nécessite une installation préalable, car les stations n'émettent qu'à une centaine de kilomètres.

Le plus précis de ces systèmes est le RTK (*cinématique temps réel* [Langley, 1998]), qui permet d'améliorer le système de poistionnement par satellites pour obtenir une précision de l'ordre du centimètre. Pour ce faire, la station de référence compare une copie interne du signal avec le signal récupéré du satellite. La station et le satellite possédent des horloges synchronisées ce qui permet de calculer une distance à partir du décalage des signaux. Ainsi, le RTK peut corriger le décalage sur les balises aux sol. Ce système nécessite des bases fixes de références, ici tous les 10 kilomètres, il se base donc sur un réseau dense d'antennes fixes (à titre d'exemple le réseau Teria compte 200 antennes en France [Teria, 2015]).

Une autre solution pour augmenter la précision a été de lancer des satellites en orbite géostationnaire (*Satellite Based Augmentation System* « SBAS » [EGNOS, 2015]). Ces satellites réduisent

l'erreur à 3 mètres et nécessitent d'être beaucoup moins nombreux (une quarantaine par continent).

Le GPS est une technologie américaine, elle est soumise à des réglementations très strictes sur le plan militaire. Certes, un accord multilatéral est signé avec ses alliés de l'OTAN [Defense.gouv.fr, 2012], mais réaliser un système en étant indépendant est une perspective intéressante.

Le *motion capture* ou « mocap » [Moeslund and Granum, 2001] est le second système de positionnement par balises. C'est un procédé permettant de récupérer le mouvement d'un objet dans l'espace à l'aide de balises ou de marqueurs. Couramment le *mocap* est réalisé en filmant avec une caméra infrarouge un objet sur lequel est installé un ensemble de marqueurs réfléchissants. Dans la scène seront aussi visibles des marqueurs de référence. Tous ces marqueurs permettront de retranscrire le mouvement de l'objet sous la forme d'un mouvement de points, beaucoup plus simple à analyser [Moeslund and Granum, 2001]. Cette technique est très utilisée par l'industrie cinématographique et vidéoludique afin de capturer des mouvements humains pour les transposer dans leurs environnements virtuels. Cette technique peut aussi être utilisée pour récupérer le mouvement d'un véhicule [Thorel, 2014], mais elle a une forte contrainte spatiale. En effet, pour que le *mocap* fonctionne l'objet en mouvement doit être équipé de marqueurs et rester continuellement entre les balises de référence, le tout visible par une ou plusieurs caméras filmant la scène en intégralité. Cette approche, envisageable pour réaliser des acquisitions sur de faibles surfaces n'est pas exploitable dans notre application.

Pour conclure, le défaut du positionnement absolu est la contrainte engendrée par la nécessité de références externes aux véhicules. Cette contrainte n'apparaît pas dans le positionnement relatif par calcul de déplacement présenté par la suite.

2.2.2.2 Positionnement relatif par calcul de déplacement incrémental

Le positionnement par calcul de déplacement appelé odométrie est une méthode très couramment utilisée [Swank, 2012; Howard, 2008; Zhang and Singh, 2014a,a; Kazik et al., 2012; Wirth et al., 2013]. Elle sert à calculer la position d'un objet en mouvement grâce à l'étude de son déplacement entre deux instants proches. L'odométrie fournit un déplacement incrémental ; la trajectoire peut être obtenue en sommant l'ensemble de ces incréments. Contrairement au positionnement par balises, cette méthode n'utilise pas de référence spatiale, elle a, par contre, besoin de connaître sa situation initiale (sa vitesse, sa position et son orientation) pour pouvoir fournir une estimation précise de sa position. Elle a l'avantage de ne pas nécessiter d'installation particulière sur la trajectoire de déplacement. Ce calcul peut être réalisé en fonction :

- des roues du véhicule
- de ce que « voit » le véhicule
- du mouvement inertiel du véhicule

Odométrie par mouvement des roues Sur des véhicules à roues, l'odométrie par calcul différentiel de la vitesse des roues permet de déduire le mouvement qu'effectue l'objet.

Ce type d'odométrie est très utilisé en robotique mobile sur de petites plateformes, ou pour des déplacements sur de très faibles distances à faible vitesse [Steux and ElHamzaoui, 2010; Jung et al., 2008; Terrien et al., 2000]. En effet, à faible vitesse les risques de glissement sont amoindris.

Cette odométrie simple et rapide à calculer n'est utilisable que dans des conditions idéales où aucun glissement n'a lieu. Dans notre cas, le sol n'étant pas plan et notre véhicule effectuant des glissements, cette odométrie n'est pas utilisable.

La navigation inertielle La navigation inertielle se base sur l'utilisation d'une centrale inertielle [Wendel et al., 2012] qui est un ensemble de capteurs permettant de retranscrire un mouvement.

Elle s'intéresse à l'acquisition sensorielle du mouvement réalisé par le robot (comme l'oreille interne [Aldebaran, 2014]). Les capteurs qui la composent sont :

- Trois accéléromètres, qui capturent les accélérations linéaires sur chaque axe
- Trois gyroscopes qui capturent les accélérations angulaires autour de chaque axe
- Un magnétomètre qui permet de calculer l'orientation, comme une boussole.

Sans entrer dans les détails, pour retrouver la position du capteur inertiel dans l'espace il faut intégrer deux fois les accélérations acquises par les accéléromètres et une fois celles acquises par les gyroscopes [Shen et al., 2011]. Le problème de ces calculs est la dérive qu'ils engendrent. En effet, en intégrant deux fois l'accélération on intègre aussi deux fois le bruit qui n'est donc plus négligeable. Le bruit peut par exemple représenter les vibrations subies par l'appareil ou le bruit électromagnétique. Cette dérive est inhérente à tout système inertiel, mais des centrales inertielles très performantes (et aussi très chères) peuvent, en limitant le bruit, améliorer grandement les résultats. Malgré tout, ces systèmes sont très sensibles aux vibrations et aux petits chocs [Braman and Grossman, 2006] très présents dans notre application, et qui expliquent pourquoi nous ne les avons pas utilisés comme capteur principal de positionnement.

En revanche, la fusion avec d'autres méthodes de positionnement [Diesel, 1987] peut permettre de réinitialiser le bruit en continu afin d'obtenir des résultats performants. Il est courant de voir des fusions de centrale inertielle avec des GPS par exemple [Jakubowski, 2008; Geiger et al., 2012].

Les deux méthodes précédentes faisaient appel à des capteurs proprioceptifs [3] qui ne semblent pas être idéaux dans notre application, alors que l'analyse de notre mouvement par la modification de l'environnement nous semble plus adaptée. Au travers des deux prochaines méthodes présentées, nous allons expliquer la raison de ce choix.

L'Egomotion « L'Egomotion » [Gluckman and Nayar, 1998] est un procédé d'odométrie visuelle [4], qui peut fonctionner sur des caméras monoculaires ou stéréoscopiques et peut faire appel à différentes méthodes. En effet, pour calculer une transformation liant deux images on peut faire appel à la mise en correspondance :

- d'imagettes,
- de points caractéristiques,
- de contours,
- de pixels,
- ou encore de longueurs d'onde.

Comme l'explique la figure 2.6 page suivante, l'objectif de cette méthode et de retrouver la transformation T_R qui lie le repère de la caméra à l'instant puis à l'instant $t + 1$ en observant le déplacement T_d des objets dans l'image.

Pour réaliser l'*egomotion*, la méthode la plus courante est l'utilisation du « flux optique » [Tian et al., 1996; Kitt et al., 2010; Alcantarilla et al., 2012; Lu et al., 2009, 2010; Geiger et al., 2011b; Kazik et al., 2012; Gluckman and Nayar, 1998; Zhang and Singh, 2014b]. Elle permet de différencier différentes vitesses d'objets dans une scène [Horn and Schunck, 1981]. Comme expliqué dans [Beauchemin and Barron, 1995], quand une caméra bouge dans un environnement rigide, l'image change au cours du temps de manière similaire sur tous les objets fixes de la scène. L'étude du mouvement des objets de la scène permet alors d'estimer le mouvement réalisé par l'observateur (Figure 2.6).

3. Un capteur proprioceptif est un capteur qui mesure des données internes au véhicule, comme la vitesse angulaire des roues.

4. L'*egomotion* permet donc de trouver le mouvement d'un capteur ayant acquis des images d'une scène.

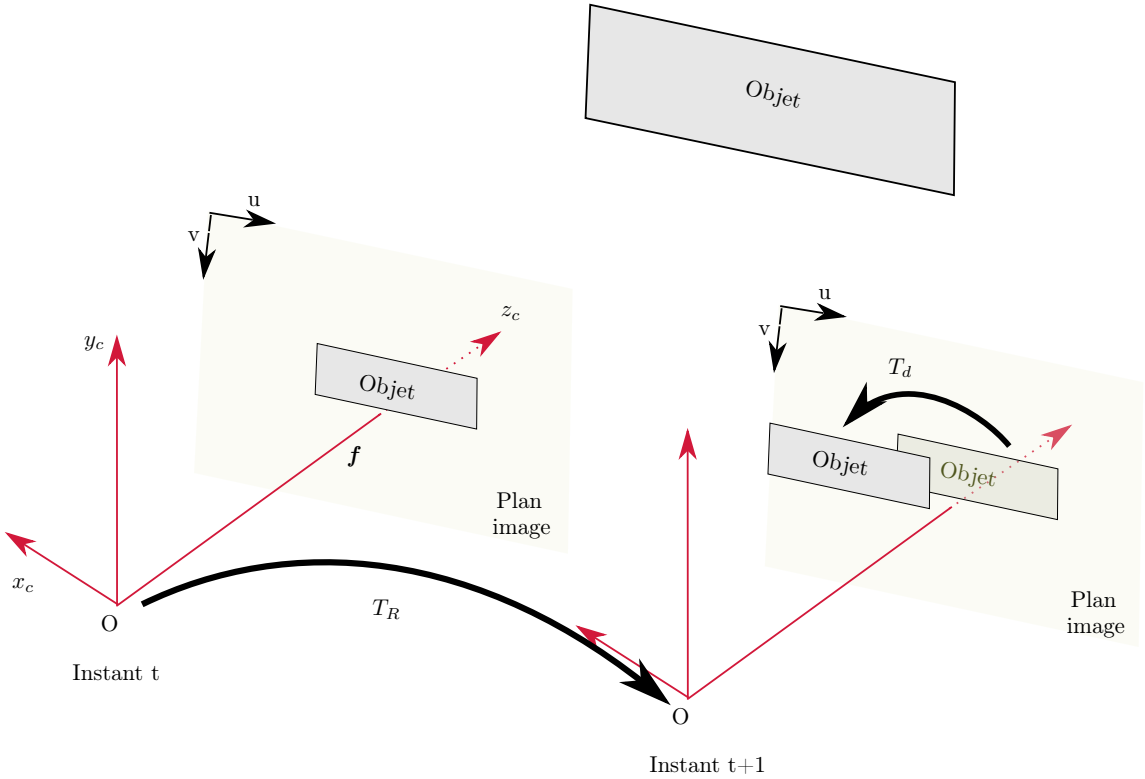


FIGURE 2.6 – Le calcul d'odométrie par l'estimation du déplacement des objets dans une image.

Cette approximation peut être réalisée par l'analyse de « l'intensité » $I(x,t)$ de la région d'une image x au temps t.

Cette méthode se base sur l'hypothèse Lambertienne que l'intensité est constante (au moins localement) entre deux images d'une même scène prises à un intervalle très court :

$$I(x,t) \simeq I(x+\delta x, t+\delta t) \tag{2.1}$$

Où δx est le déplacement de la zone (x,t) à l'ordre 1 de l'image après un temps δt. Les termes d'ordre supérieur sont négligeables. Les séries de Taylor nous permettent d'écrire :

$$I(x,t) = I(x_0,t_0) + \nabla I \cdot \delta x + \delta t I_t \tag{2.2}$$

Avec $\nabla I = (I_x, I_y)$ et I_t les dérivées partielles au premier ordre de $I(x,t)$.

On a donc :

$$\begin{aligned} I(x,t) &= I(x,t) + \nabla I \cdot \delta x + \delta t I_t \\ 0 &= \nabla I \cdot \delta x + \delta t I_t \\ 0 &= \nabla I \cdot \frac{\delta x}{\delta t} + I_t \\ \nabla I \cdot \mathsf{v} + I_t &= 0 \end{aligned} \tag{2.3}$$

Où ∇I est le gradient d'intensité spatial et $\mathsf{v} = (u,v)$ est la vélocité de la caméra. L'équation 2.3 est connue comme étant l'équation de contrainte du flux optique, mais elle n'est pas suffisante pour trouver les deux composants de v. Par conséquent, on réalise l'approximation de cette vélocité en mesurant le déplacement de petites zones de l'image par des méthodes de mise en correspondance. Voyons plus en détail la mise en correspondance de points d'intérêt et d'imagettes.

Mise en correspondance de points d'intérêt Dans une image les points d'intérêts sont sélectionnés à des positions caractéristiques tels que des coins, des joints en T [Gizzare, 2004] ou des changements de texture flagrants [Rao et al., 2014]. Les détecteurs servent à identifier des zones caractéristiques d'une image. Ils doivent être marquants afin de pouvoir être retrouvés malgré des changements de position, d'échelle, d'orientation, de luminosité ou de bruit. Un détecteur doit être répétable, c'est-à-dire qu'un même détecteur doit être trouvé avec différentes conditions de visibilité.

Ensuite, le voisinage de chaque point d'intérêt est représenté comme un vecteur de caractéristiques d'apparence, cette représentation est le descripteur. L'appariement entre deux détecteurs sur deux images différentes est possible grâce au descripteur. Les descripteurs doivent être distinctifs et robustes au bruit, aux erreurs de détections et au changement de point de vue et d'illumination.

L'extraction des points d'intérêt grâce aux détecteurs est réalisée dans les deux images à comparer entre elles, puis les descripteurs des points d'intérêt alors extraits sont comparés afin de trouver des ressemblances [Klein and Murray, 2008].

Afin d'illustrer les détecteurs et descripteurs, voici deux exemples, le détecteur de Harris et le descripteur SIFT.

Harris Le détecteur de Harris [Harris and Stephens, 1988] utilise les coins comme critère de comparaison entre deux images. Ces coins sont définis par deux traits qui se croisent et un changement brusque de texture. L'image est alors divisée en zones représentant des coins (deux traits qui se croisent), des traits uniques, ou des zones vides. Ce détecteur est invariant à la rotation, mais pas au changement d'échelle.

Il existe des alternatives au détecteur de Harris tel que FAST [Rublee and Bradski, 2011] qui s'exécute plus rapidement en utilisant des zones circulaires au lieu de carrés et des algorithmes d'apprentissage afin d'accélérer les extractions de détecteurs.

Sift (Scale Invariant Feature Transform) [Gizzare, 2004] est un procédé double qui réalise l'extraction de détecteurs et leur description.

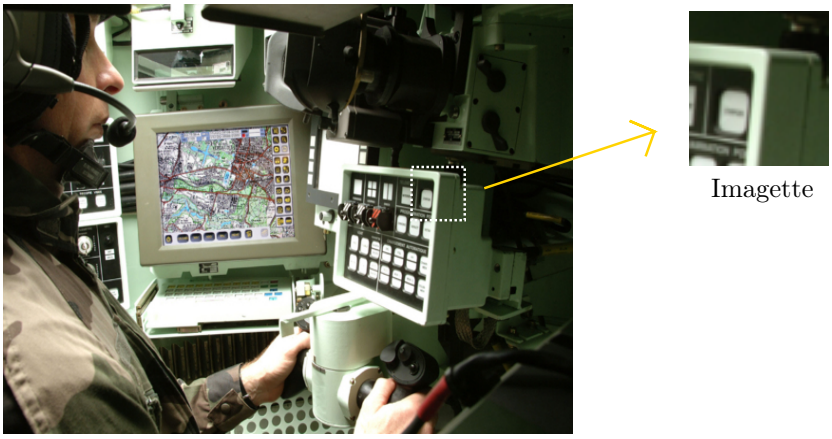
SIFT utilise comme détecteurs les extrema locaux d'une différence de fonctions gaussiennes appliquées sur une série d'une même image à différentes échelles. Ces détecteurs sont invariants à l'échelle et à la rotation. De plus, ils sont partiellement invariants aux changements de luminosité et aux changements de point de vue. Il en résulte des points d'intérêt bien localisés à la fois spatialement et sur le domaine des fréquences, réduisant ainsi les éventuels problèmes liés aux occlusions, ou au bruit. Un grand nombre de ces points d'intérêt peut être extrait d'une seule image en restant malgré tout, tous pertinents. Leur coût d'extraction est minimisé grâce à l'utilisation d'une approche de filtrage en cascade [5].

Ensuite, les gradients de la région entourant chaque point d'intérêt sont mesurés à une échelle sélectionnée. Le descripteur est alors construit grâce au calcul des histogrammes d'orientation sur ces gradients. Ce descripteur sera de dimension 128 [Lowe, 2004].

Des améliorations de l'algorithme SIFT existent, tel que SURF [Bay et al., 2006] qui accélère le calcul des descripteurs jusqu'à un facteur 4 ou ORB [Rublee and Bradski, 2011], utilisable à des fréquences permettant l'exécution temps réel et moins sensible au bruit. En outre, on trouve aussi des alternatives telles que BRIEF [Bay et al., 2006] qui possède par contre les performances moins bonnes que SIFT sur le plan de l'invariabilité à la rotation.

Mise en correspondance d'imagettes Pour faire du recalage d'imagettes, on détermine une imagette (T) de taille t et de coordonnées (u_1, v_1) dans une image plus grande (I_1) qui va être

5. Un filtrage en cascade est un filtrage appliqué sur différentes échelles d'une image afin de réduire la dimension de l'image à traiter.



(a) Image et son imagette associée

(b) On recherche dans une nouvelle image (la première, mais tronquée) où se trouve l'imagette

FIGURE 2.7 – Mise en correspondance d'imagettes.

recherchée dans une autre image (I_2) (Figure 2.7). Cette imagette va être comparée à différentes positions de l'image (I_2) pixel par pixel afin de déterminer le degré de ressemblance (par exemple par le calcul de la corrélation croisée normalisée R, équation 2.4) dans toute l'image. Chaque coordonnée (u_2, v_2) de (I_2) va donner une valeur de corrélation R en fonction de l'intensité de la correspondance entre le template (T) de (I_1) et le template (T_2) de taille t mais aux coordonnées (u_2, v_2) de (I_2).

La corrélation croisée normalisée n'est qu'un exemple de détecteur de recalage d'imagettes. Elle est définie par R. Bracewell dans [Bracewell, 1965] par l'équation suivante :

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y') \cdot I(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}} \tag{2.4}$$

Après avoir parcouru toute l'image (I_2), on récupère le résultat le plus élevé qui indique les coordonnées (u_{max}, v_{max}) ou devrait se placer l'imagette (T) dans l'image (I_2) pour maximiser la correspondance entre (T) et (T_2).

Le vecteur de coordonnées $(u_{max} - u_1, v_{max} - v_1)$ donne alors le déplacement entre les deux images (en pixels). Cependant, le défaut de cette méthode, bien que très rapide et simple, et qu'elle ne prend pas en compte les changements d'échelle ni les rotations en roulis.

Ces exemples offrent un aperçu des étapes possibles pour le calcul de l'*egomotion*. Néanmoins, notons que des méthodes similaires à l'*egomotion* s'appliquent pour les données autres que des images. Ces méthodes de mise en correspondance de données sont communément appelées méthodes de recalage « matching methods », et peuvent traiter des nuages de points ou des cartes entières.

L'utilisation du recalage Le recalage ou « matching » consiste à mettre en correspondance deux données. Dans le cas de la cartographie mobile, les données peuvent être soit des cartes (en graphe [Estrada et al., 2005], en grille [Grisetti et al., 2007] ou autre), soit des nuages de points [Steux and ElHamzaoui, 2010]. Plusieurs méthodes existent telles que le « Map matching » [Bosse and Zlot, 2008], qui consiste à mettre en correspondance des cartes, le « Scan matching » [Zhao et al., 2010] qui corrèle des trames laser 2D ou encore le « Point Cloud matching » analysant deux nuages de points 3D [Fioraio and Konolige, 2011].

Le but du recalage est double, il permet à la fois de retrouver une transformation appliquée entre deux données et/ou de créer une nouvelle donnée composée des deux données d'entrées corrélées. Il existe des algorithmes plus ou moins complexes et performants, du plus simple et léger [Steux and ElHamzaoui, 2010] récupérant un scan 2D (ou trame) et l'odométrie du robot pour calculer le mouvement réalisé par la plateforme d'acquisition grâce un filtre particulaire et construire une carte, aux plus lourds tel que l'ICP [Segal et al., 2009]. L'ICP est une méthode très consommatrice en temps de calcul et c'est pourquoi elle est essentiellement utilisée en post-traitement. Simple dans son architecture (elle se résume en deux points), elle nécessite un très grand nombre de calculs récursifs sur l'ensemble des points du nuage de points. Par conséquent, son temps de calcul augmente au fur et à mesure de l'acquisition. En effet, sa complexité algorithmique est de l'ordre de $O(N_p N_x)$ avec N_p et N_x le nombre de points à comparer entre eux.

[Zhang and Singh, 2014a] traite quant à lui d'un recalage non pas de trames laser entières, mais de caractéristiques structurelles d'une trame laser dans les articles abordant son algorithme LOAM (Lidar Odometry and Mapping in Real-time). Il y explique que les caractéristiques structurelles d'une trame laser sont les coins, les segments et les plans, et qu'ils seront comparés avec les trames suivantes pour retrouver la transformation liant ces deux trames. Cette transformation va ensuite permettre de retrouver le mouvement effectué par le capteur, comme le ferait une méthode traitant des images.

Comme expliqué dans les sections précédentes, dans le cadre de la cartographie mobile, la cartographie est soumise au positionnement qui lui permet de se construire. En revanche, dans le cadre des méthodes dites de cartographie et de localisation simultanée (SLAM), cette relation n'est plus unidirectionnelle. La section suivante va vous présenter cette théorie et les algorithmes qui s'y rattachent.

2.3 Cartographie par méthode SLAM

2.3.1 Principe

Le SLAM (*Simultaneous Localization And Mapping*) consiste à permettre à un robot ou à un véhicule autonome de construire et de mettre à jour une carte d'un environnement inconnu (sans *a priori*), et de s'y localiser [Durrant-Whyte and Bailey, 2006a]. Contrairement à la cartographie mobile où la localisation est utilisée unidirectionnellement pour faire de la cartographie, le SLAM utilise en plus la cartographie pour se localiser. Souvent assimilé au problème de « l'œuf ou la poule ? » ce sujet a commencé à trouver réponse auprès de la communauté scientifique depuis une

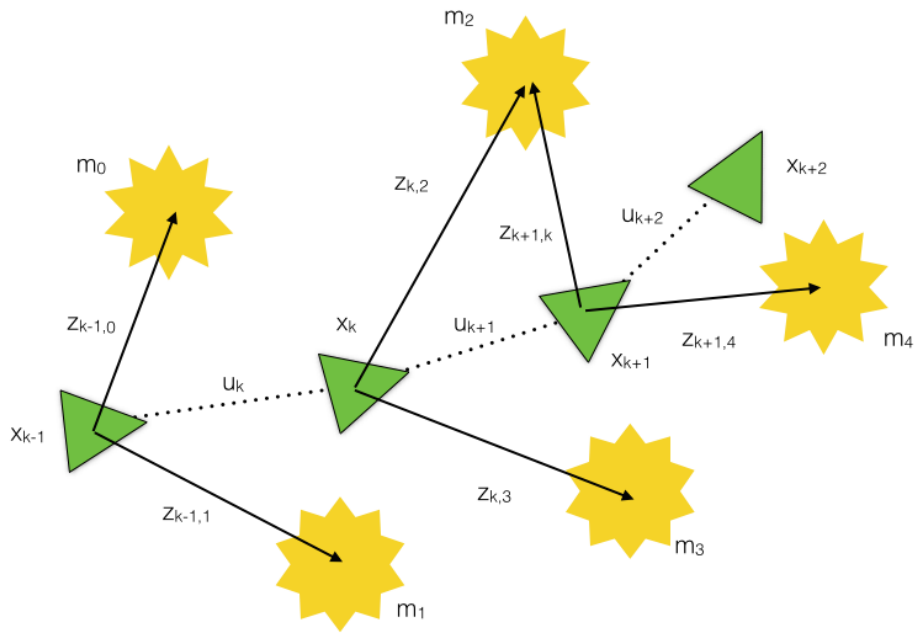


FIGURE 2.8 – Le problème essentiel du SLAM : Une estimation simultanée de la position du robot et de son environnement.

trentaine d'années [Hager and Durrant-Whyte, 1986]. L'automatisation de robots en dépendant beaucoup, le SLAM est un problème majeur en robotique [Durrant-Whyte and Bailey, 2006a].

Formulation probabiliste initiale La suite de cette section décrit les équations régissant la problématique de la cartographie et de la localisation en simultanée par un système robotique réalisant des observations dans un environnement inconnu et sans *a priori*.

En considérant un robot se déplaçant dans un environnement en faisant des observations grâce à ses capteurs, on peut définir le système suivant (Figure 2.8 [Durrant-Whyte and Bailey, 2006a,b]) :

- x_k : position et orientation du robot (attitude)
- u_k : le vecteur de contrôle appliqué au temps $k-1$ qui permet de conduire le robot à l'état x_k au temps k
- m_i : un vecteur décrivant la position du $i^{\text{ème}}$ amer [6] dont la position réelle est indépendante du temps
- $z_{k,i}$: une observation faite par le robot de l'amer i au temps k. Dans la suite de la discussion nous parlerons plutôt de z_k pour désigner la mesure de tous les amers en k
- $X_{0:k} = [x_0, ..., x_k]$: Historique de positions du robot
- $U_{0:k} = [u_0, ..., u_k]$: Historique des entrées de contrôle
- $m = [m_1, ..., m_k]$: Les paramètres de tous les amers qui forment la carte
- $Z_{0:k} = [z_0, ..., z_k]$: Les paramètres de toutes les observations des amers

Le problème du SLAM est défini par l'estimation de la densité de probabilité suivante :

$$p(x_k, m | Z_{0:k}, U_{0:k}, x_0) \tag{2.5}$$

6. Les amers sont des points de repère fixes non ambigus. Ce terme emprunté à la navigation maritime est très utilisé en robotique.

On cherche à calculer la densité de probabilité traduisant les enregistrements des observations et des contrôles pour tout instant k ainsi que l'état initial du robot. Pour le calcul on a besoin qu'un modèle de transition d'état et d'observation soient définis. Le modèle d'observation décrit la densité de probabilité de faire une observation z_k quand la position du robot et la position du point de repère sont connues. Les observations sont décrites par la densité de probabilité suivante :

$$p(z_k|x_k, m) \tag{2.6}$$

De plus, l'hypothèse gaussienne étant appliquée, les observations peuvent s'écrire sous la forme d'une gaussienne telle que :

$$p(z_k|x_k, m) \sim \mathcal{N}(h(x_k, m), R_k) \tag{2.7}$$

Où $h(.)$ représente la géométrie de l'observation et R_k sa covariance. Le mouvement est quant à lui décrit par :

$$p(x_k|x_{k-1}, u_k) \sim \mathcal{N}(f(x_{k-1}, u_k), Q_k) \tag{2.8}$$

Où $f(.)$ représente le modèle dynamique ou cinématique du robot et Q_k sa covariance. L'état x_k dépend seulement de l'état qui le précède x_{k-1} ayant subi le contrôle u_k et pas ceux qui précédent. Cet état de transition est un « processus de Markov caché en ligne à canal sans mémoire ».

Le calcul de la densité de probabilité (2.5) s'effectue alors grâce aux deux étapes suivantes répondant au processus des chaînes de Markov :

1) La mise à jour temporelle :

$$\begin{aligned} p(x_k, m|Z_{0:k-1}, U_{0:k}, x_0) &= \int p(x_k|x_{k-1}, U_k) \\ &\times p(x_{k-1}, m|Z_{0:k-1}, U_{0:k-1}, x_0)\ dx_{k-1} \end{aligned} \tag{2.9}$$

2) La mise à jour des mesures :

$$p(x_k, m|Z_{0:k}, U_{0:k}, x_0) = \frac{p(z_k|x_k,m)\ \mathrm{x}\ p(x_k,m|Z_{0:k-1},U_{0:k},x_0)}{p(Z_k|Z_{0:k-1},U_{0:k})} \tag{2.10}$$

La création de la carte peut alors être formulée comme le calcul de la densité conditionnelle :

$$p(m|X_{0:k}, Z_{0:k}, U_{0:k}) \tag{2.11}$$

La localisation peut être formulée ainsi :

$$p(x_k|Z_{0:k}, U_{0:k}, m) \tag{2.12}$$

Par ces deux équations, on comprend mieux la problématique du SLAM et ses difficultés. D'une part, la création de la carte (équation (2.11)) suppose que la position du robot x_k soit connue à chaque instant. Ensuite, une carte m au temps k est créée en fusionnant les paramètres précédents d'observations $Z_{0:k}$ et de mouvements $U_{0:k}$. D'autre part, l'équation de localisation (équation (2.12)) suppose, elle, que tous les amers soient connus avec certitude et l'objectif est de calculer la position du véhicule avec le respect de ces points de repère.

Trouver des solutions pour le problème de SLAM impose de trouver des représentations appropriées des observations (équation (2.6)) et du mouvement (équation (2.8)) qui acceptent des calculs antérieurs et postérieurs (équations (2.9) et (2.10)), efficaces et robustes.

Grâce à sa faculté d'outrepasser les problèmes de non-linéarités, l'utilisation du filtre de Kalman étendu pour résoudre les problèmes de SLAM est de plus en plus commune. En effet, ce filtre fait

l'hypothèse que les estimations postérieures sont des gaussiennes de hautes dimensions [S. Thrun M. Montemerlo, 2004]. Par contre, il est nécessaire de mentionner qu'une alternative importante est de décrire le mouvement du robot comme un ensemble d'échantillons d'une probabilité non gaussienne plus générale. Dans ce cas le filtre de Kalman n'est plus utilisable. Cette approche nécessite l'utilisation de filtres à particules de « Rao-Blackwell » tels que l'algorithme FastSLAM pour résoudre le problème de SLAM. Dans la suite de ce chapitre, nous allons détailler ces différentes approches.

2.3.2 Algorithmes de filtrage associés au SLAM

Deux principaux algorithmes de filtrage sont utilisés dans les méthodes de SLAM. Il s'agit des algorithmes utilisant le filtrage de Kalman et les algorithmes de filtrage particulaire. Dans cette section nous allons présenter l'approche dite de filtre de Kalman étendu et le filtre de Kalman inodore qui traite la non linéarité autrement, améliorant ainsi la précision des estimations. Dans un deuxième temps nous aborderons le sujet du filtre particulaire via l'explication de FastSLAM.

2.3.2.1 EKF-SLAM

Le filtre de Kalman simple s'applique dans le cas linéaire gaussien où $f(.)$ et $h(.)$ sont linéaires et que les fonctions de densité de probabilité citées dans la section précédente sont gaussiennes. Mais la plupart des systèmes physiques sont non-linéaires donc le filtre de Kalman simple n'est plus applicable. L'EKF SLAM est une méthode ayant pour but de traduire les équations d'observation et de mouvement sous la forme d'un espace d'états avec un bruit gaussien afin d'obtenir la meilleure estimation de x_k, ainsi qu'une estimation de l'erreur d'estimation d'état. Cependant, dans ce cas, $f(.)$ et $h(.)$ seront non linéaires et ne pourront pas être appliquées directement au calcul de la covariance. Pour régler ce problème, la stratégie de l'EKF est de linéariser le système et d'en calculer les jacobiens [S. Thrun M. Montemerlo, 2004].

Le filtre de Kalman étendu s'inspire du filtre de Kalman pour calculer par récurrence une estimation des moyennes et variances $\hat{z}_k$, $\hat{x}_k$, $\hat{x}_{k+1}$, $p(x_k|x_{k-1}, u_k)$, $p(x_{k+1}|x_k, u_{k+1})$. Ainsi, d'après l'hypothèse Gaussienne, l'équation de mouvement $p(x_k|x_{k-1}, u_k)$ devient :

$$p(x_k|x_{k-1}, u_k) \sim \mathcal{N}(f(x_{k-1}, u_k), Q_k) \tag{2.13}$$

De la même manière on pour définir l'équation d'observation $p(z_k|x_k, m)$ devient :

$$\begin{aligned} p(z_k|x_k, m) \text{ avec } z_k &= h(x_k, m) + v_k \\ \Leftrightarrow z_k &= h(x_k, m) + v_k \text{ avec } v_k \sim \mathcal{N}(0, R_k) \end{aligned} \tag{2.14}$$

$h(.)$ décrit quant à elle la géométrie de l'observation et v_k est le bruit de mesure gaussien de moyenne nulle et de covariance R_k.

La linéarisation de l'EKF est donnée par l'estimation bayésienne optimale de variance minimale qui est le maximum a *posteriori*, à savoir, la moyenne et la covariance linéarisées de la densité de probabilité $p(x_k, m|Z_{0:k}, U_{0:k}, x_0)$. La moyenne s'écrit :

$$\begin{bmatrix} \hat{x}_{k|k} \\ \hat{m}_k \end{bmatrix} = E \begin{bmatrix} x_k \\ m \end{bmatrix} | Z_{0:k}, U_{0:k} \tag{2.15}$$

La covariance est quant à elle définie par :

$$
\begin{aligned}
P_{k|k} &= \begin{bmatrix} P_{xx} & P_{xm} \\ P_{xm}^T & P_{mm} \end{bmatrix}_{k|k} \\
&= E\left[\begin{pmatrix} x_k - \hat{x}_k \\ m - \hat{m}_k \end{pmatrix} \begin{pmatrix} x_k - \hat{x}_k \\ m - \hat{m}_k \end{pmatrix}^T | Z_{0:k}, U_{0:k} \right]
\end{aligned} \tag{2.16}
$$

L'EKF offre un estimateur linéaire sous-optimal. Son utilisation est très répandue, car il est simple d'utilisation d'autant plus que sa problématique reste la même que le Kalman simple, à savoir, calculer la covariance et l'espérance de la densité de probabilité $p(x_k, m|Z_{0:k}, U_{0:k}, x_0)$ aux nouvelles observations. L'EKF SLAM présente cependant quatre problèmes majeurs [Ray, 1997] :

- Le premier, pas particulier à l'EKF, est un problème de convergence, l'incertitude associée à la carte converge vers une borne inférieure, mais jamais l'incertitude ne peut être nulle. En effet, la variance des amers individuels converge vers une borne inférieure déterminée par les incertitudes initiales de position du robot et des observations [Julier and Uhlmann, 1997].
- Le second défaut de cette méthode est l'effort de calcul nécessaire à son fonctionnement de complexité $O(n^2)$ [Paz et al., 2007]. En effet, la mise à jour des observations nécessite que la matrice de covariance soit recalculée à chaque fois qu'une observation est faite. Cela signifie que le temps de calcul augmente quadratiquement avec le nombre d'amers [Tanizaki and Mariano, 1996]. Par contre, de nombreuses variantes de l'EKF ont été développées afin de fonctionner avec des milliers d'amers.
- Le troisième défaut de cette méthode est sa fragilité face aux associations d'observations incorrectes [Montemerlo and Thrun, 2003]. En atteste la difficulté de réaliser une fermeture de boucle (*Loop Closure*) avec certains algorithmes utilisant des EKF. En effet, l'EKF ne peut pas oublier ses fausses associations, il lui est donc difficile de reconnaître quand il revoit le même amer. Ce problème est d'autant plus important quand les amers sont complexes et que leur apparence dépend du point de vue.
- Le quatrième problème est que le filtre de Kalman étendu utilise des modèles linéaires pour résoudre des problèmes qui ne le sont pas (observation et mouvement). Cette non-linéarité peut mener à des incohérences dans les solutions obtenues. La convergence ainsi que la consistance [7] ne peuvent être garanties que dans les cas linéaires [Julier et al., 2001].

Pour résoudre ces défauts, des alternatives ont été trouvées, comme le « filtre inodore » présenté dans la section suivante qui s'intéresse à résoudre le quatrième défaut de l'EKF.

2.3.2.2 Filtre de Kalman « inodore » — UKF

Comme expliqué dans [Julier and Uhlmann, 1997], l'« Unscented Kalman Filter » (UKF) a pour objectif de régler les problèmes d'approximation de l'EKF. L'état est toujours représenté par une variable gaussienne aléatoire, mais dans l'UKF elle est définie en utilisant un ensemble minimal de points échantillonnés, soigneusement choisis, appelés σ-points. Ces points sont sélectionnés afin de capturer de manière complète (la gaussienne est modélisée par un nombre fini de particules [Lefaudeux, 2013]) et précise les véritables moyenne et covariance de la variable gaussienne, jusqu'au troisième ordre et pour n'importe quelle non-linéarité. Cet UKF se base sur le principe de la transformation inodore comme l'explique [Wan and Van Der Merwe, 2002].

La transformation inodore La transformation inodore (*Unscented Transform*) est une méthode qui sert à calculer des moments statistiques d'une variable aléatoire sous-jacente d'une transformation non linéaire [Wan and Van Der Merwe, 2002]. Pour la calculer, on considère une propagation

7. Comme l'explique [Joly, 2010], la consistance *peut être vue comme la qualité de l'incertitude fournie par le filtre (à travers la matrice de variances-covariances) par rapport à l'erreur réelle.*

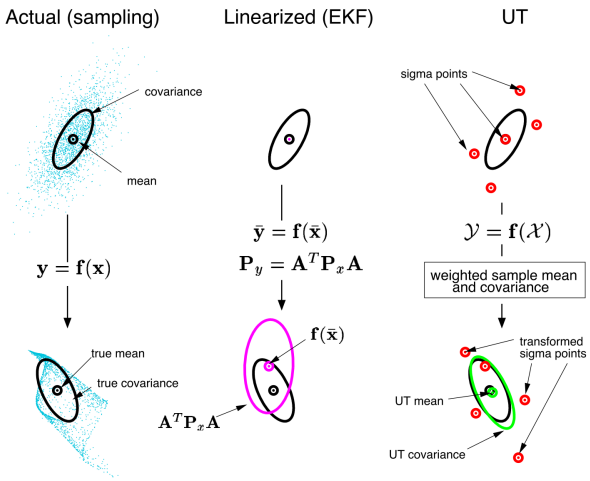


FIGURE 2.9 – Comparaison de la moyenne et de la covariance données par la transformation inodore face aux résultats donnés par un EKF, le tout mit en perspective des moyennes et covariances véritables. [Wan and Van Der Merwe, 2000]

d'une variable aléatoire $x \in \mathbb{R}^n$ [Abbeel, 2015] de dimension L dans une fonction non linéaire, $y = g(x)$. De plus, on définit $\bar{x}$, sa moyenne et P_x, sa covariance.

Pour calculer les moments statistiques de y, on considère la matrice χ de $(2L+1)$ σ-points χ_i. Ces particules ont un poids W_i^m pour le calcul de la moyenne et W_i^c pour le calcul de la covariance. Ces σ-points sont définis par les formules suivantes :

$$\begin{aligned} & \chi_0 = \bar{x} \\ \forall i \in [\![1; L]\!] \quad & \chi_i = \bar{x} + (\sqrt{(L+\lambda)P_x})_i \\ \forall i \in [\![L+1; 2L]\!] \quad & \chi_i = \bar{x} - (\sqrt{(L+\lambda)P_x})_{i-L} \end{aligned} \tag{2.17}$$

Leurs poids sont définis par :

$$\begin{aligned} W_0^m &= \frac{\lambda}{L+\lambda} \\ W_0^c &= \frac{\lambda}{L+\lambda} + (1-\alpha^2+\beta) \\ \forall i \in [\![1; 2L]\!] \quad W_i^m &= W_i^c \frac{1}{2(L+\lambda)} \end{aligned} \tag{2.18}$$

De plus, les paramètres définissant ces équations sont tel que :

- $\lambda = (\alpha^2(L+\kappa) - L)$ est un paramètre d'échelle.
- α détermine la dispersion des σ-points autour de $\bar{x}$, et il est habituellement défini comme une faible valeur positive (10^{-3}).
- κ est le deuxième paramètre d'échelle souvent fixé à 0
- β est utilisé afin d'insérer une connaissance *a priori* de la distribution de x (dans le cas d'une distribution gaussienne, la valeur optimale de β est 2).
- $(\sqrt{(L+\lambda)P_x})_i$ est la i^{eme} ligne de la racine carrée de la matrice formée par $A = (L+\lambda)P_x$. En effet, la matrice de covariance étant symétrique, et semi-positive, elle admet une racine carrée telle que $BB^T = A$. B est sa racine carrée [UJF-Grenoble, 2015].

Les σ-points sont propagés suivant la fonction non linéaire :

$$\mathcal{Y}_i = g(\chi_i), \forall i \in [\![0; 2L]\!] \tag{2.19}$$

La moyenne et la covariance pour y sont estimées en utilisant une moyenne et une covariance des échantillons pondérés des σ-points postérieurs appelée $\mathcal{Y}_i$.

$$\begin{aligned} \bar{y} &= \sum_{i=0}^{2L} W_i^m \mathcal{Y}_i \\ P_y &= \sum_{i=0}^{2L} W_i^c [\mathcal{Y}_i - \bar{y}][\mathcal{Y}_i - \bar{y}]^T \end{aligned} \tag{2.20}$$

Cette transformation appliquée aux équations de l'EKF donne lieu au Filtre de Kalman inodore.

De la transformation inodore au filtre de Kalman inodore (UKF) Le filtre de Kalman inodore (UKF) découle directement de la transformation inodore vers l'estimation récursive de l'équation (2.13) de l'EKF où la variable aléatoire x_k est redéfinie comme la concaténation de la variable d'état et des variables de bruit w_k et v_k [Wan and Van Der Merwe, 2002] :

$$x_k^a = [x_k^T w_k^T v_k^T]^T \tag{2.21}$$

La sélection des σ-points (équation (2.17)) est alors appliquée à cette nouvelle variable aléatoire pour calculer la nouvelle matrice de σ-points correspondante χ_i^a. L'utilisation des σ-points permets de calculer directement les transformations non linéaires de l'EKF avant d'appliquer la transformation inodore tel qu'exprimé dans l'algorithme 1 [Wan and Van Der Merwe, 2002].

Ainsi la linéarisation et le calcul des jacobiens sont évités, gagnant ainsi en précision et en temps de calcul. Par contre, l'UKF ne résout pas le problème d'inconsistance de l'EKF [Wan and Van Der Merwe, 2002]. On peut noter que cette méthode diffère de la méthode courante de l'échantillonnage telle que la méthode particulaire de Monte-Carlo [Moulines, 2012] par l'utilisation de la notion d'échantillonnage inodore qui implique un côté aléatoire et un système de pondération « dynamique » pour propager une distribution d'états. Comme le montre la figure 2.9 pour un simple exemple en deux dimensions, la supériorité des précisions des estimations réalisées par la transformation inodore face à l'EKF est avérée, même si elle ne résout pas tous les défauts de l'EKF, et ce sans nécessiter un nombre de calculs plus élevé que l'EKF.

Même si l'UKF utilise des « échantillons », ce n'est pas un filtre dit « à particules » [Ulas and Temeltas, 2012]. En effet, les particules σ-points sont choisies de manière déterministe et non par des tirages aléatoires comme dans les filtres à particules.

2.3.2.3 Le filtre particulaire FastSLAM

Le filtre particulaire FastSLAM a été le premier à représenter directement les modèles de mouvements non linéaires et les modèles d'observations non gaussiens. Cependant, il linéarise tout de même les équations non linéaires d'observation, mais en réalisant une estimation déterministe de l'attitude du véhicule.

FastSLAM utilisant comme base de récursivité l'échantillonnage de Monte-Carlo [Moulines, 2012], l'application directe d'un filtre à particules n'est pas envisageable sur le plan calculatoire au vu de la grande dimension du problème de SLAM. Par contre, il est possible de réduire l'échantillonnage de l'espace de calcul en appliquant la méthode de « Rao-Blackwell » (R-B) [Wu et al., 2008], où la probabilité conjointe est divisée suivant la loi de Bayes :

Algoritme 1 Calcul de l'UKF

On initialise le système avec :

$$\hat{x}_0 = E[x_0]$$
$$P_0 = E[(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T]$$
$$\hat{x}_0^a = E[x^a] = \begin{bmatrix}\hat{x}_0^T & 0 & 0\end{bmatrix}^T$$

$$\hat{P}_0^a = E[(x_0^a - \hat{x}_0^a)(x_0^a - \hat{x}_0^a)^T] = \begin{bmatrix} P_0 & 0 & 0 \\ 0 & P_{v_k} & 0 \\ 0 & 0 & P_{w_k} \end{bmatrix}$$

Avec, P_{v_k} la covariance du calcul de commande et P_{w_k} la covariance du bruit de mesure.
Ensuite, $\forall k \in [\![1;\infty]\!]$, on calcule les σ-points :

$$\chi_{k-1}^a = \left[\hat{x}_{k-1}^a \quad \hat{x}_{k-1}^a \pm \sqrt{(L+\lambda)P_{k-1}^a}\right]$$

Où, λ est un paramètre d'échelle et L la dimension de la variable aléatoire x_k.
La mise à jour temporelle est alors calculée par :

$$\chi_{k|k-1}^{x_k} = f[\chi_{k-1}^{x_k}, \chi_{k-1}^{v_k}]$$
$$\hat{x}_k^- = \sum_{i=0}^{2L} W_i^m \chi_{i,k|k-1}^{x_k}$$
$$\bar{P}_k = \sum_{i=0}^{2L} W_i^c [\chi_{i,k|k-1}^{x_k} - \hat{x}_k^-][\chi_{i,k|k-1}^{x_k} - \hat{x}_k^-]^T$$
$$\mathcal{Y}_{k|k-1} = h[\chi_{i,k|k-1}^{x_k}, \chi_{i,k|k-1}^{w_k}]$$
$$\hat{y}_k^- = \sum_{i=0}^{2L} W_i^m \mathcal{Y}_{i,k|k-1}$$

Dont, W_i^m et W_i^c sont les poids des particules i pour les calculs respectifs de la moyenne et de la covariance.
Enfin, la mise à jour des observations est alors calculée par :

$$P_{\bar{y}_k\bar{y}_k} = \sum_{i=0}^{2L} W_i^c [\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-][\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-]^T$$
$$P_{x_k y_k} = \sum_{i=0}^{2L} W_i^c [\chi_{i,k|k-1} - \hat{x}_k^-][\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-]^T$$
$$\mathcal{K} = P_{x_k y_k} P_{\bar{y}_k\bar{y}_k}^{-1}$$
$$\hat{x}_k = \hat{x}_k^- + \mathcal{K}(y_k - \hat{y}_k^-)$$
$$P_k = P_k^- - \mathcal{K} P_{\bar{y}_k\bar{y}_k} \mathcal{K}^T$$

Avec $x_k^a = \begin{bmatrix}x_k^T v_k^T w_k^T\end{bmatrix}$ et $\chi_k^a = \begin{bmatrix}(\chi^{x_k})^T (\chi^{v_k})^T (\chi^{w_k})^T\end{bmatrix}$

$$p(x_1, x_2) = p(x_2|x_1)p(x_1) \tag{2.22}$$

Cependant, si $p(x_2|x_1)$ peut être représenté analytiquement, seul $p(x_1)$ a besoin d'être échantillonné selon :

$$x_1^{(i)} \sim p(x_1) \tag{2.23}$$

Dans ce cas, la densité de probabilité conjointe est représentée par l'échantillon suivant :

$$\{x_1^{(i)}, p(x_2|x_1^{(i)})\}_i^N \tag{2.24}$$

Alors, la probabilité conjointe du SLAM peut être factorisée au travers d'une composante véhicule et d'une composante de carte conditionnelle :

$$\begin{gathered} p(X_{0:k}, m|Z_{0:k}, U_{0:k}, x_0) = \\ p(m|X_{0:k}, Z_{0:k})p(X_{0:k}|Z_{0:k}, U_{0:k}, x_0) \end{gathered} \tag{2.25}$$

Grâce à cette représentation, la densité de probabilité est sur la trajectoire $X_{0:k}$ plutôt que sur le simple état x_k car les amers sont indépendants quand ils sont conditionnés sur la trajectoire [Durrant-Whyte and Bailey, 2006a]. Cette propriété est le principal atout de FastSLAM et c'est aussi la raison de sa rapidité [Montemerlo et al., 1999], car la carte est représentée comme un ensemble de gaussiennes indépendantes avec une complexité linéaire $O(M \log(K)$ (avec K amers et M particules) plutôt qu'une covariance de carte conjointe avec une complexité quadratique $O(K^2)$ [Calonder, 2006].

Le reste de l'algorithme est un état de *Rao-Blackwell*, où la trajectoire est représentée par des échantillons pondérés dont l'estimation récursive est réalisée par un filtrage particulaire. La carte, quant à elle, est calculée de façon analytique en général par un filtre de Kalman ou un EKF. Pour simplifier, FastSLAM emploie $M \times K$ « petits » EKFs (K EKF sur chaque particule $M_{(i)}$) alors que EKF SLAM en utilise un de grande dimension [Calonder, 2006].

Malgré ce gros avantage calculatoire, le problème de FastSLAM est qu'il n'est pas capable d'oublier les erreurs du passé, à l'instar de l'EKF, les erreurs d'estimation des précédentes positions sont enregistrées dans la carte. D'ailleurs, à chaque fois qu'un rééchantillonnage est réalisé et qu'une particule n'est pas choisie, tout un historique de positions et d'hypothèse de carte disparaît avec elle. Par conséquent, il en résulte une perte statistique pour l'estimation de l'amer amputé des particules éjectées [Calonder, 2006]. Ce défaut est problématique dans le cas des fermetures de boucles où l'atténuation de l'erreur n'est pas possible. Ainsi, les filtres particulaires sont idéaux lorsque le système est réellement à « oubli du passé » comme c'est le cas dans les systèmes d'exploration. Dans ce cas d'application, les résultats montrent que FastSLAM est efficace pour réaliser de la cartographie, que ce soit sur le plan de la précision ou de la rapidité d'exécution [Lin et al., 2011]. En effet, [Calonder, 2006] montre que le filtre à particules, en plus d'être plus rapide, est plus robuste que l'EKF car il fait disparaître les mauvaises association grâce au principe de rééchantillonage en fonction de la pondération des particules.

Cette section a présenté les différentes manières d'obtenir les informations de localisation et de cartographie, mais des algorithmes de plus hauts niveaux permettent de hiérarchiser ses données afin de créer des cartes plus cohérentes. La prochaine section va en présenter deux.

2.3.3 SLAM : correction d'erreurs et simplification

Des méthodes d'architecture de données permettent de corriger des erreurs d'estimation au travers de l'étude ou de la simplification de la carte ou de sa structure. Dans cette section nous allons en présenter deux très connues, la méthodologie en graphe et la fermeture de boucle (qui peut en découler).

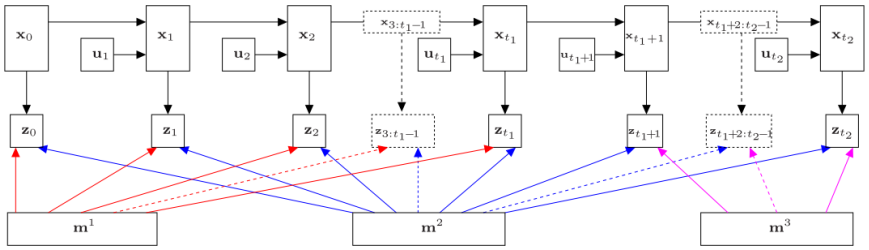


FIGURE 2.10 – Dans cette figure on peut voir le réseau bayésien du SLAM où les nœuds $x_{3:t_1-1}$ et $x_{t_1+2:t_2-1}$ sont calculés grâce aux données des variables d'états de la chaîne de Markov. [Joly and Rives, 2009]

2.3.3.1 Les méthodes en graphe

Les méthodes en graphe peuvent construire leur graphe de deux manières, soit elles utilisent les données capteurs, soit elles le composent de « sous-cartes ».

Dans une méthode en graphe utilisant les données du capteur [Grisetti and Kuummerle, 2010], chaque nœud du graphe indique la position d'acquisition à un instant t et les mesures capturées à cette position. Dans ce cas, l'arrête entre deux nœuds représente la contrainte spatiale liant ces deux nœuds. Ces contraintes sont données par les observations et les mouvements [Kümmerle et al., 2011].

Une autre possibilité est de construire un graphe composé de sous-cartes à partir d'une carte globale [Pini et al., 2008]. Ici, l'intérêt va être de réduire la dimension des données à traiter et ainsi de réduire le coût de calcul. Il est aussi prouvé que ce type de méthode peut augmenter la cohérence de la carte créée. Dans le cas de [Joly and Rives, 2009], la structure en graphe est caractérisée par cette cohérence qui est un paramètre pour créer une nouvelle sous-carte (un nœud). Quand une sous-carte arrive en limite de cohérence, une nouvelle est créée, ainsi la carte est globalement cohérente grâce à ses multiples cohérences locales.

La méthodologie en graphe se prête bien au problème du SLAM. En effet, la figure 2.10 montre que de la structure en graphe expliquée précédemment s'échappe de manière assez évidente la structure markovienne de la problématique du SLAM.

La mise en correspondance d'acquisitions passées et présentes est appelée la fermeture de boucle et permet d'augmenter encore la cohérence de la carte en réduisant les incertitudes d'estimation.

2.3.3.2 La fermeture de boucle

La fermeture de boucle ou « Loop Closure » [Triggs et al., 2000] regroupe les méthodes permettant de produire des structures qui se rejoignent de manière optimale, c'est une approche globale, qui répercute les corrections sur l'ensemble des données. Par exemple, lors de la cartographie de l'intérieur d'un bâtiment, si l'on passe deux fois par le même couloir, la fermeture de boucle a pour but de les repositionner de manière optimale au même endroit.

L'optimale sera obtenue par la minimisation de fonctions de coûts. Ces méthodes sont très pratiques dans le cadre de cartographie à grande échelle, ou le biais des capteurs ou des approximations peut, en augmentant au fil du temps, devenir très important et supprimer la cohérence d'une carte. La fermeture de boucle permet alors de corriger l'erreur acquise et de repropager la correction à l'ensemble de la carte (Figure 2.11 page ci-contre). Cette étape est bien sûr simplifiée par une structure en graphe ou les liens entre les différents nœuds sont plus simples à corriger que l'ensemble des variables d'états du système indépendamment. **Dans le cadre de notre étude, la fermeture**

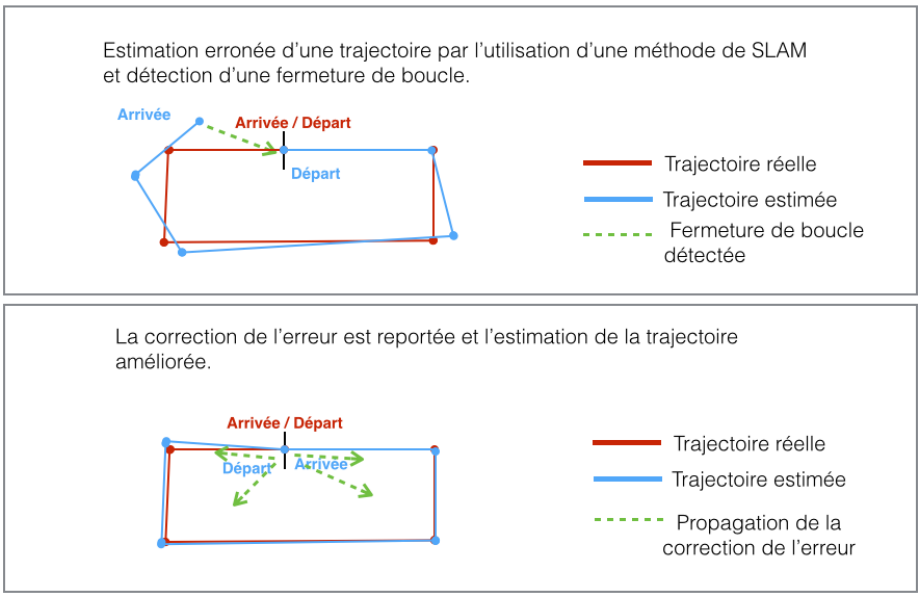


FIGURE 2.11 – Explication schématique d'une fermeture de boucle. Premièrement, une estimation de trajectoire erronée est réalisée, mais une fermeture de boucle est détectée. Dans un second temps, cette fermeture de boucle permet de corriger l'erreur accumulée.

de boucle ne nous intéresse pas, car nous ne nous orientons pas vers une optique de cartographie à grande échelle.

2.3.4 Conclusion sur la cartographie

Dans cette section nous avons étudié les différentes méthodes qui composent le SLAM. Le SLAM est une méthode intéressante surtout dans le cas où une carte de grande dimension doit être acquise, ou que plusieurs passages doivent être réalisés au même endroit.

Cependant, pour notre application, les approches utilisées en cartographie mobile nous semble plus pertinente. En effet, le SLAM apporte une forte charge calculatoire au système, charge qui dans notre cas ne serait pas justifiée par la seule volonté d'une carte courante et de faible dimension. Ainsi, la cartographie mobile, plus simple dans sa construction, nous semble être une meilleure réponse à notre problématique actuelle. De plus, au regard des vibrations du véhicule, de l'environnement et de notre volonté d'indépendance vis-à-vis du GPS, nous nous orientons vers un positionnement par étude de déplacement utilisant le recalage ou l'*egomotion*.

La cartographie mobile nous permettra donc, au travers d'un positionnement par calcul de déplacement, de reconstruire une carte de notre environnement courant. Dans la section suivante, nous présentons les méthodes qui permettent l'analyse d'un tel environnement ainsi construit afin d'en extraire les dangers potentiels.

2.4 Analyse du terrain

Le moyen de cartographie étant posé nous allons maintenant voir comment analyser la carte qui sera produite pour en extraire les dangers. En effet, l'objectif de notre système et d'aider le pilote à visualiser les dangers se trouvant sur son chemin, il est donc nécessaire de mettre en exergue ces obstacles par une extraction pertinente.

Ainsi, nous présentons les attributs utiles à la détermination de ces obstacles puis, nous verrons comment les détecter.

2.4.1 Franchissement : une définition approfondie des obstacles

2.4.1.1 Généralités

D'après le dictionnaire Larousse, un obstacle est « ce qui empêche d'avancer » mais on peut transposer cette définition à notre application en précisant que **les obstacles sont tous les objets ou aspérités du sol empêchant un véhicule de poursuivre une trajectoire sur laquelle un obstacle se trouve.** On peut classifier les obstacles en deux types :

- Les obstacles « positifs » qui se trouvent au-dessus du sol par exemple, les objets imposants, les bâtiments, les autres véhicules, les personnes ou les marches montantes.
- Les obstacles « négatifs » qui se trouvent en dessous du sol et qui représentent les trous (nids de poule, cassis, fossés) ou les marches descendantes.

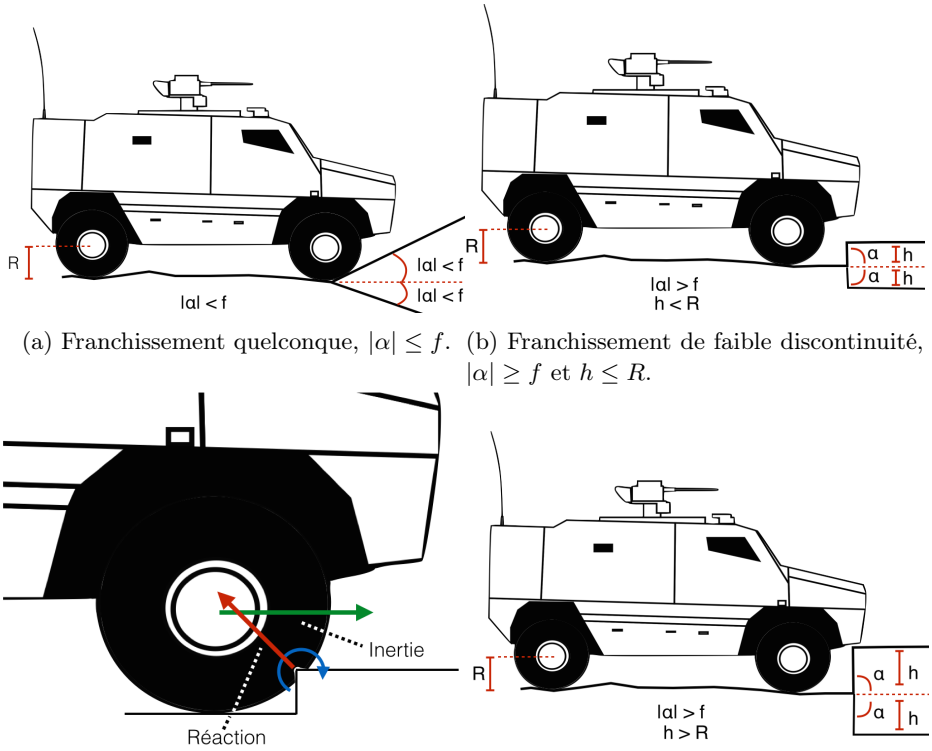
Afin de définir au mieux les obstacles, il faut savoir ce qu'un véhicule peut, ou ne peut pas, surmonter. Néanmoins, dans le cadre de cette thèse nous ne nous intéressons qu'au cas nominal de conduite, à savoir, un sol relativement dur et des conditions d'adhérence normales. En effet, nous ne traitons de la conduite ni dans le sable, ni dans la neige, ni dans la boue profonde, ni des passages à gué.

Dans un premier temps, explicitons certains termes :

1. Le **franchissement** représente la capacité d'un véhicule ou d'un robot à franchir un obstacle sans se détériorer ou mettre en danger la vie de ses occupants. Comme le souligne [Jarrault, 2013], dans le cas d'un franchissement de face, il existe deux types d'obstacles dépendant à la fois de l'angle α que forme leur pente avec la direction du mouvement du véhicule et de leur hauteur h. Ces obstacles sont qualifiés de *discontinuités* et classifiés comme *faible* ou *forte*. À cela s'ajoute le cas des obstacles négatifs et des fosses, mais aussi des bosses non définis dans [Jarrault, 2013]. La suite de cette section en apportera une définition.
2. Une **discontinuité** est un changement dans la pente du sol. En outre, une discontinuité positive est une discontinuité dont l'angle α formé entre sa pente et la direction du mouvement est positif et pour une discontinuité négative l'angle α est négatif. Une discontinuité est qualifiée de notable (forte) si α est supérieur à l'angle limite d'adhérence roues / sol f ($\simeq 45°$).
3. Une **fosse** est un obstacle négatif qui est composé d'une suite de deux discontinuités, négative puis positive. Dans la suite de cet exposé, on qualifiera de fosse simple une fosse dont une seule des discontinuités est notable et double quand les deux le sont. De plus, une fosse est définie par les paramètres de ses discontinuités et l'écart entre elles. De sorte que, l'angle α caractérise la pente de sa première discontinuité, négative, avec la direction du mouvement du véhicule et que l'angle β définit la pente de la seconde discontinuité, positive, avec le fond de la fosse ou la première paroi si il n'existe pas de fond. Puis, w représente la largeur de la fosse et h sa profondeur (Figure 2.13a).

2.4.1.2 Les franchissements d'obstacles positifs ou négatifs simples

1. Si $|\alpha|$ est inférieur à l'angle limite d'adhérence roues / sol f ($\simeq 45°$), le franchissement est qualifié de quelconque (Figure 2.12a) et ne pose pas de problème particulier. Si au contraire $|\alpha| \geq f$ alors la discontinuité est qualifiée de notable. En effet, du fait des hypothèses exprimées sur l'adhérence et la nature du sol, le véhicule ne devrait rencontrer des difficultés de franchissement qu'à partir d'une valeur supérieure à 45° [Amar et al., 2009].



(a) Franchissement quelconque, $|\alpha| \leq f$.

(b) Franchissement de faible discontinuité, $|\alpha| \geq f$ et $h \leq R$.

(c) Franchissement de faible discontinuité, $|\alpha| \geq f$ et $h \leq R$. Zoom sur le franchissement.

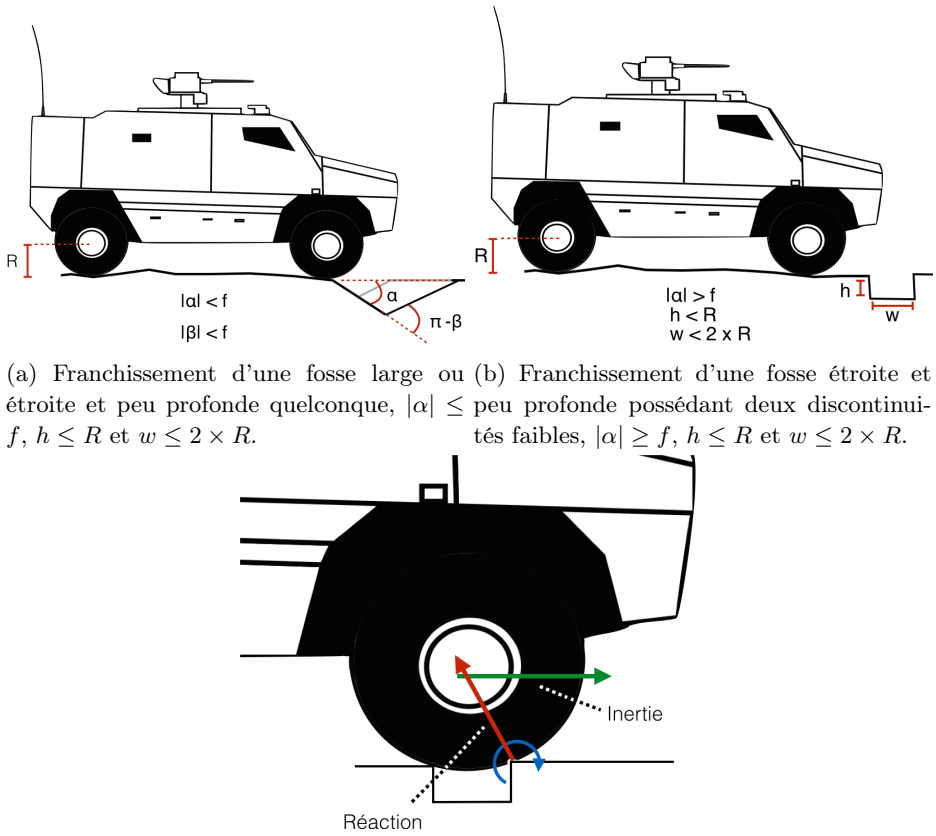
(d) Franchissement de forte discontinuité, $|\alpha| \geq f$ et $h \geq R$. L'obstacle est infranchissable.

FIGURE 2.12 – Les franchissements de discontinuités simples.

2. Un obstacle de faible discontinuité est une discontinuité dont la pente est supérieure à f et la hauteur h est inférieure au rayon des roues R. Cet obstacle est franchissable (Figure 2.12b) [Jarrault, 2013]. En effet, dans le cas des obstacles positifs deux paramètres permettent au véhicule de franchir cet obstacle. Premièrement, l'inertie produite par la vitesse du véhicule engendrera, lors de la collision roues / obstacle, un moment cinétique entraînant les roues sur le sommet de l'obstacle. Deuxièmement, les pneus déformables et les suspensions créeront une poussée verticale à leur détente aidant ainsi le véhicule à monter (Figure 2.12c).
3. Un obstacle de forte discontinuité est quant à lui une discontinuité définie à la fois par $\boldsymbol{|\alpha| \geq f}$ et par $\boldsymbol{h \geq R}$ pour les obstacles positifs et par $h \geq P$ pour les obstacles négatifs (P étant la hauteur du pont qu'on considère égale à R dans la suite du rapport) (Figure 2.12d). **C'est par ces paramètres que nous définissons nos obstacles « positifs » et « négatifs simples »**.

2.4.1.3 Les franchissements d'obstacles négatifs doubles : les fosses et positifs : les bosses

4. Le cas des bosses n'est pas notable car il n'apporte pas de problématique supplémentaire au franchissement consécutif d'un obstacle positif simple puis d'un obstacle négatif simple.
5. Le premier type de fosse est une fosse dont les deux discontinuités sont quelconques, la fosse est alors quelconque, quelles que soient sa largeur et sa profondeur (Figure 2.13a).
6. Deuxièmement, on définit la fosse étroite et peu profonde de discontinuités simples ou doubles, mais faibles (Figure 2.13b). Notons qu'une fosse est qualifiée d'étroite si sa largeur est inférieure au diamètre des roues ($w \leq 2 \times R$) et large dans le cas contraire.



(a) Franchissement d'une fosse large ou étroite et peu profonde quelconque, $|\alpha| \leq f$, $h \leq R$ et $w \leq 2 \times R$.

(b) Franchissement d'une fosse étroite et peu profonde possédant deux discontinuités faibles, $|\alpha| \geq f$, $h \leq R$ et $w \leq 2 \times R$.

(c) Franchissement d'une fosse étroite et peu profonde possédant deux discontinuités faibles, $|\alpha| \geq f$, $h \leq R$ et $w \leq 2 \times R$. Zoom sur le franchissement.

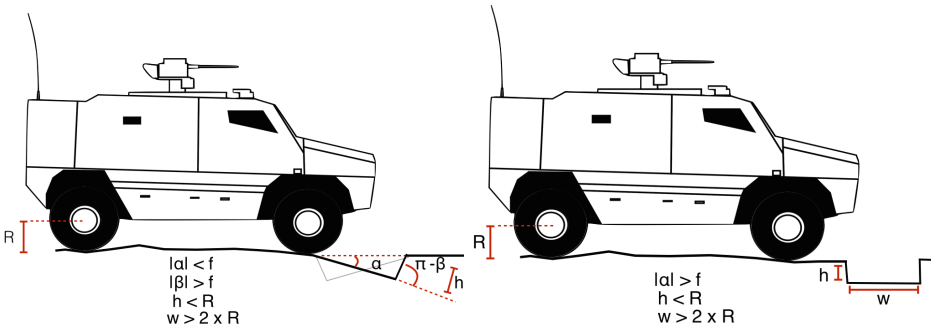
FIGURE 2.13 – Le franchissement d'une fosse étroite et peu profonde.

Dans ce cas de franchissement, l'obstacle est franchissable suivant la même explication que pour le cas 2. avec une difficulté moindre. En effet, dans le cas d'une fosse étroite, les roues n'entrent pas complètement dans la fosse donc la seconde discontinuité est « raccourcie » (Figure 2.13c).

7. Ensuite, on trouve les fosses larges et peu profondes à discontinuités faibles simples (Figure 2.14a) et doubles (Figure 2.14b). Ces cas reprennent exactement l'explication fournie en 2. car ce sont des cas de franchissement de discontinuités faibles (Figure 2.14c).

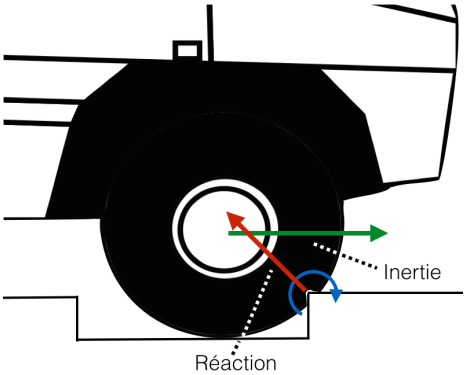
8. Le cas des fosses étroites et profondes à discontinuités fortes simples et doubles (Figure 2.15a) est intéressant, car la profondeur n'a pas d'impact sur le franchissement. Dans la mesure où la fosse est étroite, on retrouve le cas 5. qui reprend exactement l'explication fournie en 2. (Figure 2.15b).

9. Enfin, si la fosse est profonde, large et de fortes discontinuités simples (Figure 2.16a) ou doubles (Figure 2.16b), elle est infranchissable ($\boldsymbol{|\alpha| \geq f}$ et $\boldsymbol{|\beta| \geq f}$ et $\boldsymbol{h \geq R}$ et $\boldsymbol{w \geq 2R}$). **C'est par ces paramètres que nous définissons nos « obstacles négatifs »**.



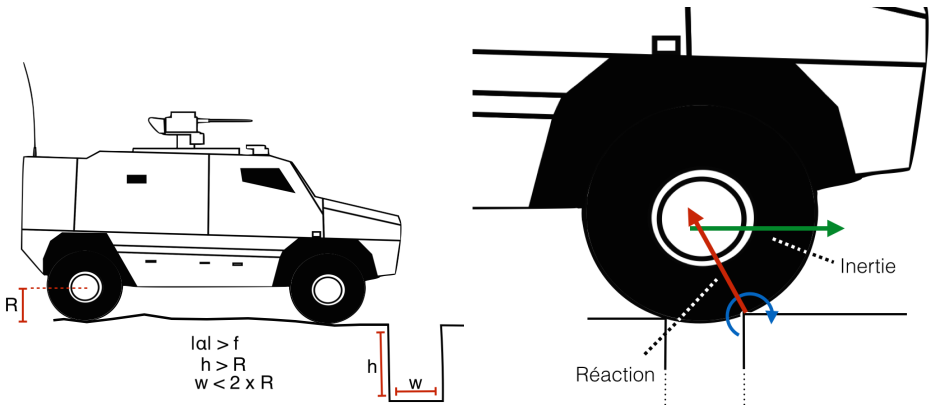
(a) Franchissement d'une fosse large et peu profonde possédant une discontinuité faible, $|\alpha| \geq f$ et $|\beta| \leq f$ ou inversement et $h \leq R$ et $w \geq 2 \times R$.

(b) Franchissement d'une fosse large et peu profonde possédant deux discontinuités faibles, $|\alpha| \geq f$, $|\beta| \geq f$, $h \leq R$ et $w \geq 2 \times R$.



(c) Franchissement d'une fosse large et peu profonde possédant deux discontinuités faibles, $|\alpha| \geq f$, $|\beta| \geq f$, $h \leq R$ et $w \geq 2 \times R$. Zoom sur le franchissement.

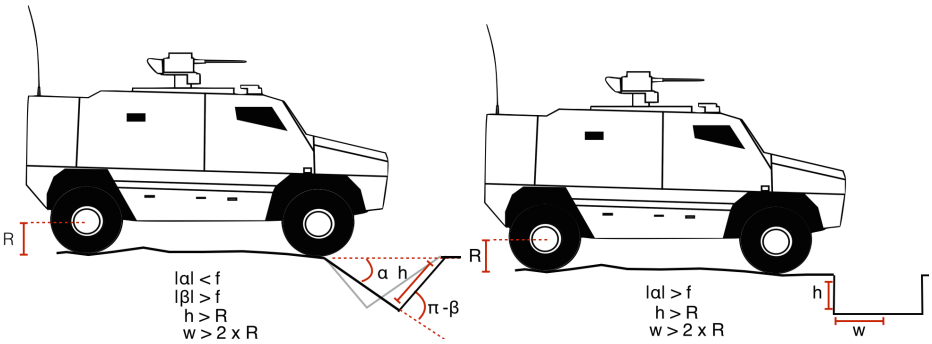
FIGURE 2.14 – Le franchissement d'une fosse large et peu profonde.



(a) Franchissement d'une fosse étroite et profonde possédant deux discontinuités fortes, $|\alpha| \geq f$, $|\beta| \geq f$, $h \geq R$ et $w \leq 2 \times R$.

(b) Franchissement d'une fosse étroite et profonde possédant deux discontinuités fortes, $|\alpha| \geq f$, $|\beta| \geq f$, $h \geq R$ et $w \leq 2 \times R$. Zoom sur le franchissement.

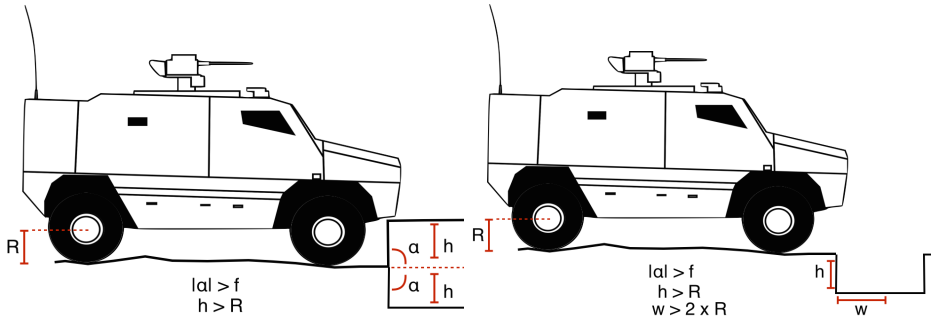
FIGURE 2.15 – Le franchissement d'une fosse étroite et profonde.



(a) Franchissement d'une fosse large et profonde possédant 1 discontinuité forte, $|\alpha| \geq f$ ou $|\beta| \geq f$ ou inversement, $h \geq R$ et $w \geq 2 \times R$. L'obstacle est infranchissable.

(b) Franchissement d'une fosse large et profonde possédant deux discontinuités fortes, $|\alpha| \geq f$ et $|\beta| \geq f$ ou inversement, $h \geq R$ et $w \geq 2 \times R$. L'obstacle est infranchissable.

FIGURE 2.16 – Le franchissement d'une fosse large et profonde.



(a) L'obstacle positif qui se définit par un le franchissement de forte discontinuité, $|\alpha| \geq f$ et $h \geq R$. L'obstacle est infranchissable.

(b) L'obstacle négatif qui se définit par le franchissement d'une fosse large et profonde possédant deux discontinuités fortes, $|\alpha| \geq f$ et $|\beta| \geq f$ ou inversement, $h \geq R$ et $w \geq 2 \times R$. L'obstacle est infranchissable.

FIGURE 2.17 – Les obstacles positifs et négatifs en environnement naturel.

2.4.1.4 Conclusion

Pour conclure cette section, on peut définir les obstacles positifs et négatifs simples comme étant des discontinuités répondant aux paramètres suivants (Figure 2.17a) :

$$\begin{aligned} |\alpha| &\geq f \text{ avec } f \simeq 45^\circ \\ h &\geq R \end{aligned} \tag{2.26}$$

Et les obstacles négatifs doubles (les fosses) comme ceux répondant à (Figure 2.17b) :

$$\begin{aligned} |\alpha| &\geq f \text{ avec } f \simeq 45^\circ \\ \text{et/ou } |\beta| &\geq f \\ h_\alpha &\geq R \\ \text{et/ou } h_\beta &\geq R \\ w &\geq 2 \times R \end{aligned} \tag{2.27}$$

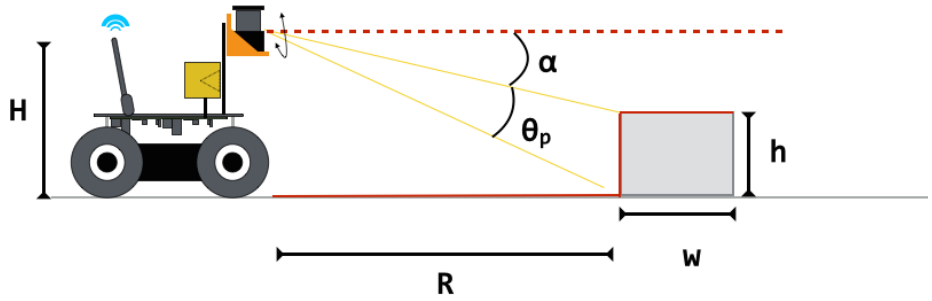
Cette section nous a permis de définir les obstacles dans notre cas d'utilisation. Grâce à ces informations, nous savons ce que nous cherchons à détecter et nous pouvons tenter de le faire. La prochaine section aborde le sujet de la détectabilité des obstacles en environnement naturel ainsi que des manières de les détecter.

2.4.2 Détection d'obstacles

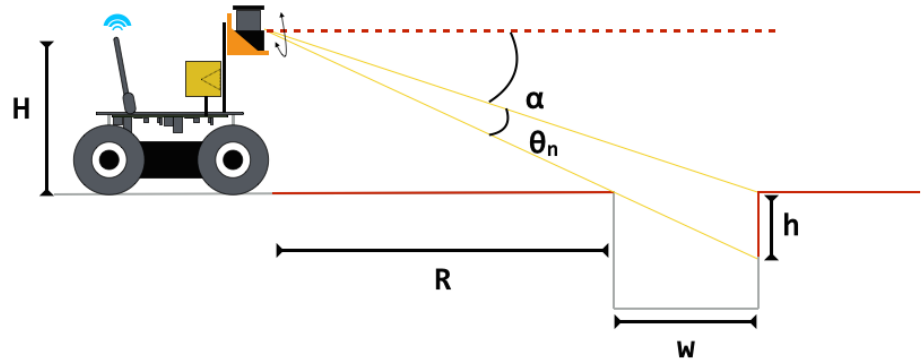
Cette section aborde le sujet de la détection d'obstacles, en définissant tout d'abord les conditions de détectabilité des obstacles. Ensuite nous étudierons plusieurs méthodes de détection d'obstacles et enfin nous nous intéresserons à la problématique de simplification des données acquises pour améliorer les détections.

2.4.2.1 Visibilité d'un obstacle

Les obstacles positifs sont généralement détectés plus facilement par les algorithmes de détection. En raison de leur configuration, ils apparaissent sur des images de profondeur des caméras stéréoscopiques ou de laser de manière plutôt évidente. Au contraire, les obstacles négatifs sont souvent



(a) Géométrie de la détection d'un obstacle positif.



(b) Géométrie de la détection d'un obstacle négatif.

FIGURE 2.18 – La détection d'obstacles avec un robot mobile possédant un capteur réalisant des acquisitions en dessous de la ligne d'horizon. Le capteur est positionné à la hauteur H est perçoit pour la première fois un obstacle à une distance R. W est la largeur de l'obstacle et h sa hauteur détectable. α correspond à l'angle nécessaire au capteur pour détecter le sommet de l'obstacle. θ représente l'angle de visibilité de l'obstacle par le capteur. On cherche θ_p et θ_n qui vont indiquer l'angle de détectabilité respectivement pour un obstacle négatif et positif. En gris, l'environnement et en rouge l'environnement cartographiable par le laser.

cachés par la surface du sol et ils ne se découvrent qu'en arrivant au-dessus d'eux. De plus, ils ne sont pas forcément visibles à l'œil et reste invisibles aux traitements stéréoscopiques [Dubbelman, 2006].

Afin d'illustrer la différence de détectabilité entre les obstacles négatifs et les obstacles positifs, nous allons nous intéresser à leur géométrie. Comme l'illustre [Matthies and Rankin, 2003], une bonne méthode de comparaison est l'étude de l'angle de détectabilité θ que forme l'obstacle avec le capteur.

Pour les obstacles se trouvant à des distances élevées par rapport à la taille du véhicule ou du robot ($R \gg H$), on peut supposer que l'angleθ sont très petits et qu'ainsi $\sin\theta = \tan\theta = \theta$ et $\cos\theta = 1$. Donc, pour un obstacle positif (Figure 2.18a), on peut déterminer l'angle de détectabilité θ_p suivant :

$$\begin{aligned} \alpha &= \frac{H-h}{R} \\ \alpha + \theta_p &= \frac{H}{R} \\ \text{D'où } \theta_p &= \frac{h}{R} \end{aligned} \tag{2.28}$$

Ainsi, θ_p est dépendant de la hauteur de l'obstacle h et de la distance le séparant du véhicule

R (H est la hauteur du capteur). C'est logiquement qu'on constate que l'angle de détectabilité est inversement proportionnelle à la distance. Plus l'objet est loin moins il est détectable.

Pour les obstacles négatifs (Figure 2.18b), le résultat n'est pas identique et θ_p est alors défini par :

$$\begin{aligned} \alpha &= \frac{H}{R+w} \\ \alpha + \theta_n &= \frac{H}{R} \\ \text{D'où } \theta_n &= \frac{Hw}{R^2+Rw} \end{aligned} \tag{2.29}$$

Comme le souligne ces équations, la détectabilité θ_n est désormais dépendante de la hauteur du capteur H, de la distance au véhicule R et de la largeur de la fosse w. Ce qui est remarquable est le dénominateur de ce ratio qui croît désormais de façon quadratique avec la distance qui sépare la fosse du véhicule. Ce $\frac{1}{R^2+Rw}$ explique donc pourquoi les obstacles négatifs sont plus difficilement détectables que les obstacles positifs.

Dans la suite de ce chapitre, nous présentons deux méthodes de détection d'obstacles courantes utilisant la vision, il apparaît que les méthodes de traitements d'images sont limités quand à la détection des obstacles positifs.

2.4.2.2 Le traitement d'images comme moyen de détection

Les méthodes de traitement d'images sont couramment utilisées dans le cadre de la détection d'obstacles. Au travers des deux méthodes présentées nous faisons ressortir leur difficulté à détecter des obstacles négatifs.

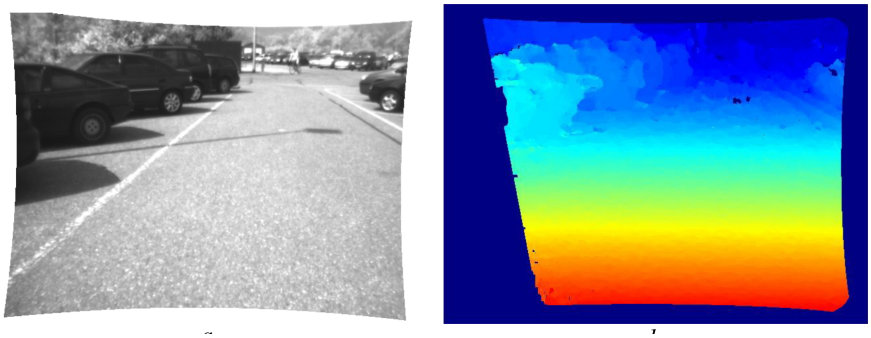
La V-disparité La V-disparité, qui s'intéresse à la géométrie des images pour définir ses primitives, peut servir à détecter des obstacles en utilisant la disparité offerte par les caméras stéréoscopiques [Labayrade, 2003]. Une image de disparité est créée grâce à deux images stéréoscopiques et offre une image colorée représentant les distances pour chaque pixel des images. Dans la figure 2.19a, l'image est encodée sous le format de couleurs HSV et implique que la température de la couleur [8] est inversement proportionnelle à la distance du pixel.

Ensuite, de cette image on peut calculer la V-disparité qui représente chaque pixel en fonction de sa place dans l'image (en ligne) et de sa disparité (Figure 2.19a). Grâce à cette représentation on peut analyser l'image en fonction des distances. De plus, la transformée de Hough nous permet d'extraire le sol et déduire la position des obstacles dans l'image (Figure 2.19b).

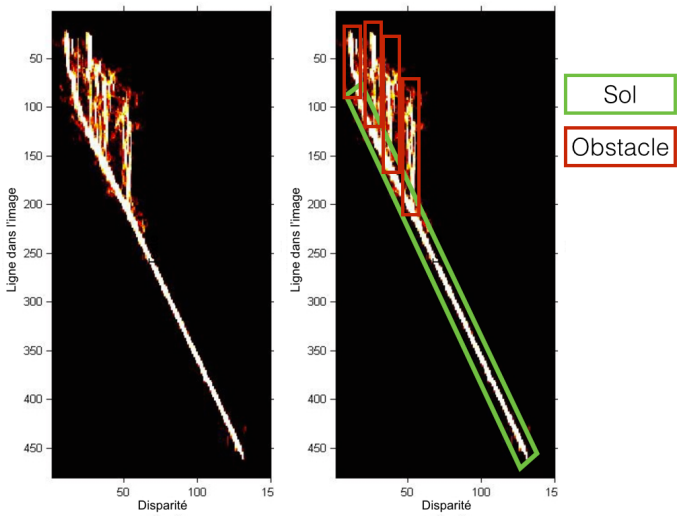
Bien que cette méthode soit rapide (elle nécessite un nombre d'opérations en O(Hauteur $\times$ Largeur)) et efficace pour extraire les obstacles même en milieu naturel [Broggi et al., 2005], non seulement elle fait l'hypothèse d'un sol plan [Labayrade, 2003] mais en plus elle ne considère pas les obstacles négatifs. Cette méthode ne peut donc pas être utilisée seule pour fonctionner en environnement naturel.

Nous continuons ce tour d'horizon avec une autre méthode d'analyse d'images, le traitement par colonne. Cette méthode bien que plus lourde en temps de calcul a l'avantage de pouvoir aussi être utilisée avec des capteurs laser comme nous le verrons.

8. Dans l'encodage HSV une haute valeur de teinte (chaude) tendra vers le rouge alors qu'une faible valeur tendra vers du violet (froid). Le vert est une valeur moyenne.



(a) L'image de disparité associée à l'image.



(b) Le calcul de la V-disparité et son interprétation.

FIGURE 2.19 – Les étapes de l'algorithme de V-disparité. [Dubbelman, 2006]

L'analyse par colonne L'analyse par colonne est une autre approche de traitement d'images stéréoscopiques ayant pour but d'extraire des obstacles positifs en environnement naturel [Manduchi et al., 2015]. Cette méthode se base sur le fait que chaque image est divisée par définition en ligne et en colonne de pixels. Ici les pixels vont être analysés les uns par rapport aux autres au sein d'une même colonne. Cette méthode s'exécute suivant trois étapes. Premièrement, les pixels d'une colonne sont analysés deux à deux afin de ressortir leur différence en profondeur puis dans un second temps la hauteur les séparant va être calculée. Ainsi, les valeurs enregistrées vont pouvoir être comparées aux seuils définissant les obstacles pour extraire les pixels appartenant à des obstacles éventuels.

Cette méthode est intéressante car elle peut être transposée à l'utilisation des nuages de points à l'instar de [Larson and Trivedi, 2011b] travaillant non plus sur les colonnes d'une image, mais sur les lignes d'un nuage de points.

La simplification de ces données pourront alors donner des cartes 3D simplifiées.

Il apparaît avec ces deux méthodes représentatives des algorithmes de détection d'obstacles par traitement d'images que la vision est suffisante pour la détection d'obstacles simples. Cependant, le problème de la détection des obstacles négatifs n'est pas évident. En effet, la sous-section 2.4.2.1 page 47 montre que la visibilité des obstacles négatifs est moins évidente que celle des obstacles positifs. Dans le cas de l'utilisation de la vision, un paramètre supplémentaire augmente la difficulté de détection : la précision des estimations de profondeur.

Contrairement aux capteurs actifs, les capteurs passifs nécessitent d'estimer la profondeur et

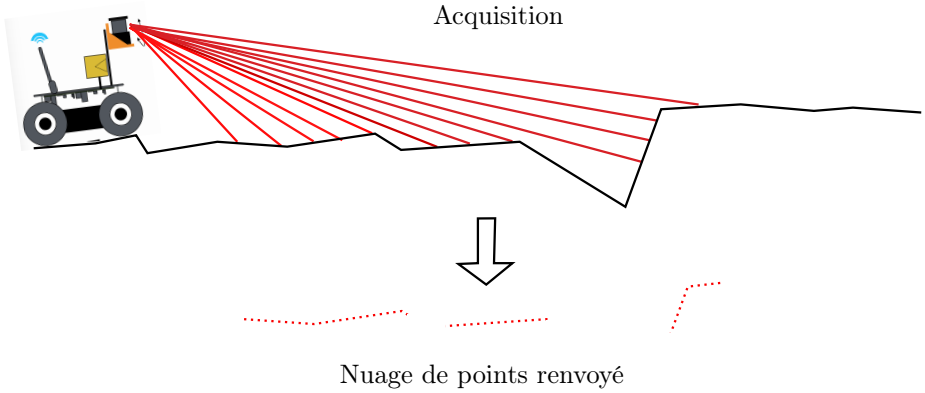


FIGURE 2.20 – Schéma d'acquisition d'une trame laser dans un environnement possédant un trou. Des sauts dans les données apparaissent au niveau des obstacles négatifs.

non de la mesurer. Comme nous l'exposons précédemment plusieurs méthodes existent, mais toutes nécessitent de différencier des pixels les uns par rapport aux autres. Cependant, quand la texture de l'image n'est pas suffisante pour différencier deux profondeurs, les profondeurs sont confondues et l'obstacle est alors invisible [Broggi et al., 2005].

Ainsi, bien que les capteurs de vision puissent, dans une certaine mesure, détecter les obstacles négatifs, les capteurs actifs utilisant la simple comparaison de mesures sont plus à même de réaliser cette fonction.

Détection d'obstacles négatifs Nous avons vu dans la section 2.4.2.1 page 47 que les obstacles négatifs sont plus difficilement identifiables que les obstacles positifs. De plus, les méthodes monoculaires et stéréoscopiques peinent à les détecter de manière robuste [Larson and Trivedi, 2011b]. En effet, ces obstacles ne sont pas visuellement marquants. À longue distance, les fosses peuvent tout simplement ne pas apparaître et le sol peut sembler régulier.

Les méthodes couramment utilisées [Winslow, 2007; Larson et al., 2011; Heckman et al., 2007; Dubbelman et al., 2007] pour la détection des fosses sont des méthodes de détection de « saut » dans un ensemble de données. Comme le montre la figure 2.20, lors de l'acquisition d'un environnement comportant une discontinuité avec des capteurs actifs, comme un laser, un « saut » va apparaître dans les données récupérées. Nous avons montré plus tôt que l'apparition de tels sauts n'est pas triviale (voir section 2.4.2.1 page 47), cependant ils apparaîtront de manière plus évidente qu'une différence de texture dans une image [Broggi et al., 2005]. Certes, d'autres méthodes de traitement d'image telle que la détection de fosse dans une image thermique [Matthies and Rankin, 2003] existent aussi, mais ne rivalisent pas avec les systèmes utilisant des données laser.

Ainsi, pour pouvoir traiter l'intégralité des obstacles présents dans l'environnement, il semble évident que la vision seule ne pourra pas répondre à notre problématique. Les différentes informations établies dans ce chapitre nous montrent qu'un capteur actif semble être indispensable. Malgré tout, les méthodes de traitement d'images, par leur densité d'informations, apportent elles aussi des informations importantes. La preuve est qu'elles peuvent s'avérer performantes dans la détection et le suivi d'objets mobiles.

2.4.2.3 Détection d'obstacles mobiles

Les objets mobiles sont des sources de dangers importantes dans la conduite de véhicules. En effet, en situation réelle, les véhicules doivent veiller continuellement à éviter des piétons, des voitures,

des animaux, etc.

Dans le cadre de la cartographie par méthode SLAM, les méthodes détectant les objets mobiles sont appelées SLAMMOT [Marquez, 2012; Lin and Wang, 2010; Vivet, 2011; Narayana et al., 2010; Wu and Sun, 2010; Chung and Huang, 2008]. Mais comme indiqué dans le chapitre relatif à ces méthodes, le SLAM n'est pas nécessaire à notre application, c'est pourquoi, dans la suite de cette partie nous allons développer deux méthodes compatibles avec de la « simple » cartographie mobile.

Sur un sol dur, les obstacles négatifs ne se déplacent pas, ce qui n'est pas toujours le cas de obstacles positifs. De ce fait, la détection des objets mobiles peut se faire s'en avoir à s'inquiéter du sol. Les méthodes de traitements d'images sont donc viables ainsi que les méthodes utilisant des lasers monocouche.

- L'egomotion permet en plus du calcul d'odométrie, la détection d'objets mobiles. Effectivement, connaissant la vitesse des pixels de l'ensemble de la scène, la vitesse générale de déplacement de la scène peut être calculée par estimation de l'egomotion. Ainsi, dans le cas où un ensemble de pixels formant une forme continue montre un déplacement dans un sens différent du mouvement de l'image ou une vitesse différente de celle de l'image, on peut déterminer que cet ensemble est un objet en mouvement dans la scène. Grâce à cette méthode ou peut segmenter des objets mobiles en temps réel de manière relativement robuste [Lefaudeux, 2013]. Évidement, cette méthode ne s'applique qu'au traitement d'images, dans le cas de l'utilisation de capteurs de distance actifs on utilisera plutôt une grille d'occupation [Homm et al., 2010].
- Les grilles d'occupation sont souvent utilisées avec des lidars. Par leur utilisation [Bobruk and Austin, 2002], on peut détecter les mouvements dans une scène. En effet, cette grille a pour but de garder une trace des points laser repérés dans une certaine zone appelée « cellule ». Si cette case change d'état (de vide à occupée ou l'inverse) alors les points à l'intérieur sont considérés en mouvement. De surcroît, avec cette méthode, on peut réaliser le suivi des objets en mouvement et on peut calculer leur vitesse de déplacement. Ce calcul peut alors permettre d'éviter des collisions s'il est couplé avec un modèle de déplacement des objets mobiles.

Ainsi, il apparaît que la détection des objets mobiles n'est pas problématique en fonction des capteurs utilisés au contraire des obstacles négatifs qui nécessitent l'utilisation d'un capteur actif.

2.5 Conclusion sur la définition de notre système

L'étude précédente nous a permis de découvrir le panel d'aides au pilotage offertes aux pilotes militaires ainsi qu'aux conducteurs civils. Aucune ne répond pleinement à nos besoins. Néanmoins, leur étude nous oriente vers des méthodes plus complexes de perception d'environnement.

En effet, nous avons démontré l'intérêt que possède la cartographie mobile pour notre application. Adapter de telles méthodes permettrait d'obtenir les données nécessaires à des analyses de terrain. De plus, contrairement au SLAM, qui engendre une trop grande complexité algorithmique, la cartographie mobile répondra à nos besoins sans en faire trop. En outre, souhaitant rester indépendant des données GPS, et évitant d'utiliser une centrale inertielle, comme base de notre concept, pouvant être biaisée par les chocs et vibrations (Tableau 2.1 page 54), nous choisissons de réaliser le positionnement de nos données par calcul visuel de déplacement. Ce calcul bien que relatif sera suffisant compte tenu de l'application de courte durée que nous ferons de nos données. Le choix de la méthode de calcul utilisée se porte vers des méthodes de traitement d'images, dont les performances semblent répondre à nos exigences (Tableau 2.1 page 54). De plus, la richesse d'informations contenue dans les images nous permettront de réaliser des traitements complémentaires de sémantique.

Ensuite, dans ce chapitre nous avons défini de manière claire et détaillée les obstacles que nous cherchons à identifier. Cette étude nous permettra de paramétrer au mieux les algorithmes de

détection d'obstacles. Effectivement, les données obtenues grâce à la cartographie mobile devront être analysées afin d'extraire les obstacles positifs et négatifs pour évaluer les dangers présents dans la scène. Il apparaît qu'un capteur d'acquisition actif soit indispensable pour obtenir les informations nécessaires à la détection des obstacles négatifs (Tableau 2.1). Parmis les méthodes évoquées, celle basée sur les détections par « sauts » nous semble être prometteuse.

Grâce aux conclusions établies avec cet état de l'art nous pouvons définir plus en détails les capteurs de visions et télémétriques qui composeront notre système. Le chapitre suivant présente ces capteurs en détails afin d'expliquer le choix des capteurs LIDAR et stéréoscopiques comme base de notre concept de reconstruction.

Capteurs	Contraintes				Besoins			
	Résistance au climat	Résistance aux vibrations	Nécessité d'une calibration	Discrétion	Odométrie en env. naturel	Cartographie 3D en env. naturel	Détection des obstacles positifs	Détection des obstacles négatifs
Centrale inertielle	Oui	Non	Non	Oui	Non (pas seul)	Non	Non	Non
Capteurs télémétriques	Oui	Oui	Non	Non (jusqu'à la limite de portée)	Difficilement, car manque de contexte	Oui	Oui	Oui
Capteurs de vision	Limite de visibilité	Oui	Oui (au montage)	Oui	Oui	Limitée par l'estimation de profondeur	Oui	Très difficilement

Table 2.1 – Les capteurs inertiels face à notre problématique et les contraintes rattachées.

Chapitre 3

Définition des capteurs de notre système

Sommaire

3.1 Capteurs passifs **56**
3.1.1 Caméras monoculaires 56
3.1.2 Caméras stéréoscopiques 58
3.2 Capteurs actifs **63**
3.2.1 Caméra 3D active 63
3.2.2 Télémètres 66
3.3 Conclusion **70**
3.4 Conclusion sur la définition de notre système **74**

Le chapitre précédent nous a permis de définir quelles méthodes seront à même de répondre à notre problématique. Pour réaliser le positionnement relatif, nous nous orientons vers des capteurs passifs de vision. Pour la cartographie 3D de l'environnement, un capteur actif semble indispensable afin de permettre une détection efficace des obstacles négatifs. Dans un premier temps, nous exposons les capteurs passifs au travers des différences de fonctionnement entre les caméras monoculaires et stéréoscopiques et ce qu'elles apportent à notre solution. Ensuite, nous étudierons les capteurs actifs permettant de réaliser l'acquisition exhaustive de notre environnement. Ainsi, nous conclurons sur l'explication du choix des capteurs stéréoscopiques et LIDAR 2D comme capteurs retenus pour notre système.

Ainsi, dans le cadre de la cartographie en deux dimensions et quand un laser est utilisé, la carte sera calculée en deux dimensions et les amers aussi [Espino-Corsino et al., 2011]. Ici, l'hypothèse de monde plan permet de simplifier les calculs. Néanmoins, cette hypothèse peut aussi être utilisée avec un laser 3D ou une caméra et la carte pourra alors toujours être construite en 2D, mais les amers seront alors traités en 3D [Cole and Newman, 2006; Wirbel, 2014].

Le SLAM 3D est en principe une suite directe au SLAM 2D. Cependant, il implique une augmentation de la complexité due entre autres à l'augmentation de la complexité du modèle de mouvement, mais aussi, la complexité des données acquises par les capteurs et celle des modèles de points d'intérêt [Durrant-Whyte and Bailey, 2006b]. Quand la cartographie mobile passe du monde plan au monde tridimensionnel une augmentation de la complexité intervient aussi. Une des formes les plus simples de la cartographie 3D est la cartographie 2D en couches. Cette méthode consiste en une succession sur un troisième axe (généralement z) de scans 2D [Surmann et al., 2001], qu'il est nécessaire d'associer afin d'obtenir une reconstitution 3D de l'environnement capturé [Gidel et al., 2010].

Plus récemment des méthodes utilisant des LIDAR monocouches fixés sur des servomoteurs sont apparues. Appelés « tilting LIDAR » [Larson et al., 2011; Larson and Trivedi, 2011b] (LIDAR à balancier), ces systèmes sont capables de recréer des scènes 3D de manière très précise, et à moindre coût [Zhang and Singh, 2014a].

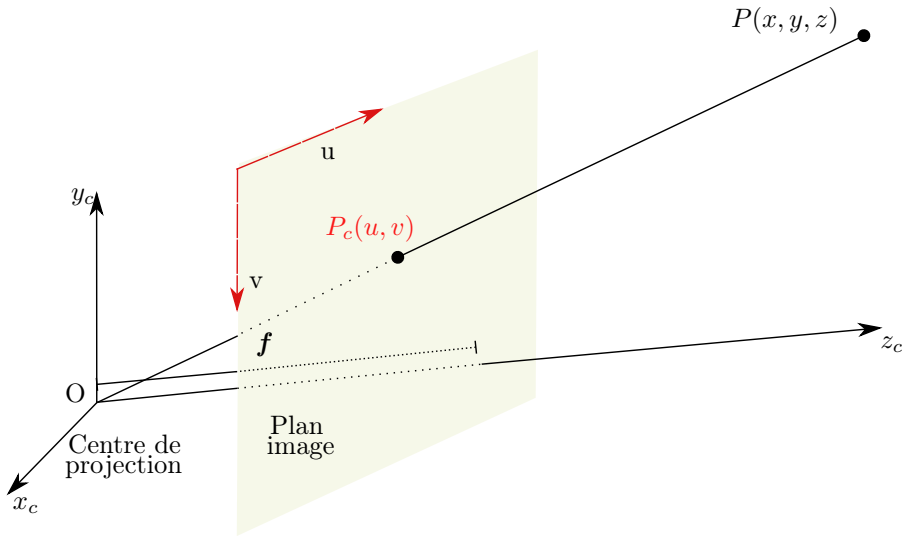


FIGURE 3.1 – Représentation schématique de l'acquisition d'une image d'un objet par une caméra.

À ces méthodes on peut rajouter les méthodes utilisant des LIDAR multicouches [Roß ler et al., 2012], chers, mais très performants et les systèmes de vision utilisés pour réaliser de la 3D. Appelées « Structure From Motion », ces méthodes utilisent des points d'intérêt qu'elles replacent en 3D dans l'environnement [Geiger et al., 2011a,b; Lefaudeux, 2013], cependant cette méthode garde les limitations des systèmes de vision : dépendance à la texture de la scène.

La suite de ce chapitre explique en détail le fonctionnement des capteurs passifs utilisés pour le calcul de la position relative du véhicule et les capteurs actifs utilisés pour la cartographie.

3.1 Capteurs passifs

Les capteurs passifs sont des capteurs pouvant acquérir une scène sans avoir besoin d'agir sur celle-ci. Dans notre système, ils servent à calculer la position relative de notre plateforme par rapport à sa position initiale.

3.1.1 Caméras monoculaires

Une caméra monoculaire fonctionne sur le même principe que son ancêtre, le sténopé [1]. Les caméras sont donc régies par le même modèle optique que celui-ci. Cette section va expliquer le fonctionnement de ce modèle.

Prenons la figure 3.1, dans cette figure un objet se trouvant au point $P(X, Y, Z)$ se retrouve positionné dans l'image à la position $P_c(u, v)$. Pour les obtenir, il faut réaliser plusieurs transformations [Boland, 2002]. Il faut passer du point dans le repère monde $P(X, Y, Z)$ au point dans le repère caméra $P(X_c, Y_c, Z_c)$ pour enfin obtenir le point dans l'image en coordonnées métriques tout d'abord $P_c(x, y)$ par une projection sur le plan Z=1, puis en coordonnées pixeliques $P_c(u, v)$:

$$P(X, Y, Z) \underset{T}{\rightarrow} P(X_c, Y_c, Z_c) \xrightarrow[Projection]{} P_c(x, y) \underset{A}{\rightarrow} P_c(u, v) \tag{3.1}$$

1. Le sténopé est une boite possédant un petit trou par lequel on peut récupérer une image complète.

avec :

$$
\begin{aligned}
T &= \begin{bmatrix} R_{3\times3} & t_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix} \\
A &= KF \\
A &= \begin{bmatrix} k_x & k_x\cos\theta & c_x + c_y\cos\theta & 0 \\ 0 & k_y\sin\theta & c_y\sin\theta & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \\
A &= \begin{bmatrix} f_x & f_x\cos\theta & c_x + c_y\cos\theta & 0 \\ 0 & \frac{f_y}{\sin\theta} & \frac{c_y}{\sin\theta} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}
\end{aligned}
\tag{3.2}
$$

$A = KF$ est alors défini par les cinq « paramètres intrinsèques » de la caméra tels que :

- c_x et c_y sont les coordonnées de l'intersection de l'axe optique (z) avec le plan image. Ils sont idéalement égaux à zéro ;
- f est la distance focale de la caméra ;
- f_x et f_y représentent la focale de la caméra en nombre de pixels suivant les directions x et y ;
- k_x et k_y désignent le nombre de pixels par unité de longueur suivant les directions x et y. Dans le cas de pixels carrés $k_x = k_y$;
- θ est le « skew factor » qui représente la non-orthogonalité éventuelle entre x_c et y_c. Idéalement, x_c et y_c sont orthogonaux et θ est égale à $\frac{\pi}{2}$;

Pour passer de $\begin{bmatrix} X & Y & Z & 1 \end{bmatrix}^T$ à $\begin{bmatrix} X_c & Y_c & Z_c & 1 \end{bmatrix}^T$ on applique la transformation T représentée par les paramètres extrinsèques de la caméra.

$$
\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = T \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}
\tag{3.3}
$$

Ensuite, pour obtenir $\begin{bmatrix} x & y & 1 \end{bmatrix}^T$ à partir de $\begin{bmatrix} X_c & Y_c & Z_c & 1 \end{bmatrix}^T$, il faut appliquer une projection sur le plan $z = 1$. Ainsi les coordonnées métriques dans l'image vont être exprimées en fonction de X_c et Z_c tel que :

$$
\begin{bmatrix} x & = \frac{X_c}{Z_c} \\ y & = \frac{X_c}{Z_c} \\ 1 & = \frac{Z_c}{Z_c} \end{bmatrix}
\tag{3.4}
$$

Et enfin, pour passer des coordonnées métriques aux coordonnées pixelliques on applique la transformation A ainsi :

$$
\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}
\tag{3.5}
$$

Donc, dans le cas où $\theta = \frac{\pi}{2}$ on obtient :

$$
\begin{aligned}
u &= f_x \frac{r_{11}x + r_{12}y + r_{13}z + t_x}{r_{31}x + r_{32}y + r_{33}z + t_z} + c_x \\
v &= f_y \frac{r_{21}x + r_{22}y + r_{23}z + t_y}{r_{31}x + r_{32}y + r_{33}z + t_z} + c_y
\end{aligned}
\tag{3.6}
$$

Ce modèle sténopé est à la base des algorithmes de traitements d'images monoculaires. Avec des images stéréoscopiques, ce modèle peut aussi être utilisé sous une forme arrangée.

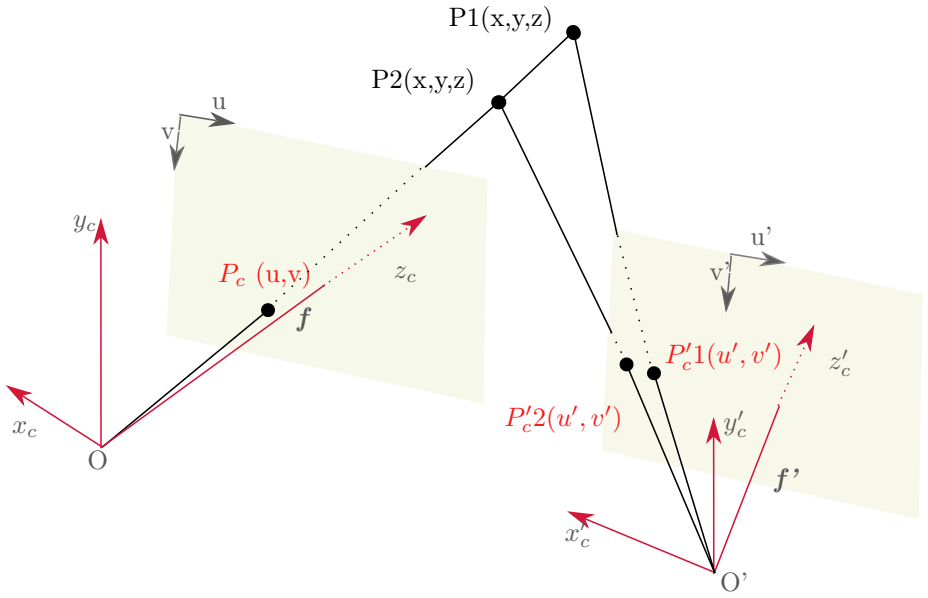


FIGURE 3.2 – Schématisation de l'acquisition de deux objets par une paire de caméras stéréoscopiques.

3.1.2 Caméras stéréoscopiques

Critères de définition La stéréoscopie est un procédé de mise en correspondance de deux images qui a pour but d'extraire les informations de profondeur de la scène. Ils sont définis par les caméras qu'ils utilisent, la baseline [2] qui les sépare et la densité de leur image de profondeur.

Premièrement, la taille de baseline est choisie en fonction de l'application pour laquelle la paire stéréoscopique est utilisée. À large baseline (plus d'un mètre), les deux caméras vont être éloignées l'une par rapport à l'autre. Ce type de système va permettre de récupérer la profondeur sur des distances caméras-objet relativement lointaines. En effet, comme l'explique [Gallup et al., 2008], pour une résolution fixe, la précision maximale est atteinte pour une baseline telle que :

$$b = \frac{z_{loin}^2}{f \times \varepsilon_z} \tag{3.7}$$

où ε_z est l'erreur en profondeur, z_{loin} la distance calculable la plus lointaine, b la baseline et f la distance focale des caméras en pixels. En outre, f(en pixels) se calcule de la manière suivante :

$$f(\text{en px}) = w_{\text{cam}}(\text{en px}) \times \frac{f(\text{en mm})}{w_{\text{CCD}}(\text{en mm})} \tag{3.8}$$

Où w_{cam} est la largeur de la caméra et w_{CCD} est la largeur du capteur CCD de la caméra.

Donc plus la baseline est large plus la distance idéale d'appariement est grande (là où l'erreur d'appariement est la plus faible). Malgré tout, ces méthodes sont limitées en distance d'acquisition et comme l'indique [Chang and Chatterjee, 1992], l'erreur de détermination de la profondeur est définie par :

$$\varepsilon_z = -\frac{z^2}{b \times f} \times \varepsilon_d \tag{3.9}$$

2. La baseline est la droite reliant les centres optiques des deux caméras.

où ε_d est l'erreur en disparité et z la distance. Donc plus on s'éloigne moins l'estimation sera précise.

Par contre, un système à large baseline ne pourra pas extraire des profondeurs d'objets proches. En effet, la distance où le chevauchement des champs de vision des caméras commence est tel que [Gallup et al., 2008] :

$$z_{proche} = \frac{b}{\tan(\frac{\theta_{fov}}{2})} \tag{3.10}$$

Où θ_{fov} est le champ de vision des caméras.

Ainsi, ce type de méthode à large baseline peut être utilisé pour des applications où il n'est pas nécessaire de voir ce qu'il se passe proche de la caméra par exemple la navigation automobile autonome [Broggi et al., 2005]).

Au contraire à courte baseline, les deux caméras vont être relativement proches l'une par rapport à l'autre et on va récupérer les profondeurs des objets proches, voire très proches. Ces méthodes suivent les mêmes équations que celles expliquées précédemment.

Enfin, la densité d'un système stéréoscopique correspond au nombre de primitives utilisées pour réaliser l'appariement des images.

La stéréo « dense » a pour but de fournir la profondeur de tous les pixels (ou presque) de la scène avec une grande précision dans des scènes complexes. L'inconvénient de ce type de méthode est le temps de calcul qu'elle requiert pour réaliser l'appariement des pixels des deux images [Lazaros et al., 2007]. En effet, les volumes de données importants impliquent une complexité algorithmique importante elle aussi (à hauteur de $O(Hauteur \times Largeur)$) [Olivieira, 2005], soit plus de 900 000 calculs à chaque itération dans le cas d'une paire stéréo de résolution 1280×960.

Dans le cas des méthodes éparses, on ne va appareiller que des points d'intérêt comme des coins, des lignes ou de forts contrastes (voir chapitre 2.2.2.2 page 29) [Witt and Weltin, 2012]. L'avantage de cette méthode se traduit par les volumes de données beaucoup moins importants (entre 100 et 4000 points [Lefaudeux, 2013]). Par contre, les calculs des primitives plus lourds que le seul appariement pixel à pixel, la perte en précision quant à l'appariement de la profondeur aux pixels, et l'estimation potentiellement moins précise dans les scènes complexes rendent le choix de la méthode d'appariement dépendante de l'application finale.

Calculs géométriques Les propriétés géométriques des caméras stéréoscopiques permettent de gagner en informations par rapport aux caméras monoculaires. En effet, comme l'indique la figure 3.2, avec une seule caméra l'information de distance est perdue entre $P1$ et $P2$ qui sont alors confondus. Grâce à une deuxième caméra décalée par rapport à la première, on peut trouver cette information de profondeur qui les différencie. Cette information est obtenue par triangulation des points, et n'est possible qu'avec une référence commune aux deux caméras c'est pourquoi le positionnement relatif des caméras l'une par rapport à l'autre est nécessaire.

Voyons maintenant les calculs géométriques qui définissent la stéréoscopie. Comme l'explique [Orteu, 2008], et en prenant la figure 3.3 comme illustration, on peut définir les relations suivantes.

Soit $P(x, y, z)$ un point 3D dans le repère monde, $P_c(x, y, z)$ et $P'_c(x, y, z)$ le même point respectivement dans le repère caméra gauche et droite et $P_c(u, v)$ et $P'_c(u', v')$ ce point positionné dans les images gauche et droite respectivement, alors :

$$\begin{aligned} P(x, y, z) &= TP_c(x, y, z) \\ P(x, y, z) &= T'P'_c(x, y, z) \\ P_c(x, y, z) &= T_s P'_c(x, y, z) \end{aligned} \tag{3.11}$$

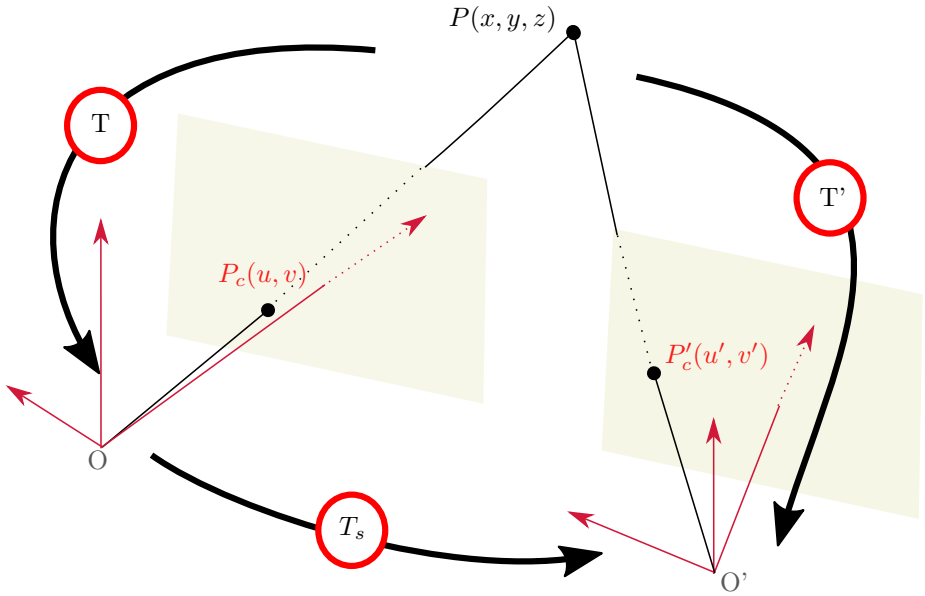


FIGURE 3.3 – Les transformations liant les caméras d'une paire stéréoscopique et un objet réel $P(x, y, z)$.

Il apparaît que les trois transformations ne sont pas indépendantes, on peut donc établir des relations entre elles comme suit :

$$\begin{aligned} T &= T_s^{-1} T' \\ T' &= T_s T \\ T_s &= T' T^{-1} \end{aligned} \tag{3.12}$$

En considérant que :

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ et } T' = \begin{bmatrix} R' & t' \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r'_{11} & r'_{12} & r'_{13} & t'_x \\ r'_{21} & r'_{22} & r'_{23} & t'_y \\ r'_{31} & r'_{32} & r'_{33} & t'_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.13}$$

On peut trianguler [Hartley and Sturm, 1997] le point, c'est à dire, écrire le système de quatre équations suivant :

$$\begin{cases} u = f_x \dfrac{r_{11}x + r_{12}y + r_{13}z + t_x}{r_{31}x + r_{32}y + r_{33}z + t_z} + c_x \\ v = f_y \dfrac{r_{21}x + r_{22}y + r_{23}z + t_y}{r_{31}x + r_{32}y + r_{33}z + t_z} + c_y \end{cases} \text{ et } \begin{cases} u' = f'_x \dfrac{r'_{11}x + r'_{12}y + r'_{13}z + t'_x}{r'_{31}x + r'_{32}y + r'_{33}z + t'_z} + c'_x \\ v' = f'_y \dfrac{r'_{21}x + r'_{22}y + r'_{23}z + t'_y}{r'_{31}x + r'_{32}y + r'_{33}z + t'_z} + c'_y \end{cases} \tag{3.14}$$

Ainsi, si les caméras sont calibrées, les variables $(R, R', t, t', f_x, f_y, f'_x, f'_y, c_x, c_y, c'_x, c'_y)$ sont connues et les trois coordonnées de notre point $P(x, y, z)$ peuvent alors être déterminées par l'application d'un moindre carré sur ce système à 3 inconnues et 4 équations. Ainsi, la profondeur du point par rapport au centre de la paire stéréoscopique et la disparité [3] peuvent être calculées. A l'inverse, ces équations permettent aussi de calibrer les caméras entre elles quand la distance au point est connue.

3. La disparité correspond à la distance séparant les points dans l'images de gauche par rapport aux même points dans l'image de droite. [Devernay et al., 2011]

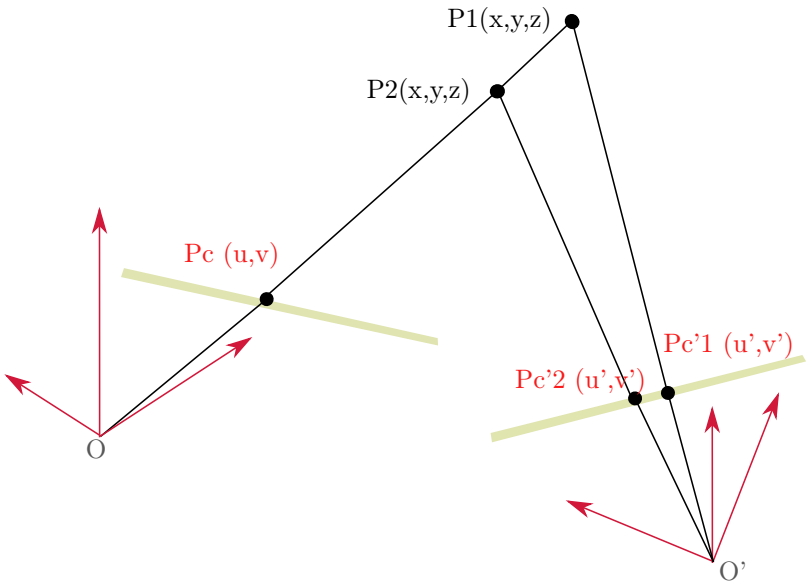


FIGURE 3.4 – Contraintes d'appariement stéréoscopique : La contrainte d'unicité. Une primitive d'une image ne peut correspondre qu'à un seul appariement dans l'autre image.

Appariement algorithmique Étudions maintenant le principe de fonctionnement d'un algorithme d'appariement d'images stéréoscopiques. Comme l'explique [Kramm, 2011], un traitement stéréoscopique se déroule en trois étapes. :

1. Extraction de primitives : Les primitives, ou points d'intérêt peuvent être extraits de manière éparse ou dense dans les images, elles peuvent être représentées par des lignes, des coins, de forts contrastes ou dans le cas des méthodes denses représenter tous les pixels de l'image (voir chapitre 2.2.2.2 page 29).
2. L'appariement entre les deux images.
3. Calcul de la valeur de profondeur pour chaque paire de primitives par reconstruction 3D ou calcul de la disparité.

Selon [Labayrade et al., 2002] l'appariement doit répondre à plusieurs contraintes :

- La contrainte d'opacité : les objets observés ne doivent pas posséder de surfaces semi-transparentes, c'est-à-dire qu'à chaque $P_c(u, v)$ doit correspondre un seul point $P(x, y, z)$.
- La contrainte de cohérence photométrique régie par l'hypothèse lambertienne [Grimson, 1981] qui implique (qu'idéalement) les objets doivent renvoyer la même intensité lumineuse dans toutes les directions. Donc, l'intensité d'un objet sur l'image de gauche $I_g(u, v)$ doit être la même que sur l'image de droite $I_d(u', v')$. $I_g(u, v) = I_d(u', v')$.
- La contrainte d'unicité [Marr and Poggio, 1979] : une primitive d'une image ne peut correspondre qu'à un seul appariement dans l'autre image. (Figure 3.4).
- La contrainte d'ordre : si un point $P_c1(u, v)$ est à gauche d'un point $P_c2(u, v)$ dans l'image gauche, alors la même chose doit se retrouver dans l'image de droite [Baker and Binford, 1981] (Figure 3.5).
- La contrainte épipolaire qui ramène l'appariement 2D à un appariement 1D dans chaque image. En effet, un point dans une image n'a de correspondant dans l'autre image que suivant la droite épipolaire qui lui correspond. Cette relation n'est pas réversible, une droite dans une image ne correspond pas à un point dans une autre. Simplement, cela signifie qu'un point réel

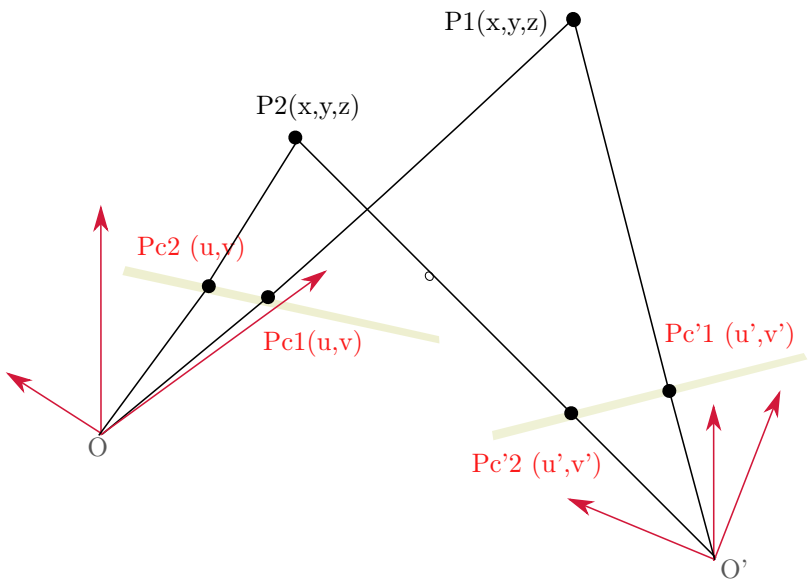


FIGURE 3.5 – Contraintes d'appariement stéréoscopique : La contrainte d'ordre. Si un point $P_c1(u, v)$ est à gauche d'un point $P_c2(u, v)$ dans l'image gauche, alors la même chose doit se retrouver dans l'image de droite.

ne trouvera correspondance que sur la droite issue de l'intersection du plan image avec le plan formé par la baseline [4] et le point réel. (Figure 3.6).

Estimation de la profondeur L'appariement va permettre d'obtenir des points d'intérêts à comparer pour réaliser l'estimation de la profondeur de la scène. En effet, grâce aux calculs géométriques expliqués au paragraphe 3.1.2, on peut calculer la position 3D d'un point en fonction de ses positions connues dans les images gauche et droite, $P_c(u, v)$ et $P'_c(u', v')$. Pour ce faire, on va chercher à calculer le point d'intersection des deux droites $(O, P_c(u, v))$ et $(O', P'_c(u', v'))$.

Le problème est qu'en raison des approximations des mesures, les droites ne se rencontreront certainement pas [Geiger et al., 2011b]. La plupart du temps, la solution consiste alors à considérer que le point 3D se trouve au centre du segment reliant les deux droites à l'endroit ou celles-ci sont les plus proches [Geiger et al., 2011a]. Ainsi on peut déduire une estimation de la profondeur et de la disparité.

La profondeur va être indiquée par une « carte de profondeur » indiquant la valeur de profondeur de chaque pixel suivant une échelle de couleur (HSV) ou d'intensité (nuances de gris) (Figure 3.7 page 64).

Comme nous avons pu le constater dans cette section, malgré la simplicité des données acquises, un grand nombre d'informations peuvent en être extraites. Ces capteurs ont l'avantage d'être discrets et rapides. Malheureusement, ils sont dépendants de la visibilité dans la scène et ne sont pas capables de détecter de subtiles variations de profondeur. Cependant la densité de leurs données leur permet d'extraire le contexte de la scène et d'ainsi offrir de riches informations pour réaliser des traitements d'images.

Au contraire, les capteurs actifs (ou du moins la part active des capteurs actifs) ne peuvent que difficilement extraire le contexte d'une scène. Par contre, ils sont très performants quand il s'agit de télémétrer un objet, comme nous allons le voir dans la section suivante.

4. La baseline est la droite reliant les centres optiques des deux caméras.

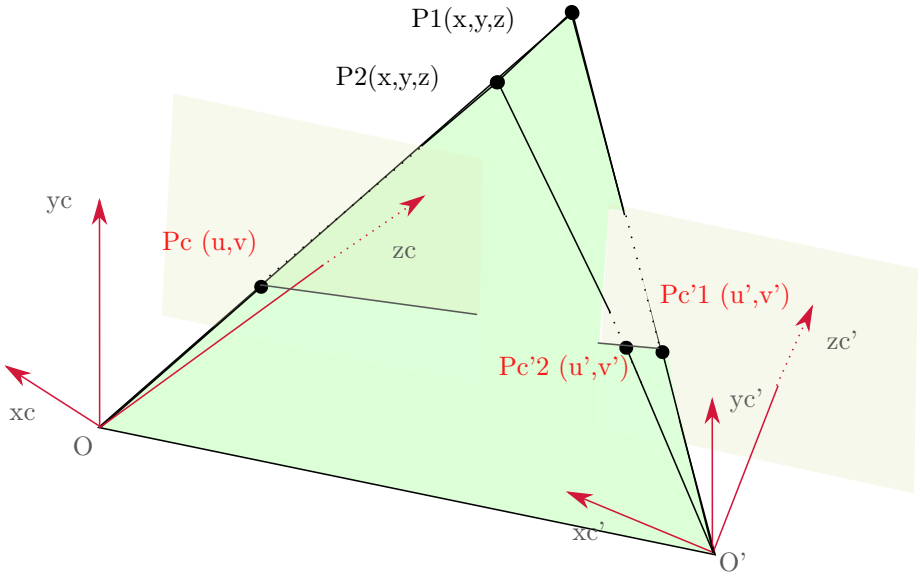


FIGURE 3.6 – Contraintes d'appariement stéréoscopique : La contrainte épipolaire. Un point réel ne trouvera correspondance que sur la droite issue de l'intersection du plan image avec le plan formé par la baseline et le point réel.

3.2 Capteurs actifs

Contrairement aux capteurs passifs, les capteurs actifs agissent sur une scène pour en acquérir des données. Dans notre système, ce type de capteurs est utilisé pour l'acquisition des données permettant de réaliser la cartographie. Cette section va présenter les différents capteurs actifs présents sur le marché permettant de réaliser de la cartographie mobile et le choix réalisé. Dans un premier temps nous étudierons les différents types de caméras actives puis nous verrons les télémètres.

3.2.1 Caméra 3D active

Contrairement aux caméras 3D passives comme les caméras stéréoscopiques, les caméras actives n'utilisent qu'une caméra, mais possèdent un système de projection leur permettant de déceler la profondeur de la scène qu'elles filment.

Il y a deux approches employées actuellement pour les caméras 3D actives. La première appelée temps de vol ou « Time Of Flight (TOF) », utilise de la lumière modulée et incohérente et se base sur les calculs de phase qui peuvent être implémentés dans les capteurs des types CMOS ou CCD [Xu et al., 1998; Oggier et al., 2006]. La deuxième approche utilise le principe de la lumière structurée qui se base sur l'utilisation d'un motif de lumière qui se déforme au contact des objets permettant ainsi de déceler les objets par l'étude de la déformation.

« Temps de vol (TOF) » Le principe de cette technologie [Lange, 2000] est l'utilisation de la corrélation des signaux optiques incidents venant d'une illumination proche infrarouge réfléchis par la scène afin de récupérer une carte de profondeur utilisable pour des post-traitements. Les deux principales méthodes utilisées pour calculer la distance sont : La modulation d'intensité et l'obturateur optique.

– Modulation d'intensité : Dans les méthodes utilisant la modulation d'intensité, la distance aux

FIGURE 3.7 – Image de profondeurs, aussi appelée carte de profondeurs, provenant de [Pujades et al., 2012]

objets est calculée suivant :

$$d = \frac{c}{4\pi\omega}\phi$$

Avec ω la modulation de fréquence, ϕ la compensation de phase et c la célérité.

– Obturateur optique : Le concept de base des caméras à obturateur optique est d'utiliser des impulsions lumineuses proches infrarouges qui représentent une profondeur donnée. Le signal optique est réfléchi par la scène avec une certaine intensité donnant une indication de la profondeur de l'objet [Piatti, 2010; Kilpela, 2004] (Figure 3.8). Un obturateur positionné devant le capteur CCD de la caméra s'ouvre pendant une fenêtre temporelle $\tau = [t_{min}, t_{max}]$ relative aux distances minimales et maximales désirées $[d_{min}, d_{max}]$. L'intensité des pixels nommés I_{obt} représente l'intensité de lumière réfléchie pendant τ et indique de manière linéaire la distance des objets de la scène.

Comme expliqué dans [Kolb et al., 2009], l'utilisation de caméras temps de vol est rendue difficile par plusieurs limitations :

– **Ils sont faiblement résolus**, entre 64×64 (PMD) et 512×424(Kinect 2) pixels.
– **Une distance systématiquement erronée**, aussi appelée « wiggling », due au fait que pour fonctionner parfaitement, le signal doit être parfaitement sinusoïdal, ce qui n'est pas atteignable en situation réelle.
– De plus, une **erreur de distance liée à l'intensité** s'ajoute à l'erreur de signal, les calculs de distance sont soumis à l'éclairage général de la scène, ce qui provoque des biais. Ce défaut limite considérablement leur utilisation en extérieur, et encore plus dans des zones géographiques très ensoleillées.
– **La mauvaise détection des bords** des objets ayant une profondeur non homogène.
– **Les artefacts de mouvement** dus à l'acquisition séquentielle des phases de l'image.

FIGURE 3.8 – Les « murs de profondeurs » liés aux impulsions lumineuses avec une *Kinect 2*. http ://www.bryancook.net

FIGURE 3.9 – Exemple d'utilisation de lumière 3D.[Camera, 2015]

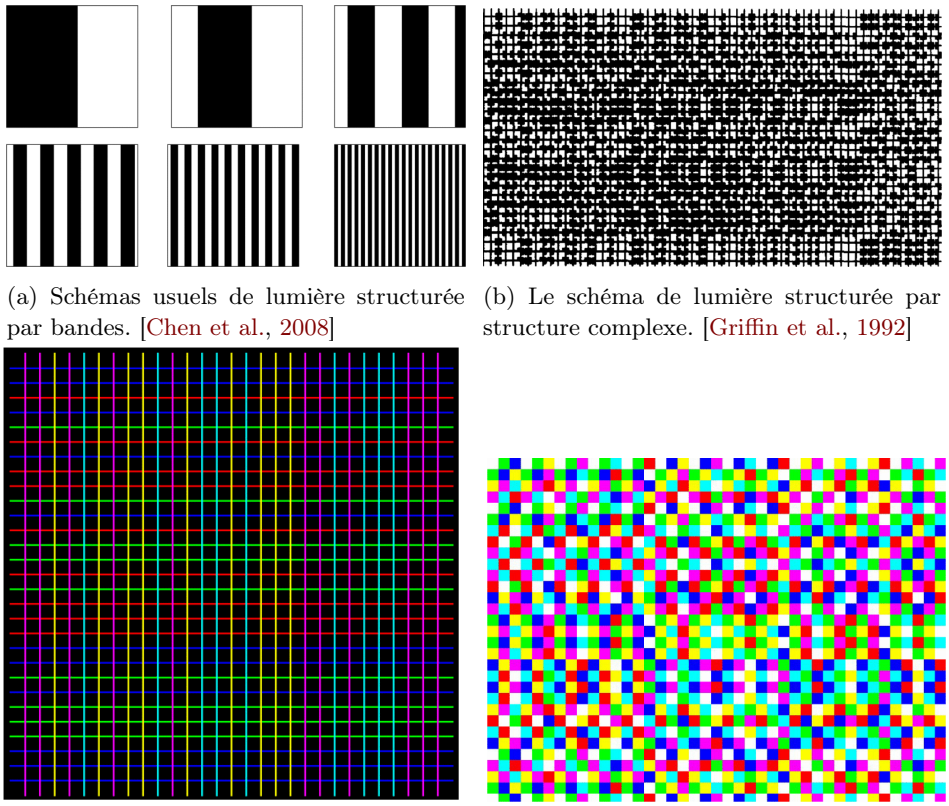
– **Les réflexions multiples** liées aux lumières extérieures qui peuvent rentrer en conflit avec les réflexions des lumières de la caméra.

En plus de ces défauts, des interférences sont possibles entre plusieurs caméras 3D [Piatti, 2010]. Le dernier problème est lié à la nature des objets qui peuvent fausser les distances en ayant une réflexion plus ou moins forte que la moyenne, ou en réfléchissant la lumière dans d'autres directions [Foix and Aleny, 2011].

Lumière structurée Comme les technologies temps de vol, les technologies utilisant la lumière structurée utilisent une projection de lumière. Dans le cas de la lumière structurée, la projection va se faire par bandes étroites. En effet, une bande de lumière projetée sur un objet apparaît déformée lorsqu'on la visualise avec un point de vue différent de celui du projecteur [Scharstein and Szeliski, 2003] (Figure 3.9). L'étude de cette déformation permet alors de déterminer les contours des formes en présence et ainsi une carte de disparité entre les différents objets en présence peut être créée.

Le schéma de lumière le plus simple est celui utilisant des bandes parallèles (Figure 3.10a). Cependant, la lumière n'est pas forcément projetée en bandes [Salvi et al., 2004] (Figure 3.10), les formes peuvent être :

- des grilles [Chen et al., 2008]
- des grilles infrarouges (Primesense Kinect)
- des grilles de couleurs [Salvi et al., 2004; Chen et al., 2008] (Figure 3.10b et Figure 3.10c)
- des formes complexes [Griffin et al., 1992] (Figure 3.10d)
- des variations d'intensité lumineuse [Caspi et al., 1998]



(a) Schémas usuels de lumière structurée par bandes. [Chen et al., 2008]

(b) Le schéma de lumière structurée par structure complexe. [Griffin et al., 1992]

(c) Le schéma de lumière structurée par grille de couleurs. [Salvi et al., 2004]

(d) Le schéma de lumière structurée par grille de couleurs. [Chen et al., 2008]

FIGURE 3.10 – Les différents types de lumières structurées. [Chen et al., 2008]

Comme les caméras fonctionnant sur le principe du temps de vol, celles utilisant la lumière structurée ne fonctionnent pas ou peu en extérieur. Ainsi, il apparaît que les caméras 3D actives ne sont pas encore assez matures pour être utilisées dans des problématiques d'environnement naturel.

Contrairement à ces systèmes, les télémètres sont régulièrement utilisés en extérieur. La prochaine section présente les exemples du RADAR et du SONAR déjà utilisés dans le milieu automobile et le LIDAR qui est la solution choisie.

3.2.2 Télémètres

Contrairement aux caméras 3D actives, les télémètres sont souvent utilisés en extérieur. Les cadres d'application peuvent être : la construction, la sécurité routière, la défense, ou même les domaines de l'agroalimentaire. Dans cette section nous allons présenter le RADAR, le SONAR et le LIDAR.

RADAR Le RADAR (Radio Detection And Ranging) est un type de télémètre utilisant les radios fréquences. Sa création date des années 30 et comme beaucoup de télémètres, il est composé d'un

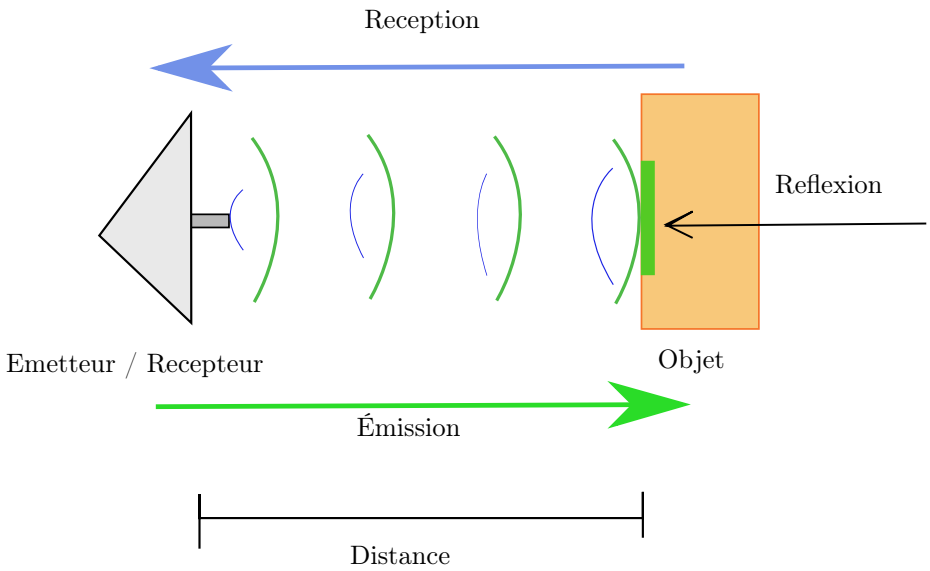


FIGURE 3.11 – Système émetteur / récepteur de télémétrie. L'exemple du RADAR ou du SONAR.

émetteur et d'un récepteur, l'émetteur envoyant un signal (onde électromagnétique radioélectrique) et le récepteur attendant de recevoir le même signal après réflexion sur un objet (Figure 3.11). Il existe deux technologies, les RADAR à ondes pulsées et les RADAR à émission continue. Le RADAR à ondes pulsées va, avec la même antenne, émettre un signal et attendre son retour. En émission continue, l'antenne va émettre continuellement et une deuxième antenne va être nécessaire pour recevoir les réflexions. Néanmoins, dans les deux cas, la distance est trouvée en mesurant le temps que l'onde met pour revenir compte tenu de sa vitesse de départ.

En outre, les systèmes RADAR fonctionnent dans la gamme des ondes électromagnétiques radioélectriques, ou ondes radio, proches des micro-ondes entre $3Mhz$ et $110Ghz$. La fréquence ainsi que la longueur d'onde sont définies vis-à-vis de l'application pour laquelle le RADAR est prévu. En effet, une haute fréquence va permettre une forte résolution et une grande longueur d'onde va servir à obtenir une portée maximale élevée. Cependant, il est difficile d'obtenir les deux, à moins d'avoir un budget conséquent et un espace de stockage équivalent [5]. Par exemple, certains RADAR de recul présents sur les véhicules civils (Proxel [Proxel, 2015]), émettent sur la « bande W » et ont une fréquence entre 75 et $110Ghz$ et une longueur d'onde de $2,7$ à $4mm$, ce qui leur accorde une très haute fréquence pour des distances courtes, au contraire des RADAR de détection des missiles émettant sur la « bande UHF » $300/1000Mhz$ $0,3-1m$, qui leur donne moins de détections, mais sur des distances beaucoup plus longues, les menaces étant entre 600 et $3000km$.

Des méthodes de reconstruction utilisant des Radars existent [Natour et al., 2015; Vivet et al., 2013]. Cependant, le défaut majeur des RADAR est qu'ils sont facilement brouillables, car ils sont sensibles aux interférences. Dans notre cas d'application, un grand nombre d'interférences interviennent du, entre autres, aux brouilleurs de signaux installés sur les véhicules.

Le SONAR fonctionne sur le même principe avec des ondes différentes, nous allons en expliquer le concept.

SONAR Comme le Radar, un Sonar actif émet et réceptionne des ondes, mais ici, ces ondes sont des ondes sonores. L'onde sonore est émise grâce à une antenne et son écho est réceptionné avec cette même antenne ou par une antenne déportée.

5. http ://www.smdc.army.mil/KWAJ/RangeInst/ALTAIR.html

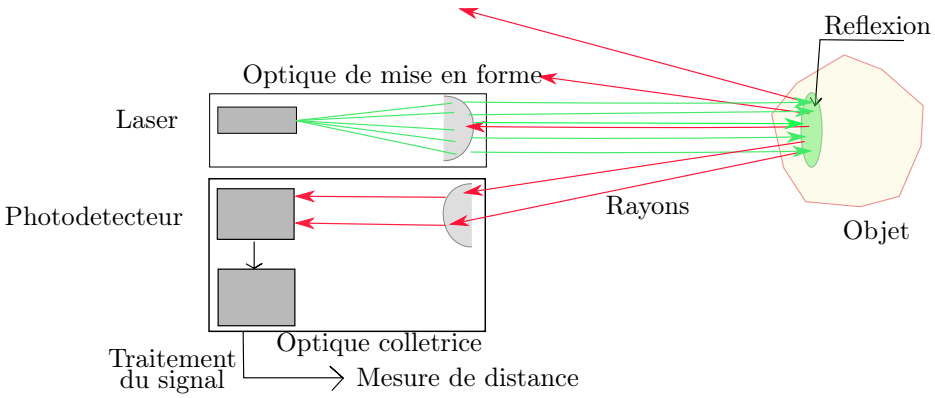


FIGURE 3.12 – Schématisation de fonctionnement d'un LIDAR.

L'antenne peut être orientée soit omnidirectionnellement, dans ce cas le calcul du déphasage entre les échos permet de connaître l'orientation de départ, soit directionnellement.

Les SONAR passifs sont quant à eux uniquement utilisés pour l'écoute et la détection d'objets faisant du bruit, ce qui dans notre cas ne nous intéresse pas.

Les SONAR possèdent plusieurs défauts, mais le défaut majeur est leur faible résolution directionnelle causée par l'interaction des réflexions. Il est alors très compliqué d'interpréter des données SONAR pour réaliser une reconstruction [Siciliano and Khatib, 2008].

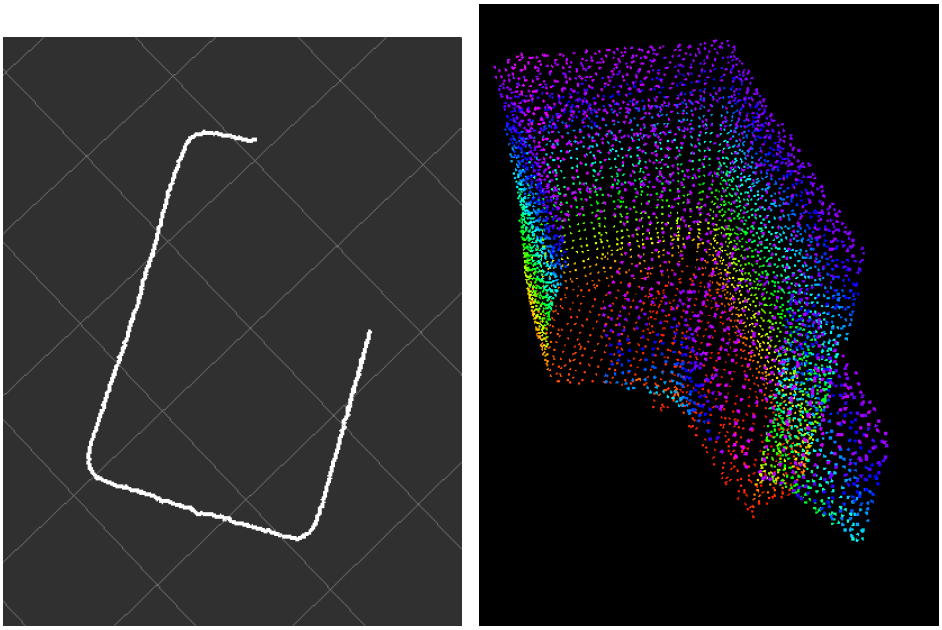
Intéressons nous aux ondes lumineuses aussi utilisées pour réaliser de la télémétrie, mais dans notre cadre d'application, de manière plus performante que leurs homologues.

LIDAR Les lasers, quand ils sont utilisés pour faire de la télémétrie, sont généralement des LIDAR [Jakubowski, 2008] pour « LIght Detection And Ranging ». Ces LIDAR fonctionnent en sondant une cible avec de la lumière de façon similaire à un SONAR ou un RADAR.

Premièrement, le LIDAR génère une courte impulsion de lumière à une longueur d'onde spécifique dans une direction donnée. Pour des raisons de coût et de sécurité, la plupart du temps les longueurs d'ondes utilisées sont dans des régions proches de l'infrarouge voire du visible. En effet, des LIDAR couramment utilisés tels que l'*Hokuyo UTM-30LX* ou le *Velodyne HDL-64e* utilisent une longueur d'onde de 905nm ($0,905\mu m$, l'infrarouge commençant à $1\mu m$). L'impulsion lumineuse voyage alors de manière rectiligne jusqu'à une cible potentielle et interagit avec elle, la lumière pouvant être réfléchie ou absorbée par le matériau. Ensuite, la lumière réfléchie dans la direction du récepteur est capturée par l'optique collectrice qui transmet le signal au reste du système pour traitement. Le temps entre l'émission de l'impulsion et la réception de la réflexion est mesuré et converti en distance suivant la vitesse de la lumière (Figure 3.12).

La plupart des LIDAR utilisés dans le contexte de robotique sont des LIDAR à balayage qui envoient des impulsions en direction d'un miroir tournant. Ce miroir permet alors d'envoyer les impulsions tout autour du LIDAR (Figure 3.14). Un LIDAR à balayage réalise ces télémétries un grand nombre de fois (6000 fois par seconde pour l'*Hokuyo UTM-30LX*). On peut trouver deux types de LIDAR, les lidars 2D aussi appelés monocouche ou monofibre et les LIDAR 3D appelés multicouches ou multifibres.

Un LIDAR 2D va acquérir des données uniquement sur un plan perpendiculaire à l'axe de rotation de son miroir (Figure 3.14). Concernant les LIDAR multicouches, ils vont acquérir des points de la même manière, sauf que le nombre d'émetteurs et de récepteurs est augmenté et que leur angle d'émission varie des uns aux autres. Par exemple, on peut citer le *Velodyne HDL-32e* qui va émettre 32 impulsions à chaque télémétrie (22000 par seconde) ou le *Velodyne HDL-64e* qui va en émettre 64. Cependant, bien que la résolution des LIDAR multicouches soit plus élevée (700 000



(a) Scan transmis par un LIDAR 2D. (b) Nuage de points 3D.

FIGURE 3.13 – Nuages de points 2D et 3D.

points par seconde pour le *Velodyne HDL-32e* contre 6000 pour l'*Hokuyo UTM-30LX* (2D)) leur prix est aussi beaucoup plus élevé (32k€ pour le *Velodyne HDL-32e* contre 1500€ pour l'*Hokuyo UTM-30LX*).

De plus, le prix des LIDAR est de moins en moins élevé et leur taille est de plus en plus réduite [Holmstrom et al., 2014] ce qui les rend extrêmement attractifs dans toutes les recherches ayant pour objet la cartographie ou la détection d'obstacles. Cependant, leur prix reste le point bloquant de beaucoup de véhicules autonomes pour pouvoir être commercialisé auprès du grand public [RAYNAL, 2015].

Enfin, pour qualifier la qualité des données Lidar, deux mesures sont utilisées : la précision et la répétabilité [Grussenmeyer et al., 2013]. La précision représente le degré de rapprochement avec une valeur issue d'une « vérité terrain ». La répétabilité correspond à la régularité avec laquelle les mesures sont renvoyées (Figure 3.15).

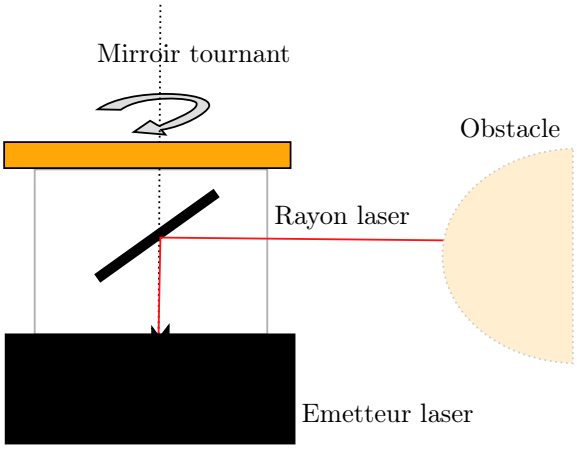


FIGURE 3.14 – LIDAR mono couche à miroir tournant (balaiement).

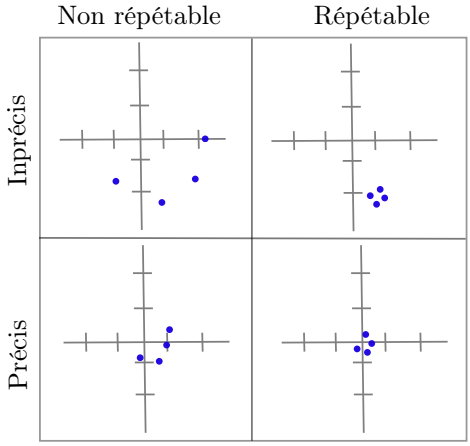


FIGURE 3.15 – Précision et répétabilité.

Ainsi, les LIDAR ont les avantages d'être petits et d'avoir des prix acceptables (pour les LIDAR 2D), de fonctionner en extérieur de manière relativement discrète et de fournir un très grand nombre de données fidèles. **Ces avantages nous ont conduits à choisir cette solution comme moyen d'acquisition du terrain.**

3.3 Conclusion

Suite aux études présentées dans les chapitres précédents, nous avons choisi d'utiliser deux capteurs dans notre système : une caméra stéréoscopique pour réaliser le calcul du déplacement relatif de la plateforme et un LIDAR nous permettant de récupérer les aspérités du terrain de manière plus efficace que ne l'aurait fait un capteur passif.

Premièrement, comme expliqué dans le chapitre 2.5 page 52, nous ne souhaitons pas utiliser de GPS afin de réaliser un système indépendant. De plus, nous avons aussi expliqué qu'une centrale inertielle ne convient pas à notre application de par sa sensibilité aux vibrations qui provoque une dérive importante.

Ensuite, les capteurs passifs de vision (Tableau 3.1 page 72) ont l'avantage d'être peu chers et offrent de bons résultats en calcul de déplacement relatif. De plus, ces capteurs sont peu encombrants et fonctionnent à une haute fréquence d'acquisition. En outre, le faible coût de ces capteurs permet d'imaginer l'évolution du système pour le rendre utilisable de nuit en redondant le système de capteurs passifs infrarouges fonctionnant de nuit.

Concernant les capteurs actifs (Tableau 3.2 page 73), les caméras actives ne conviennent pas à notre application. En effet, la plupart de ces systèmes ne fonctionnent pas ou très mal en extérieur par temps ensoleillé. Bien que cette solution soit très performante en environnement intérieur grâce à la combinaison de la vision et des informations de profondeur, elle n'est pas envisageable dans notre cas. De plus, le SONAR, de par sa très faible discrétion, n'était pas une solution envisageable. Ensuite, comme le SONAR, le RADAR est peu discret en plus d'être facilement brouillable par les systèmes utilisés par le véhicule d'application. Nous n'avons donc pas choisi cette technologie.

Aussi, le LIDAR multicouche convenait parfaitement. La grande quantité de données acquises aurait été une source d'information très riche. Malheureusement, le prix de cette technologie était trop élevé pour notre étude, c'est pourquoi nous n'avons pas utilisé de LIDAR multicouche.

Enfin, le LIDAR monocouche est une technologie intéressante. En effet, son indépendance à la luminosité et sa possibilité de récupérer les informations de distance de manière absolue en font une technologie très performante pour la détection d'obstacles positifs et négatifs. De plus, son prix est

abordable, les portées conviennent à notre application, mais son fonctionnement sur un mode plan limite grandement son utilisation. Néanmoins, des solutions existent pour faire fonctionner le laser en 3D. [Larson et al., 2011] montent leur LIDAR 2D sur un servomoteur et récupère une information 3D par la rotation du LIDAR. [Bosse et al., 2012] quant à eux, installes le LIDAR sur un ressort et récupère les mouvements grâce à une centrale inertielle.

Ainsi, les analyses précédentes (Tableau 3.2 page 73 et tableau 3.1 page suivante) nous ont permis de définir notre système. Il se compose d'une caméra stéréoscopique, d'un LIDAR 2D et d'un servomoteur. Avec ce système nous réalisons de la cartographie en environnement naturel par acquisition 3D et calcul de déplacement relatif par odométrie visuelle. Nous n'avons pas utilisé un véhicule représentatif de notre application pour plusieurs raisons que nous développerons dans le chapitre suivant. Nous développerons aussi l'installation réalisée sur la base roulante et la mise au point du système. Enfin, le système d'acquisition de données 3D avec le LIDAR 2D sera expliqué en détail.

Capteur	Prix	Taille	Fragilité	Sensibilité à la luminosité	Discrétion	Portée	Fréquence	Angle de vue
Caméra monoculaire	Faible ($\leq$ 1000€)	Petit	Peut être durcie	Inefficace dans le noir	Oui (capteur passif)	Jusqu'à $200m$ (Dépend de la résolution et de la focale)	Jusqu'à $100Hz$ (couramment $30Hz$)	Dépend de l'optique
Caméra monoculaire proche infrarouge	Moyen ($\leq$ 2000€)	Petit	Peut être durcie	Inefficace si la luminosité est trop forte	Oui (capteur passif)	Jusqu'à $200m$ (Dépend de la résolution et de la focale)	Jusqu'à $100Hz$ (couramment $30Hz$)	Dépend de l'optique
Caméra stéréoscopiques	Moyen ($\leq$ 2000€)	Moyen	Peut être durcie	Inefficace dans le noir	Oui (capteur passif)	Jusqu'à $100m$ (dépend de la résolution, de l'écartement et de la focale. Portée soit proche soit lointaine)	Jusqu'à $100Hz$ (couramment $30Hz$)	Dépend de l'optique
Caméra stéréoscopiques proche infrarouge	Moyen ($\leq$ 2000€)	Moyen	Peut être durcie	Inefficace si la luminosité est trop forte	Oui (capteur passif)	Jusqu'à $100m$ (dépend de la résolution, de l'écartement et de la focale. Portée soit proche soit lointaine)	Jusqu'à $100Hz$ (couramment $30Hz$)	Dépend de l'optique

Table 3.1 – Récapitulatif des conclusions fonctionnelles sur les capteurs passifs

Capteur	Prix	Taille	Fragilité	Sensibilité à la luminosité	Discrétion	Portée	Fréquence	Angle de vue
Caméra 3D active	Faible ($\leq$ 1000€)	Moyen	Peut être durcie	Inefficace si la luminosité est trop forte, **ne fonctionne pas en extérieur**	Visible grâce à une caméra infrarouge (très longue portée)	Jusqu'à $10m$	Jusqu'à $30Hz$	Dépend de l'optique
LIDAR monocouche	Moyen ($\leq$ 2000€)	Petit	Sensible au fortes vibrations et aux chocs, peut être durcie	Aucune	Capteur actif, Détectable souvent jusqu'au double de leur portée. Jusqu'à $60m$ pour l'*Hokuyo UTM-30LX*	Jusqu'à $100m$. (*Hokuyo UTM-30LX* : $30m$)	Jusqu'à $40Hz$	jusqu'à $360° \times 0°$
LIDAR multicouches	Très cher ($\geq$ 30000€)	Moyen (*Velodyne HDL 32e* : $25cm \times 15cm$)	Très sensible au fortes vibrations et aux chocs, peut être durcie	Aucune	Capteur actif, Détectable souvent jusqu'au double de leur portée. Jusqu'à $60m$ pour l'*Hokuyo UTM-30LX*	Jusqu'à $200m$ (*Velodyne HDL 32e* : $120cm$)	Dépend des modèles. Jusqu'à $15Hz$	$360° \times 27°$ (pour le *Velodyne RX-64*)
SONAR	Faible ($\geq$ 1000€)	Dépend de la portée	peut être durcie	Aucune	Capteur actif, Bruyant, détectable à très longue portée	Jusqu'à plusieurs kilomètres	Dépend de la portée	Dépend de la portée. Ouverture conique. Jusqu'à 60°
RADAR	Faible ($\geq$ 1000€)	Dépend de la portée	peut être durcie	Aucune	Capteur actif, onde électromagnétique détectable à très longue portée et brouillable facilement	Jusqu'à plusieurs kilomètres	Jusqu'à plusieurs kilomètres	Dépend de la portée. Ouverture conique. Jusqu'à 60°

TABLE 3.2 – Récapitulatif des conclusions fonctionnelles sur les capteurs actifs

3.4 Conclusion sur la définition de notre système

Cette partie présente l'état de l'art des méthodes et des capteurs en lien avec notre problématique. Dans ce cadre, l'analyse des aides au pilotage existantes met en exergue la nécessité de réaliser notre propre système étant donné qu'aucune aide au pilotage ne répond pleinement à notre problématique. Ensuite, l'étude des méthodes et des capteurs possibles pour percevoir au mieux l'environnement nous conduit à un concept mélangeant plusieurs principes existants.

Afin de percevoir de manière exhaustive l'environnement, nous réalisons l'acquisition de celui-ci par des capteurs stéréoscopiques et LIDAR. Les données acquises par le LIDAR servent à construire une cartographie mobile sous la forme de nuages de points 3D positionnés dans l'espace de manière relative par l'estimation du déplacement au travers des images stéréoscopiques. Ce concept répond théoriquement à notre problématique en créant une reconstruction numérique de l'environnement où les dangers sont extractibles.

La partie suivante présente la validation de ce concept par le développement d'une solution algorithmique et son évaluation grâce à une plateforme d'expérimentations.

Deuxième partie

Système de perception d'un environnement naturel

Introduction à la partie II La première partie de ce manuscrit a exposé les raisons du choix des méthodes et des capteurs utilisés dans notre système. Le chapitre 2 - *État de l'art* - explique la décision d'utiliser la cartographie mobile plutôt que le SLAM et le choix du calcul de déplacement relatif par rapport au calcul de déplacement absolu, mais aussi l'intérêt des capteurs actifs pour la création d'une cartographie dans laquelle les obstacles négatifs seront théoriquement détectables. Ensuite, le chapitre 3 - *Définition des capteurs de notre système* - a montré que la caméra stéréoscopique et le LIDAR 2D pouvaient satisfaire nos contraintes et fournir une reconstruction satisfaisante. Ainsi, cette première partie définit précisément notre système de perception et de reconstruction de l'environnement sur le plan théorique et matériel.

Cette seconde partie présente les développements algorithmiques réalisés pour reconstruire l'environnement avec les données transmises par ces capteurs et la création d'une plateforme d'expérimentation permettant la validation du concept.

Le chapitre 4 - *Une plateforme mobile d'expérimentations* - présente la plateforme d'expérimentations. Cette plateforme est un robot mobile sur laquelle sont installés les capteurs choisis suite à la partie précédente. Elle nous a permis de réaliser une évaluation des algorithmes que nous avons mis en place pour répondre à notre problématique.

Le chapitre 5 - *Reconstruction par données LIDAR et odométrie visuelle* - présente les développements qui font suite à l'absence dans l'état de l'art de solutions répondant pleinement à notre application. Ainsi, nous abordons d'abord la reconstruction 3D basée sur l'acquisition de données du LIDAR 2D tournant et le déplacement relatif estimé par odométrie visuelle. Nous exposons son principe de fonctionnement général puis nous détaillons le fonctionnement de *Libviso2*[6], l'algorithme d'odométrie visuelle utilisé. Puis, les métriques permettant de quantifier la réponse de notre méthode et de nos algorithmes aux contraintes posées en introduction de ce manuscrit sont présentées. Ensuite, nous comparons les trajectoires et les cartographies estimées ainsi que la charge matérielle avec des données de références et de « vérité terrain » publiques et d'autres acquises dans notre environnement de test. Finalement, une discussion sur ces résultats expliquera la nécessité d'optimiser cette méthode par l'optimisation réalisée par l'ajout d'une estimation de la rotation antérieurement à l'odométrie stéréoscopique. Ce second algorithme possédant un certain nombre d'hypothèses sur le mouvement permet d'obtenir de manière rapide une estimation du mouvement de rotation réalisé par la caméra. Cette estimation offre une initialisation suffisante à l'odométrie stéréoscopique pour améliorer sa vitesse d'exécution et son nombre d'itérations. Finalement, cette optimisation améliore la précision de l'estimation du mouvement réalisé. Les résultats présentés dans cette section quantifient ces améliorations.

Enfin, le chapitre 6 - *Segmentation des données acquises* - expose les travaux d'analyse de la cartographie ainsi créée par nos algorithmes. Ces traitements visent à extraire de cette cartographie des pistes pouvant mener à la détection de dangers pour le véhicule. Les travaux réalisés dans ce cadre sont des travaux prospectifs offrant des éléments pour la réalisation d'une aide au pilotage. Nous présentons une méthode de segmentation visuelle de chemin dont nous avons amélioré le temps d'exécution jusqu'à un facteur 32 puis nous étudions les développements préliminaires réalisés dans le cadre de la segmentation d'obstacles d'un nuage de points 3D.

6. http ://www.cvlibs.net/software/libviso/

Chapitre 4

Une plateforme mobile d'expérimentations

Sommaire

4.1 Création de la plateforme **80**
4.1.1 Le choix de la plateforme mobile 80
4.1.2 Le système de communication ROS 81
4.1.3 Le contrôle de plateforme 82
4.2 Les capteurs **84**
4.2.1 Caméras 84
4.2.2 Laser 3D à balancier 84
4.2.3 Calibration 88
4.3 Conclusion **91**

La partie précédente a présenté notre concept de perception d'environnement. Une caméra stéréoscopique permet le positionnement relatif des données acquises par le LIDAR 2D.

Ce chapitre présente la mise en place d'une plateforme d'expérimentation permettant la validation de ce concept. De plus, nous expliquerons ici la particularité de notre système d'acquisition 3D basé sur un LIDAR 2D tournant sur un servomoteur.

Nous expliquerons le choix de la plateforme Wifibot, sa création et sa mise au point. Ensuite, nous nous intéresserons à la mise au point du système d'acquisition 3D LIDAR créé par la mise en correspondance de données d'un servomoteur et d'un LIDAR 2D.

La plateforme robotique utilisée pour nos expériences n'est pas équivalente à la plateforme d'acquisition et pour cause, c'est un petit robot de quelques kilos seulement. Dans un premier temps le choix de cette plateforme va être expliqué. Ensuite, nous allons aborder le sujet de l'installation des capteurs sur cette plateforme et de sa mise au point. Finalement, nous expliquerons en détail la manière dont nous obtenons des informations 3D avec le LIDAR 2D choisi.

Pourquoi une petite plateforme robotique mobile ? L'aide au pilotage finale a pour objectif d'être portée sur des véhicules militaires blindés, mais dans le cadre de la thèse et pour la mise au point des algorithmes composant la reconstruction, de nombreuses contraintes nous ont convaincus de réaliser nos développements sur une plateforme robotique plutôt que sur un véhicule.

1. Les véhicules militaires sont difficiles d'accès. Pour réaliser une manipulation, ou une acquisition, il faut faire appel à du personnel militaire de la DGA, seuls autorisés à conduire de tels véhicules.
2. Le véhicule définitif n'ayant pas été défini, il n'était pas impératif de réaliser des montages sur une plateforme militaire.
3. Une plateforme robotique offre l'avantage d'être déjà électronisée.

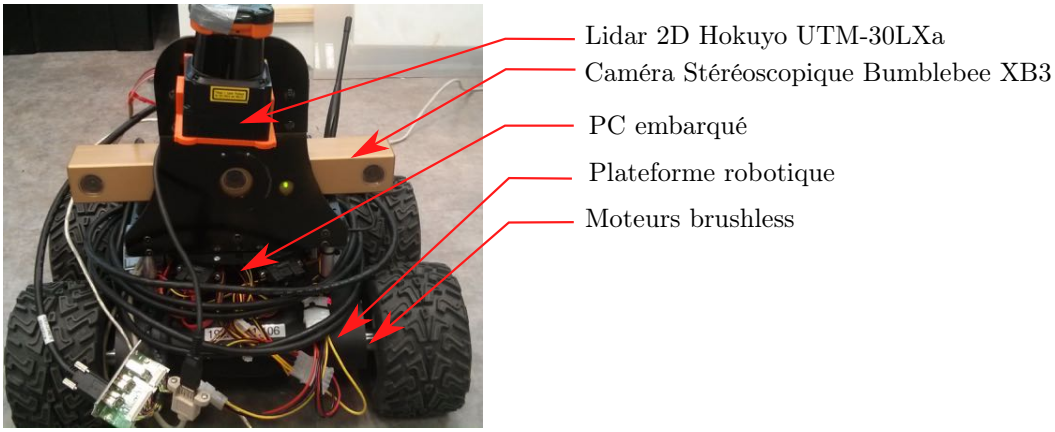


FIGURE 4.1 – Notre plateforme robotique vue de face. La plateforme est un *Wifibot Lab V4* sur lequel sont montés une caméra stéréoscopique et un LIDAR 2D lui-même monté sur un servomoteur.

4. La modularité d'une plateforme robotique nous permet d'installer les capteurs choisis comme nous le souhaitons pour la mise au point des algorithmes.
5. Le prix d'une plateforme est sans conteste plus attractif que le prix d'un véhicule de grande taille.

Ainsi, nous avons choisi de réaliser notre système d'acquisition sur une plateforme robotique mobile terrestre. La section suivante explique pourquoi nous avons choisi d'utiliser le *Wifibot Lab V4*, l'installation des capteurs, le système de contrôle et celui de communication.

4.1 Création de la plateforme

Notre système de cartographie est une plateforme robotique utilisant un LIDAR monté sur un servomoteur pour réaliser l'acquisition 3D et une caméra stéréoscopique pour réaliser le positionnement relatif de la plateforme. Détaillons le choix du modèle *Wifibot Lab V4*.

4.1.1 Le choix de la plateforme mobile

Bien que la plateforme ne soit pas un véhicule équivalent à l'application finale, nous souhaitions nous en rapprocher le plus possible, c'est pourquoi la plateforme choisie devait répondre à plusieurs critères :

1. La plateforme devait être « tout terrain », c'est-à-dire qu'elle devait être capable de rouler en extérieur et avoir un couple suffisant pour rouler sur des sols non plats et escalader de petits obstacles / pentes.
2. Elle devait être modulaire pour permettre l'installation des capteurs et avoir une puissance motrice assez forte pour pouvoir se mouvoir toute équipée. De plus, elle devait être assez stable pour ne pas risquer d'endommager les capteurs.
3. Il était nécessaire qu'elle soit équipée d'un PC embarqué assez puissant pour pouvoir réaliser l'acquisition de données, les transférer vers une autre machine à distance et réaliser une partie des traitements de données.
4. Le dernier critère était que sa taille devait permettre un stockage et un transport aisés.

Après une comparaison des modèles disponibles sur le marché, des plus grands, comme le Robucar [Robosoft, 2013] aux plus petits équivalents à [DFRRobot, 2013], nous nous sommes orientés

vers le *Wifibot Lab V4 + LIDAR* [Wifibot, 2013]. Premièrement, Wifibot appartient désormais à Nexter Robotics, une filiale de Nexter Systems, par conséquent, le prix des plateformes est bon marché pour Nexter Systems. Par ailleurs, comme la suite de cette section le montre, cette plateforme répond à tous les critères sus-cités et est fournie avec un LIDAR *Hokuyo UTM-30LX*.

Notre plateforme Le *Wifibot Lab V4* (Fig. 4.1) est un robot mobile terrestre à 4 roues motrices. Il n'est pas équipé de roues directionnelles, aussi, le contrôle de trajectoire est réalisé par un différentiel de vitesses appliqué aux roues étant donné que les vitesses des roues gauches et droites sont réglables indépendamment. De plus, les roues sont équipées de pneus qui donnent à cette plateforme une bonne adhérence en extérieur et de moteurs *brushless* d'un couple de $4N.m$ largement suffisant pour tirer la plateforme et son matériel.

Par ailleurs, la base roulante est surmontée d'un socle sous lequel est installé le PC embarqué. Il possède un processeur *core i5* cadencé à $1,9Ghz$, $4G$ de RAM et un SSD qui permettent de réaliser l'ensemble des traitements envisagés sans problème. En effet, la RAM et le SSD permettent de traiter l'acquisition de données et de les stocker sans limitation de débit et le processeur permet de réaliser les traitements primaires (Transformation des scans 2D en 3D et appariement des images stéréoscopiques). De plus, la liaison entre le chassis et le PC embarqué se fait par liaison série. Cette liaison permet de contrôler les roues et de récupérer les informations électriques de la plateforme.

Ainsi, cette configuration permet de monter sur le socle la caméra stéréoscopique et le LIDAR nécessaires à notre système.

Enfin, la communication entre le PC embarqué, les capteurs et le PC distant se fait par le biais du middleware *ROS* que nous allons présenter.

4.1.2 Le système de communication ROS

Le *Wifibot Lab V4* étant une plateforme robotique « Open-Source » possédant un PC embarqué et plusieurs capteurs, nous devions mettre en place un système de communication entre ces différents éléments. En effet, un des points cruciaux d'un système robotique faisant intervenir plusieurs capteurs est le système de communication et c'est pourquoi nous avons décidé de faire appel à un système de communication performant, ROS (*Robot Operating System*) [Quigley et al., 2009]. ROS est un middleware très utilisé en robotique, sa simplicité et son efficacité l'ont rendu célèbre auprès de la communauté robotique mondiale [Institute, 2015].

Ce « middleware » [1] développé en collaboration par « Willow Garage » [2] et le laboratoire « STAIR » de l'université de STANFORD [3] qui permet de faire communiquer les éléments d'un système robotique (que ce soient des capteurs, des actionneurs ou simplement des PC). En effet, fonctionnant sur la technologie « Peer to Peer » [4] il est le lien entre le matériel (bas niveau) et les logiciels (haut niveau). Pour cela, son fonctionnement est basé sur un annuaire central où sont stockées et mises à jour les adresses vers tous les éléments communiquant sur le système.

Parallèlement, ROS posséde une communauté offrant un ensemble d'outils et de bibliothèques de programmation permettant de faire fonctionner des capteurs et des algorithmes entre eux. Notons qu'à la base il offrait seulement un système de messagerie entre les applications et un horodatage centralisé des messages. Désormais, cette communauté partage en plus des fonctions de base, des algorithmes complets et des méthodes permettant d'utiliser des bibliothèques spécialisées telles

1. Un middleware est un logiciel permettant la communication entre les couches basses (capteurs, actionneurs) et les couches hautes de traitements.
2. http://pr.willowgarage.com
3. http://stair.stanford.edu
4. Une architecture « Peer to Peer » ou Pair à Pair, est une architecture informatique permettant de communiquer directement d'un client à un autre client. Ces clients peuvent être des matériels, des programmes ou des machines.

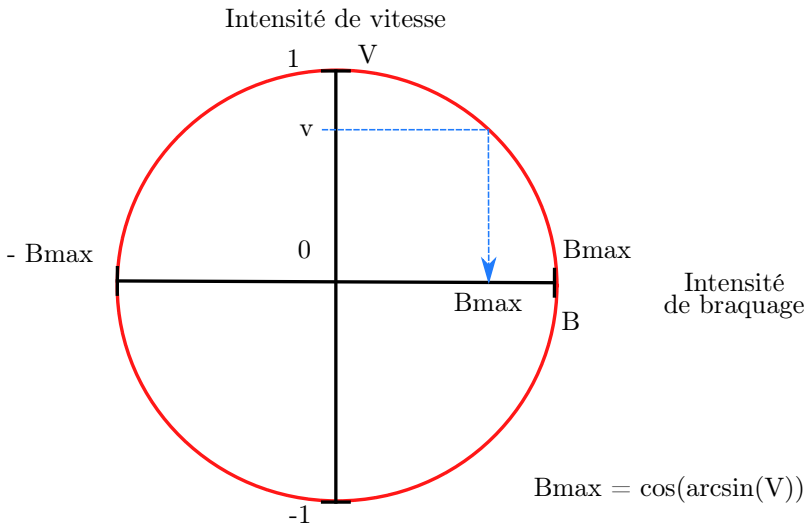


FIGURE 4.2 – Schématisation de l'obtention de l'intensité de braquage maximale B_{max} en fonction de l'intensité de poussée en vitesse V.

qu'OpenCV [Garage, 2011] pour le traitement d'images, ou PCL [Rusu et al., 2010] pour le traitement des nuages de points.

Ce système de communication permet donc à notre système de synchroniser toutes les données acquises. Comme le reste des modules du système, le pilotage à distance de la plateforme transite par ROS. En effet, la manette et le logiciel de pilotage se situent sur une machine distante. Le fonctionnement de ce logiciel de pilotage est expliqué par la suite.

4.1.3 Le contrôle de plateforme

Contrairement à un véhicule à taille humaine, une plateforme robotique ne peut être pilotée que de l'extérieur. ROS nous permet de pouvoir transmettre à distance des ordres de pilotage au robot. Ainsi, nous avons dû développer des contrôles à appliquer au robot pour qu'il réponde correctement à nos souhaits.

La plateforme n'est pas fournie avec un système de pilotage. Il est donc necessaire d'en développer un pour pouvoir réaliser des acqusitions. Par ailleurs, nous désirions que ce pilotage ne mette pas en danger l'intégrité de nos capteurs.

Afin de piloter la plateforme, une interface « Qt » [5] a été réalisée indiquant différentes informations à propos du robot. Ces informations sont : la vitesse des roues, la charge de la batterie et l'état d'alimentation électrique du LIDAR et de la caméra. Nous avons aussi ajouté un interrupteur électronique sur la plateforme afin de limiter l'utilisation de la batterie quand le système d'acquisition de données LIDAR n'est pas actif.

Le contrôle de la plateforme se fait grâce à une manette *XBOX 360* [6] qui permet à la fois de conduire le robot et de gérer l'alimentation de la caméra et du laser. De plus, nous avons implémenté un système de pilotage, utilisant cette manette, que nous souhaitions fluide et qui permettait d'éviter le renversement du robot en limitant l'intensité des virages à pleine vitesse. En effet, sans un contrôle équivalent, le robot pourrait entamer l'équivalent d'un demi-tour à pleine vitesse et provoquer la perte d'équilibre voire le retournement. Ainsi, le pilotage développé permet un pilotage précis de la plateforme sans risque de tête à queue.

5. http://www.qt.io/
6. https://www.microsoft.com/hardware/fr-fr/p/xbox-360-controller-for-windows

La commande en direction est réalisée grâce aux vitesses angulaires des roues gauches et droites ω_g et ω_d, le changement de direction du robot n'étant possible que par la variation du différentiel de vitesses comme les roues ne sont pas directionnelles. De surcroît, les vitesses angulaires sont calculées en fonction des intensités de poussée globale gauche et droite, respectivement $V_g \in [-1; 1]$ et $V_d \in [-1; 1]$, et de la vitesse angulaire maximale ω_{max}. Cette intensité de poussée globale est elle-même dépendante de l'intensité de poussée en vitesse et en braquage. Notons que ces intensités de poussée en vitesse et en braquage retranscrivent respectivement les pressions longitudinales et latérales exercées par l'utilisateur sur le joystick. Contrairement à un bouton où la valeur serait binaire, ici la valeur peut être répartie sur une plage et traduit ainsi l'intensité avec laquelle l'utilisateur souhaite que l'action soit réalisée. Cette intensité n'a pas d'unité, car elle résulte du rapport entre la position maximale du joystick et sa position actuelle.

Le système d'équations (4.1) qui suit le contrôle appliqué sur le différentiel en fonction de l'intensité de poussée appliquée sur le joystick.

Soit :

- ω_g et ω_d, les vitesses appliquées respectivement sur les roues gauches et droites
- ω_{max}, la vitesse maximale que l'on peut appliquer sur les roues
- $V_g \in [-1; 1]$ et $V_d \in [-1; 1]$, les intensités de poussée globales gauche et droite respectivement
- $V \in [-1; 1]$, l'intensité de poussée en vitesse
- $B \in [-B_{max}; B_{max}]$, la valeur de braquage, fonction de l'intensité de poussée latérale appliquée sur le joystick. Où B_{max}, la valeur de braquage maximale est définie en fonction de l'intensité de poussée en vitesse.

On a :

$$\begin{aligned} \omega_g &= V_g \times \omega_{max} \\ \omega_d &= V_d \times \omega_{max} \\ V_g &= V - B \times \cos(\arcsin(V)) \\ V_d &= V + B \times \cos(\arcsin(V)) \end{aligned} \tag{4.1}$$

Plus l'intensité de poussée en vitesse est élevée, plus le braquage possible est faible (Figure 4.2). Par exemple, si l'intensité de poussée en vitesse V est à $|0,9|$, $B \times \cos(\arcsin(V))$ ne pourra pas dépasser la valeur $|0,44|$.

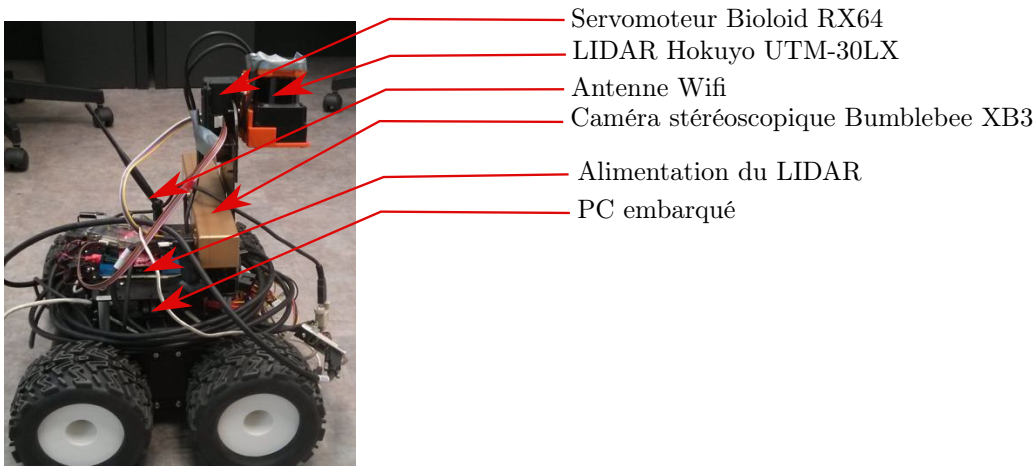


FIGURE 4.3 – Notre plateforme robotique de profil.

FIGURE 4.4 – Le champ de vision de la caméra stéréo n'est pas altéré par le laser ou la plateforme.

Ce contrôle permet de réduire l'amplitude des virages et d'ainsi mieux gérer sa trajectoire. A vitesse nulle, le robot peut réaliser un virage très court où seules les roues d'un côté s'activent. En revanche, à vitesse maximale, le robot ne pourra appliqué qu'un différentiel à 50%. Ainsi, il permet d'éviter toute dégradation de la caméra et du LIDAR installés que nous allons vous présenter.

4.2 Les capteurs

Notre approche mêle vision par caméra et télémétrie laser. Dans ce cadre, la caméra choisie est une caméra stéréoscopique *Point Grey XB3 Bumblebee* et le LIDAR est un *Hokuyo UTM-30LX*. Ce LIDAR 2D est monté sur un servomoteur *Dynamixel RX-64* permettant de transformer les informations 2D données par le LIDAR en informations 3D dont nous avons besoin (Fig. 4.3).

4.2.1 Caméras

La caméra choisie est une caméra stéréoscopique *Point Grey XB3 Bumblebee*. Cette caméra est tropicalisée et peut de ce fait être utilisée en extérieur et montée sur un véhicule qui voyage par mauvais temps ou en environnement humide. De par sa conception en boîtier, elle ne se décalibre pas. De plus, sa résolution est de 1280×960 (960p) ce qui est suffisant pour réaliser de la visualisation et qui permet d'espérer des traitements performants jusqu'à des distances suffisament élevées pour cette plateforme. En effet, on peut évaluer que la distance maximale d'acquisition de la profondeur est d'environ 17 mètres avec la baseline de 24cm car sa distance focale est de $3.8mm$ pour une ouverture de 66°. Enfin, le capteur est un CCD fonctionnant avec un « Global shutter » (voir section 3.1.1 page 56). En outre, elle transporte les données en Firewire 800 ce qui permet d'obtenir jusqu'à 16 images par seconde. Cette caméra est montée à l'avant du robot sous le LIDAR, ainsi sa vision n'est pas entravée par le robot ou le LIDAR (Figure 4.1 et 4.4). Étudions maintenant notre système LIDAR d'acquisition 3D.

4.2.2 Laser 3D à balancier

Avec l'aide des images extraites de la caméra stéréoscopique, nous pouvons calculer le positionnement relatif de la plateforme par rapport à son point de départ. Ainsi, nous pouvons positionner

les données 3D acquises par le LIDAR. Comme expliqué dans le chapitre précédent, les télémètres laser 3D étant trop coûteux pour notre application, nous nous sommes orientés vers une solution 2D. Cependant, nous n'avons pas abandonné l'idée d'obtenir des données 3D, c'est pourquoi ce LIDAR monocouche mis en rotation par un servomoteur nous permettant de transformer les informations 2D en informations 3D. La suite de ce chapitre va expliquer le matériel et le fonctionnement de ce système d'acquisition.

4.2.2.1 Matériel

Nous avons utilisé un LIDAR *Hokuyo UTM-30LX*. Ce LIDAR est un laser à balayage monocouche qui peut télémétrer jusqu'à 30 mètres. Cependant les faisceaux laser voyagent jusqu'à une distance de 60 mètres. Cette portée est acceptable pour notre application, car elle ne trahira pas la position du véhicule sur de longues portées. De plus, le LIDAR a une résolution angulaire de $0,25°$ et un champ d'acquisition de 270° vers l'avant soit 1080 points d'impact [7] maximum lors d'un balayage complet. L'ensemble des impacts lasers sont transmis sous la forme d'un « scan » qui est un ensemble de couples (distance, pas), à une fréquence de $40Hz$ soit 40 scans par seconde équivalent à 43200 points par seconde.

4.2.2.2 Principe

Pour faire tourner le LIDAR nous avons utilisé un servomoteur *Dynamixel RX-64* supportant le poids du LIDAR qui pèse plus de 200g et pouvant le faire tourner à une vitesse suffisante. En effet, le servomoteur *Dynamixel RX-64* a un couple de $4N.m$ qui permet de supporter l'*Hokuyo UTM-30LX* tout en tournant à une vitesse angulaire de $90°/s$.

Malgré l'utilisation d'un LIDAR monocouche, nous souhaitions réaliser de la reconstruction en trois dimensions. Pour cela nous avons positionné le LIDAR sur un servomoteur, inspirés par des articles traitant du sujet [Larson and Trivedi, 2011b; Larson et al., 2011].

Aussi, pour construire un scan 3D nous récupérons un scan 2D que nous mettons en correspondance avec l'angle du servomoteur avec lequel il a été acquis. En effet, le servomoteur est numérique et retourne sa position. Les scans acquis par le LIDAR et les angles du servomoteur sont horodatés. Par conséquent, la vitesse angulaire peut être calculée à tout moment pour pouvoir approximer l'angle exact avec lequel le scan a été acquis. Cette vitesse angulaire est le résultat d'un filtre passe-bas appliqué sur les vitesses angulaires les plus récemment mesurées à partir des données du servomoteur.

Pour la rotation du LIDAR sur le servomoteur, deux approches étaient possibles. La première (Figure 4.5) consistait à faire tourner le servomoteur suivant l'axe latéral du robot, nous la nommons « approche proche-loin ». Dans cette approche les données sont acquises en partant des distances proches pour acquérir à la fin des données lointaines. L'autre approche (Figure 4.6) consistait à faire tourner le servomoteur suivant l'axe longitudinal du robot. Cette méthode acquiert des données hori gauche du véhicule jusqu'à sa droite en passant puis inversement. Nous appelons cette approche, « l'approche gauche-droite ».

Ces deux méthodes diffèrent par la résolution qu'elles offrent. En effet, l'approche proche-loin offre une forte résolution sur toute la largeur de l'environnement, mais la résolution dans le sens de la marche est limitée. Comme le montre la figure 4.7 page 88, au-delà de 10 mètres devant le véhicule, l'écart entre deux scans dépasse les 2 mètres. Par contre, tout au long de l'acquisition les informations sont bien réparties latéralement. Cette situation s'explique par la résolution angulaire liée à la vitesse de rotation du servomoteur qui n'est que de 2.25° et qui provoque un grand écart entre

7. Les points d'impact sont des points dont on récupère la distance. Si un objet est à moins de 30 mètres et qu'il est impacté par une impulsion lumineuse envoyée par le LIDAR, il en résultera la mesure d'une distance jusqu'à ce point d'impact.

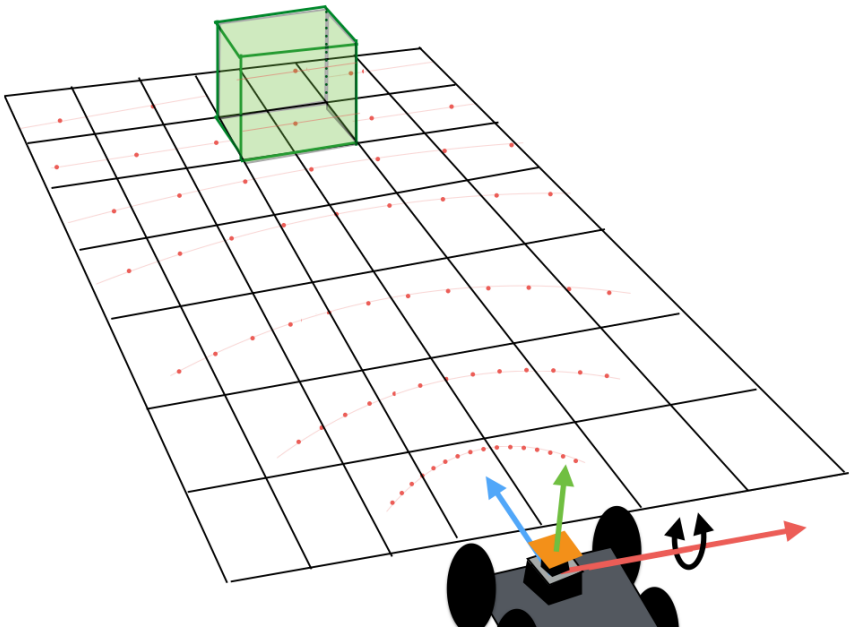


FIGURE 4.5 – Schéma d'acquisition de données avec l'approche « proche-loin » de rotation du servomoteur.

les données. Au contraire, la méthode gauche-droite garde une forte résolution jusqu'à 30 mètres frontalement au véhicule et perd rapidement de l'information sur les côtés (Figure 4.7 page 88).

Toutefois, la problématique de notre reconstruction 3D étant de permettre la détection des dangers sur le chemin du véhicule, la priorité est l'arc avant. La résolution doit donc être la plus importante suivant l'axe longitudinal du véhicule ce qui explique le choix de l'approche gauche-droite.

Enfin, comme le montre le schéma (Figure 4.5), le problème de l'approche proche-loin est que l'écart entre les scans varie beaucoup entre les acquisitions proches du robot et les acquisitions lointaines. D'autre part, nous n'aurions pas à tout moment un aperçu du terrain proche et lointain ce qui poserait donc des problèmes de viabilité vis-à-vis de la détection d'obstacles. Néanmoins, ce type d'approche semble plutôt efficace pour les acquisitions immobiles, comme la reconstruction d'une salle ou d'un objet [Espino-Corsino et al., 2011].

Ainsi, dans les 5 mètres de part et d'autre du centre du véhicule la résolution est assez élevée jusqu'à des distances éloignées (30 mètres), voyons ceci en détail.

4.2.2.3 Résolution

Comme l'indique la figure 4.8 page 89, un grand nombre de points peuvent être acquis par notre système autour de l'axe longitudinal. Effectivement, à la base un *Hokuyo UTM-30LX* acquière des données avec une résolution angulaire de $0,25°$ séparant deux points dans un scan. Pour un scan acquis à 10 mètres, cette résolution de $0,25°$ équivaut à un écart de 4 centimètres entre deux points et à 30 mètres on obtient un écart de 13 centimètres. Notons qu'à proximité du véhicule l'écart est inférieur au centimètre (tableau 4.1 et 4.6).

Cependant, dans notre approche le servomoteur tourne à une vitesse angulaire de 90° par seconde. Le LIDAR ayant une fréquence d'acquisition des scans de $40Hz$, la résolution angulaire

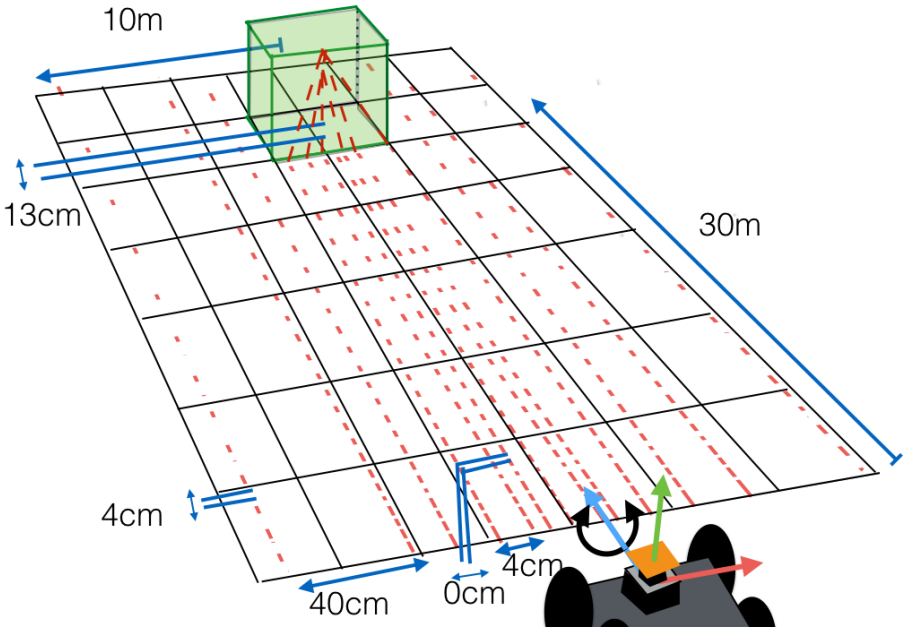


FIGURE 4.6 – Schéma d'acquisition de données avec l'approche « gauche-droite » de rotation du servomoteur. Cette approche est l'approche choisie dans notre système.

obtenue est égale à $2,25°$ ($=\frac{90}{40}$ obtenu après mise au point du système, voir paragraphe 4.2.2.4) entre chaque scan de la reconstruction. De plus, comme nous l'indique le tableau 4.1, cette résolution équivaut à un écart latéral de 4 centimètres dans la proximité du véhicule, d'environ 40 centimètres à une distance d'acquisition de 10 mètres et de $1,2$ mètres à 30 mètres (Figure 4.6).

Ainsi, sur le plan surfacique, jusqu'à 5 mètres devant le véhicule la densité de points au mètre carré décroit de 200 pts/m^2 sur l'axe longitudinal à 19 pts/m^2 quand on s'éloigne de 5 mètres de cet axe. Puis, entre 5 et 10 mètres en avant du véhicule la densité est entre 50 pts/m^2 au centre et 9 pts/m^2 sur les côtés. À 20 mètres cette densité est de 6 pts/m^2 au centre de l'axe de 1.5 pts/m^2 sur les côtés. Enfin, à 30 mètres la densité reste à 6 pts/m^2 au centre, mais tombe à 1 pts/m^2 sur les côtés (Figure 4.9 page 89).

Cette densité surfacique n'est certes pas très élevée à plus de 20 mètres, elle s'avère cependant représentative dans le cadre de notre preuve de concept sur une petite plateforme mobile pour réaliser de la détection d'obstacles et réagir en cas d'alerte.

Écart entre deux points de points à	$\leq$ 1m	10m	30m
Axe gauche-droite	$\sim$ 0cm	4cm	13cm
Axe proche-loin	4cm	40cm	1.2m

TABLE 4.1 – Écart entre deux points acquis par le système 3D à différentes distances

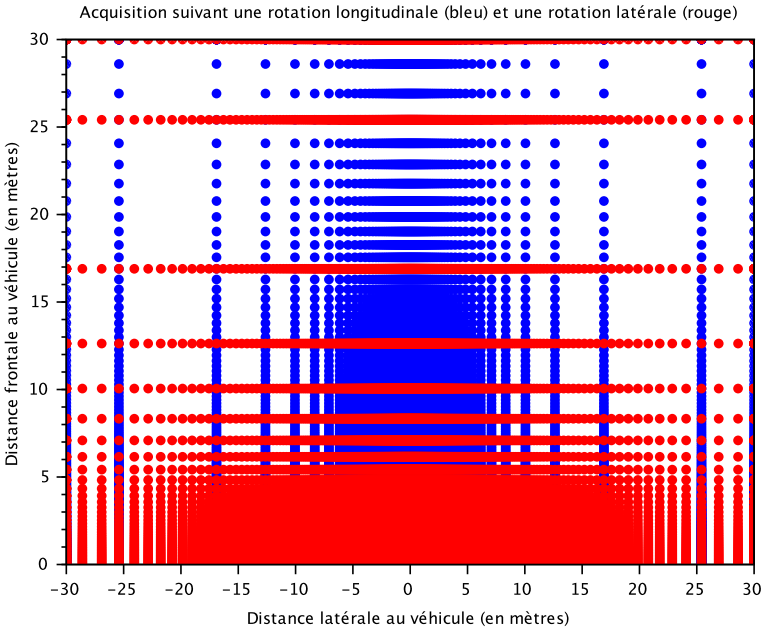


FIGURE 4.7 – Répartition des points selon l'approche proche-loin (rouge) et l'approche gauche-droite (bleu).

4.2.2.4 Mise au point

Une fois l'installation du LIDAR sur le servomoteur réalisée, il a été nécessaire de passer par une période de mise au point afin de rechercher les erreurs de reconstruction et de les corriger. Pour ce faire nous avons effectué des acquisitions immobiles dans une petite salle vide et régulière. En effet, comme le montre la figure (Figure 4.10b) cette salle a des propriétés idéales pour la recherche d'erreurs : ses murs sont lisses et perpendiculaires au sol, perpendiculaires les uns aux autres et de tailles équivalentes. De plus, le plafond est lui aussi lisse et perpendiculaire aux murs. Ces propriétés sont idéales pour la recherche d'erreurs, car la moindre irrégularité dans la reconstruction ressort clairement. Par conséquent, cette salle nous a permis de tester la reconstruction avec un robot immobile et de faire ressortir aisément des erreurs de reconstruction en repérant les erreurs orthogonalité.

Ainsi, des tests avec les jeux de données acquis dans cette salle nous ont permis de trouver les limites de notre système quant à la vitesse de rotation du servomoteur. En faisant tourner le servomoteur à une vitesse angulaire proche de $180°/s$, les données ne sont plus transmises assez rapidement et les scans ne sont pas positionnés avec les bons angles ce qui provoque des erreurs de reconstruction (Figure 4.10a). Ces tests nous ont permis de valider une valeur de $90°/s$ comme vitesse de rotation définitive pour le servomoteur (Figure 4.10b). En effet, cette vitesse est la plus élevée atteignable en gardant des données de reconstruction cohérentes. Car, au-dessus de cette vitesse, les échanges de données (durant $8ms$) sont trop longs et faussent le positionnement des scans. Notre sustème acquière donc l'intégralité de l'environnement frontal en 2 secondes.

4.2.3 Calibration

Dans notre cadre expérimental, nous n'avons pas établi de système de calibration automatique ni de recalage de la calibration. La calibration de la plateforme est réalisée à la main. Ainsi, nous

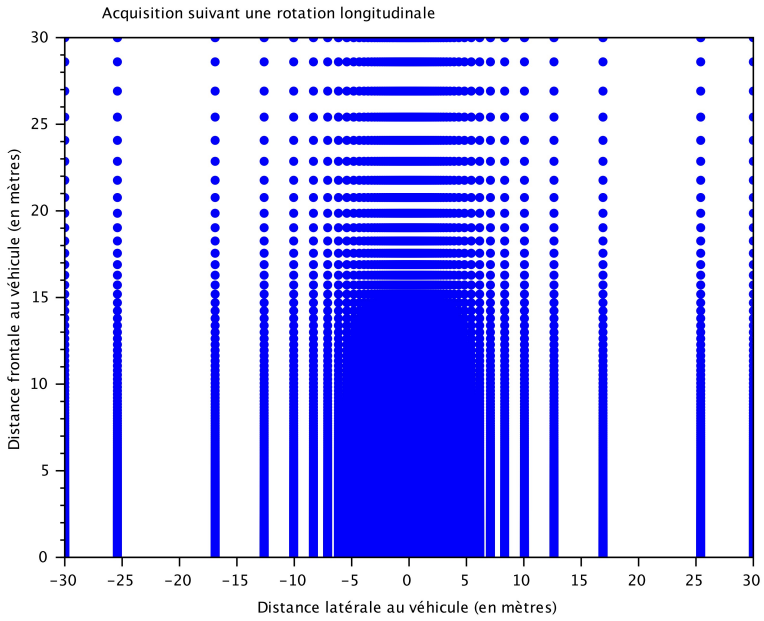


FIGURE 4.8 – Répartition des points selon l'approche gauche-droite entre 0 et 30m.

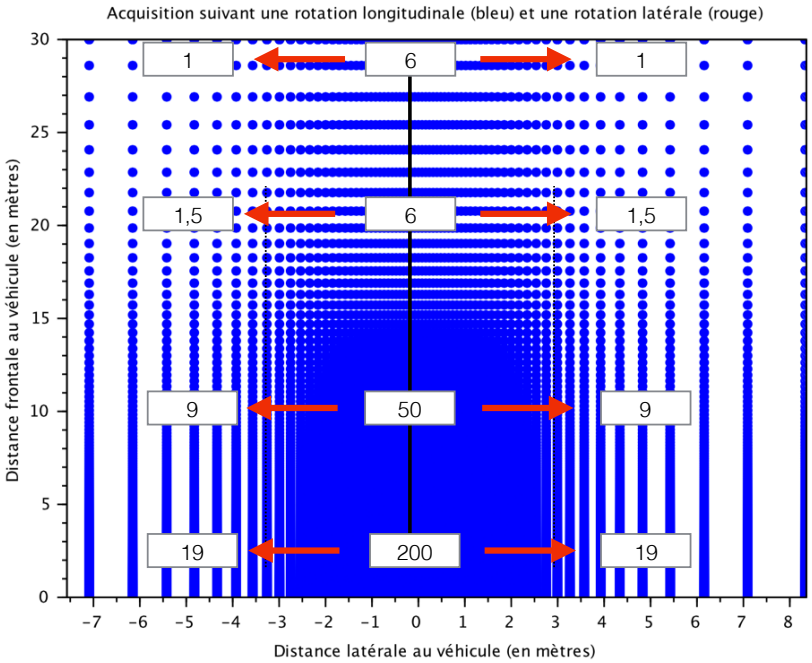
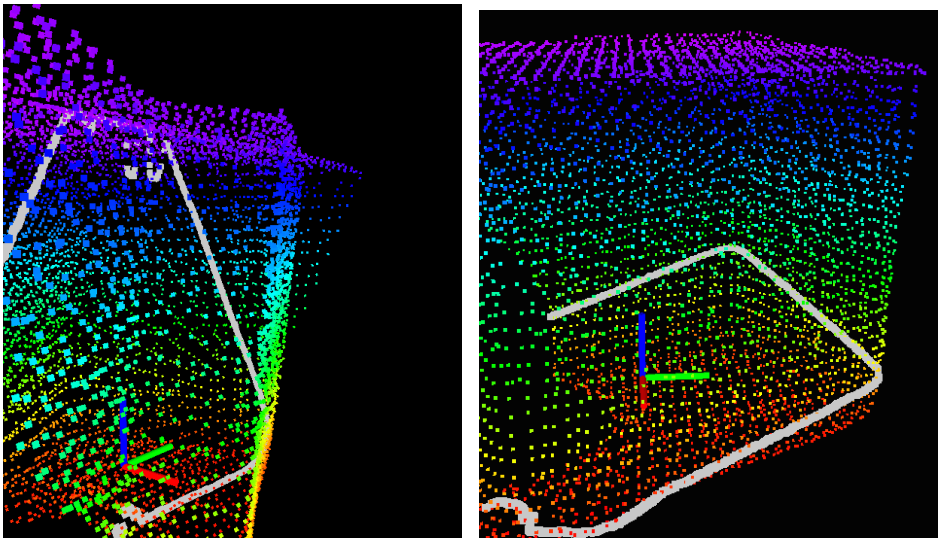


FIGURE 4.9 – Évolution de la densité surfacique (indiquée en points par mètre carré dans les rectangles) en fonction de la distance d'acquisition.



(a) Erreur de reconstruction repérée quand la fréquence est trop élevée. On constate une irrégularité dans la reconstruction.

(b) Reconstruction à 90°/s qui permet d'obtenir une reconstruction fidèle à la vérité.

FIGURE 4.10 – Mise au point de la reconstruction 3D immobile et détection d'erreur.

présentons les paramètres définissant les différents repères de la plateforme. Elle est composée d'un repère par capteurs soit quatre repères. Le repère principal est le repère robot R_r situé au centre de la plateforme roulante. Le repère caméra R_c est situé sur l'image de référence de l'odométrie, soit l'image de gauche. Le servomoteur R_s est placé sur l'axe central de la plateforme et le Lidar R_l est dans l'axe du servomoteur.

Si l'on considère un repère monde R_m on a :

$$
\begin{aligned}
R_r &= \begin{bmatrix} R_{m3\times3} & \begin{bmatrix} 0 \\ 0 \\ t_{zr} \end{bmatrix} \\ 0 & 1 \end{bmatrix} \\
R_c &= \begin{bmatrix} R_{r3\times3} & \begin{bmatrix} t_{xc} \\ t_{yc} \\ t_{zc} \end{bmatrix} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \text{ Rotation de } 90^\circ \text{ autour de l'axe x} \\
R_s &= \begin{bmatrix} R_{r3\times3} & \begin{bmatrix} t_{xs} \\ 0 \\ t_{zs} \end{bmatrix} \\ 0 & 1 \end{bmatrix} \\
R_l &= \begin{bmatrix} R_{r3\times3} & \begin{bmatrix} t_{xl} \\ 0 \\ t_{zl} \end{bmatrix} \\ 0 & 1 \end{bmatrix}
\end{aligned}
\tag{4.2}
$$

Où t_{zr}, t_{xc}, t_{yc}, t_{zc}, t_{xs},t_{zs}, t_{xl}, t_{zl} sont mesurés à la main.

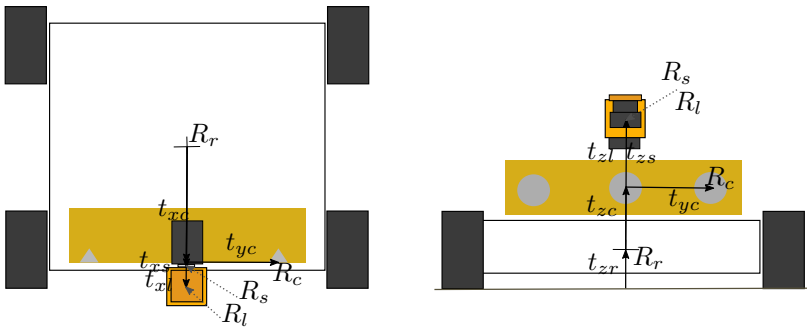
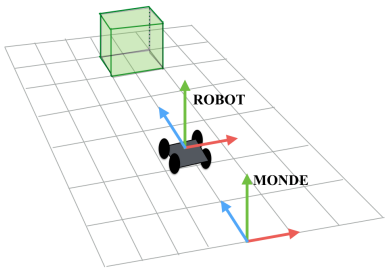
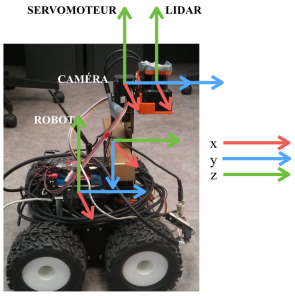


FIGURE 4.11 – Schématisation des transformations liant nos repères sur la plateforme.



(a) Le repère robot mobile et le repère monde immobile



(b) Les différents repères intervenant dans l'élaboration de notre plateforme. On notera particulièrement le repère caméra qui est « z devant ».

FIGURE 4.12 – Schémas représentatifs des différents repères de notre système.

4.3 Conclusion

Ce chapitre présente la création de la plateforme d'acquisition développée sur le concept de notre système de perception d'environnement. Cette plateforme n'est pas représentative des véhicules de notre application, néanmoins, elle est équipée de l'intégralité des capteurs de notre concept. Ainsi, elle offre la capacité de valider le concept au travers d'expérimentations menées avec les capteurs choisis. Cette plateforme est utilisée pour l'acquisition de jeux de données permettant l'évaluation du concept et des algorithmes développés en réponse à notre problématique.

La chapitre suivant présente les développements algorithmiques permettant la validation du concept au travers de résultats obtenus sur les jeux de données acquis grâce à notre plateforme et sur des jeux publics.

Chapitre 5

Reconstruction par données LIDAR et odométrie visuelle

Sommaire

5.1 Acquisition de données LIDAR et odométrie stéréoscopique 94
5.1.1 Reconstruction par acquisition de données LIDAR et odométrie visuelle . . . 94
5.1.2 Résultats 99
5.1.3 Conclusion 109
5.2 Optimisation par ajout d'une estimation de la rotation 109
5.2.1 Initialisation par estimation de la rotation 109
5.2.2 Etude du déplacement d'une image entre deux instants 110
5.2.3 Résultats 120
5.2.4 Conclusion quant à l'estimation de la rotation 125
5.3 Conclusion quant à notre algorithme 127

Comme le rappelle la présentation de la partie, afin de répondre à la problématique : *sécuriser le pilotage des véhicules militaires avec un système de perception d'environnement* nous réalisons une reconstruction de l'environnement frontal du véhicule pour alimenter une future aide au pilotage de détection d'obstacles. Pour ce faire, nous avons conçu un système n'utilisant que la perception de l'environnement extérieur au véhicule. Ainsi, cette reconstruction est créée avec les données issues d'un capteur LIDAR et une caméra stéréoscopique.

En effet, le chapitre 2 - *État de l'art* - et le chapitre 3 - *Définition des capteurs de notre système* - ont respectivement montré qu'au vu de l'état de l'art, une solution nouvelle était nécessaire pour répondre à notre problématique. Ainsi, le chapitre 3 a développé le concept de notre reconstruction d'environnement au travers de la solution matérielle. Compte tenu des contraintes de coût, nous avons réalisé notre propre système d'acquisition 3D sur la base d'un LIDAR 2D monté sur un servomoteur. Le chapitre 5 - *Reconstruction par données LIDAR et odométrie visuelle* - présente le système qui peut acquérir avec une bonne résolution (entre $200pts/m^2$ et $9pts/m^2$) l'environnement jusqu'à 20 mètres devant le véhicule et jusqu'à 5 mètres sur les côtés. Cependant, deux secondes sont nécessaires pour acquérir l'intégralité de l'environnement frontal. Ce système d'acquisition 3D est couplé à une caméra stéréoscopique permettant le calcul des déplacements réalisés par le véhicule.

Ce chapitre détaille les solutions algorithmiques développées à la suite de notre définition matérielle. Dans ce cadre, des résultats d'expériences démontrent la viabilité du concept face aux contraintes posées dans le chapitre 1. Ainsi, pour valider la robustesse du système à tout type d'environnements, nous avons évalué son moyen de positionnement sur des jeux de données tout terrain et des jeux de données routiers. De même, nous mesurons la qualité de la cartographie construite en environnement naturel. Enfin, les résultats évaluent l'impact des algorithmes le calculateur.

Les algorithmes présentés transmettent une cartographie 3D de l'environnement frontal au véhicule. Comme le montre le schéma fonctionnel (Figure 5.1), l'algorithme fonctionne en trois étapes :

(1) Le système acquiert des scans et les angles correspondants du servomoteur pour créer des nuages de points 3D instantanés

(2) Le système acquiert des images stéréoscopiques afin de calculer l'odométrie visuelle

(3) La mise en correspondance de l'odométrie et des nuages de points 3D instantanés permet la construction d'une cartographie 3D de l'environnement frontal au véhicule

Cette section détaille cet algorithme suivant les interfaces (1), (2) et (3). Ensuite, nous expliquons les résultats obtenus vis-a-vis des contraintes de notre application, en détaillant les moyens d'évaluations puis en présentant les résultats numériques. Enfin, une discussion expliquera en quoi ce premier algorithme ne satisfait pas nos contraintes, impliquant la recherche d'une optimisation.

Dans la section suivante, nous abordons l'optimisation apportée au calcul de l'odométrie stéréoscopique *LibViso 2* par l'ajout d'une estimation de la rotation ayant pour but d'accélérer l'odométrie stéréoscopique. Nous en détaillons le principe ainsi que les développements mathématiques et algorithmiques réalisés. Ensuite, nous étudions les nouveaux résultats obtenus en trajectoire et en reconstruction. Nous terminons ce chapitre par une discussion autour des résultats après optimisation et les contributions apportées.

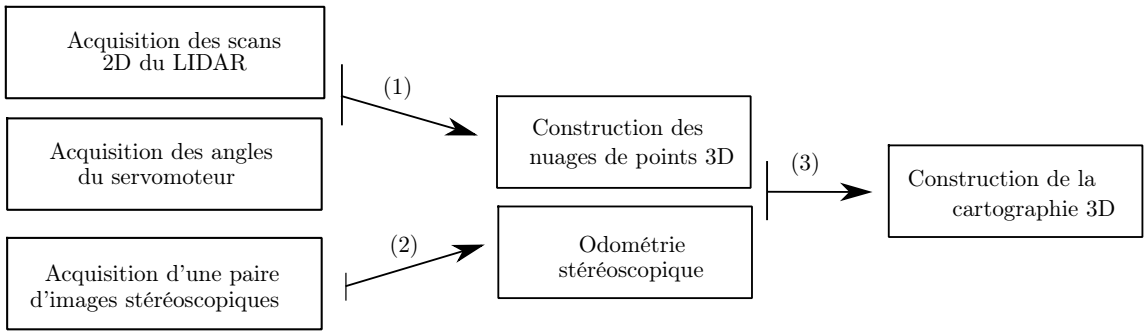


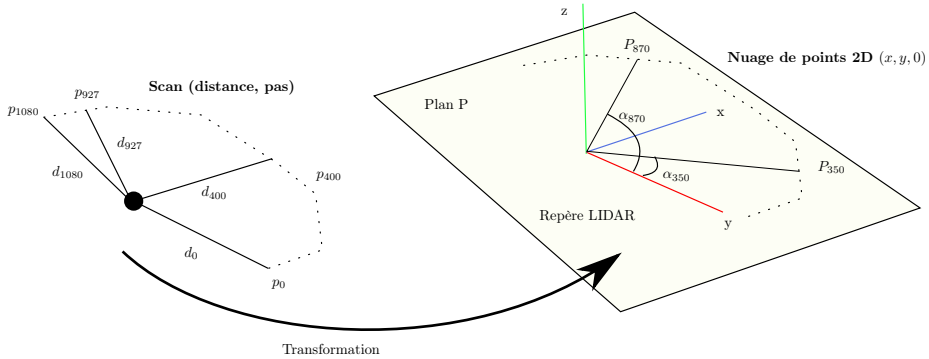
FIGURE 5.1 – Schéma fonctionnel de notre approche. Elle se compose en trois parties : l'acquisition de données, le prétraitement et la construction de la cartographie 3D de l'environnement.

5.1 Acquisition de données LIDAR et odométrie stéréoscopique

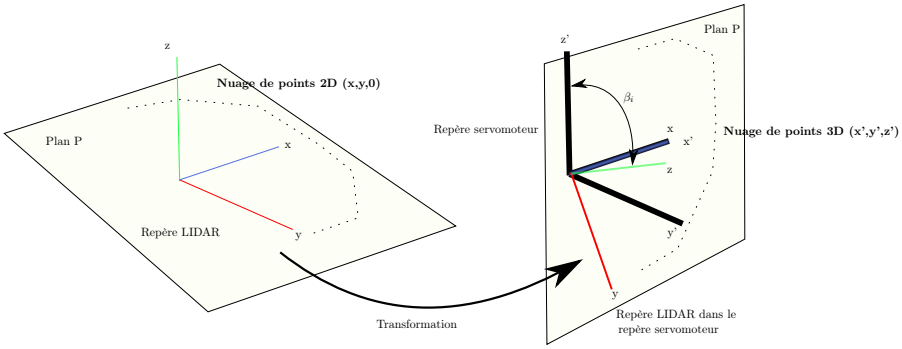
La présente section détaille le principe général des algorithmes développés dans notre méthode suivant le schéma fonctionnel de la figure 5.1. Elle détaille dans un premier temps comment nous créons des données 3D à partir de notre capteur LIDAR et du servomoteur (Figure 5.1 (1)). Au travers de plusieurs changements de repères (Figure 4.12b page 91), les scans transmis par le LIDAR sont transformés en nuages de points 3D dans le repère du robot. Ensuite, la section détaille le fonctionnement de l'algorithme d'odométrie visuelle *LibViso2* sur lequel notre algorithme se base (Figure 5.1 (2)). Cet algorithme utilise les paires des images stéréoscopiques pour calculer le mouvement que réalise le robot au travers du mouvement réalisé par la caméra. Enfin, nous expliquons la reconstruction de l'environnement résultant de la fusion de ces deux précédentes étapes (Figure 5.1 (3)). Cette reconstruction résulte du placement des nuages de points 3D instantanés entre les différentes positions du véhicule.

5.1.1 Reconstruction par acquisition de données LIDAR et odométrie visuelle

Notre système évolue avec différents repères, comme l'illustre les figures 4.12 page 91 qui présente les deux repères principaux que sont le repère monde et le repère robot. Le repère monde représente



(a) Transformation du scan en nuage de points 2D par la transformation du couple (distance, pas) en coordonnées 2D



(b) Transformation du nuage de points 2D en nuage de points 3D par l'application d'un changement de repère entre le repère LIDAR et le repère servomoteur.

FIGURE 5.2 – Illustration des étapes permettant de transformer le scan en un nuage de points 3D. Premièrement, le scan est transformé en nuage de points 2D par la mise en correspondance des pas avec la résolution angulaire et des distances en coordonnées. Puis le repère du nuage de points 2D est transformé grâce à l'angle du servomoteur donnant ainsi un nuage de points 3D dans le repère du servomoteur.

le repère immobile dans lequel la reconstruction d'environnement est réalisée. Le repère robot est quant à lui le repère dans lequel les acquisitions sont réalisées et les nuages de points construits et stockés en attendant d'être positionnés dans le repère monde grâce au calcul d'odométrie.

Le repère robot est le repère de référence pour tous les capteurs (caméra, LIDAR et servomoteur). La suite de ce chapitre détaille la création des données dans les repères capteurs au travers des interfaces (1) et (2) du schéma fonctionnel, ces repères capteurs étant ensuite exprimés dans le repère robot. L'interface (3) présente le passage du repère robot au repère monde.

5.1.1.1 Du scan 2D à un nuage de points 3D

Cette sous-section détaille l'interface (1) du schéma fonctionnel qui traduit la transformation du scan en un nuage de points 3D (Figure 5.1). Les nuages de points 3D nécessaires à la reconstruction ne sont pas des données provenant directement des capteurs mais des données que nous créons. Effectivement, notre approche utilise un LIDAR 2D monté sur un servomoteur et, à la base, le LIDAR transmet des scans 2D qui sont un ensemble de couples (distance, pas). Pour chaque couple, le pas correspond à l'indice de l'impulsion lumineuse projetée par le LIDAR dans les airs compris, dans notre cas, entre 0 et 1080. Ainsi, sachant que le LIDAR a un débattement angulaire constant et

défini (270° pour l'*Hokuyo UTM-30LX*), il suffit de mettre en correspondance le pas et la résolution angulaire pour obtenir l'angle d'acquisition. Par ailleurs, la distance est calculée en fonction du temps que met l'impulsion lumineuse envoyée pour revenir après avoir percuté un objet. Donc, avec cette distance (d) et l'angle (α) obtenue grâce au pas, nous pouvons transformer le scan en un nuage de points 2D dans un plan (x, y). En effet, en considérant que le LIDAR envoie des impulsions autour de son axe de lacet (z), que l'axe de roulis est (x) et que l'axe de tangage est (y) on obtient :

$$\begin{aligned} x &= d\cos(\alpha) \\ y &= d\sin(\alpha) \end{aligned} \tag{5.1}$$

En parallèle de l'acquisition LIDAR, nous envoyons des requêtes toutes les $8ms$ au servomoteur afin de connaître sa position. En contrepartie, le servomoteur transmet le pas auquel il se trouve. La mise en relation du pas, de l'ouverture angulaire du servomoteur (270°), du pas initial et du pas maximal permet d'obtenir l'angle actuel du servomoteur (β). Aussi, avec cet angle nous pouvons calculer la transformation qui lie le repère du servomoteur et le repère du LIDAR par une rotation autour de l'axe de roulis ru robot (x). Ainsi, en réalisant un changement de repère entre le repère du LIDAR et le repère du servomoteur on peut obtenir le nuage de points 3D (x', y', z') du scan acquis à l'angle β (Figure 5.2), tel que :

$$\begin{cases} x' = x \\ y' = \cos(\beta)y - \sin(\beta)z \\ z' = \sin(\beta)y + \cos(\beta)z \end{cases} \tag{5.2}$$

Pour réaliser la cartographie, tous les nuages de points 3D (chaque nuage représentant à un seul scan en 3D) ainsi créés sont placés dans l'environnement grâce à l'étude du déplacement de la plateforme. Cette étude de déplacement est réalisée grâce aux traitements des images stéréoscopiques. Cette méthode s'appelle l'odométrie visuelle et nous avons choisi d'utiliser une méthode existante proposée dans la bibliothèque communautaire de *ROS LibViso 2*.

5.1.1.2 Odométrie visuelle stéréoscopique : LibViso2

Cette sous-section détaille l'interface (2) du schéma fonctionnel qui représente le passage des images stéréoscopiques à l'odométrie (Figure 5.1 page 94). Comme expliqué dans [Kitt et al., 2010], *LibViso 2* est un algorithme permettant de calculer une estimation du mouvement réalisé par la caméra grâce à l'étude de deux paires d'images consécutives. Les paramètres du mouvement sont ensuite intégrés temporellement en appliquant un filtrage par « Iterated Sigma Point Kalman Filter » (ISPKF) [1][Sibley et al., 2006].

Premièrement, dans chaque paire d'images stéréoscopiques, on extrait des points d'intérêt de *Harris* (des coins) qu'on met en correspondance entre les deux images de la paire stéréoscopique.

Bucketing Une fois cette correspondance faite, on divise l'image en rectangles qui ne se recoupent pas (Figure 5.3). [Kitt et al., 2010] appelle cela le « Bucketing ». Dans ces rectangles ne sont gardés qu'un certain nombre de points d'intérêt afin de réduire la complexité algorithmique et améliorer qualité des échantillons. Ces points sont équitablement distribués sur l'axe de roulis (z) [2] pour distribuer équitablement les points d'intérêt sur les objets proches et lointains. C'est une étape

1. Comme expliqué dans [Sibley et al., 2006], l'*ISPKF* offre des propriétés intéressantes quand il est utilisé pour de la stéréoscopie à longue portée. De plus, son fonctionnement est proche de celui de l'*UKF*(Chapitre 2.3.2.2 page 35) à ceci prêt qu'il est itératif lors du choix des échantillons sigma.

2. En effet, le repère stéréoscopique est de type (z) devant, et dans ce cas (z) est l'axe de roulis. (Figure 4.12b page 91). La distribution est garantie équitable par l'estimation de la profondeur des pixels séléctionnés suite à l'appariement des deux images.

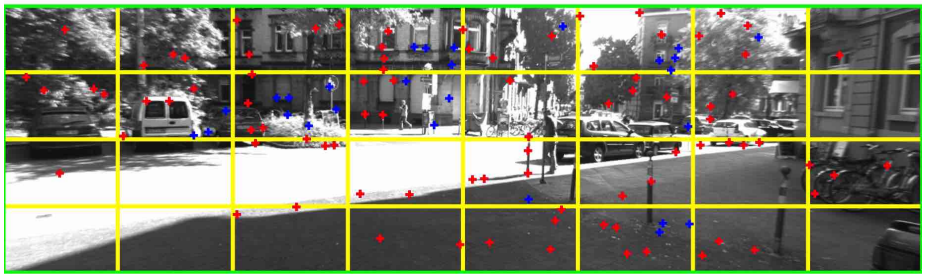


FIGURE 5.3 – Cette figure représente la manière dont fonctionne le « bucketing ». Les lignes jaunes montrent les « bucket » ; les croix représentent les appariements trouvés entre les deux images de la paire stéréo : les croix rouges sont les points d'intérêt sélectionnés et les bleus sont ceux rejetés. Extrait de [Kitt et al., 2010]

importante pour une bonne estimation de la vitesse angulaire et linéaire [Kitt et al., 2010]. En effet, sans cette démarche un grand nombre de points d'intérêt pourraient se retrouver sur un objet proche et mobile quand peu seraient positionnés sur le fond fixe et lointain. Il en résulterait une fausse estimation du mouvement majoritairement influencée par ces points d'intérêt sur l'objet mobile.

Ensuite, les points d'intérêt sont uniformément distribués dans toute l'image afin d'obtenir une distribution équitable dans toutes les dimensions et ainsi maximiser les chances d'une bonne estimation du mouvement. Cette technique de « bucketting » réduit la dérive de cette approche même dans des situations dégradées [Kitt et al., 2010].

Filtrage Dans cette approche le but du filtrage est de supprimer les points d'intérêt appartenant à des objets mobiles restant afin qu'ils ne faussent pas l'estimation du mouvement de la caméra. Il fonctionne de la manière suivante :

1. Un sous-échantillon de n de points d'intérêt est choisi de manière aléatoire suivant :

$$n = \frac{\log(1-p)}{\log(1-(1-\varepsilon)^s)} \tag{5.3}$$

   Où n est le nombre de points d'intérêt, $s = 3$ est le nombre minimal de points d'intérêt requis pour l'estimation du mouvement, p est la probabilité qu'au moins un échantillon contienne un point d'intérêt valide et ε définit le pourcentage de données aberrantes [Civera et al., 2009].
2. Sur cet échantillon, le mouvement est estimé, tel qu'expliqué dans le paragraphe 2.2.2.2 page 27, à l'aide d'un filtre de Kalman *ISPKF*
3. Une fois que ce filtre a convergé, l'erreur euclidienne de reprojection [3] est calculée. Les points d'intérêt dont l'erreur est supérieure à un seuil sont considérés comme des valeurs aberrantes et supprimés de l'échantillon sur lequel sera appliqué le calcul final.
4. Une dernière estimation du mouvement est réalisée avec les points d'intérêt validés précédemment (Figure 5.4).

L'estimation du mouvement nous permet de récupérer la nouvelle position et la nouvelle orientation de notre véhicule. Comme indiqué précédemment, dans cet algorithme, c'est le filtre de Kalman *ISPKF* qui est utilisé pour le calcul. Nous allons le présenter brièvement.

3. L'erreur euclidienne de reprojection consiste à calculer à partir d'une image la projection d'un point de l'image dans l'espace. Puis ce point de l'espace est reprojeté dans l'image. Idéalement, les points de départ et d'arrivée dans l'image devraient être confondus, mais avec les approximations de calculs ils seront décalés. Aussi, la distance euclidienne entre ces deux points dans l'image correspond à l'erreur euclidienne de reprojection.

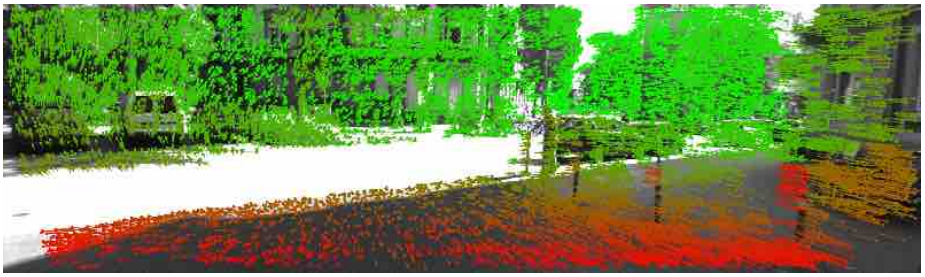


FIGURE 5.4 – Le flux optique - qui représente l'ensemble des vecteurs de déplacement des points d'intérêt - calculé par *Libviso 2*. On remarque les vecteurs de mouvements des points d'intérêt dont la couleur indique la distance vis-à-vis de l'observateur. Le rouge indique une distance proche et le vert une distance lointaine. L'*egomotion* estime l'inverse du mouvement moyen de ce flux optique (chapitre 2.2.2.2 page 27) . Extrait de [Kitt et al., 2010]

Filtre de Kalman (ISPKF) Cette approche utilise le filtre de Kalman *ISPKF*[Sibley et al., 2006] dont le fonctionnement est similaire à l'*UKF*, détaillé dans le chapitre 2.3.2.2 page 35, à la différence près que l'*ISPKF* est itératif. De surcroît, [Kitt et al., 2010] font l'hypothèse d'une vitesse constante entre deux images consécutives.

Bien que cette hypothèse soit fausse dans le cas d'une accélération, d'une décélération ou d'un virage, elle est une bonne approximation du mouvement réel [Kitt et al., 2010]. En effet, le temps est très court entre deux images consécutives, car les caméras ont des fréquences d'acquisition élevées, par conséquent, la vitesse n'a pas le temps d'évoluer plus que d'une valeur négligeable. Par contre, la vitesse change à chaque image, l'hypothèse de vitesse constante n'est faite que dans le temps qui sépare ces images.

Comme l'*UKF*, le filtre *ISPKF* réduit l'erreur de linéarisation vis-à-vis d'un *EKF*, cependant il converge aussi beaucoup plus vite que l'*EKF* contrairement à l'*UKF*. [Kitt et al., 2010] annonce une convergence accélérée d'un facteur 60 qui s'explique par le caractère itératif de la méthode [Sibley et al., 2006].

Si nous avons choisi d'utiliser *LibViso 2* c'est parce qu'il présente de bons résultats. Cet algorithme est classé dans le benchmark *Kitti* [Geiger et al., 2012] avec une erreur de seulement $2,4\%$ [4]. De plus, cet algorithme est open-source et disponible au sein de la communauté *ROS* sous le nom *viso2_ROS*[5].

Ainsi, grâce à nos nuages de points 3D et à l'odométrie visuelle calculée par *LibViso 2* nous pouvons reconstruire notre environnement. La sous-section suivante détaille l'approche utilisée pour cette reconstruction.

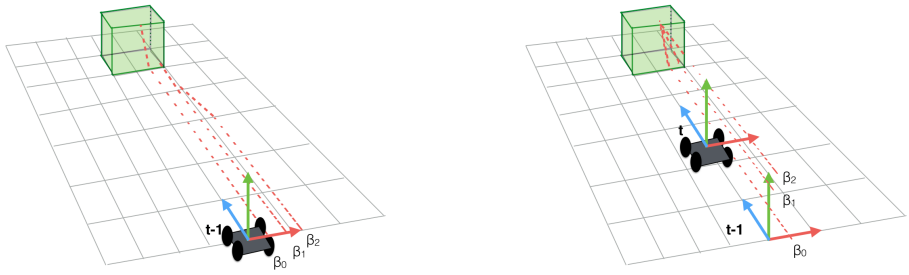
5.1.1.3 Reconstruction de l'environnement 3D

Cette sous-section détaille l'interface (3) du schéma fonctionnel qui représente la reconstruction de l'environnement avec l'odométrie et les nuages de points 3D. Notre étape de reconstruction est simple, nous n'utilisons que deux entrées :

- les nuages de points 3D créés à partir des scans et des angles du servomoteur ;
- la position et l'orientation actuelles de notre véhicule calculées par *LibViso 2* et la précédente estimation.

L'estimation du mouvement s'exécute une fréquence moins élevée que l'acquisition des scans et la création des nuages de points 3D, aussi les nuages de points 3D sont stockés en attendant d'être

4. http ://www.cvlibs.net/datasets/kitti/eval_odometry.php
5. http ://wiki.ros.org/viso2_ros



(a) Dans un premier temps les nuages de points LIDAR de tous les angles β(Un angle par scan) du servomoteur sont stockés dans le repère robot.

(b) Ensuite, ils sont replacés équitablement dans le repère monde entre la dernière position connue et la nouvelle position estimée par l'odométrie visuelle.

FIGURE 5.5 – Reconstruction 3D de l'environnement frontal par répartition temporelle des données LIDAR entre les différentes positions calculées par l'odométrie visuelle.

utilisés. Puis, quand une nouvelle odométrie est calculée, les nuages de points sont équitablement répartis sur le vecteur séparant la position actuelle et la position précédente dans le repère monde (Figure 5.5) et la structure de stockage se vide pour accueillir les nouveaux nuages.

Comme pour l'odométrie visuelle, notre reconstruction fait aussi l'hypothèse d'une vitesse constante entre chaque position calculée. De plus, le LIDAR obtient des scans à fréquence constante, tous les quarantièmes de seconde. Les changements de vitesse peuvent donc aussi être considérés comme négligeables et les scans sont censés être séparés d'une distance égale entre deux odométries.

Enfin, dans notre approche nous ne souhaitons pas stocker nos données une fois qu'elles ne servent plus à la détection. Par conséquent, les points dépassés par le robot seront supprimés du nuage de points après plusieurs mètres.

Théoriquement, les performances de *LibViso 2* nous permettent d'obtenir des résultats très fiables grâce à son faible taux d'erreurs annoncé et sa vitesse d'exécution. Ainsi, nous espérons une trajectoire très fiable permettant d'obtenir une reconstruction cohérente spatialement. De plus, nous esperons que la distance séparant cette reconstruction et la vérité terrain soit très faible, ce qui indiquerait que la reconstruction est très fiable. Nous vous présentons maintenant les résultats obtenus avec cette méthode.

5.1.2 Résultats

L'algorithme dont le principe est présenté précédemment a été installé sur la plateforme d'acquisition afin d'expérimenter les algorithmes en situation réelle. Nous évaluons les résultats de cette expérience suivant plusieurs critères. Premièrement, afin d'évaluer la robustesse de notre approche face à différents environnements nous utilisons la précision de la trajectoire (pourcentage et mètres) et la reconstruction 3D (écart en mètres) comme métriques. Deuxièmement, nous évaluons aussi le pourcentage d'utilisation des CPUs comme métrique pour l'impact de nos algorithmes sur le système.

Pour pouvoir évaluer l'algorithme en situation réelle nous avons réalisé une acquisition de l'environnement dans lequel la plateforme a évolué. Cette acquisition nous a servie au titre de vérité terrain et à été réalisée avec un scanner laser 3D fixe. Cette acquisition nous sert de référence pour l'évaluation que nous effectuons. Nous présentons d'abord cette acquisition de « vérité terrain » afin de mieux comprendre les résultats obtenus dans notre évaluation des algorithmes.

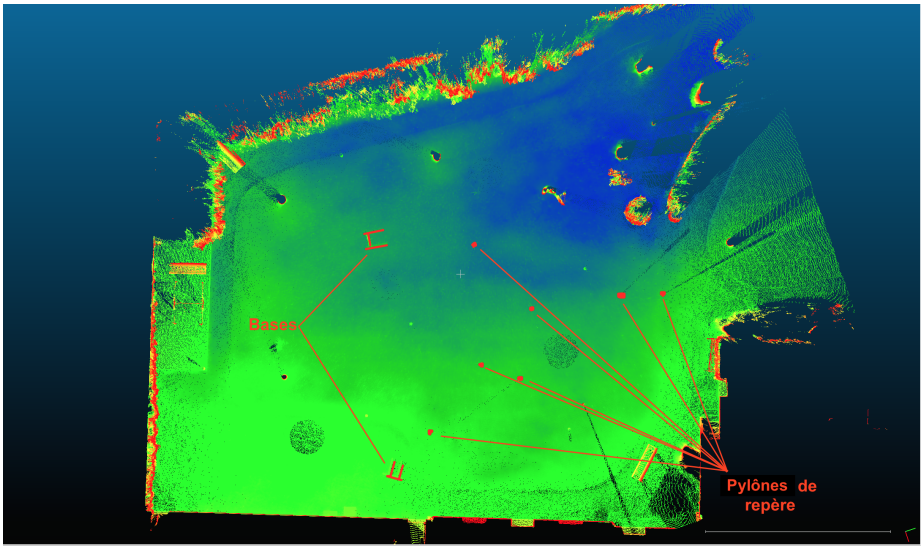


FIGURE 5.6 – Acquisition réalisée avec le scanner laser *FARO*. La couleur indique la hauteur (le bleu représente à une altitude basse et le rouge une altitude élevée). De plus, sont indiqués en rouge, les bases de départ et d'arrivée ainsi que les pylônes de repère (amers).

5.1.2.1 Vérité terrain

Notre environnement de test La thèse s'est déroulée essentiellement au laboratoire du CAOR à MINES ParisTech. Le laboratoire possède un espace extérieur composé de zone d'herbe et de gravier. C'est dans cet environnement que nos expériences ont pris place. La surface d'expérimentation était d'environ 20 mètres par 15 en face du laboratoire. Le robot n'est pas équipé de système de géolocalisation précis, c'est pourquoi nous avons positionné des amers physiques dans l'environnement. Par ailleurs, nous avons aussi fait appel à la vérité terrain fournie avec les jeux de données *Kitti* pour l'évaluation des trajectoires en environnement routier.

Vérité terrain Ces amers représentés par dix pylônes et deux bases de départ et d'arrivée nous permettent de reconstituer le chemin emprunté par le robot et d'ainsi pouvoir évaluer la trajectoire calculée par l'algorithme d'odométrie (Figure 5.6). La trajectoire de vérité terrain n'est donc pas précise au centimètre mais permet tout de même d'évaluer la précision de la trajectoire calculée par nos algorithmes.

De plus, nous avons utilisé un scanner laser *FARO Focus 3D* pour réaliser une acquisition précise de l'environnement de test sous la forme d'un nuage de points 3D colorisé. Ce scanner fixe permet de réaliser une cartographie très précise ($\pm 2mm$) avec une très forte résolution (inférieure au centimètre). Le nuage de points de 70 millions de points ainsi créé nous a servi de vérité terrain pour évaluer les nuages de points recréés avec notre approche sur les différentes trajectoires réalisées (Figure 5.7).

C'est en comparaison avec ces vérités terrain que nos algorithmes sont évalués grâce à des métriques détaillées dans la section suivante.

5.1.2.2 Critères d'évaluation et métriques

Notre concept a pour objectif de répondre aux contraintes présentées dans le paragraphe 1.1.3 page 5. Aussi, nous évaluons notre système sur deux types d'environnement : l'environnement routier au travers du jeu de données *Kitti* et en environnement naturel grâce à notre acquisition dans le

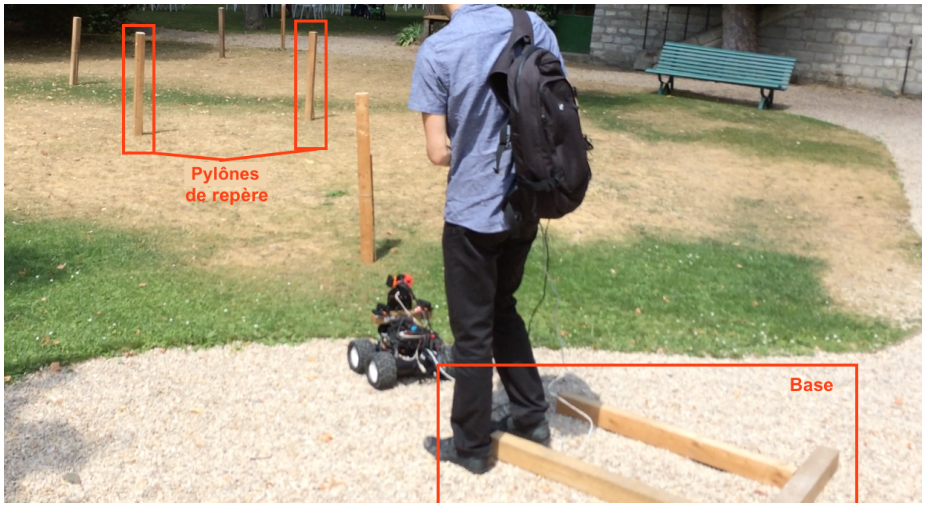


FIGURE 5.7 – Acquisition par le robot en cours. L'opérateur contrôle le robot avec une manette XBOX 360. Il est noté que la plateforme est reliée par un câble Ethernet à l'opérateur en raison d'un problème de connectivité WiFi. Mais, les contrôles se font « à distance » aussi faible soit-elle. Sur cette image on voit en détail à quoi correspondent les bases ainsi que les pylônes de repère (amers).

jardin des MINES ParisTech. La robustesse dans ces environnements est mesurée grâce à deux évaluations : la trajectoire calculée par le système d'odométrie et la distance entre les nuages de points reconstruits et ceux de la vérité terrain. Par ailleurs, nous avons aussi mesuré l'impact calculatoire de la méthode au travers de la charge CPU et de la mémoire RAM utilisée lors de l'exécution des traitements.

Pour chaque critère nous donnons aussi la métrique nous permettant de comparer notre méthode à la vérité terrain.

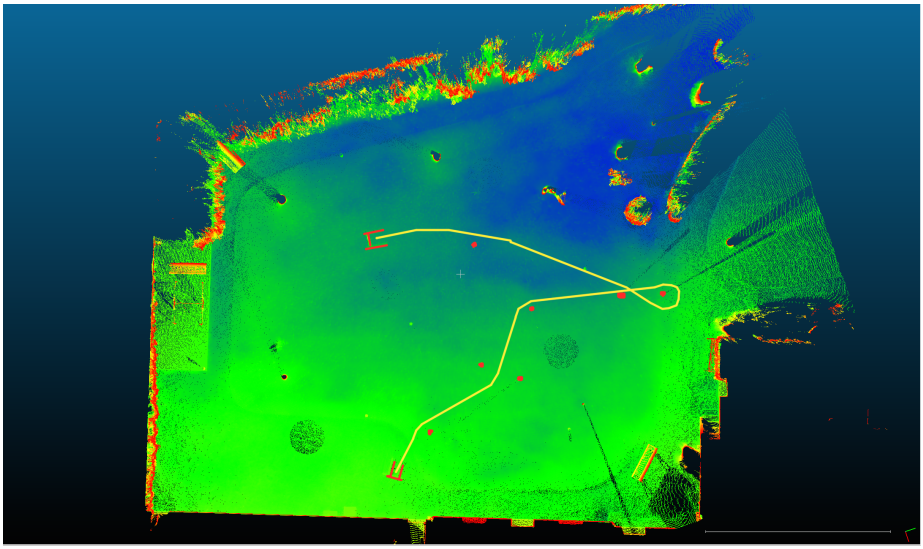
La trajectoire Premièrement, afin d'évaluer la qualité de l'odométrie visuelle avec notre caméra et dans notre environnement, nous comparons la trajectoire calculée par *LibViso 2* à la trajectoire de la vérité terrain. Cette comparaison permet d'évaluer les performances de *LibViso 2* en mètres. Ce critère permet, en plus de valider les performances de l'odométrie visuelle, de donner par la suite des éléments d'explication de la qualité de la reconstruction de l'environnement.

De surcroît, l'évaluation de l'odométrie est réalisée sur notre jeu de données mais aussi sur le jeu de données *Kitti* pour évaluer d'une manière plus universelle la qualité de l'odométrie et valider la robustesse de notre approche à tout type d'environnements.

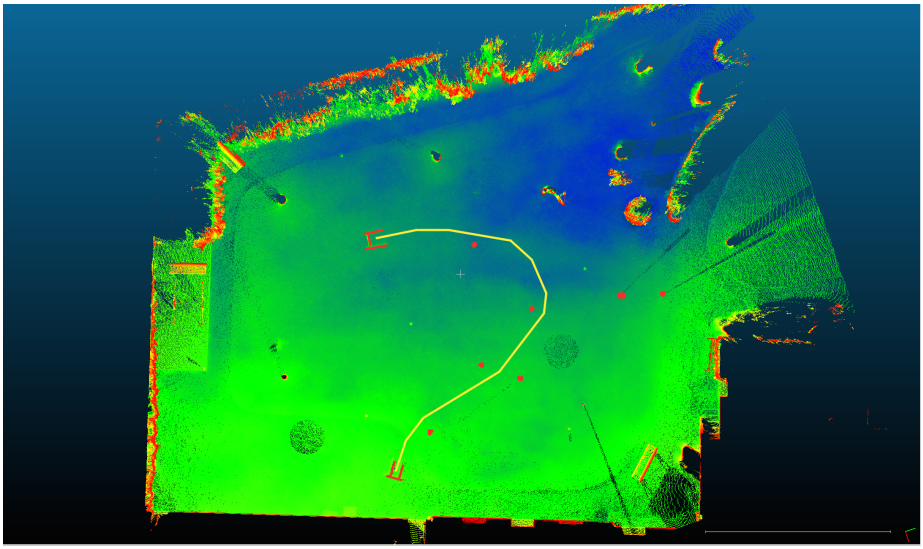
Nous avons réalisé sept jeux de données séparées en deux groupes. Le premier groupe comprend les acquisitions à faibles vitesses ($1m/s$) sur une distance de 40 mètres et le deuxième groupe comprend les acquisitions réalisées à des vitesses supérieures ($5m/s$) sur 25 mètres (Figure 5.8).

Kitti [Geiger et al., 2012] est un jeu de données stéréoscopiques dont l'acquisition s'est faite sur route. Il comprend 10 jeux de données d'évaluation de 25 000 images qui sont fournis avec les positions relatives géolocalisées où ont été prises les images par rapport à la position de départ. Ces données bien qu'intéressantes pour l'environnement routier ne sont pas pleinement représentatives de notre application et reflètent d'une situation idéale.

La métrique utilisée pour la comparaison en trajectoire est la distance euclidienne des points d'arrivée. En considérant $P_{a,v}$ le point d'arrivée de la vérité terrain, $P_{a,e}$ le point d'arrivée estimé



(a) En jaune est indiqué le circuit réalisé à faible vitesse ($1m/s$) dans les deux sens.



(b) En jaune est indiqué le circuit réalisé à haute vitesse ($5m/s$) dans les deux sens.

FIGURE 5.8 – Acquisition réalisée avec le scanner laser *FARO*. De plus, sont indiquées en rouge les bases de départ et d'arrivée ainsi que les pylônes de repère. Sur les deux images sont aussi indiqués les circuits réalisés au cours de l'expérience.

et D la distance totale parcourue, on obtient les erreurs E_{traj} par :

$$\begin{aligned} E_{traj} &= ||P_{a,v} - P_{a,e}||, \text{ en mètres} \\ E_{traj,p} &= \frac{||P_{a,v} - P_{a,e}||}{D}, \text{ en pourcentage} \end{aligned} \tag{5.4}$$

C'est une métrique de distance exprimée en mètres et en pourcentage par rapport à la distance parcourue. À ce stade, cette métrique nous permet de visualiser clairement si la méthode est efficace ou non et le pourcentage nous permet d'évaluer la méthode sur des jeux de données de longueurs différentes. Pour cette métrique la référence est le point de départ de la séquence.

Pour le jeux de données du jardin des Mines, des métriques permettent d'évaluer l'intégralité de la trajectoire, mais les trajectoires de notre vérité terrain n'étant pas géoréférencées, ces techniques ne semblent pas pertinentes.

La distance entre les nuages de points Deuxièmement, le nuage de points de la reconstruction d'environnement acquis par la plateforme est comparé au nuage de points acquis par le scanner laser *FARO Focus 3D*. La métrique utilisée est la distance moyenne séparant les nuages de points reconstruits (en mètres). Cette comparaison est réalisée en calculant les distances séparant les deux nuages de points, point à point. Cette métrique correspond à la moyenne des distances de chacun des points du nuage reconstruit avec respectivement un point (le plus proche) de l'autre nuage (vérité terrain). En considérant $P_{v,i}$ le i^{ème} point du nuage de point reconstruit, $P_{e,\text{proche}}$ le point le plus proche associé du nuage de point de la vérité terrain et N le nombre de points du nuage à évalué, l'erreur de distance point à point E_{pp} est définie telle que :

$$E_{pp} = \frac{1}{N} \sum_{i}^{N} ||P_{v,i} - P_{e,\text{proche}}||, \text{ en mètres} \tag{5.5}$$

Pour cette métrique la référence est aussi le point de départ de la séquence.

Le défaut de cette métrique est qu'elle est sensible à la répartition des points dans les nuages de points. En effet comme l'illustre la figure 5.9, avec une répartition différente entre deux nuages de points identiques, on obtiendra une distance non nulle car les points du premier nuage ne seront pas forcément juxtaposés avec les points de l'autre nuage, et ceux même avec deux nuages identiques.

Dans notre cas, ce problème n'a pas de réel impact. Effectivement, nous l'utilisons pour visualiser de grandes différences entre des nuages de points où la résolution n'a pas d'impact. Par contre, Cette méthode ne pourra pas être utilisée pour comparer deux nuages de grande précision.

Charge CPU et mémoire RAM Troisièmement, nous présentons la charge CPU et la charge en mémoire RAM de la machine lors du lancement de la méthode sur le rejeu des données.

Pour ce critère d'évaluation, la métrique utilisée est le pourcentage d'utilisation des CPU et le nombre de mega-octets utilisés par les différentes méthodes.

5.1.2.3 Résultats

La trajectoire Nous évaluons l'odométrie *LibViso 2* sur les dix jeux de données offerts par *Kitti*. De même, nous l'évaluons sur les sept jeux de données que nous avons acquis.

Les résultats (Tableau 5.3) montrent une erreur de 8% sur la trajectoire lorsque la vitesse de progression du robot est faible (inférieure à 1 mètre par seconde). À vitesse plus rapide (supérieure à 5 mètres par seconde), l'erreur passe à 66%. Ces résultats sont éloignés des résultats sur les jeux de données *Kitti* où une erreur de seulement 2.47% apparaît.

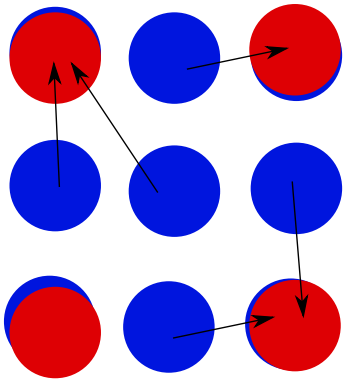


FIGURE 5.9 – Schéma démontrant le problème intrinsèque à la méthode de la distance point à point. La distance ne peut pas être nulle en fonction de la résolution des nuages, une distance peut intervenir même si les deux plans sont identiques.

Erreur moyenne en trajectoire	en mètres	en %
Données Kitti Séquences (0 à 9) à plus de $5m/s$	$\sim$ 1m pour 40m	2.47%
Notre jeu de données Sur 40 mètres à $\sim 1m/s$ Sur 24.5 mètres à $\sim 5m/s$	 $\sim$3 mètres $\sim$16 mètres	 8% 66%

TABLE 5.1 – Résultats en trajectoire donnant l'erreur finale de position calculée selon le principe de la distance euclidienne.

Malgré ces résultats, cette expérience vise à prouver l'intérêt d'une telle méthode pour la reconstruction d'environnement dans le but de faire de la détection d'obstacles devant le véhicule. Comme le montre la figure 5.10, l'approche à faible vitesse s'avère être à moins de 1% d'erreur avant le virage et on retrouve les différents changements d'orientation même s'ils sont décalés. Par conséquent, la cohérence globale de la trajectoire est présente. Par contre à plus haute vitesse, l'erreur est telle qu'elle ne peut pas être utilisée par la suite pour construire un nuage de points (Figure 5.11).

De plus, la différence avec les résultats de *Kitti* s'explique par différents facteurs. Dans *Kitti*, le système n'est pas le même, ils n'utilisent pas les mêmes caméras, la *baseline* de leur système stéréoscopique est plus large (54cm) et il est situé plus haut (1,64m). En outre, les jeux de données sont réalisés sur des routes où la progression est régulière et stable. Enfin, le calcul de l'odométrie est réalisé image par image, contrairement à la situation réelle où les images arrivent en flux continu et certaines images ne peuvent pas être prises en compte. Ce dernier point est important car il reflète un des principaux défauts de *LibViso 2*, sa vitesse d'exécution. Ainsi, les résultats offerts par les données *Kitti* offrent un aperçu des résultats atteignables dans une situation idéale, mais nos jeux de données montrent la réalité de la situation.

Au vu de cette différence entre les deux vérités terrain, le taux d'erreurs semble lié à la fréquence de calcul des odométries. En effet, à ce stade l'algorithme calcule seulement six odométries par seconde soit une odométrie toutes les deux à trois images. Or, plus le nombre d'odométries calculées est grand plus la précision entre les positions est forte. De plus, cette limite peut s'avérer gênante quand le véhicule exécute une trajectoire relativement rectiligne, mais lorsqu'il effectue des virages importants comme sur nos jeux de données, ce problème n'est qu'augmenté. Ainsi, il nous semble évident que pour améliorer notre approche il faille améliorer la vitesse de calcul de l'algorithme d'odométrie *LibViso 2* surtout pour l'estimation des rotations.

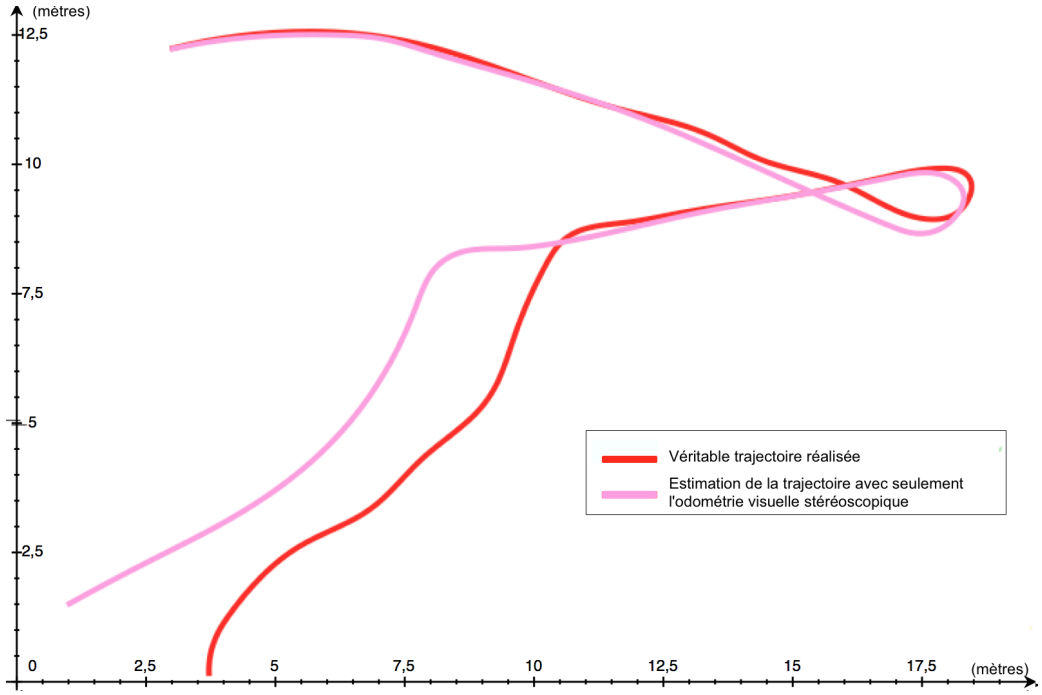


FIGURE 5.10 – Comparaison des trajectoires sur notre jeu de données à basse vitesse. En rouge, le véritable chemin réalisé par le robot dans le jardin. En rose, la trajectoire calculée par l'algorithme *Libviso 2*.

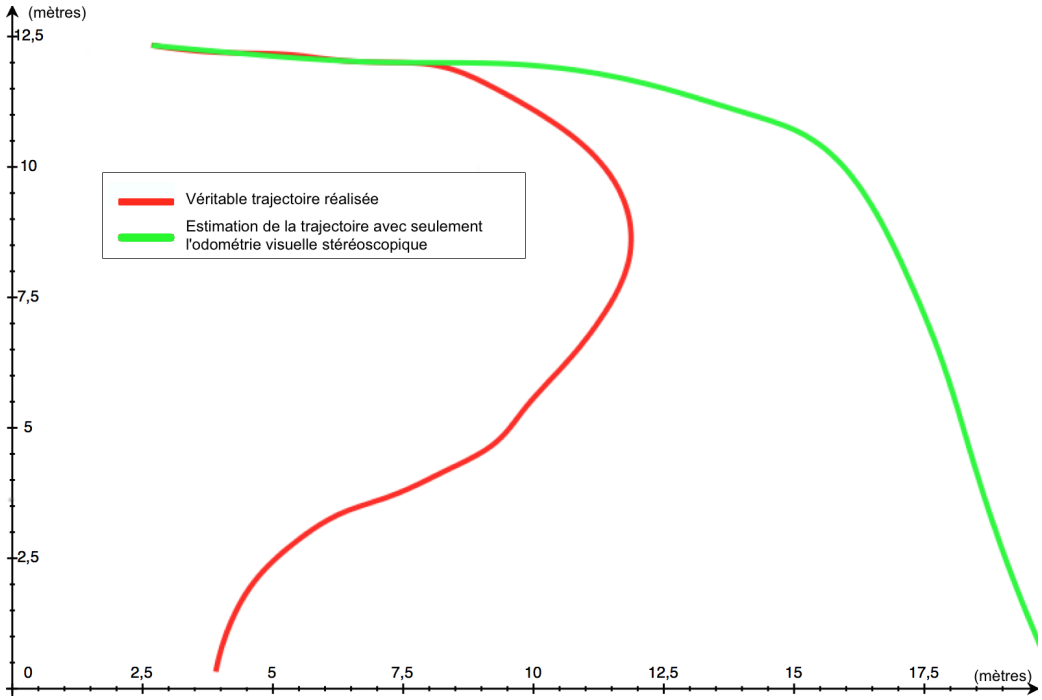


FIGURE 5.11 – Erreur de calcul de trajectoire quand le robot roule à haute vitesse. En rouge, le véritable chemin réalisé par le robot dans le jardin. En vert, la trajectoire calculée par l'algorithme *LibViso 2*.

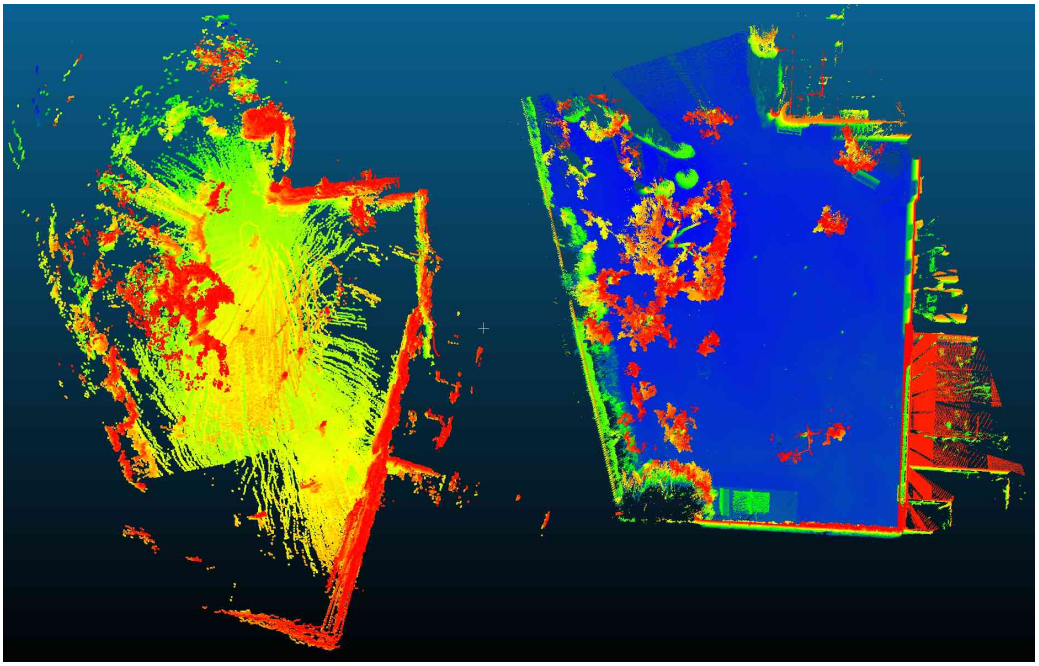


FIGURE 5.12 – Comparaison des nuages de points reconstruits. Le nuage de point reconstruit (gauche) à côté de celui de la vérité terrain (droite). On constate que la reconstruction n'est pas parfaite. Bien que la structure s'apparente à celle de la vérité terrain, on voit clairement une cassure dans le mur à droite liée à l'erreur d'odométrie.

La distance entre les nuages de points La précision de la reconstruction 3D de l'environnement est dépendante de la précision du calcul de trajectoire car elle est construite en fonction des déplacements estimés par l'odométrie.

Afin de qualifier la reconstruction créée nous avons calculé la distance moyenne entre les points du nuage de points de vérité terrain et celui reconstruit. Cette distance est mesurée en calculant pour chaque point du nuage de vérité terrain, la distance avec le point le plus proche du nuage de points reconstruit. Pour réaliser cette étude, nous avons comparé les nuages en utilisant comme référence la base de départ de l'acquisition.

En outre, nous ne nous sommes intéressés qu'aux nuages construits à faible vitesse car les 66% d'erreur moyenne pour les autres ne permettaient pas de garder de cohérence globale, donc comparer les nuages point à point n'a pas de sens.

Ainsi, la distance moyenne obtenue entre les nuages de points reconstruits et le nuage de points de vérité terrain est de **50 cm** en moyenne avec un écart type de $0,55cm$. Ce résultat est bien la preuve que la cohérence est présente même avec une erreur finale de 8% sur la trajectoire (Figure 5.12).

Par contre, pour une reconstruction visant à permettre la détection d'obstacles, 50 cm d'écart en moyenne est bien trop élevé. En effet, comme nous l'avons défini dans le chapitre 2.4.2.1 page 47, un objet devient un obstacle quand sa taille dépasse le rayon des roues. Même pour un véhicule militaire, une différence de 50 cm sur la taille d'un objet est problématique quand il faut le surmonter.

Finalement, il apparaît qu'à ce stade la reconstruction établie n'est pas suffisante pour notre application.

Charge CPU et mémoire RAM Un véhicule militaire est équipé de machines sur lesquelles un certain nombre d'applications sont lancées. En effet, ces machines servent à la visualisation des

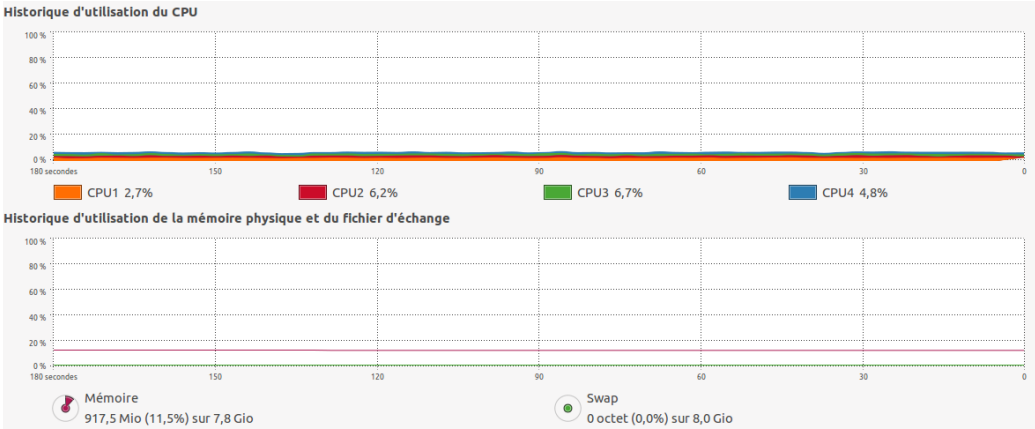


FIGURE 5.13 – Situation de référence sur notre machine pour ce qui est de la consommation CPU et de la charge mémoire. On constate que les quatre cœurs virtuels sont disponibles.

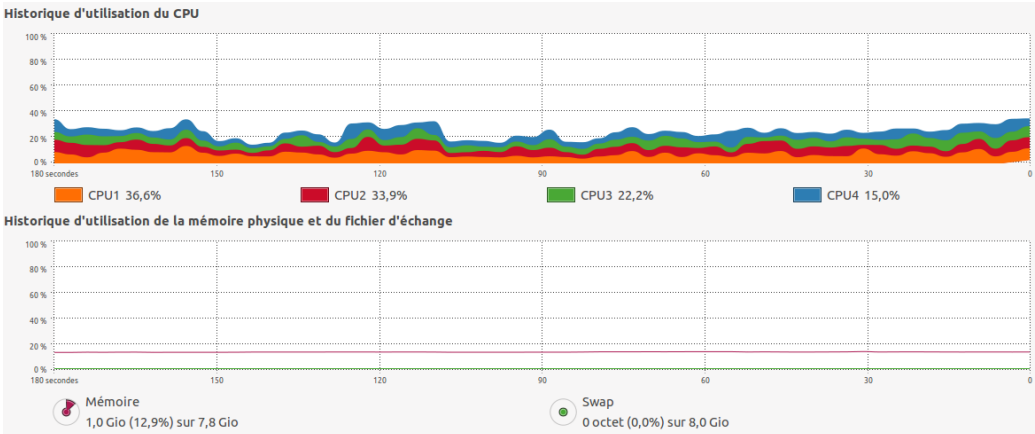


FIGURE 5.14 – Situation de la machine quand l'odométrie *LibViso2* est lancée. La charge est repartie sur les différents cœurs virtuels de la machine pour atteindre une consommation moyenne de $26,9\%$ de charge CPU contre $5,1\%$ en situation de référence. Concernant la charge en mémoire le lancement de l'algorithme consomme $82,5$mo soit 1.4% de la mémoire totale.

caméras, la situation GPS, et le lancement des autres aides au pilotage (chapitre 2.1.2 page 21). Par conséquent notre algorithme ne doit pas consommer l'intégralité des ressources de la machine pour permettre le lancement en parallèle des autres applications.

Pour évaluer la consommation de notre algorithme, nous avons pris comme référence notre machine dans un cadre nominal, c'est à dire au démarrage. Comme le montre la figure 5.13, en situation nominale la consommation n'est pas nulle. La consommation CPU est aux alentours de $5,1\%$ et la mémoire RAM est utilisée à hauteur de $917,5$mo soit 11.5% de la mémoire totale.

En comparaison, nous avons capturé la consommation lorsque l'algorithme *LibViso2* est lancé. Nous voyons sur la figure 5.14 que la consommation est répartie sur les différents cœurs virtuels de notre machine jusqu'à atteindre une consommation moyenne de $26,9\%$ et une charge mémoire de $82,5$ mo soit $1,4\%$. Cette consommation n'est pas très importante.

Ainsi, au niveau de la consommation, l'algorithme ne pose pas de problème particulier et répond correctement à nos contraintes.

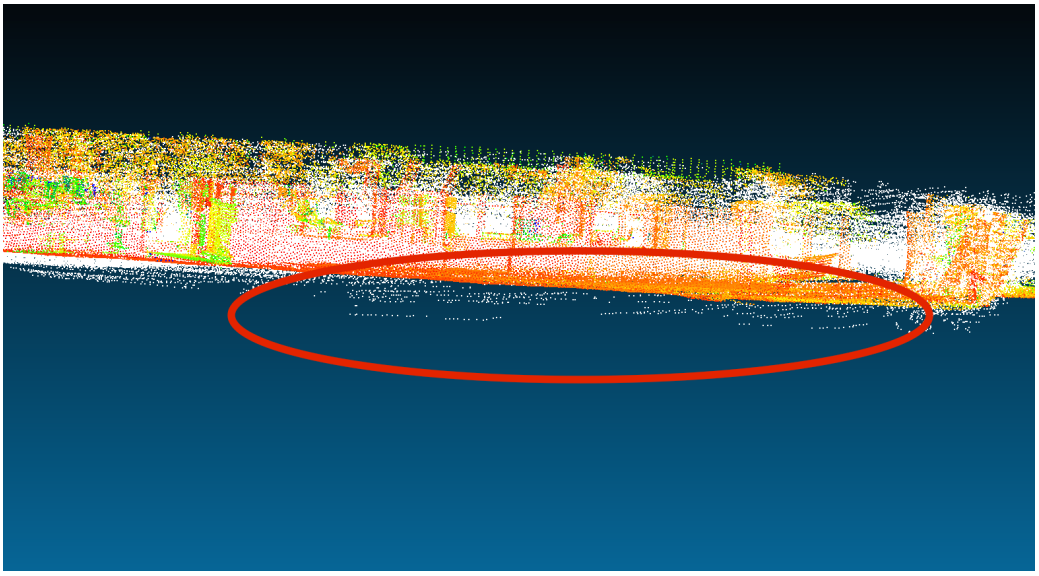


FIGURE 5.15 – Les erreurs causées par les vibrations. En blanc, sous le nuage de points coloré, on voit les points LIDAR laser mal replacés dans l'environnement qui « plongent » dans le sol.

5.1.2.4 Limites de cette approche

En attestent les résultats de cette expérience, cette approche n'est pas pleinement satisfaisante. Dans cette sous-section nous détaillons certaines limites apparues à la suite de ces expériences et les solutions pour y remédier.

Vitesse de déplacement La première et principale limite de cette approche est la vitesse de déplacement possible. En effet, la section précédente a montré qu'à des vitesses supérieures à $1m/s$ l'odométrie subissait une erreur très importante. De plus, notre système LIDAR ne tournant qu'à $90°/s$ et n'acquérant des données qu'à une distance de 30 mètres, la vitesse admissible pour une reconstruction d'une résolution suffisante est limitée.

Ce problème de vitesse de déplacement possède plusieurs solutions. Pour le problème de résolution du LIDAR, le système final devra remplacer le servomoteur par un système plus performant comme par exemple une roue codeuse qui est équipée d'un boîtier électronique renvoyant continuellement la position de la roue et pouvant résuire le tant nécessaire à l'acquisition des positions. De plus, la fréquence du LIDAR pouvant devenir limitante si le système de rotation accélère, il est envisageable de réduire l'amplitude de la rotation pour se concentrer sur une zone frontale plus restreinte.

Par ailleurs, pour régler le problème de l'erreur liée à l'odométrie, nous pensons qu'accélérer l'algorithme qui la calcule pourrait permettre d'améliorer les résultats. La section suivante va présenter une optimisation de cet algorithme qui améliore ses performances et qui améliore par la même occasion la qualité de la reconstruction.

Vibrations Actuellement, le nombre d'odométries par seconde implique des écarts importants entre les positions calculées. Dans le cas de notre plateforme, qui est légère est basse, un grand nombre de vibrations et de sursauts interviennent. Par conséquent, les scans acquis ne sont pas tous alignés les uns aux autres et leur placement sur le vecteur séparant deux positions provoque des aberrations dans la reconstruction. En effet, un scan capturé lors d'un sursaut n'aura pas la même orientation qu'un scan acquis avant le sursaut, mais lors du replacement ces deux scans seront

replacés comme ayant été pris avec la même orientation. On retrouve dans la reconstruction des scans qui « plongent » sous terre et d'autres qui « sortent » de terre (Figure 5.15).

Ces erreurs de reconstruction sont gênantes pour l'application que nous souhaitons faire de cette reconstruction. Effectivement, la détection d'obstacles nécessite que les scans soient correctement placés dans l'environnement sans quoi aucune analyse ne pourra en être faite.

Comme pour la vitesse de déplacement, une solution est l'amélioration de la vitesse de calcul de l'odométrie. Cette accélération permettrait d'obtenir plus de positions, donc un positionnement plus précis des scans qui éviterait les aberrations sus-citées.

5.1.3 Conclusion

La section qui précède présente l'approche algorithmique choisie pour reconstruire l'environnement en trois dimensions en utilisant seulement le positionnement relatif visuel stéréoscopique et l'acquisition 3D par un LIDAR 2D tournant. Pour évaluer notre approche, nous avons réalisé une expérience utilisant un robot mobile équipé d'une caméra stéréoscopique et d'un LIDAR tournant. De plus, l'acquisition de notre terrain de test avec un scanner laser fixe très précis nous offre une vérité terrain servant de référence à nos comparaisons.

Cette approche très simple dans son fonctionnement offre des résultats intéressants. En effet, malgré une vitesse de déplacement très faible et un taux d'erreur encore relativement important il apparaît qu'une cohérence globale se dégage des trajectoires calculées à faible vitesse et qu'ainsi, la reconstruction n'est pas très éloignée de la vérité. Malgré tout, la précision de cette reconstruction ne répond pas aux contraintes de notre application.

Enfin, les limites de cette approche ont été développées et les solutions pour y remédier convergent vers la nécessité d'accélérer la méthode de calcul d'odométrie surtout dans le cas des rotations afin de limiter les erreurs de reconstruction. Ainsi, dans la section suivante nous détaillons l'optimisation apportée à cette approche.

5.2 Optimisation par ajout d'une estimation de la rotation

Au vu des résultats précédent, nous avons tenter d'améliorer notre algorithme en initialisant *LibViso 2* avec un autre algorithme afin de l'aider dans ses estimations en tentant de le faire converger vers une solution plus rapidement.

Cette section présente cet algorithme d'optimisation au travers d'une explication générale puis le détail des deux étapes qui la compose : le calcul du déplacement dans l'image et le calcul du mouvement de la plateforme qui en découle. Ensuite nous étudions les résultats obtenus en comparaison avec la méthode stéréoscopique au niveau du temps de calcul et en trajectoire sur les jeux de données *Kitti*. Puis, nous nous intéressons aux résultats obtenus sur nos jeux de données en trajectoire, en reconstruction et en charge.

5.2.1 Initialisation par estimation de la rotation

À l'image du schéma fonctionnel (Figure 5.16), la nouvelle approche proposée diffère dans les prétraitements réalisés. En effet, après l'acquisition des images, nous réalisons indépendamment, sur l'image gauche et l'image de droite, notre pré-traitement. Les résultats de cet algorithme sont envoyés à *LibViso 2* comme paramètres d'initialisation du mouvement. Cette initialisation permet de réduire le nombre d'itérations nécessaires à *Libviso 2* pour converger vers la solution. Nous appliquons ces traitements sur les deux images afin de vérifier la répétabilité des résultats et ainsi valider notre estimation.

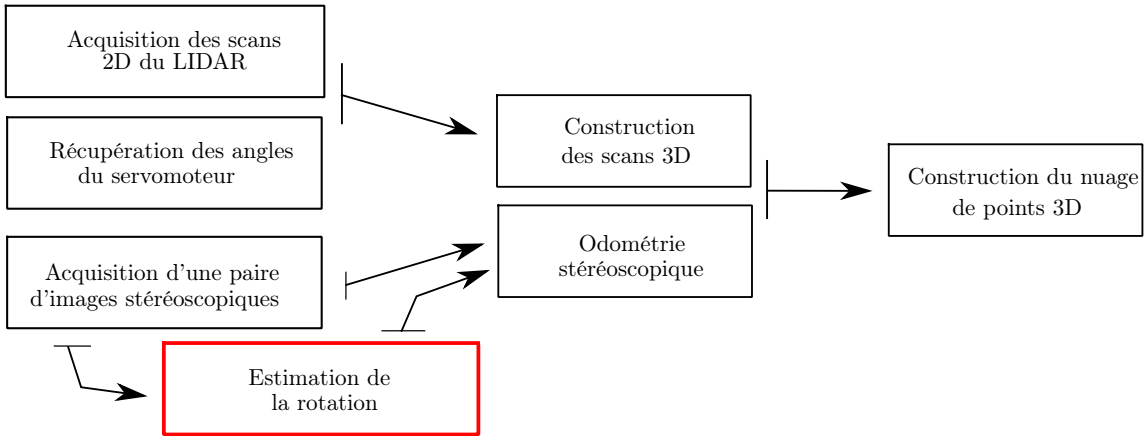


FIGURE 5.16 – Schéma fonctionnel de notre approche avec l'optimisation. Elle se compose en trois parties : l'acquisition de données, le prétraitement et la construction de la cartographie 3D de l'environnement. Désormais le prétraitement odométrique se compose en deux parties : l'estimation de la rotation puis l'odométrie stéréoscopique initialisée par ce pré-traitement.

Pour la réalisation de cet algorithme, nous avions deux contraintes. Premièrement l'algorithme devait être assez rapide pour pouvoir être lancé en amont de *Libviso 2* et être plus rapide que celui-ci. Deuxièmement, l'algorithme devait être suffisamment précis pour pouvoir initialiser *Libviso 2* correctement et réduire le nombre d'itérations nécessaires pour obtenir une odométrie précise.

Nous avons choisi d'utiliser une méthode d'estimation de la rotation, pour supprimer le temps nécessaire aux appariements. Pour estimer le déplacement entre les images, nous avons choisi d'utiliser la méthode du recalage d'imagettes (expliquée dans le chapitre 2.2.2.2 page 29) par corrélation croisée normalisée. Cette méthode a l'avantage d'être relativement peu coûteuse en temps de calcul tout en étant efficace. En effet son efficacité est prouvée lorsqu'elle est utilisée comme odométrie visuelle [Wirbel et al., 2013; Wirbel, 2014].

Cette odométrie se compose de deux étapes : l'algorithme débute en estimant le déplacement en pixels entre deux images acquises à deux instants consécutifs. Puis ce déplacement est utilisé pour estimer la rotation effectuée par la caméra entre ces deux instants. Comme l'explique la figure 5.17, par cette méthode on cherche la transformation T_R qui lie le repère de la caméra à l'instant t puis à l'instant $t+1$ en observant le déplacement T_d des objets dans l'image.

Ainsi, la prochaine sous-section détaille le calcul du déplacement d'une image entre deux instants, suivra le détail du calcul de l'estimation du mouvement réalisé par la caméra.

5.2.2 Etude du déplacement d'une image entre deux instants

Notre algorithme d'optimisation se compose en deux parties, l'estimation du déplacement dans l'image puis nous en déduisons le mouvement réalisé dans le repère monde. Ici, nous détaillons la première étape.

Lorsqu'une caméra bouge et filme une scène fixe, la scène se déplace du point de vue de la caméra et ce déplacement se répercute sur l'image. L'étude du mouvement de cette scène peut être réalisée de plusieurs manières déjà expliquées au paragraphe 2.2.2.2 page 29. Nous avons choisi la méthode du « recalage d'imagettes » aussi détaillée au paragraphe 2.2.2.2, par le calcul de la corrélation croisée normalisée pour l'estimation du déplacement dans le plan image. Cette estimation nous permet ensuite de déduire le mouvement réalisé par la caméra. Le déplacement dans le plan image correspond à un vecteur à deux dimensions exprimé en pixels séparant les deux images.

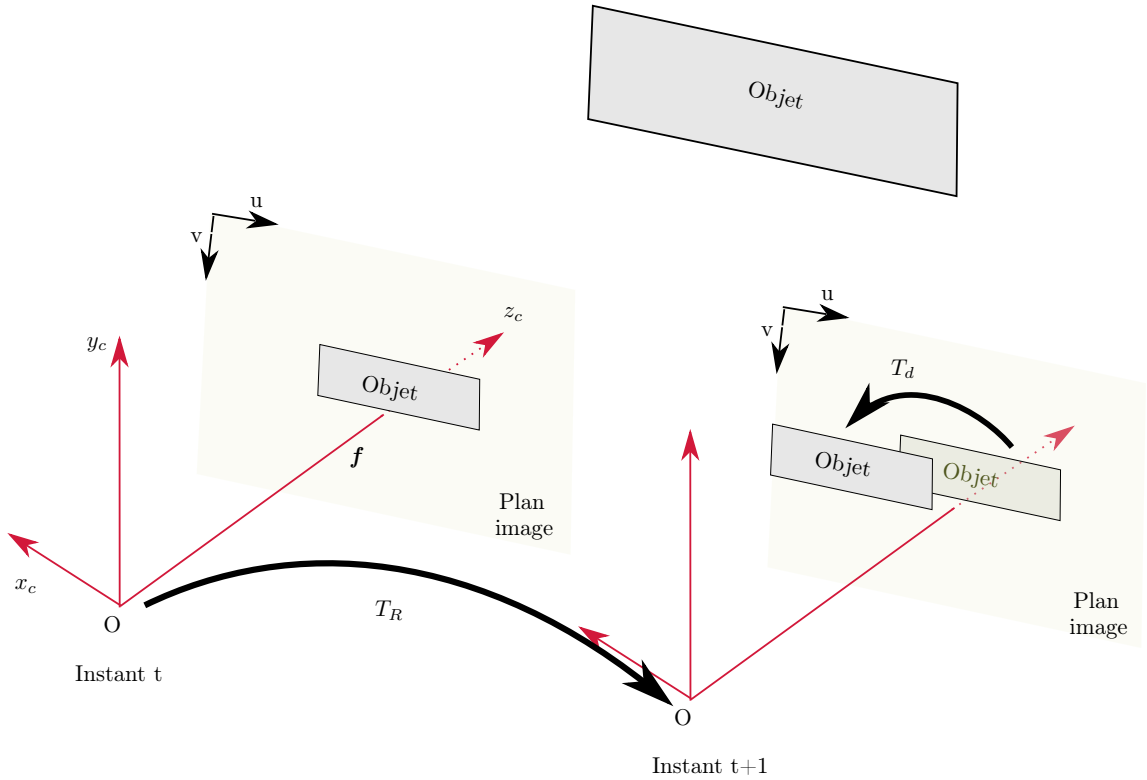


FIGURE 5.17 – Schéma d'un mouvement réalisé par une caméra filmant un objet. T_d représente le déplacement de l'image et T_R le mouvement de la caméra. Ces deux transformations sont liées.

Corrélation croisée normalisée L'utilisation de la corrélation croisée normalisée n'est pas nouvelle pour réaliser de recalage d'imagettes [Lewis, 1995], mais ses performances et sa simplicité en font une méthode efficace est peu coûteuse en temps de calcul.

La corrélation croisée normalisée permet de déterminer la position (u_{pos}, v_{pos})(position en pixels) du centre d'une imagette t dans une image 2D I.

Soit $f(x, y)$ l'intensité [6] de l'image I de taille $M_x \times M_y$ au point (x, y) en niveau de gris. L'imagette T est une image plus petite que l'image I, son intensité au point (x, y) est dénotée $t(x, y)$ et elle est de taille $N_x \times N_y$. d est la zone de recouvrement de T et D est la zone de recherche dans laquelle T va être recherchée (Figure 5.18). Pour calculer (u_{pos}, v_{pos}), la corrélation croisée notée γ va être calculée pour tous les points (x, y) de d pour f et pour t. De plus, d va être déplacée pour recouvrir l'ensemble des pixels de D au moins une fois, la position de d est notée (u, v). Enfin, $\hat{f}_d$ est la moyenne des intensités de l'image dans d et $\hat{t}$ est la moyenne des intensités de l'imagette.

Ainsi, la corrélation croisée normalisée γ est calculée grâce à la formule :

$$\gamma = \sum_{u,v \in D} \frac{\sum\limits_{x,y \in d} (f(x,y) - \hat{f}_d)(t(x,y) - \hat{t})}{\sqrt{\sum\limits_{x,y \in d} (f(x,y) - \hat{f}_d)^2 \sum\limits_{x,y \in d} (t(x,y) - \hat{t})^2}} \tag{5.6}$$

6. L'intensité d'un pixel en niveau de gris (8 bit) représente son intensité lumineuse est correspond à une valeur comprise entre 0 et 255.

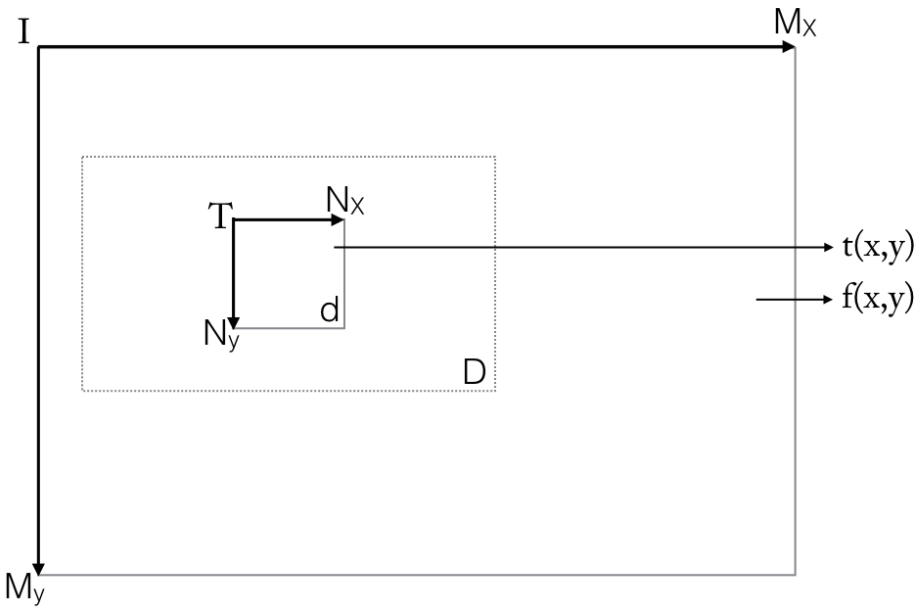


FIGURE 5.18 – Une image I de taille $M_x \times M_y$ et son imagette T de taille $N_x \times N_y$. L'intensité de I au point (x, y) est notée $f(x, y)$ et l'intensité de T au même point est notée $t(x, y)$. La zone d représente la zone dans laquelle vont être comparées t et f afin d'obtenir la corrélation croisée normalisée. Cette comparaison va être réalisée un certain nombre de fois jusqu'à ce que d est recouvert entièrement D.

Par ailleurs, $\hat{f}_d$ et $\hat{t}$ sont définis par :

$$\begin{aligned} \hat{f}_d &= \frac{1}{d} \sum_{x,y \in d} f(x, y) \\ \hat{t} &= \frac{1}{d} \sum_{x,y \in d} t(x, y) \end{aligned} \tag{5.7}$$

Le dénominateur de l'équation (5.6) la rend insensible aux changements d'illumination et de contraste car, il normalise le calcul sur l'ensemble de l'image. Ainsi, même si une image est plus claire que la précédente, la normalisation permet que l'image et l'imagette aient des valeurs comparables.

La position recherchée de l'imagette (u_{pos}, v_{pos}) est obtenue en calculant la position où l'on obtient la valeur maximale de γ (Figure 5.19). En effet, cette valeur maximale représente l'endroit où la similitude est la plus élevée entre l'imagette et l'image. Cette position est ensuite comparée à la position de l'imagette dans l'image précédente. Cette comparaison permet d'obtenir le vecteur de déplacement de l'imagette noté $\begin{bmatrix} d_u & d_v \end{bmatrix}^T$.

Ce calcul peut être réalisé sur plusieurs imagettes afin d'augmenter la chance d'une bonne estimation du déplacement entre deux images. Dans notre cas, ce calcul est réalisé pour 10 imagettes tirées aléatoirement dans l'image (Figure 5.20). Ces imagettes ont une taille $N_x \times N_y$ de 64×64 pixels. De plus, nous avons défini $M_x \times M_y$ à 128×128 pixels. Effectivement, les mouvements n'étant pas très importants entre deux images, il n'est pas nécessaire d'avoir une zone de déplacement de l'imagette très grande.

De surcroît, nous réalisons cette estimation sur l'image gauche et l'image droite pour comparer les résultats obtenus et les rejeter en cas d'anomalie. En effet, bien que les résultats ne soient pas censés être identiques entre les deux images, le sens et l'ordre de grandeur doivent correspondre. Si les résultats ne concordent pas, nous réitérons les calculs. Ce test permet de vérifier la répétabilité de l'estimation entre les deux images et ainsi contrer des estimations aberrantes.

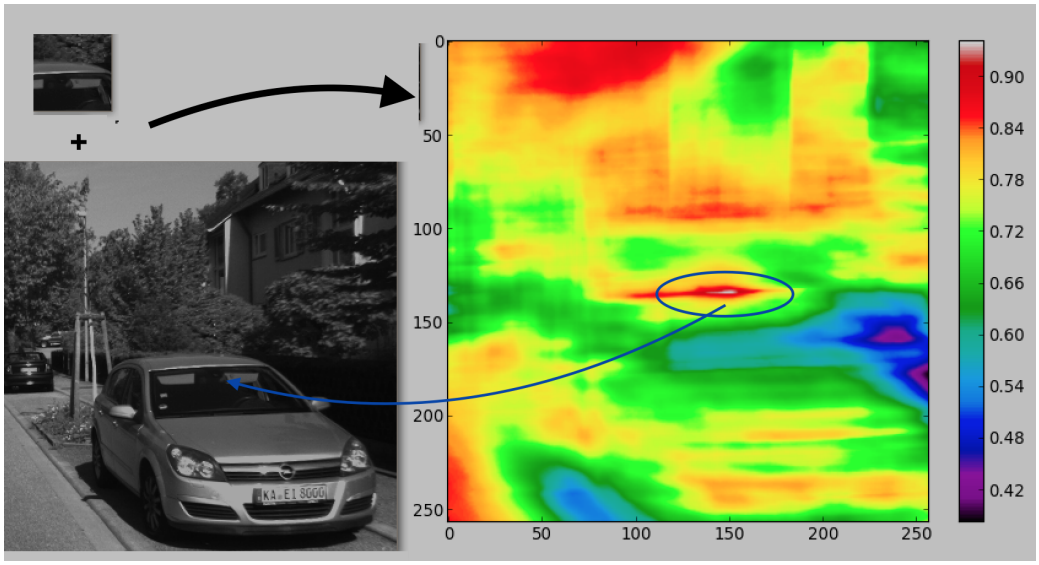


FIGURE 5.19 – La mise en correspondance d'une imagette avec une image donne par le calcul de la corrélation croisée normalisée l'emplacement où l'on retrouve cette imagette dans l'image. En effet, l'étude de l'image de corrélation permet de trouver la zone dont l'imagette semble appartenir. Une couleur chaude indique une ressemblance forte et une couleur froide une faible ressemblance. L'image subissant le traitement est tirée du jeu de données *Kitti*

FIGURE 5.20 – Cette figure montre les emplacements des imagettes choisies aléatoirement (carrés rouges) dans l'image précédente qui ont été repositionnées (carrés verts) dans l'image courante. L'image est extraite du jeu de données *Kitti*

Enfin, toutes les imagettes ne sont pas viables pour réaliser le calcul de déplacement. Si une imagette est extraite d'une partie homogène d'une image, le calcul de déplacement qui va en suivre en sera altéré.

Homogénéité Lors du choix aléatoire des imagettes il est important que celles-ci soit viables. En effet, si des imagettes sont homogènes (pixels de couleur uniforme), le calcul de la corrélation croisée normalisée ne pourra pas donner de valeur cohérente. La figure 5.21 illustre les erreurs liées à l'estimation des déplacements quand interviennent des imagettes homogènes. Les positions des imagettes dans l'image précédente sont indiquées en rouge et en violet sont indiqués leur positionnement dans l'image actuelle. On constate que sur la zone sombre et homogène, les imagettes subissent un déplacement chaotique, alors que dans les autres zones, elles se déplacent de manière similaire les unes par rapport aux autres.

Pour régler ce problème, un test antérieur à l'estimation du déplacement est réalisé pour chaque imagette. Ce test consiste à comparer l'intensité du pixel du centre de l'imagette avec les pixels du

bord de l'imagette. Si tous ces pixels sont trop similaires (différence entre les intensités inféreieure à 20) au pixel central, l'imagette est rejetée et une nouvelle imagette est sélectionnée aléatoirement.

Ainsi, grâce au déplacement obtenu avec cet algorithme nous pouvons estimer le mouvement de rotation réalisé par la caméra. La sous-section suivante détaille les principes mathématiques permettant l'estimation du mouvement.

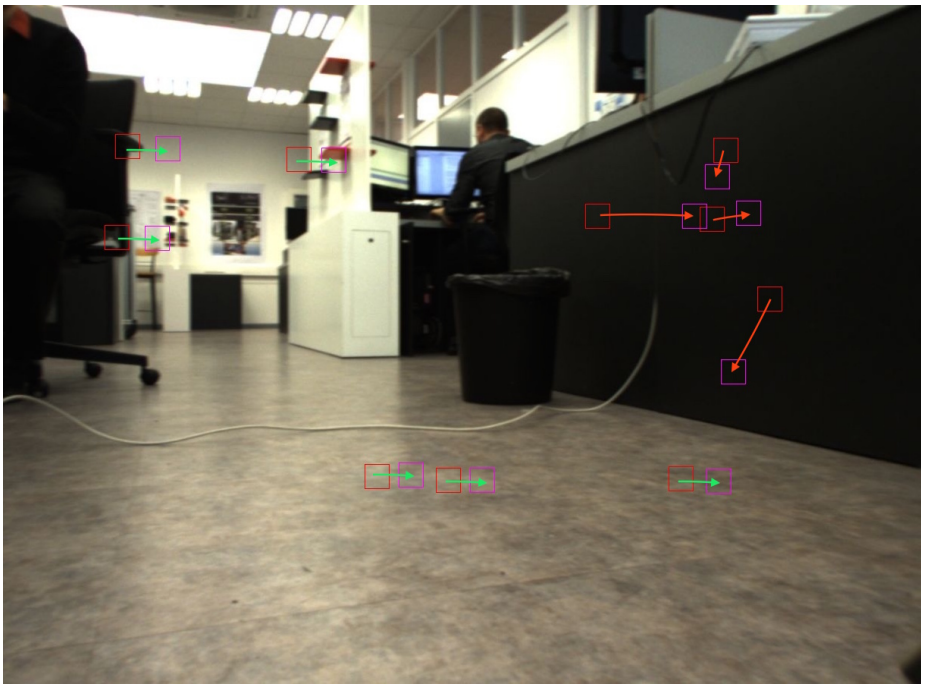


FIGURE 5.21 – Cette figure montre les erreurs liées au choix d'imagettes homogènes. En rouge les positions des imagettes dans l'image précédente et en violet, leur positionnement dans l'image actuelle. Les flèches permettent de voir les correspondances entre les carrés.

5.2.2.1 Estimation de la rotation de la caméra à partir du déplacement de l'image

Comme énoncé précédemment, notre algorithme d'optimisation fonctionne en deux parties. La sous-section précédente détaille la première partie, le calcul du déplacement entre deux images. Cette sous-section explique la seconde partie, le calcul du mouvement de la caméra à partir de ce déplacement. En effet, la mise en correspondance de deux images permet d'évaluer la distance en pixels qui les sépare l'une de l'autre. Cette distance est relative au mouvement réalisé par la caméra entre les deux acquisitions d'image. Nous allons expliquer ce lien sur le plan mathématique. Suivra le détail des résultats obtenus avec l'application de cette optimisation dans l'algorithme complet.

Cette méthode monoculaire a pour but d'initialiser la méthode stéréoscopique. À ce titre, la méthode monoculaire doit donner une estimation rapide du mouvement réalisé par la caméra, c'est pourquoi nous avons appliqué trois approximations sur le mouvement que nous recherchons. Ces approximations sont justifiées dans l'annexe B page 171.

1. Les images étant acquises à des fréquences élevées (plus de 10 images par seconde), le mouvement entre chaque image est très faible. Ainsi, les rotations considérées entre les images peuvent être associées à des rotations suivant de « petits angles » et permettant d'utiliser les propriétés : $\cos\theta = 1$ et $\sin\theta = \tan\theta = \theta$. Ces rotations offrent des propriétés intéressantes simplifiant les estimations et sont une très bonne approximation de la réalité.
2. Les translations sont dépendantes de la distance des points considérés. Nous considérons les points comme étant à l'infini. Ainsi, les translations entre deux images peuvent être négligées.

3. Nous supposons le roulis négligeable. Bien que le roulis ne soit pas véritablement négligeable, cette approximation donne de bons résultats.

Rotations aux petits angles Détaillons le calcul avec ces hypothèses. Ce calcul nous permet d'estimer la rotation réalisée entre chaque image à partir du déplacement calculé par la mise en correspondance de ces images. Suivant les angles d'Euler, une rotation peut s'écrire comme le produit des rotations sur les différents axes d'un repère. De plus, le repère caméra est de type z devant, une rotation R s'écrit donc suivant :

$$\begin{aligned} R &= \text{Roulis} \cdot \text{Tangage} \cdot \text{Lacet} \\ R &= R_z(\theta_z) \cdot R_y(\theta_y) \cdot R_x(\theta_x) \end{aligned} \tag{5.8}$$

Or, d'après notre troisième hypothèse, nous négligeons le roulis qui s'apparente donc à une matrice identité. Ce roulis équivaut aux rotations autour de l'axe z. Ainsi, la rotation n'est plus composée que des deux rotations en tangage et en lacet suivantes :

$$R = R_y(\theta_y) \cdot R_x(\theta_x) \tag{5.9}$$

De plus, le détail de ces rotations suivant les angles θ_y et θ_x respectivement donne :

$$R_y(\theta_y) = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \text{ et } R_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix} \tag{5.10}$$

La première hypothèse considérant les angles comme des petits angles permet d'approximer les sinus et cosinus de ces angles. Le cosinus est alors équivalent à 1 et le sinus est équivalent à l'angle qu'il décrit. Ainsi, on peut considérer les matrices de rotation précédentes sous la nouvelle forme suivante :

$$R_y(\theta_y) = \begin{bmatrix} 1 & 0 & e_{\theta_y} \\ 0 & 1 & 0 \\ -e_{\theta_y} & 0 & 1 \end{bmatrix} \text{ et } R_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -e_{\theta_x} \\ 0 & e_{\theta_x} & 1 \end{bmatrix} \tag{5.11}$$

Avec e_{θ_y} l'approximation de l'angle de tangage θ_y et e_{θ_x} l'approximation de l'angle de lacet θ_x. Ainsi :

$$\begin{aligned} R &\simeq I_3 + R_x(\theta_x) + R_y(\theta_y) \\ R &\simeq I_3 + \sum R_i(\theta_i) \end{aligned} \tag{5.12}$$

Principe du calcul de la rotation de la caméra Du recalage d'image on obtient un vecteur de déplacement subi par l'image entre deux instants. Ce vecteur de déplacement est exprimé en pixels. Si on prend deux images I et I', avec I' l'image la plus récente, le déplacement va être exprimé pour une imagette suivant la différence entre $(u, v)^T$ et $(u', v')^T$ respectivement la position de l'imagette dans l'image d'origine et la position de l'imagette dans la nouvelle image. En effet, si on prend en compte une seule imagette, le déplacement est équivalent à $(u' - u, v' - v)^T$. Dans notre cas, nous réalisons ces calculs pour 10 imagettes dans chaque image.

D'après le chapitre 3.1.1 page 56 qui définit le modèle sténopé on sait qu'un pixel dans l'image est relié à son point dans le repère monde par les transformations suivantes :

$$P(x, y, z)^T \underset{T}{\longrightarrow} P(x_c, y_c, z_c)^T \underset{K}{\longrightarrow} P_c(u, v)^T \tag{5.13}$$

Mais le passage du point 3D dans le repère caméra $P(x_c, y_c, z_c)^T$ au point 2D dans l'image $P_c(u, v)^T$ est non linéaire. Par conséquent, le passage de l'un à l'autre de ces systèmes de coordonnées n'est pas trivial. Heureusement, il existe une relation linéarisée liant $P_c(u, v)^T$ à $P(x_c, y_c, z_c)^T$. Elle consiste à projeter le point $P(x_c, y_c, z_c)^T$ sur le plan image z_c. On obtient alors une relation entre $P_c(u, v)^T$ et $P_c(x_c/z_c, y_c/z_c)^T$.

Par ailleurs, on sait que quelque soit le mouvement réalisé par la caméra, un point dans le repère monde ne change pas. Par contre, la transformation liant ce point réel et le point dans le repère caméra change.

De plus, dans l'image, le pixel associé au point réel ne sera pas le même non plus entre les deux instants, car il est lié au point dans le repère de la caméra, donc :

$$\begin{aligned} P(x, y, z)^T &\underset{T}{\longrightarrow} P(x_c, y_c, z_c)^T \underset{K}{\longrightarrow} P_c(u, v)^T \\ P(x, y, z)^T &\underset{T'}{\longrightarrow} P(x'_c, y'_c, z'_c)^T \underset{K}{\longrightarrow} P_c(u', v')^T \end{aligned} \tag{5.14}$$

Ainsi l'étude du déplacement en pixels donné par $P_c(u, v)^T$ et $P_c(u', v')^T$ permet d'étudier le mouvement réel décrit par la relation liant $P(x_c, y_c, z_c)^T$ et $P(x'_c, y'_c, z'_c)^T$. Par contre, les transformations T et T' sont dépendantes à la fois de la matrice de rotation liant le point réel au point caméra mais aussi de la translation associée.

$$T = \begin{bmatrix} R_{3\times3} & t_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix} \tag{5.15}$$

L'application de la deuxième hypothèse nous permet de nous affranchir de ces translations. Cette approximation nous permet au travers de l'étude de la rotation liant les deux points caméra $P(x_c, y_c, z_c)^T$ et $P(x'_c, y'_c, z'_c)^T$ d'obtenir l'estimation du mouvement de la caméra tel que :

$$\begin{bmatrix} x'_c \\ y'_c \\ z'_c \end{bmatrix} = R \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{5.16}$$

Calcul de la rotation à partir du recalage d'image **Nous cherchons à obtenir l'estimation de la rotation en tangage $\hat{e}_{\theta_x}$ et en lacet $\hat{e}_{\theta_y}$.** Pour obtenir x_c, y_c et z_c à partir de $(u, v)^T$, il est nécessaire d'appliquer la projection sur le plan image z_c. On obtiendra alors $(u, v, 1)^T$ en fonction de $(\frac{x_c}{z_c}, \frac{y_c}{z_c}, 1)^T$, qui représente la projection du point 3D sur le plan image en coordonnées homogènes. Cette projection illustre la linéarisation de la relation non linéaire liant ces coordonnées. Cette projection nous donne la relation détaillée suivante :

$$\begin{aligned} F \cdot K \begin{bmatrix} \frac{x_c}{z_c} \\ \frac{y_c}{z_c} \\ 1 \end{bmatrix} &= \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \\ \begin{bmatrix} \frac{x_c}{z_c} \\ \frac{y_c}{z_c} \\ 1 \end{bmatrix} &= F^{-1} \cdot K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \\ \begin{bmatrix} \frac{x_c}{z_c} \\ \frac{y_c}{z_c} \\ 1 \end{bmatrix} &= \begin{bmatrix} \frac{u}{fk_x} - \frac{v\cos(\theta)}{fk_y\sin(\theta)} + \frac{c_y k_x \cos(\theta) - k_y(c_y\cos(\theta) + c_x)}{fk_x k_y} \\ \frac{v\sin(\theta) - c_y}{fk_y} \\ 1 \end{bmatrix} \end{aligned} \tag{5.17}$$

avec :

$$F = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.18}$$

et

$$K = \begin{bmatrix} k_x & k_x \cos\theta & c_x + c_y \cos\theta & 0 \\ 0 & \frac{k_y}{\sin\theta} & \frac{c_y}{\sin\theta} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5.19}$$

Cette relation prend en compte le « skew factor », θ qui permet de prendre en compte des pixels non carrés. Afin de simplifier les calculs nous considérons le cas idéal où les pixels sont carrés et où $\theta = \pi/2$. Ainsi, on obtient la matrice K simplifiée comme suit et la relation liant le point caméra et le point image suivante où $f_x = k_x \times f$ et $f_y = k_y \times f$:

$$K = \begin{bmatrix} k_x & k_x \cos\theta & c_x + c_y \cos\theta & 0 \\ 0 & \frac{k_y}{\sin\theta} & \frac{c_y}{\sin\theta} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \simeq \begin{bmatrix} k_x & 0 & c_x & 0 \\ 0 & k_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5.20}$$

$$\begin{bmatrix} \frac{x_c}{z_c} \\ \frac{y_c}{z_c} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{u - c_x}{f_x} \\ \frac{v - c_y}{f_y} \\ 1 \end{bmatrix} \tag{5.21}$$

De plus, d'après l'équation (5.12), on sait que la rotation qui lie $P(x_c, y_c, z_c)^T$ à $P(x'_c, y'_c, z'_c)^T$ est $(I_3 + \sum e_{\theta_i})$. Donc, suivant (5.16) on peut définir ces deux points l'un par rapport à l'autre par :

$$(I_3 + \sum R_i(\theta_i))^{-1} \begin{bmatrix} x'_c \\ y'_c \\ z'_c \end{bmatrix} = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{5.22}$$

Où :

$$(I_3 + \sum R_i(\theta_i))^{-1} = \begin{bmatrix} \frac{e_{\theta_x}^2 + 1}{e_{\theta_x}^2 + e_{\theta_y}^2 + 1} & \frac{e_{\theta_x} e_{\theta_y}}{e_{\theta_x}^2 + e_{\theta_y}^2 + 1} & \frac{-e_{\theta_y}}{e_{\theta_x}^2 + e_{\theta_y}^2 + 1} \\ \frac{e_{\theta_x} e_{\theta_y}}{e_{\theta_y}^2 + e_{\theta_y}^2 + 1} & \frac{e_{\theta_x}^2 + 1}{e_{\theta_x}^2 + e_{\theta_y}^2 + 1} & \frac{e_{\theta_x}}{e_{\theta_x}^2 + e_{\theta_y}^2 + 1} \\ \frac{e_{\theta_y}}{e_{\theta_x}^2 + e_{\theta_y}^2 + 1} & \frac{-e_{\theta_x}}{e_{\theta_x}^2 + e_{\theta_y}^2 + 1} & 1 \end{bmatrix} \tag{5.23}$$

Or, en faisant l'hypothèse des petits angles, on peut aussi considérer que les angles au second ordre sont négligeables. Par conséquent la relation (5.23) peut s'écrire :

$$(I_3 + \sum R_i(\theta_i))^{-1} = \begin{bmatrix} 1 & 0 & -e_{\theta_y} \\ 0 & 1 & e_{\theta_x} \\ e_{\theta_y} & -e_{\theta_x} & 1 \end{bmatrix} \tag{5.24}$$

On obtient finalement le système :

$$\begin{cases} x'_c - e_{\theta_y} z'_c = x_c \\ y'_c - e_{\theta_x} z'_c = y_c \\ e_{\theta_y} x'_c - e_{\theta_x} y'_c + z'_c = z_c \end{cases} \tag{5.25}$$

Ainsi, en appliquant la projection sur le plan image pour le point initial $P(x_c, y_c, z_c)^T$, on a :

$$\begin{cases} \dfrac{x'_c - e_{\theta_y} z'_c}{e_{\theta_y} x'_c - e_{\theta_x} y'_c + z'_c} = \dfrac{x_c}{z_c} \\ \dfrac{y'_c - e_{\theta_x} z'_c}{e_{\theta_y} x'_c - e_{\theta_x} y'_c + z'_c} = \dfrac{y_c}{z_c} \\ 1 = 1 \end{cases} \tag{5.26}$$

Mais, on ne connaît pas l'expression de x'_c, z'_c, z'_c en fonction de u' et v', le passage de la projection sur le plan image étant non linéaire. On peut par contre linéariser en multipliant les numérateurs et les dénominateurs par $\frac{1/z'_c}{1/z'_c}$. Ainsi on pourra exprimer $\frac{x'_c}{z'_c}$ et $\frac{y'_c}{z'_c}$ en fonction de u' et v' :

$$\begin{cases} \dfrac{\frac{x'_c}{z'_c} - e_{\theta_y} \times 1}{e_{\theta_y} \frac{x'_c}{z'_c} - e_{\theta_x} \frac{y'_c}{z'_c} + 1} = \dfrac{x_c}{z_c} \\ \dfrac{\frac{y'_c}{z'_c} - e_{\theta_x} \times 1}{e_{\theta_y} \frac{x'_c}{z'_c} - e_{\theta_x} \frac{y'_c}{z'_c} + 1} = \dfrac{y_c}{z_c} \end{cases} \tag{5.27}$$

La résolution de ce système nous donne alors e_{θ_y} et e_{θ_x} fonction de $\frac{x'_c}{z'_c} = X'$, $\frac{y'_c}{z'_c} = Y'$ et $\frac{x_c}{z_c} = X$, $\frac{y_c}{z_c} = Y$. Ainsi, nous obtenons autant de e_{θ_y} et e_{θ_x} qu'il y à d'imagettes. Nous considérons qu'il y à N imagettes et $i \in [1; N]$.

$$\begin{bmatrix} e_{\theta_x} \\ e_{\theta_y} \end{bmatrix}_i = \begin{bmatrix} YX' + 1 & Y'X \\ -YX' & -XX' - 1 \end{bmatrix}_i \begin{bmatrix} X - X' \\ Y - Y' \end{bmatrix}_i \tag{5.28}$$

De plus, grâce à l'équation (5.21) on peut obtenir e_{θ_y} et e_{θ_z} en fonction de valeur connue, $u, u', v, v', f_x, f_y, c_x, c_y$ par :

$$\begin{aligned} \begin{bmatrix} X \\ Y \end{bmatrix} &= \begin{bmatrix} \frac{u - c_x}{f_x} \\ \frac{v - c_y}{f_y} \end{bmatrix} \\ \text{et} \begin{bmatrix} X' \\ Y' \end{bmatrix} &= \begin{bmatrix} \frac{u' - c_x}{f_x} \\ \frac{v' - c_y}{f_y} \end{bmatrix} \end{aligned} \tag{5.29}$$

Ensuite, ce calcul est à appliquer à l'ensemble des déplacements calculés avec toutes les imagettes. Sur l'ensemble des imagettes testées, les rotations obtenues sont idéalement identiques d'une imagette à l'autre. Aussi, la résolution de ce système est réalisée suivant deux étapes, une étape de rejet de valeurs aberrantes et une étape d'estimation d'une valeur optimale.

Premièrement, on calcule les médianes numériques de l'ensemble des couples obtenues pour une image. Les médianes sont les valeurs de $e_{\theta_x i}$ et $e_{\theta_y i}$ se trouvant au milieu des intervalles $[\min(e_{\theta_x i}); \max(e_{\theta_x i})]$ et $[\min(e_{\theta_y i}); \max(e_{\theta_y i})]$. Les médianes sont notées $\tilde{e}_{\theta_x i}$ et $\tilde{e}_{\theta_y i}$. La médiane des couples est donc le couple $(\tilde{e}_{\theta_x i}, \tilde{e}_{\theta_y i})$, mais cette médiane ne fait référence à aucun couple existant. Par conséquent, l'ensemble des valeurs médianes noté M que nous gardons dans nos calculs postérieurs est l'ensemble des couples tel que :

$$\begin{bmatrix} e_{\theta_x} \\ e_{\theta_y} \end{bmatrix}_I \text{ avec : } I = \{i \in [1; N] | (e_{\theta_x i}, e_{\theta_y i}) \in [\tilde{e}_{\theta_x i} - \varepsilon; \tilde{e}_{\theta_x i} + \varepsilon] \times [\tilde{e}_{\theta_y i} - \varepsilon; \tilde{e}_{\theta_y i} + \varepsilon]\} \tag{5.30}$$

Sans faire de généralités, on suppose pour simplifier les notations que les M premiers couples répondent à ces conditions. Donc, $I = [1; M]$

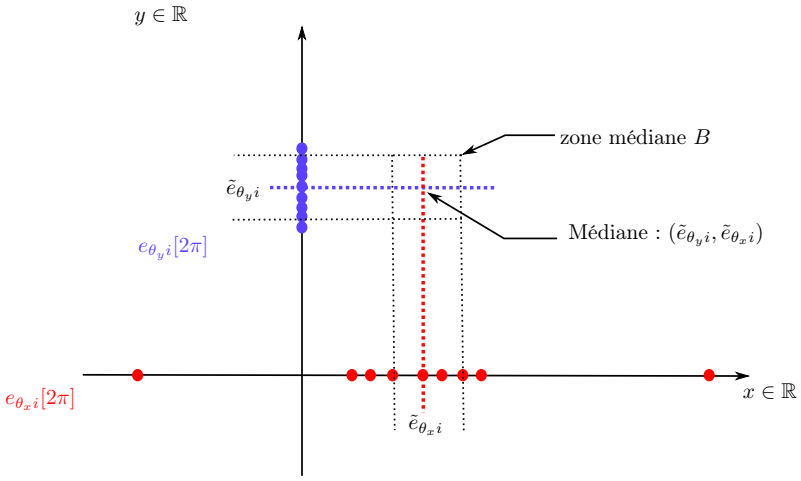


FIGURE 5.22 – Les points bleus représentent les valeurs de $e_{\theta_y i}$ et les points rouges sont les valeurs de $e_{\theta_x i}$. $\tilde{e}_{\theta_y i}$ et $\tilde{e}_{\theta_x i}$ sont respectivement les médianes des valeurs $e_{\theta_y i}$ et $e_{\theta_x i}$ et B est la zone médiane correspondant à l'intersection des ensembles $[\tilde{e}_{\theta_x i} - \varepsilon; \tilde{e}_{\theta_x i} + \varepsilon]$ et $[\tilde{e}_{\theta_y i} - \varepsilon; \tilde{e}_{\theta_y i} + \varepsilon]$. Les couples de valeurs se trouvant dans cet ensemble B sont gardés pour les calculs de moindres carrés, les autres valeurs sont rejetées.

Grâce à ce calcul de médiane (Figure 5.22), on peut rejeter les couples dont les valeurs sont aberrantes, c'est à dire des valeurs dont l'ordre de grandeur ne correspond pas au reste des estimations. En effet, contrairement à la moyenne, la médiane est très peu sensible aux valeurs aberrantes et permet d'établir des calculs de réjection fiable. Les couples gardés sont les couples se trouvant dans l'ensemble I.

Deuxièmement, il est à noter que les rotations sont censées être les mêmes quelque soit la position de l'imagette dans l'image. Cette propriété permet d'appliquer le calcul du moindre carré ordinaire afin d'extraire une valeur optimale de rotation en fonction de l'ensemble des valeurs obtenues. Le moindre carré est sensible aux aberrations, mais la réjection réalisée précédemment permet de l'utiliser de manière adaptée.

Pour réaliser ce calcul, on utilise la concaténation des systèmes ayant passé l'étape de réjection. En simplifiant la notation de l'équation (5.28) suivant la notation $A_{2M\times 2}\Theta_{2\times 1} = B_{2M\times 1}$, on obtient ce système où $\begin{bmatrix} e_{\theta_x} & e_{\theta_y} \end{bmatrix}^T$ est de nouveau inconnue :

$$\begin{bmatrix} \begin{bmatrix} Y_1X'_1 + 1 & -Y_1X'_1 \\ Y'_1X_1 & -X_1X'_1 - 1 \end{bmatrix}^{-1} \\ \dots \\ \begin{bmatrix} Y_MX'_M + 1 & -Y_MX'_M \\ Y'_MX_M & -X_MX'_M - 1 \end{bmatrix}^{-1} \end{bmatrix}_j \begin{bmatrix} e_{\theta_x} \\ e_{\theta_y} \end{bmatrix} = \begin{bmatrix} X_1 - X'_1 \\ Y_1 - Y'_1 \\ \dots \\ X_M - X'_M \\ Y_M - Y'_M \end{bmatrix}_j, \; j \in I$$

$$\begin{bmatrix} A_1^{-1} \\ \dots \\ A_M^{-1} \end{bmatrix}_j \begin{bmatrix} e_{\theta_x} \\ e_{\theta_y} \end{bmatrix} = \begin{bmatrix} B_1 \\ \dots \\ B_M \end{bmatrix}_j, \; j \in I \tag{5.31}$$

$$A\Theta = B$$

On obtient le moindre carré $\hat{\Theta}$ de Θ, **l'estimation de la rotation en tangage $\hat{e}_{\theta_x}$ et en lacet** $\hat{e}_{\theta_y}$ par la formule suivante :

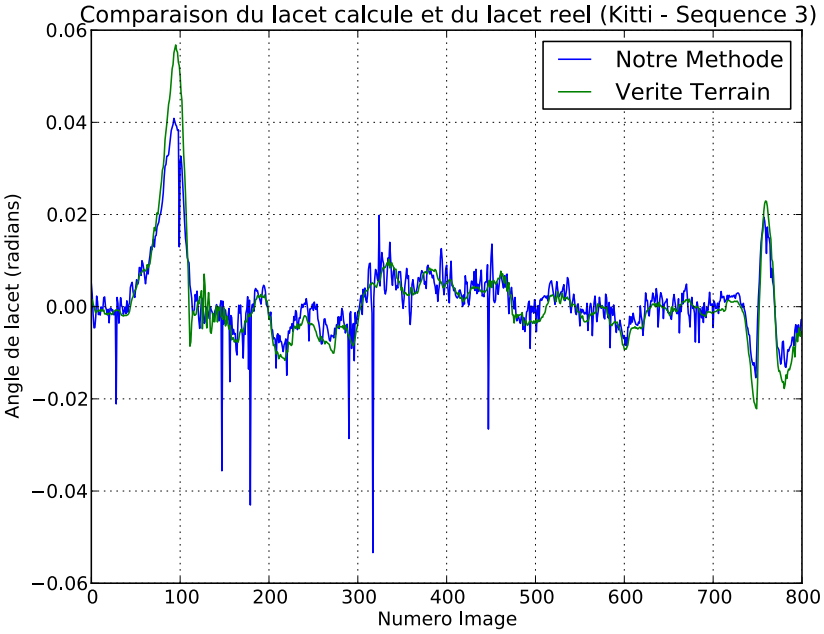


FIGURE 5.23 – Estimation de la rotation en lacet sur la séquence 3 du jeu de données *Kitti*. Cette séquence comprend 800 images et a été acquise en environnement routier.

$$\hat{\Theta} = (A^T A)^{-1} A^T B \tag{5.32}$$

Le moindre carré est le meilleur estimateur non biaisé. Il permet d'obtenir une estimation sur l'ensemble des estimations réalisées dans l'image. À la suite du moindre carrée, on obtient alors une estimation de l'angle de tangage et de l'angle de lacet. Comme les résultats le montrent dans la suite, cette méthode offre une estimation correcte des rotations réalisées par la caméra. Cette estimation permet à la méthode stéréoscopique d'accélérer son calcul et améliore ainsi l'ensemble des résultats.

5.2.3 Résultats

Cette méthode a été implémentée afin d'initialiser l'estimation réalisée par *LibViso2*. La suite de ce chapitre détaille les résultats obtenus avec cette optimisation. Ce nouvel algorithme est évalué sur les mêmes métriques que le précédent, mais nous rajoutons deux résultats montrant les particularités de cette optimisation : la précision de la rotation calculée et le nombre d'itérations de *LibViso2* avec et sans l'optimisation.

5.2.3.1 Précision de la rotation calculée

Notre algorithme d'optimisation réalise une estimation de la rotation de la caméra entre deux instants. Ici, nous présentons les résultats de cette estimation sur l'ensemble des jeux de données *Kitti* soit 25000 images. La figure 5.23 illustre au travers de la comparaison du lacet estimé sur une séquence que notre méthode offre une estimation relativement proche de la vérité. En effet, on constate que les deux courbes possèdent la même tendance malgré des valeurs aberrantes. Ce résultat est confirmé par le tableau 5.2 qui présente l'ensemble des données. Les corrélations présentées

prouvent que les valeurs calculées sont proches des valeurs réelles bien que la variance illustre la dispersion causée par de mauvaises estimations.

Ainsi, il apparaît que malgré l'utilisation d'approximations dans cette méthode, les résultats obtenus sont performants. En effet, les rotations estimées sont fortement corrélées aux rotations réelles. Ce résultat est illustré par la figure 5.23, mais on note sur cette figure des valeurs erronées. Ces valeurs ne sont pas aberrantes en ordre de grandeur, ce qui explique leur non-élimination par nos systèmes de réjection.

La suite des résultats présentés dans ce chapitre prouvent que ces rotations sont suffisantes pour permettre à notre méthode d'optimiser les calculs stéréoscopiques.

Séquence	Corrélation	Erreur d'approximation moyenne (radians)	Variance (radians)
00	0.86	0.011	0.16
01	0.86	0.008	0.12
02	0.79	0.01	0.15
03	0.88	0.003	0.07
04	0.47	0.001	0.05
05	0.83	0.008	0.13
06	0.85	0.008	0.15
07	0.88	0.01	0.15
08	0.81	0.008	0.13
09	0.78	0.011	0.14

Table 5.2 – Résultats d'estimation de la rotation en lacet en comparaison avec les valeurs réelles. La corrélation proche de 1 montre une forte ressemblance entre les données estimées et les données réelles. L'erreur d'approximation montre l'erreur moyenne par rapport aux données réelles. La variance montre la dispersion des valeurs autour des valeurs réelles.

5.2.3.2 Temps d'exécution avec et sans initialisation

L'objectif principal de cette méthode est d'optimiser l'estimation du mouvement en accélérant le calcul stéréoscopique. Comme le montre la figure 5.24, il s'avère que l'utilisation de cette méthode réduit le nombre d'itérations nécessaire au calcul dans certains cas, surtout visible quand la méthode est en difficulté. La réduction du nombre d'itérations dans les calculs de *LibViso2* permet d'accélérer l'ensemble des calculs et d'augmenter le nombre d'odométries estimées. En effet, comme le montre la figure 5.24, avec l'initialisation par notre algorithme *LibViso2* n'a besoin que de 3 ou 4 itérations pour obtenir une solution, alors que sans, *LibViso2* oscille entre 3 et 14 itérations pour obtenir ses solutions. À l'origine l'algorithme atteignait une moyenne de 6 odométries par seconde, avec notre optimisation ce nombre porte maintenant à 9. La suite de cette partie démontre que cette accélération améliore aussi la trajectoire calculée.

5.2.3.3 La trajectoire

La trajectoire calculée est un point crucial de la méthode de reconstruction d'environnement. En effet, c'est par cette trajectoire que les nuages de points 3D sont repositionnés. Meilleure est la trajectoire, meilleure est la reconstruction. Afin d'évaluer notre optimisation par rapport à la méthode non optimisée nous avons mesuré les performances de cette solution selon les mêmes critères que *LibViso2* dans la section précédente. On constate une nette amélioration à la fois sur les jeux de données à basse et à plus haute vitesse. En effet, à basse vitesse ($1m/s$) l'erreur n'est plus que de 3% contre 8% à l'origine. Ainsi, on se retrouve avec un résultat proche de celui obtenu dans le jeu de données *Kitti* qui est de 2.44% avec ce nouvel algorithme optimisé. En effet, sur les

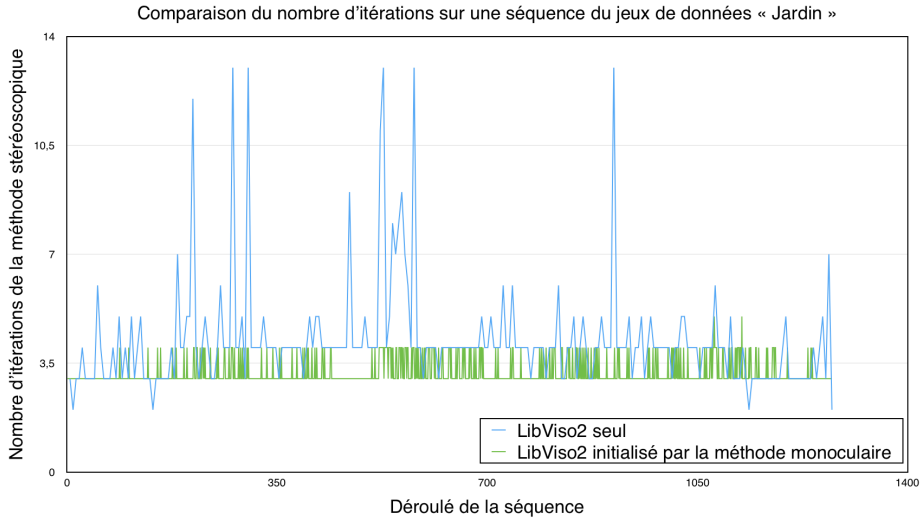


FIGURE 5.24 – Comparaison du nombre d'itérations nécessaire à *LibViso2* pour converger vers une solution dans le cas original et avec notre méthode d'optimisation.

jeux de données *Kitti*, on remarque une légère amélioration de 0.03%. Par contre, à plus de 5 m/s les résultats bien qu'améliorés d'un facteur supérieur à 3, ne sont pas suffisants pour obtenir une reconstruction performante, en effet 20% d'erreur sur la trajectoire montrent que l'algorithme n'est pas suffisament robuste pour suivre des changements plus importants entre deux images. La figure 5.26 montre bien ce fait.

La figure 5.25 et le tableau 5.3 nous permettent de constater que le gain apporté par cette optimisation est visible sur les trajectoires calculées. Sur le jeu de données acquis dans le jardin grâce au scanner laser fixe, nous obtenons une réduction de l'erreur d'un facteur 3 par rapport à la trajectoire calculée avec la stéréo seule. De même, il apparaît que notre trajectoire est désormais très proche de la vérité terrain sur l'ensemble de la trajectoire. Cette amélioration s'explique par notre optimisation qui aide *LibViso2* à mieux estimer le mouvement réalisé par la caméra.

	LibViso2		Notre méthode	
Erreur moyenne en trajectoire	en mètres	en %	en mètres	en %
Données Kitti Séquences (0 à 9) à plus de $5m/s$	0.988m pour 40m	2.47%	0.976m pour 40m	2.44%
Notre jeu de données Sur 40 mètres à $\sim 1m/s$ Sur 24.5 mètres à $\sim 5m/s$	 $\sim$3 mètres $\sim$16 mètres	 8% 66%	 $\sim$1 mètre $\sim$5 mètres	 3% 20%

TABLE 5.3 – Résultats en trajectoire donnant l'erreur finale de position calculée selon le principe de la distance euclidienne sur les jeux de données *Kitti* et notre jeu de données Jardin.

5.2.3.4 La distance entre les nuages de points

Comme expliqué précédemment, la reconstruction de l'environnement est dépendante de la trajectoire. Ainsi ces améliorations permettent d'améliorer grandement la reconstruction.

Afin d'évaluer la reconstruction réalisée par l'algorithme optimisé, nous avons comparé la reconstruction réalisée avec la vérité terrain au même titre que la comparaison réalisée pour *LibViso2*

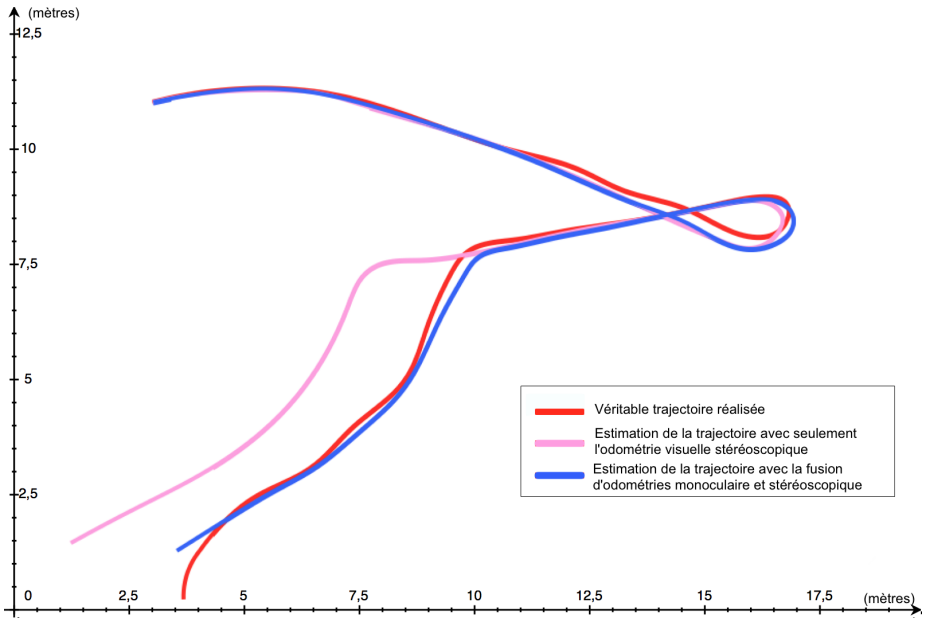


FIGURE 5.25 – Comparaison des trajectoires sur notre jeu de données à basse vitesse. En rouge apparaît la véritable trajectoire réalisée par le robot dans le jardin. En rose est tracée la trajectoire estimée par l'odométrie *libviso2* et en bleu, la trajectoire calculée par l'odométrie *libviso2* initialisée avec notre estimation de la rotation.

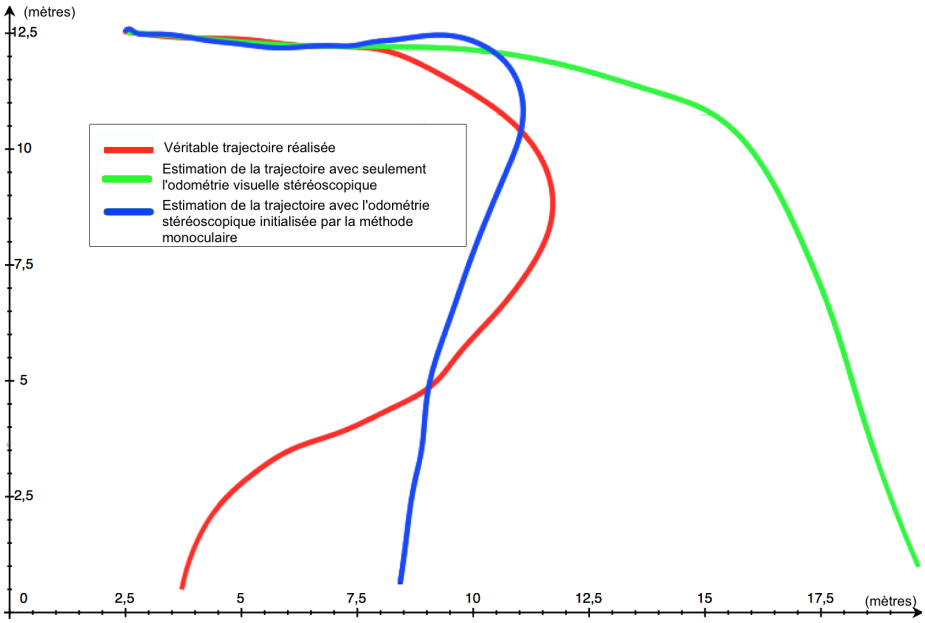


FIGURE 5.26 – Erreur de calcul de trajectoire quand le robot roule à « haute vitesse ». En rouge est tracé le véritable chemin réalisé par le robot dans le jardin. En vert est tracée la trajectoire estimée par l'odométrie *libviso2* et en bleu, la trajectoire calculée par l'odométrie *libviso2* initialisée avec notre estimation de la rotation.

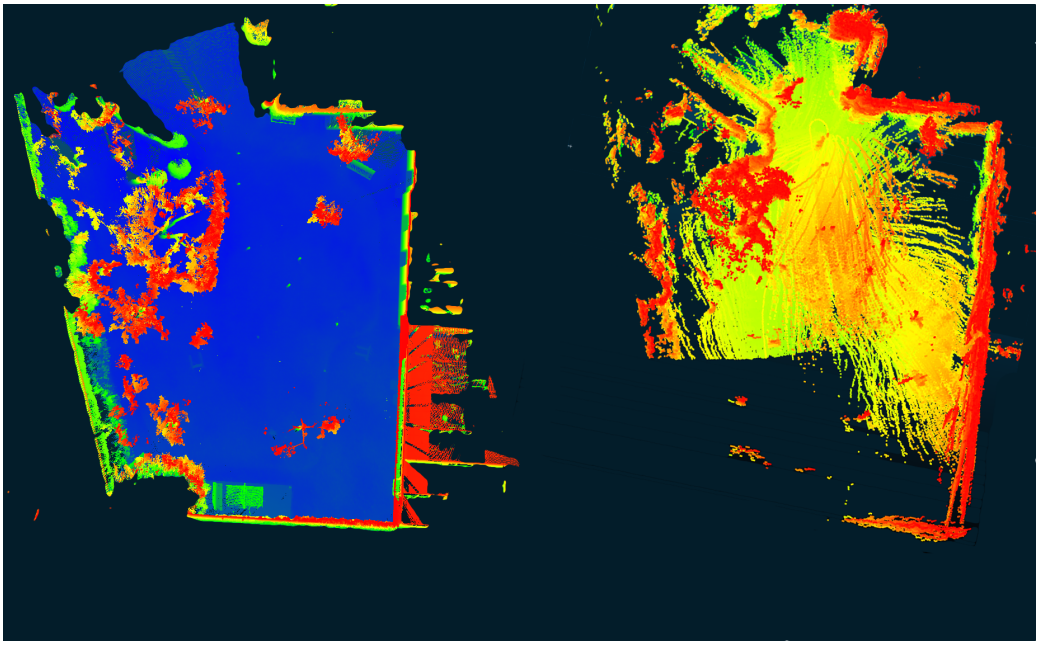


FIGURE 5.27 – Comparaison des nuages de points reconstruits à basse vitesse. Le nuage de point de la vérité terrain (gauche) à côté du nuage de points reconstruit (droite). Contrairement au nuage de la reconstruction par *LibViso2* seul, le nuage semble ici très proche de la vérité terrain. On aperçoit clairement l'enceinte du jardin. Cependant, on constate que le sol n'est pas homogène.

seul. Nous avons donc comparé les distances point à point du nuage reconstruit avec le nuage de points réel.

Ainsi, la distance moyenne obtenue entre les nuages de points reconstruits et le nuage de points de vérité terrain est de **34 cm** avec un écart type de $0,44cm$ à faible vitesse. Contrairement à ce qu'on aurait pu espérer, le nuage de points n'est pas trois fois plus proche, par contre on voit clairement sur la figure 5.27 que les nuages de points sont beaucoup plus ressemblants. En effet, contrairement à la comparaison réalisée pour *LibViso2* dans la figure (Figure 5.12 page 106), l'aspect général est plus cohérent. Ce fait est lié à la trajectoire ayant permis de replacer les nuages de points instantanés de manière plus précise.

De plus, la figure 5.28 illustre cette cohérence en montrant que les points jaunes estimés représentant les amers coïncident avec la position des amers dans la vérité terrain. La cohérence globale est bien présente avec cette méthode optimisée.

Il apparaît que notre algorithme améliore visiblement la reconstruction, mais une distance de 34 cm est encore importante. Cependant, même si cette distance est importante, la reconstruction est cohérente. En effet, il appraît que les amers sont réplacés correctement les uns par rapport aux autres. Cette cohérence « globale » peut permettre l'utilisation de ce système pour la détection d'obstacles positifs, mais la détéction d'obstacles négatifs est compromise par le manque de précision.

Observons maintenant la charge que cette optimisation provoque sur la machine.

Charge CPU et mémoire RAM Tel que nous l'avons expliqué précédemment, nos algorithmes ne sont pas les seuls à être exécutés sur les calculateurs d'un véhicule et c'est pourquoi nous devons évaluer la charge que cet algorithme impose aux machines.

Cette fois, l'évaluation se fait face à la consommation de *LibViso2* seul et non face à la situation de référence. Comme le montre la figure 5.29 page 126 pour *LibViso2*, la consommation est repartie sur les différents cœurs virtuels de notre machine jusqu'à atteindre une consommation moyenne de

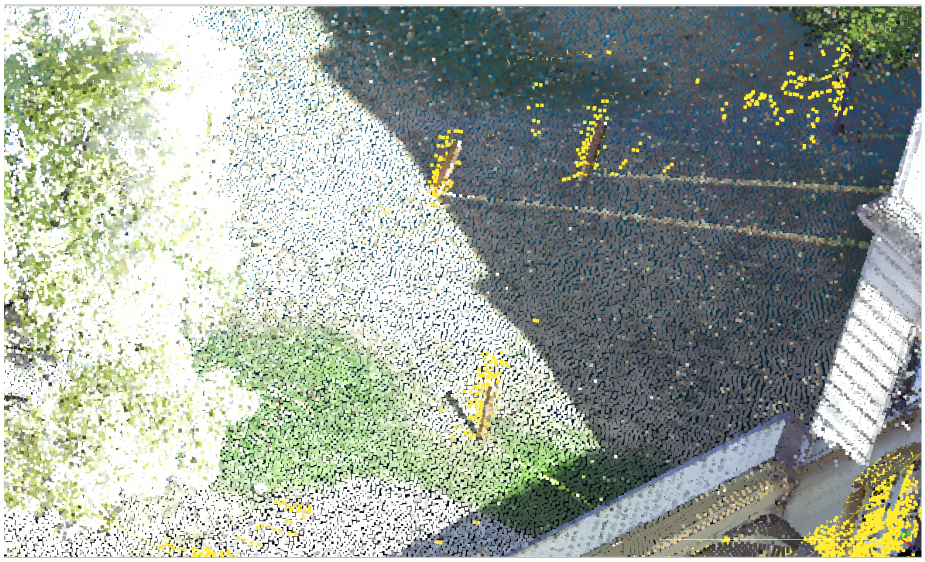


FIGURE 5.28 – Mise en exergue de la cohérence du nuage de points reconstruit (jaune) face au nuage de points de la vérité terrain. On constate que les amers reconstruits sont positionnés sur les amers réels.

$26,9\%$ et une charge mémoire RAM de $82,5$mo. Quand l'optimisation monoculaire s'exécute aussi, la charge est plus importante. Comme l'illustre la figure 5.30 la charge sur le CPU est presque doublée pour atteindre $51,3\%$ de moyenne. La mémoire RAM est, elle, utilisée de $100mo$ supplémentaires.

Notre algorithme n'est pas optimisé pour limiter la charge CPU, bien qu'une charge de $51,3\%$ soit trop importante pour être porté sur un système représentatif de notre application, une amélioration est possible. En effet, il est envisageable d'optimiser les utilisations des images, de paralléliser certains traitements, de transformer nos deux odométries en une seule combinaison. Ce résultat ne nous semble donc pas bloquant.

Actuellement, l'estimation de la rotation et le calcul d'odométrie stéréoscopique est composé de deux programmes. Chacun de ces programmes charge des images en mémoire et les traitent indépendamment. L'implémentation d'un seul accès aux images mutualisé entre l'estimation des rotations et la seconde méthode permettrait de ne réaliser qu'un accès aux images. Ensuite, les calculs appliqués sur les imagettes sont réalisés les uns à la suite des autres. Leur parallélisation est possible, car ces calculs sont indépendants. Elle permettrait alors d'optimiser le pré-traitement. Enfin, ces deux programmes échangent par le biais de ROS des informations de rotation. L'intégration de l'estimation des rotations comme un module de l'odométrie stéréoscopique s'exécutant en parallèle de l'appariement des images pourrait encore faire gagner un temps précieux.

5.2.4 Conclusion quant à l'estimation de la rotation

Cette section présente l'algorithme d'optimisation de la méthode de reconstruction par l'amélioration du calcul d'odométrie visuelle. Cette optimisation consiste à initialiser le calcul de l'odométrie stéréoscopique à une valeur approchée de la solution qu'il cherche à obtenir. Cette initialisation est réalisée par une rapide estimation de la rotation en tangage et en lacet de manière monoculaire.

Les résultats montrent une nette amélioration de la trajectoire et de la reconstruction d'environnement calculées. En effet, la trajectoire est améliorée faiblement sur les jeux de données *Kitti* mais aussi de manière importante (d'un facteur 3) sur les jeux de données acquis dans le jardin de MINES ParisTech. Cette optimisation permet, grâce à la diminution du nombre d'itérations de l'odométrie stéréoscopique *LibViso2*, d'améliorer la trajectoire. Par la même occasion la reconstruc-

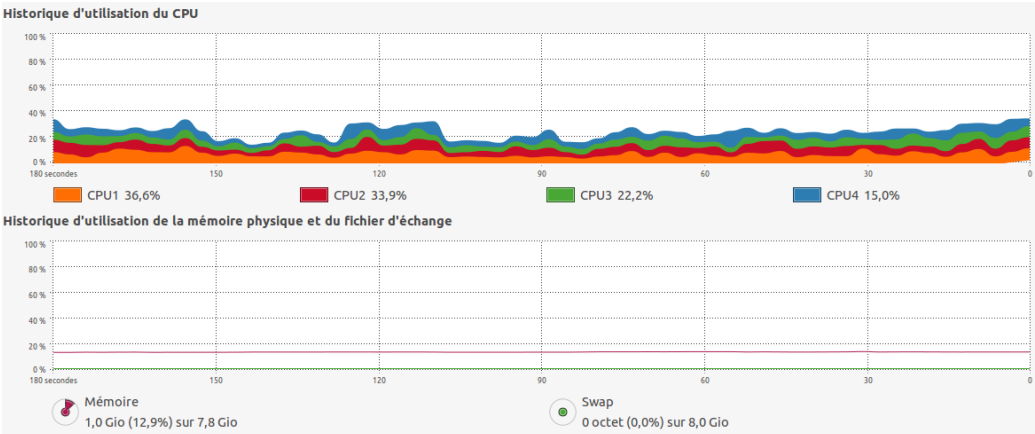


FIGURE 5.29 – Situation de la machine quand l'odométrie *LibViso2* est lancée seule. La charge est repartie sur les différents cœurs virtuels de la machine pour atteindre une consommation moyenne de $26,9\%$ de charge CPU contre $5,1\%$ en situation de référence. Concernant la charge en mémoire le lancement de l'algorithme consomme $82,5$mo.

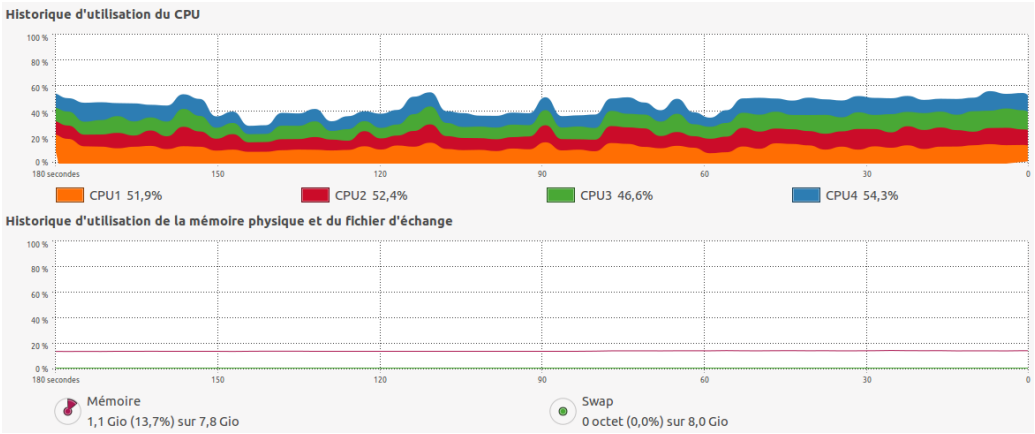


FIGURE 5.30 – Situation de la machine quand les deux odométries s'exécutent. Dans ce cas, le CPU est utilisé en moyenne à $51,3\%$ contre $5,1\%$ en situation de référence et la mémoire RAM est utilisée à hauteur de $182,5mo$.

tion est améliorée par rapport à l'utilisation de *LibViso 2* seul. De plus, bien que la charge CPU soit doublée, une large marge d'optimisation du code est possible au travers de plusieurs stratégies présentées précédemment.

Malgré ces améliorations, l'erreur de reconstruction du nuage de points est encore trop importante. En effet, comme le montre la figure 5.27 page 124, notre reconstruction n'est pas stable et le sol n'est pas homogène. Notre méthode ainsi optimisée offre donc une preuve que le concept est viable pour reconstruire l'environnement de manière grossière, mais pas suffisant pour reconstruire une carte robuste à tout type d'environnement. Ainsi, bien qu'il réponde à la nécessité de ne pas avoir à calibrer le système et à n'être trop coûteux en temps de calcul, ce concept ne répond pleinement pas à la contrainte principale qui est d'offrir la possibilité de détecter les obstacles dans tout type d'environnement. D'autant plus, que ces erreurs interviennent à faible vitesse, mais qu'à des vitesses plus élevées représentatives de notre application, une reconstruction grossière n'est pas obtenue.

On peut conclure que bien que cette optimisation fonctionne, elle n'est pas suffisante pour que notre algorithme réponde à la problématique dans son ensemble. Cependant, son optimisation est possible et les résultats montrent qu'une cohérence globale est présente à faible vitesse. Cette cohérence globale est le signe que le concept est viable et pourrait répondre pleinement à la problématique avec des ajustements et des optimisations supplémentaires.

5.3 Conclusion quant à notre algorithme

Dans ce chapitre nous avons présenté le principe de fonctionnement des algorithmes qui permettent de reconstruire l'environnement frontal au véhicule. Ces algorithmes se basent sur le placement dans l'espace des nuages de points 3D calculés grâce aux scans acquis par le LIDAR 2D tournant. Ces nuages de points sont replacés en fonction de la trajectoire calculée par l'odométrie du système. Cette odométrie est une odométrie visuelle à la base stéréoscopique et dont nous avons optimisé le fonctionnement par l'ajout d'une composante monoculaire.

En effet, dans ce chapitre nous avons montré au travers de mesures reflétant les contraintes présentées en introduction de ce manuscrit que l'algorithme *LibViso2* n'est pas suffisant au titre d'odométrie pour la reconstruction d'environnement. Suite à ces résultats nous avons développé un second algorithme d'odométrie intervenant en amont et initialisant LibViso2 avec une estimation grossière de la rotation effectuée par le véhicule.

Ainsi, les résultats de cette optimisation ont montré qu'une reconstruction proche de la vérité terrain pouvait être obtenue pour une faible vitesse de déplacement. De fait, ce chapitre a montré au travers d'expérimentations sur des jeux de données publics et sur notre propre jeu de données que l'optimisation que nous avons apportée offre une nette amélioration dans le cadre d'évaluations réalisées vis-à-vis des contraintes industrielles présentées en introduction, mais ces améliorations ne sont pas suffisantes pour répondre à nos contraintes. Cette reconstruction ne répond pas pleinement à nos contraintes de détection d'obstacles, car la précision ne permet pas de réaliser d'analyse du sol pour détecter les obstacles négatifs. Cependant, les résultats exposent tout de même que la reconstruction est viable pour la détection d'obstacles positifs. Ensuite, l'algorithme ayant une forte marge de progression pourrait permettre d'obtenir ces même résultats à des vitesses plus élevées.

Ce chapitre prouve que notre concept est viable pour la reconstruction grossière d'environnement frontal du véhicule. En effet, bien que des améliorations soient encore nécessaires pour obtenir des résultats fiables permettant la détection d'obstacles à faible et à haute vitesse, les résultats présentés ici montrent que le concept permet à lui seul d'offrir une représentation grossière de l'environnement.

Ainsi, le concept de reconstruction d'environnement par perception de l'environnement voisins est validé pour la détection d'obstacles positifs, mais nécessite encore du travail pour permettre l'obtention d'une reconstruction assez fine pour la détection d'obstacle négatifs.

Ensuite, nos travaux se sont orientés sur l'optimisation de l'odométrie, mais la création de la reconstruction pourrait aussi être optimisée en améliorant la gestion des nuages de points. Une des pistes envisagées est l'addition d'une centrale inertielle qui serait utilisée uniquement pour calculer le déplacement entre deux acquisitions LIDAR. En effet, une centrale inertielle peut atteindre des fréquences très élevées, et bien qu'elle dérive sur le long terme, son utilisation sur de courtes durées pourrait s'avérer très précise. Par cette méthode, la trajectoire approximative nécessaire et la reconstruction de l'environnement serait raffinée par l'utilisation de la centrale inertielle afin d'obtenir une forte cohérence locale de la reconstruction nécessaire à une bonne détection d'obstacles.

Le chapitre suivant présente des travaux parallèles mis en place dans le cadre de la segmentation de l'environnement. En effet, le but de ces travaux de thèse est la recherche de méthode pouvant permettre d'extraire les dangers d'un environnement par des moyens de perception. À ce titre, le chapitre suivant présente des travaux d'amélioration d'une méthode de segmentation visuelle de chemin et des travaux préliminaires de segmentation de nuages de points.

Chapitre 6

Segmentation des données acquises

Sommaire

6.1 Segmentation d'obstacles dans le nuage de points 129
6.1.1 Maniement du nuage de points 130
6.1.2 Segmentation du sol 131
6.1.3 Segmentation des obstacles 131
6.1.4 Conclusion de la segmentation du nuage de points 133
6.2 Segmentation de chemin 133
6.2.1 Segmentation de chemin par détection du point de fuite 134
6.2.2 Optimisation pour rendu temps réel 138
6.2.3 Résultats 140
6.3 Conclusion de la segmentation des données acquises 145
6.4 Conclusion sur le système de perception d'un environnement naturel . . 145

Le chapitre précédent présente les algorithmes réalisés dans le cadre de la reconstruction d'environnement. Ces algorithmes permettent d'obtenir une reconstruction cohérente globalement, mais qui manque de précision localement et ne permet pas de répondre pleinement à nos contraintes.

En parallèle du développement de ces algorithmes de reconstruction, nous avons travaillé sur des méthodes de segmentation. Les méthodes de segmentation servent à extraire des éléments pertinents d'un ensemble de données et sont cruciales pour les futures aides au pilotage, car elles permettront d'analyser la reconstruction recrée.

À ce titre, dans la première section, nous présentons les travaux réalisés dans le cadre de la segmentation du nuage de points. Ces travaux exposent des approches préliminaires pour l'extraction du sol et des obstacles positifs. En effet, comme le présente le chapitre précédent, à ce stade, la détection des obstacles négatifs n'est pas possible sur nos reconstructions d'environnement. En revanche, ces développements montrent que la cohérence globale apportée par les algorithmes de reconstruction permet déjà à ce stade une analyse de l'environnement.

Ensuite, nous nous intéressons à un algorithme de détection de chemins s'exécutant dans n'importe quel environnement. Cet algorithme pourrait s'avérer très intéressant pour réaliser une première segmentation dans l'image afin de définir une zone de recherche d'obstacles dans le nuage de points. En revanche, le défaut majeur de cet algorithme est qu'il n'est pas temps réel. Afin de contrer ce défaut, nous optimisons le cœur de cet algorithme, la détection du point de fuite.

6.1 Segmentation d'obstacles dans le nuage de points

La reconstruction d'environnement n'est pas la finalité de ce projet. Il est nécessaire d'analyser le nuage de points afin d'en extraire les dangers. Ainsi, nous présentons deux traitements dans cette section qui s'exécutent en même temps que la reconstruction du nuage de points et permettent d'extraire le sol et les obstacles positifs.

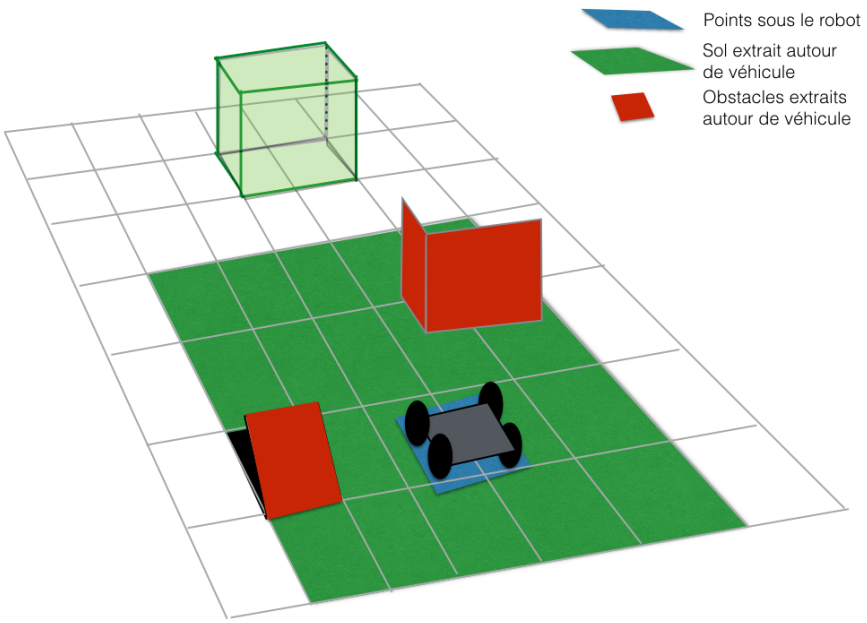


FIGURE 6.1 – Schéma de segmentation du sol et des obstacles. Le sol est extrait en fonction de l'altitude moyenne de roulage du véhicule. Les obstacles sont extraits en fonction de l'altitude du sol.

Ces traitements utilisent un ensemble de techniques de l'état de l'art et offrent ainsi une piste de réflexion pour des algorithmes de détection d'obstacles dans le cadre de notre application.

6.1.1 Maniement du nuage de points

Les nuages de points sont des structures de données lourdes à parcourir quand elles ne sont pas organisées. Une technique courante de simplification des nuages de points est l'utilisation d'arbres k-dimensionnés pour les simplifier et pour les parcourir plus rapidement. Cette méthode organise le nuage au travers de voxels qui regroupent un ensemble de points.

L'arbre k-d L'arbre k-d [Berg et al., 2008] consiste à diviser un nuage de point en une structure de données composée d'un ensemble d'espaces contenant des points. Ces espaces appelées voxels, sont créés en subdivisant le nuage en un arbre binaire suivant des hyperplans. Dans ce but, chaque point représente un nœud et chaque nœud non terminal est divisé en deux par un hyperplan. Ainsi une structure hiérarchique se crée entre les points et les recherches sont simplifiées, car on peut comparer les points par groupes hiérarchisés.

Cet algorithme est disponible dans la bibliothèque de logiciels *PCL* [1] qui permet de manipuler et de réaliser toutes sortes de traitements sur les nuages de points. Dans *PCL*, la création de l'arbre est réalisée en partant des points les plus bas pour atteindre les points les plus hauts du nuage de points.

Dans notre cas, l'arbre est de dimension 3 : x, y, z et l'arbre est donc hiérarchisé selon ces trois dimensions.

1. PCL pour Points Cloud Library : http://pointclouds.org/

Sous échantillonnage Par ailleurs, la complexité des traitements est liée à la forte densité et granularité des points composant le nuage de points. La technique de sous-échantillonnage est une technique permettant de limiter le nombre de points dans les nuages en moyennant ces points. En effet, grâce à l'arbre k-d, le nuage est organisé en arbre ce qui permet, en choisissant un niveau de profondeur de l'arbre à ne pas dépasser, de moyenner les positions des points sous le nœud terminal. Pour effectuer cette action, l'algorithme de sous-échantillonnage prend les barycentres des voxels définis par les nœuds terminaux [Pauly et al., 2002].

Ainsi, dans la suite des méthodes présentées, les nuages de points pris en compte sont des nuages de points sous échantillonnés comme sur la figure 6.1.

6.1.2 Segmentation du sol

Pour pouvoir déterminer si un objet ou une aspérité est un obstacle, il est nécessaire de pouvoir analyser sa taille et sa forme. De même, pour pouvoir l'analyser il faut connaître les limites de cet objet. Ainsi, une des méthodes les plus évidentes est de déterminer la position du sol afin de pouvoir calculer la hauteur de l'objet.

Dans ces développements préliminaires, nous avons appliqué une méthode simple d'extraction du sol basée sur l'altitude :

- On applique l'hypothèse que le véhicule ne roule pas sur une surface dangereuse pour lui même. Ainsi, on définit l'altitude du sol sur lequel il roule comme une zone de roulage sûre.
- On peut considérer que tous les points du nuage similaires, en altitude, aux points sous notre véhicule (acquis quelques instants auparavant) font partie du sol. Par conséquent, on mesure l'altitude moyenne des points sous notre véhicule. Pour ce faire, on applique la « recherche par plage ou *Range searching* » [2] sur les dimensions x et y autour de la position du véhicule (origine du repère robot), dans les limites de la taille du robot.
- Ensuite, on évalue l'altitude moyenne des points sous le robot pour obtenir l'altitude moyenne du sol environnant S_{moy}.En effet, nous considérons comme appartenant au sol tous les points assez proches du sol pour être traversés sans problème.
- Puis, pour tous les points du nuage de points, on exécute une nouvelle recherche de plage sur la dimension z. Cette fois, la plage de recherche est définie par la taille d'obstacle que nous cherchons à définir. Bien que définis de manière plus fine dans le chapitre 2.4.1 page 42, nous considérons les obstacles dans ces algorithmes comme les objets dont la taille est supérieure au rayon R des roues du véhicule .
- Ainsi, dans le cas où le sol serait à une altitude nulle, on segmenterait les points appartenant à l'intervalle compris entre $[-R; R]$. Or, comme nous l'exposons dans le chapitre 4.2.3 page 88, le repère du robot est situé approximativement à l'altitude du centre des roues. Ainsi, le sol est segmenté sur l'intervalle $[-2 \times R; 0]$. On obtient la segmentation visible sur la figure 6.2, où l'on expose la segmentation du sol suivant les points médians des voxels [3].

6.1.3 Segmentation des obstacles

Lorsque la segmentation du sol est réalisée, nous obtenons deux nuages de points distincts, le nuage de points du sol et le reste. Ce reste va être analysé afin d'extraire les obstacles potentiels.

2. La recherche par plage consiste à chercher à identifier des éléments d'un ensemble appartenant à un intervalle de valeur. Dans notre cas, on recherche dans les ensembles de points du nuage et dans un intervalle de distances à notre véhicule suivant les dimensions x et y.

3. Les points médians des voxels sont les barycentres de ces voxels.

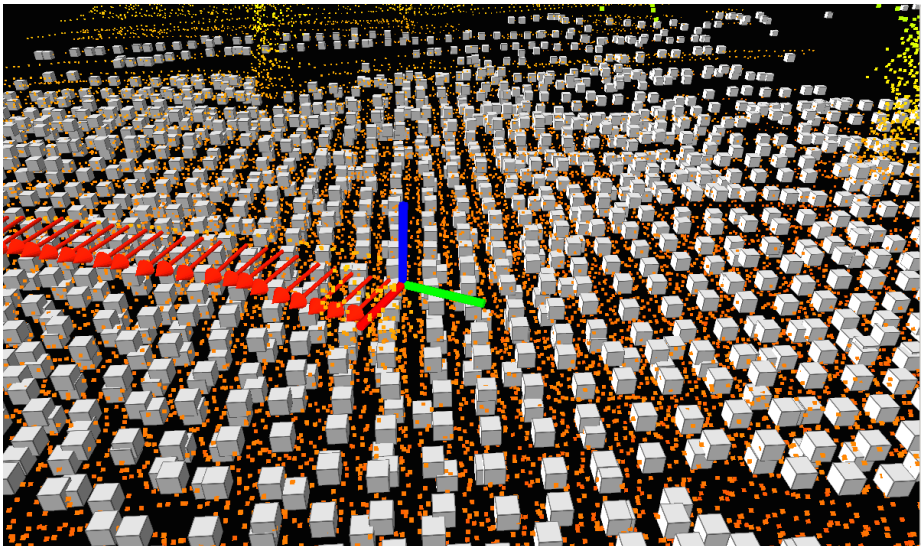


FIGURE 6.2 – Segmentation du sol dans le nuage de points. Le repère est indiqué par les traits bleu (z), rouge (x) et vert (y). Les flèches rouges représentent les positions antérieures du repère. Les cubes blancs sont les points médians des voxels segmentés comme appartenant au sol. Les points de couleur représentent l'ensemble du nuage de points non sous échantillonné.

Filtrage Pour commencer, une analyse préliminaire à la segmentation des obstacles est effectuée afin de filtrer le nuage de points et ainsi supprimer les points aberrants. Pour ce faire, les voxels vont être analysés afin de repérer les points isolés. Ainsi, on applique un filtrage sur la densité des voxels du nuage de points.

Lorsque la densité de points que renferme un voxel est inférieure à un seuil, ce voxel et tous les points qu'il renferme sont rejetés du nuage de points analysé. Le seuil est fixé en fonction de la résolution du nuage de points. Nous l'avons fixé à 6 qui représente la résolution de notre système entre 15 et 20 mètres (Figure 4.9 page 89).

Une fois ce filtrage effectué, la recherche d'obstacles peut être lancée. Pour rechercher les obstacles, nous avons choisi d'utiliser la méthode de segmentation appelée « Region Growing »[Besl, 1988] ou « croissance par région ». Cette méthode permet à partir d'un faible nombre de points d'extraire tous les points proches semblant faire partie de la même entité.

Segmentation par région L'algorithme de croissance par région développé par [Besl, 1988] est disponible dans PCL. Il définit, à partir d'une « graine » tous les points appartenant à la même entité que cette graine. On parle de croissance par région, car l'échantillon de points croît au fur et à mesure que des points proches et similaires sont identifiés. En effet, à chaque fois qu'un point connexe est identifié, on l'ajoute à la région et on cherche les points connexes avec lui, et ce jusqu'à ce que le dernier point ajouté n'ait pas d'autres points connexes que ceux qui appartiennent déjà à la région.

Dans l'approche utilisée par PCL, les paramètres d'homogénéités [4] sont définis vis-à-vis de l'angle qui sépare la normale de la graine et celle du point testé. Pour obtenir une normale à un point dans un nuage de points, on analyse ce point et ces voisins proches afin de définir la normale au barycentre de la surface créée par cet ensemble de points [Nazarian, 2008]. Cette normale est ensuite définie comme appartenant à ce point.

4. PCL Croissance par région : http://pointclouds.org/documentation/tutorials/region_growing_segmentation.php

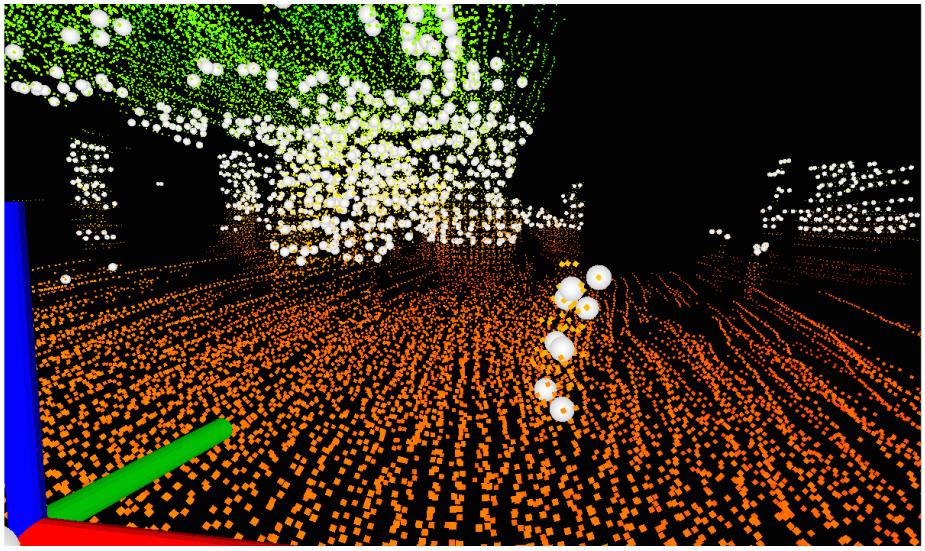


FIGURE 6.3 – Segmentation des obstacles dans le nuage de points. On retrouve le repère de notre plateforme (traits bleu, vert, rouge). Ici, les sphères blanches sont les points sous échantillonnés identifiés comme appartenant à des obstacles. Les points de couleur sont l'ensemble de points du nuage de la reconstruction.

Grâce à ces comparaisons points à points on peut définir des régions, partant du sol et segmentée dans tout leur volume comme l'illustre la figure 6.3

Par l'utilisation de cette segmentation nous avons pu visualiser que la cohérence globale de la reconstruction chapitre 5.3 page 127 permet d'extraire les obstacles positifs.

6.1.4 Conclusion de la segmentation du nuage de points

Cette section expose des travaux préliminaires sur l'analyse des nuages de points créés par l'algorithme présenté dans le chapitre précédent. Ces résultats confirment que la cohérence globale obtenue avec le système développé permet d'extraire les obstacles positifs, mais que l'incohérence locale ne permet pas d'obtenir une segmentation fine des obstacles.

Par contre, cette section expose des pistes pour la segmentation du nuage de points par la simplification de celui-ci au travers de l'utilisation d'arbre k-d et un procédé d'extraction d'objets mettant à profit cette structure. Les mêmes méthodes pourraient, avec une reconstruction plus fine, permettre d'extraire les dangers de l'environnement qu'ils soient au-dessous ou en dessus de la trajectoire du véhicule.

La section suivante présente des travaux préliminaires réalisés quant à la segmentation d'images. Cette segmentation pourrait servir à limiter la zone de traitement du nuage de points et ainsi limiter l'impact sur les performances du système.

6.2 Segmentation de chemin

Quand on souhaite analyser des données afin d'extraire des obstacles, il est important de pouvoir donner un sens à ces données. La segmentation a pour but de donner un sens à des données en définissant des zones (sol, objets, ciel) permettant la caractérisation de l'environnement. Dans leur article, [Kong et al., 2010] présentent une méthode de détection de chemin dans n'importe quel type

d'environnements. De par sa possibilité d'être utilisée dans des milieux non routiers, cette méthode possède un fort potentiel pour notre application et c'est pourquoi nous avons travaillé dessus.

L'algorithme développé par [Kong et al., 2010] se compose en trois étapes principales :

1. l'application du filtre de Gabor sur l'image
2. l'estimation du point de fuite [5]
3. l'estimation des bordures du chemin

Le filtre de Gabor permet d'estimer l'orientation des textures dans une image, grâce à ces orientations une estimation du point de fuite est possible et ce point de fuite est à l'origine de l'estimation des bordures.

Dans notre application, l'estimation visuelle des bordures d'un chemin permettrait de limiter la zone de recherche des obstacles. Cette limitation offrirait un gain calculatoire non négligeable quant à l'analyse des nuages de points. Par contre, à l'origine, cet algorithme n'est pas temps réel. Notre optimisation porte sur la deuxième partie de l'algorithme : la détection du point de fuite.

La suite de ce chapitre présente la méthode développée par [Kong et al., 2010], puis nous exposerons les méthodes testées pour optimiser la détection du point de fuite. Enfin, avec les résultats sur ces différentes méthodes nous détaillerons le choix de l'optimisation que nous avons validée.

6.2.1 Segmentation de chemin par détection du point de fuite

La détection visuelle de route est habituellement réalisée dans les environnements routiers [Risack et al., 2000; Fritz et al., 2004]. En effet, les marquages au sol sont une indication très visible des bordures de route, et leur détection permet une estimation performante de ladite route. Le problème soulevé par cet article est le cas des environnements non routiers où aucun marquage au sol n'est présent.

Dans ce cadre, les méthodes habituelles utilisant le filtre de Hough [Maitre, 1985] ne fonctionnent plus et d'autres solutions doivent être envisagées. [Kong et al., 2010] propose d'utiliser le point de fuite comme indication principale pour la détection du chemin. Les résultats sont très convaincants dans un ensemble très large d'environnements, et ce, même dans la neige où très peu d'indices permettent de détecter un chemin.

6.2.1.1 *Locally Adaptative Soft Voting method (LASV)*

Détaillons le fonctionnement de cet l'algorithme. Comme nous l'avons énoncé plus tôt, l'algorithme se compose en trois étapes principales : l'application du filtre de Gabor sur l'image, l'estimation du point de fuite et l'estimation des bordures du chemin.

En environnement routier, bon nombre d'articles traitent de cette problématique [Tardif, 2009; Taylor and Nieto, 2012] et plusieurs algorithmes existent (comme ceux de NEXYAD Automotive and Transportation™) pour y répondre. Mais, [Kong et al., 2010] réussissent eux à le détecter dans des environnements non routiers grâce à l'utilisation du filtre de Gabor. En effet, l'approche routière utilise habituellement le filtre de Hough utilisant les marquage au sol mais qui en leur absence ne peut pas fonctionner.

Le filtre de Gabor Le filtre de Gabor [Dunn et al., 1995] est un filtre utilisé en traitements d'images dont le but est d'extraire des contours répondant à des critères d'orientations et d'épaisseurs. Une image résultante d'un filtre de Gabor sera alors une image où ne subsistent que les zones avoisinant ses contours plus ou moins détaillés en fonction des paramètres choisis dans le filtre.

5. Point de fuite : De l'anglais *vanishing point*, il est l'endroit où les bords d'une route ou d'un chemin se croisent à l'horizon.

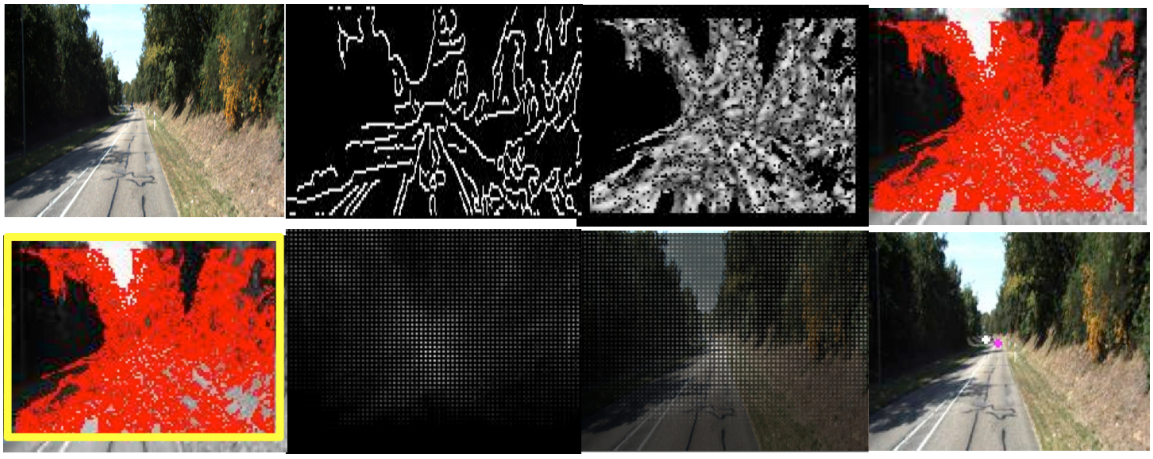


FIGURE 6.4 – Détection du point de fuite.
Ligne haute : (1) Image de départ. (2) Création de l'image de contours par filtrage de Canny. (3) Recherche des orientations des textures de l'image à l'aide d'un filtre de Gabor. (4) Création de la carte de confiance des orientations. Ligne basse : (5) Vote des pixels candidats. (6) Création de la carte de vote (7) Élection du pixel le plus populaire de la carte de vote. (8) Le point de fuite est trouvé [ROSE], il est comparé à la vérité terrain [BLANC].

Ce filtre fonctionne en appliquant des bancs de filtres passe bande sur une image. La convolution de ces bancs permet d'extraire plus ou moins de contours d'une image suivant l'orientation des bancs. Un banc s'exprime par une suite de masques identiques appliqués sur l'image et suit la sinusoïde complexe suivante :

$$g(x,y) = \exp(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}) \exp(i(2\pi \frac{x'}{\lambda} + \psi)) \tag{6.1}$$

avec :

$$\begin{aligned} x' &= x\cos\theta + y\sin\theta \\ y' &= -x\sin\theta + y\cos\theta \end{aligned} \tag{6.2}$$

Où λ représente la longueur d'onde de la sinusoide, γ paramètre l'aspect elliptique du masque, ψ est sa phase, θ correspond à l'orientation du masque et σ l'échelle à laquelle il s'applique.

Dans cet algorithme le filtre de Gabor est utilisé avec 36 orientations et suivant 5 échelles. La particularité de l'utilisation qui en est faite ici est qu'il n'est pas utilisé pour extraire des contours, mais pour évaluer l'orientations des textures de l'image. De plus, une valeur de confiance est calculée sur ces orientations. Cette confiance estime si l'orientation calculée est correcte ou non. Elle s'exprime en comparant les résultats obtenus suivant différentes orientations du filtre relatives aux maximums locaux.

De plus, cet indice de confiance est normalisé sur toute l'image et donne une valeur comprise entre 0 et 1 servant de critère de réjection des orientations calculées. Ainsi, si une orientation possède une valeur de confiance inférieure à la valeur $0,3$ (définie empiriquement par l'auteur), cette orientation est éjectée des futurs tests.

Le point de fuite Une route, courbe ou droite peut être simplifiée en deux courbes qui convergent sur la ligne d'horizon. Le point de convergence est appelé « Point de fuite ». Localiser ce point de fuite permet de répondre à trois questions : où est le sol ? Où est le ciel ? Où va la route ?

Pour détecter le point de suite, un procédé d'élection, appelé « Soft Voting Scheme » est réalisé. Cette élection a pour but d'élire le pixel de l'image semblant être le plus à même d'être le point de fuite. Pour ce faire, une première éjection est réalisée avec le calcul des orientations des textures avec le filtre de Gabor puis l'estimation de la confiance dans l'orientation. Après ces deux étapes, on élimine encore les pixels se trouvant dans les 10% bas de l'image. En effet, l'hypothèse est émise que le point de fuite ne se trouve pas tout en bas de l'image. Ce fait, vrai dans la plupart des cas ne l'est pas lorsque le véhicule dépasse une côte, mais il reste pour autant une bonne approximation de la réalité.

Pour chaque pixel restant, ses pixels voisins réalisent un vote relatif à l'angle et la distance qui les sépare de ce pixel. Il en résulte une carte des votes donnant les résultats de ces élections.

Ensuite, sur l'image d'origine est appliqué un second filtre, le filtre de Canny, transformant l'image en une image de contours. Cette image est mise en correspondance avec la carte des votes et permet d'obtenir le pixel représentant le point de fuite.

Voici un résumé des étapes réalisées dans cette première partie de la méthode :

1. Un filtre de Canny est appliqué sur l'image d'origine transformant l'image en une image de contour binaire, stockée pour être utilisée à la fin de l'algorithme. (Figure 6.4.2)
2. Un filtrage de Gabor est appliqué sur l'image d'origine. Ce filtre ne garde que les zones respectant une certaine orientation et une certaine épaisseur. (Figure 6.4.3)
3. La valeur de confiance de l'orientation estimée est calculée sur cette image filtrée. (Figure 6.4.4)
4. De cette confiance on obtient un certain nombre de candidats à l'élection de point de fuite. En effet, seuls les pixels ayant obtenu une confiance supérieure à 0,3 sont gardés. Cette valeur ayant été trouvée empiriquement par l'auteur.
5. Ensuite le vote est réalisé et on obtient la carte des résultats. (Figure 6.4.5)
6. Cette carte est mise en relation avec l'image de contour afin d'obtenir le pixel correspondant au point de fuite. (Figure 6.4.7)

Grâce à ce point de fuite, l'estimation des bords de chemin peut être réalisée.

La détection des bords de chemin Une fois le point de fuite estimé, l'estimation des bords de chemin est réalisée. Cette estimation se compose de deux étapes consistant à estimer un bord, puis l'autre.

Pour estimer le premier bord de chemin, en partant du point de fuite, un ensemble de lignes imaginaires et équitablement distribuées sont créées (Figure 6.5 (2)). Sur chacune de ces lignes, un test est réalisé afin de savoir si les pixels qui la composent possèdent des orientations proches de celle du point de fuite (Figure 6.5 (1)). Ensuite, est choisie la ligne sur laquelle il y a le plus de pixels en cohérence avec le point de fuite et dont la longueur est supérieure à un tiers de la diagonale de l'image (Figure 6.5 (4)).

Pour estimer le second bord de l'image, ce test est de nouveau réalisé, mais pour tous les pixels du premier bord (Figure 6.5 (5)). Par contre, pour chaque pixel du premier bord, un moins grand nombre de lignes imaginaires est tracé. Le deuxième bord est défini comme la ligne dont la cohérence entre le pixel du premier bord et une nouvelle ligne est la plus forte.

Une fois que ce deuxième bord est obtenu, le point de fuite final est défini comme étant l'intersection des deux bords de chemin et le chemin est la zone sous les bords. (Figure 6.5)

Performances Comme le montre l'article, cet algorithme offre de bons résultats sur tout type d'environnements. En effet, il détecte des chemins sur des terrains de poussière, de sable, de terre,

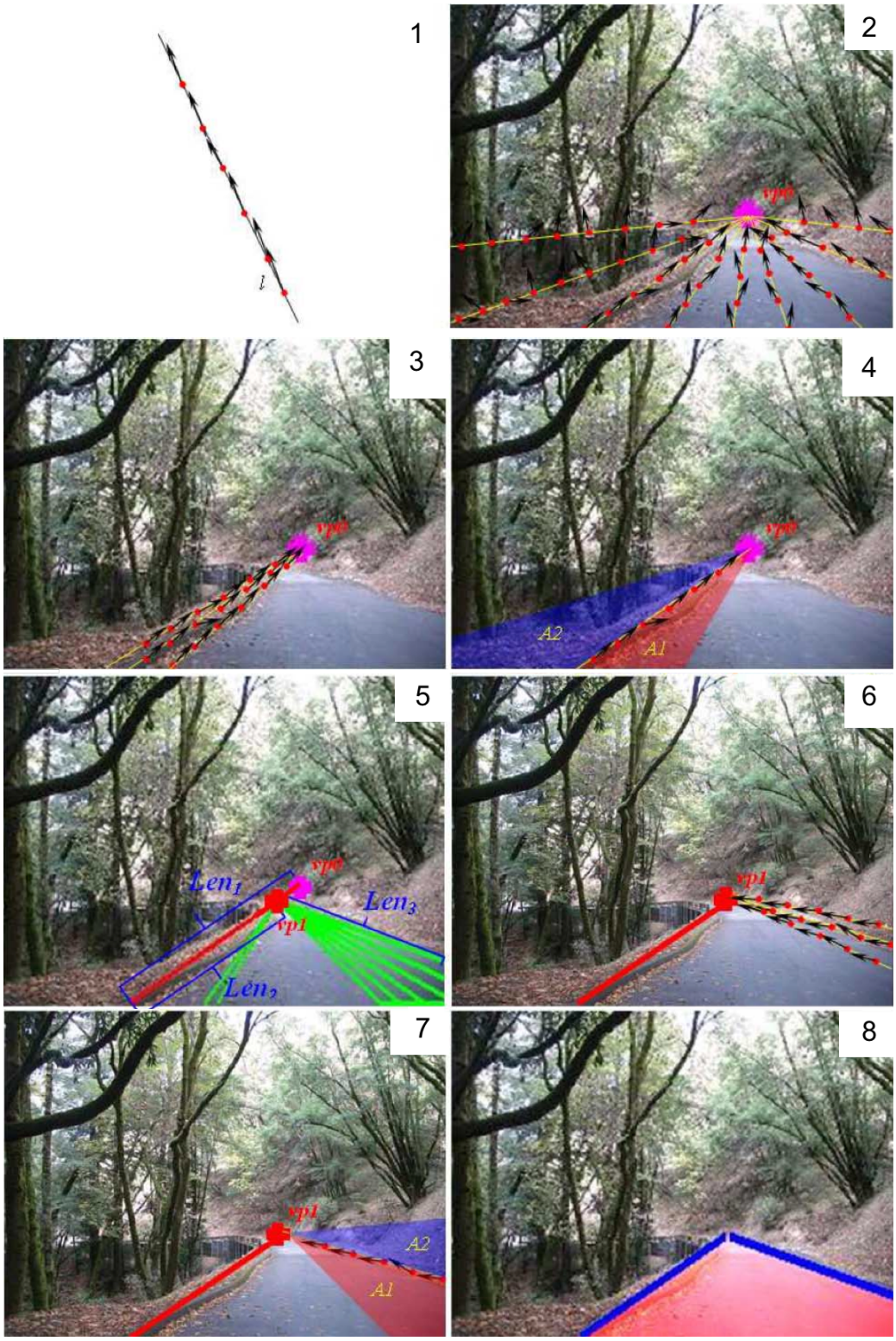


FIGURE 6.5 – Détection des bords de chemin.
(1) Une des lignes imaginaires sur laquelle les orientations des pixels sont évaluées. (2) L'ensemble des lignes imaginaires. (3) Une première estimation grossièreest réalisée, puis on réitère de manière plus fine. (4) En fonction de l'orientation de la ligne obtenue, la zone de recherche du second bord est déterminée. (5) Pour chaque pixel du premier bord on applique la recherche du second bord de la même manière. (6) On recherche la cohérence des lignes (7) On estime l'orientation du second bord. (8) et on obtient l'estimation du chemin. Schéma extrait de [Kong et al., 2010]

de goudrons et même des terrains recouverts de neige. Par contre, cette efficacité a un coût algorithmique, le code fourni fonctionne sous *MATLAB*[6] et nécessite 18 secondes sur notre machine pour calculer seulement le premier point de fuite pour une image.

Ce temps de calcul est extrêmement important est rend son utilisation inenvisageable à ce stade comme méthode de segmentation. Pour améliorer ces performances, nous avons réalisé différentes optimisations sur le calcul du point de fuite que nous présentons dans la suite de ce chapitre.

6.2.2 Optimisation pour rendu temps réel

Notre intérêt pour cet algorithme était poussé par la possibilité de l'utiliser comme une segmentation visuelle de l'environnement antérieur à la segmentation du nuage de points. Le code original est développé sous l'environnement *MATLAB* et à l'origine le temps de calcul a été évalué à 18 secondes par image pour calculer le premier point de fuite sur notre machine (processeur 2,3 GHz Intel Core i7 et mémoire 16 Go 1600 MHz DDR3). L'utilisation de cet algorithme comme moyen de segmentation n'était pas possible, mais son efficacité nous a poussés à chercher à optimiser son temps de calcul afin le rendre viable pour notre application.

Ainsi, nous exposons ici les différentes optimisations testées afin de trouver la meilleure façon d'améliorer l'algorithme. En effet, accélérer la vitesse d'exécution est une chose, mais il fallait éviter à tout prix d'en réduire son efficacité.

6.2.2.1 Les différentes méthodes évaluées

Pour optimiser l'algorithme, nous avons développé six méthodes que nous détaillons ici. Il est à noter que contrairement à l'algorithme original nous ne désirons pas traiter les images indépendamment les unes des autres. En effet, dans le jeu de données utilisé par Kong & al. toutes les images sont indépendantes et l'agorithme doit, sans contexte détecter un chemin. Dans notre cas, les images sont acquises par une caméra et font partie d'une séquence d'images qui se suivent. Par conséquent, nous utilisons le jeu de données *Kitti* [Geiger et al., 2012] en vue d'évaluer les optimisations développées dont certaines font appel à des correspondance entre des images consécutives. De plus, les optimisations sont également réalisées sous l'environnement *MATLAB*, car le but est d'optimiser la méthode et non le code en lui même.

Le temps de 18 secondes pour l'algorithme original est acquis lui aussi lors de son exécution sur les jeux de données *Kitti*. Par ailleurs, l'algorithme original réduit la taille des images à une valeur de 240×180 pixels. Dans la suite de ce chapitre quand il est fait question des images entières on parle de ces images réduites à 240×180 pixels.

Afin de justifier l'optimisation validée, nous présentons ici l'ensemble des méthodes étudiées.

Méthode I : Dépendance spatiale (Figure 6.6 I) La première méthode est une méthode où une simple dépendance spatiale est ajoutée entre chaque image. En effet, dans le cas d'application de l'algorithme original une telle optimisation n'est pas envisageable, car les images sont indépendantes. En revanche, dans notre cas, les images font partie de séquence d'images consécutives. Ainsi, l'ajout de dépendance entre les images est possible.

(a) Sur la première image de la séquence, on exécute le calcul du point de fuite en prenant en compte l'image entière.

(b) Sur les images suivantes, le vote pour l'élection du point de fuite est réalisé seulement dans une zone de $10\% \times 10\%$ de l'image autour des coordonnées du point de fuite trouvé précédemment.

6. MATLAB : http://fr.mathworks.com/index.html?s_tid=gn_logo

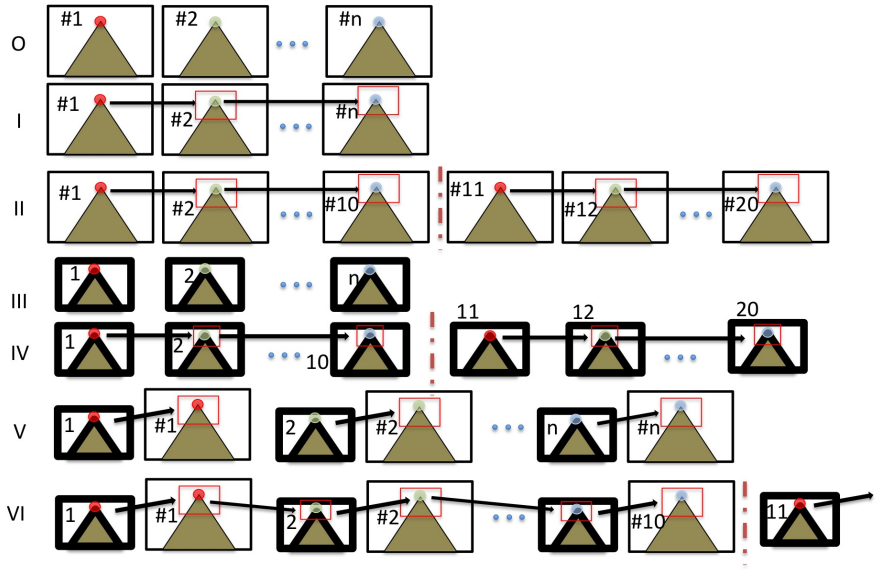


FIGURE 6.6 – Schéma des optimisations développées.
O, Méthode initiale : Calcul complet de l'image ; I, Méthode I : Les lignes illustrent la dépendance spatiale ; II, Méthode II : Les pointillés illustrent la réinitialisation de la méthode toutes les dix images ; III, Méthode III : L'image plus petite illustre la réduction de la taille de l'image.

Comme expliqué dans le chapitre précédent, les images étant acquises à 10Hz, le mouvement entre deux images n'est pas important. Ainsi, la zone de 10% × 10% s'explique par le fait que la route ne va pas se déplacer fortement entre deux images, et le point de fuite reste donc proche d'une image à l'autre.

Méthode II : Réinitialisation (Figure 6.6 II) La seconde méthode est équivalente à la première à ceci prêt que toutes les dix images, on réitère le calcul sur l'image entière afin de limiter la possible propagation d'une erreur d'estimation.

(a) La méthode I est utilisée, mais toutes les 10 images, on réitère le calcul complet I.a).

Méthode III : Réduction de la taille (Figure 6.6 III) Dans la troisième méthode, nous avons évalué la possibilité de réduire encore la taille des images quitte à perdre de l'information.

(a) La taille des images est réduite de moitié avant l'exécution du calcul sur toute l'image. Cette opération est réalisée sur toutes les images.

Méthode IV : Dépendance + Réduction + Réinitialisation (Figure 6.6 IV) La quatrième méthode est quant à elle la concaténation des trois premières.

(a) La taille des images est réduite de moitié.

(b) Sur la première image de la séquence, on exécute le calcul du point de fuite en prenant en compte l'image entière, mais réduite.

(c) Sur les images suivantes, toujours réduites, le vote pour l'élection du point de fuite est réalisé seulement dans une zone de $10\% \times 10\%$ de l'image autour des coordonnées du point de fuite trouvé précédemment.

(d) Toutes les 10 images, on réitère le calcul complet IV.b) toujours sur une image réduite puis IV.c) sur les images suivantes, et ainsi de suite.

Méthode V : Réduction de l'échelle (Figure 6.6 V) Dans cette cinquième méthode, nous appliquons une méthode de réduction d'échelle dans le but de trouver une valeur approximative du point de fuite. Ensuite, en nous basant sur les coordonnées estimées, nous recalculons le point de fuite à une échelle plus grande.

(a) La taille de l'image est réduite de moitié et le point de fuite est recherché dans l'image entière.

(b) Sur la même image à l'échelle originale, on recherche le point de fuite dans une zone de $10\% \times 10\%$ autour des coordonnées du point de fuite trouvé dans l'image réduite.

(c) On applique ces étapes sur toutes les images de la séquence.

Méthode VI : Dépendance + Réduction de l'échelle + Réinitialisation (Figure 6.6 VI) Enfin cette sixième et dernière méthode est la concaténation des méthodes V, I et II.

(a) La taille de l'image est réduite de moitié et le point de fuite est recherché dans l'image entière.

(b) Sur la même image à l'échelle originale, on recherche le point de fuite dans une zone de $10\% \times 10\%$ autour des coordonnées du point de fuite trouvé dans l'image réduite.

(c) Sur l'image suivante, le vote pour l'élection du point de fuite est réalisé seulement dans une zone de $10\% \times 10\%$ de l'image autour des coordonnées du point de fuite trouvé précédemment et pour une image de taille réduite de moitié.

(d) Sur la même image à l'échelle originale, on recherche le point de fuite dans une zone de $10\% \times 10\%$ autour des coordonnées du point de fuite trouvé dans l'image réduite.

(e) Toutes les 10 images, on réitère le calcul complet VI.a)

Par ailleurs, comme énoncé dans l'explication de l'algorithme original, l'auteur utilise dans sa méthode le filtre de Canny pour calculer une image de contours sur laquelle il base la définition du point de fuite. Nous présentons aussi les résultats avec l'utilisation du filtre de Prewitt [Maini and Aggarwal, 2009] qui transforme l'image en une image de gradient contenant plus d'information qu'une image de contours.

6.2.3 Résultats

Suite à la présentation de ces différentes méthodes, nous exposons maintenant les résultats obtenus vis-à-vis de la méthode originale. Dans ce but, nous présentons la manière dont nous avons créé nos données de vérité terrain, puis les métriques que nous avons utilisées et enfin les résultats.

6.2.3.1 Vérité terrain

Notre optimisation cherche à améliorer la détection du point de fuite. Le problème est, qu'à part les images fournies par Kong & al., il n'existe pas de base de données d'images dont les points de fuite sont labellisés afin de réaliser une vérité terrain. Par ailleurs, les images de Kong & al. sont des images indépendantes les unes des autres non représentatives du fonctionnement de notre application. Ainsi, nous avons réalisé une base de données de vérité terrain sur la base d'une séquence des jeux de données *Kitti*. Cette séquence représente une route dans un environnement périurbain.

Nous avons labellisé la position du point de fuite manuellement sur 250 images. Le point de fuite n'étant pas forcément évident à définir au pixel près, nous avons fait appel à plusieurs autres participants afin de réaliser la même tâche. Ensuite, nous avons calculé la moyenne des coordonnées labellisées pour chaque point de fuite et défini ces coordonnées comme vérité terrain .

6.2.3.2 Métriques

Afin d'estimer l'efficacité de nos optimisations et de l'algorithme original, nous avons mis en place deux métriques. La première métrique nous permet d'estimer la précision de l'estimation des points de fuite et la seconde la vitesse d'exécution des algorithmes.

Pour mesurer la précision des estimations nous calculons la distance euclidienne qui sépare les points de fuite estimés et les points de fuite de la vérité terrain. Cette distance est ensuite mise en relation avec la taille de la diagonale de l'image traitée. Une erreur de 100% équivaut donc à un point de fuite estimé à un coin de l'image et sa vérité terrain à l'opposée. Cette métrique a donc du sens pour des erreurs relativement faibles, ce qui est le cas dans la plupart des résultats.

Ensuite, nous calculons la moyenne de ces distances sur toute la séquence afin d'obtenir l'erreur moyenne de position exprimée en pourcentage. Donc, pour $P_v(x, y)$ les coordonnées du point de fuite de la vérité terrain et $P_e(u, v)$ les coordonnées du point de fuite estimé, on obtient l'erreur E_p pour N images de taille $M_x \times M_y$ par :

$$E_p = 1 - \frac{1}{N} \sum_{i}^{N} \frac{\sqrt{(x-u)^2 + (y+v)^2}}{\sqrt{M_x^2 + M_y^2}} \tag{6.3}$$

La précision est alors obtenue par :

$$P_p = (1 - E_p) \times 100, \text{ en } \% \tag{6.4}$$

Pour le calcul de la vitesse d'exécution V_e, nous calculons le temps nécessaire à l'algorithme pour estimer les coordonnées des points de fuite sur l'ensemble des images T_i et nous divisons ce temps par le nombre d'images traitées. Nous obtenons une métrique en images par seconde.

$$V_e = \frac{1}{N} \sum_{i}^{N} T_i, \text{ en images / seconde} \tag{6.5}$$

6.2.3.3 Résultats

L'évaluation porte sur l'ensemble des méthodes d'optimisation développées. Dans ce but, les métriques présentées permettent de comparer ces résultats avec l'algorithme original. Ainsi, nous classifions les méthodes entre elles et nous en validons une apparaissant comme étant la plus performante.

Premièrement, nous exposons les résultats obtenus avec la méthode originale qui servent de référence à la comparaison avec les méthodes d'optimisation.

Méthode originale (Figure 6.6.0) Selon nos métriques, la méthode originale, estimant le point de fuite sur l'intégralité de l'image, obtient des résultats très bons en précision (93.5%, Tableau 6.2). Par contre, cette méthode est extrêmement lente et s'exécute à 0.06 image par seconde. Cette vitesse même pour du code développé sous *MATLAB* est très faible. Ainsi, au travers des différentes méthodes présentées nous cherchons une méthode maximisant la vitesse d'exécution sans pour autant dégrader la précision de l'estimation.

Optimisation	Vitesse d'exécution (Image par seconde)
Méthode IV (gradient)	1.96
Méthode IV (contour)	1.66
Méthode VI (gradient)	0.71
Méthode VI (contour)	0.66
Méthode III (gradient)	0.53
Méthode I	0.42
Méthode V (gradient)	0.367
Méthode III (contour)	0.366
Méthode V (contour)	0.27
Méthode II	0.21
Originale	0.06

TABLE 6.1 – Classification des méthodes d'optimisation suivant leur performance en vitesse d'exécution indiquée en image par seconde. La méthode d'optimisation validée est indiquée en vert et le résultat de l'algorithme original est indiqué en rouge.

Optimisation	précision en position (en %)
Méthode V (gradient)	94.92%
Originale	93.50%
Méthode III (gradient)	92.66%
Méthode IV (gradient)	91.12%
Méthode V (contour)	90.28%
Méthode III (contour)	90.22%
Méthode IV (contour)	84.80%
Méthode VI (gradient)	83.26%
Méthode VI (contour)	79.74%
Méthode II	78.06%
Méthode I	30.41%

TABLE 6.2 – Classification par précision. L'erreur est la moyenne euclidienne des distances entre la vérité terrain et les points de fuite trouvés sur l'ensemble du jeu de données. La méthode d'optimisation validée est indiquée en vert et le résultat de l'algorithme original est indiqué en rouge.

Méthode I : Dépendance spatiale (Figure 6.6.I) La méthode I expose le principe de la dépendance spatiale. Cette méthode, après avoir estimé le point de fuite sur la première image, restreint les possibilités de détection du point de fuite. Ainsi, sur les images suivantes, le point de fuite ne peut appartenir qu'aux seuls 10% $\times$ 10% autour des coordonnées trouvées dans l'image précédente.

Le problème principal de cette méthode est la dérive. En effet, dans le cas où le point de fuite est toujours estimé correctement, cette méthode pourrait fonctionner, mais des erreurs successives entraînent une forte dérive de cette méthode. Dans ce cas, bien que cette dépendance spatiale apporte une bonne amélioration de la vitesse de calcul du point de fuite jusqu'à atteindre 0.42 image par secondes (Tableau 6.1), une importante perte de précision apparaît (Tableau 6.2). De fait, comme expliqué dans la section précédente, une précision de 30.41% (soit 68.59% d'erreur) est une valeur très faible. On peut conclure qu'avec la dépendance spatiale, le point de fuite est constamment perdu.

Méthode	Précision
Originale	88.46%
Méthode V (gradient)	88.01%

Table 6.3 – Classification de la précision sur le jeu de données de l'ENS (1003 images de situations difficiles).L'erreur est la moyenne euclidienne des distances entre la vérité terrain et les points de fuite trouvés sur l'ensemble du jeu de données.

Méthode II : Réinitialisation (Figure 6.6.II) La méthode II est une réponse à cette dérive induite de la méthode I. À cet effet, nous avons modifié la dépendance sur une période de seulement 10 images (Figure 6.6). Ainsi, toutes les 10 images, un calcul complet est lancé pour éviter la propagation du biais possiblement induit par les erreurs.

Le tableau 6.1 expose des résultats moins performants sur le plan de la rapidité. Effectivement, la méthode II atteint 0.21 images par seconde. Par contre, le niveau d'erreur est aussi moins prononcé. En effet, on constate une amélioration de la précision de 48 points jusqu'à atteindre 78% qui est encore une très faible valeur de précision (Tableau 6.2).

Méthode III : Réduction de la taille (Figure 6.6.III) La méthode III n'est pas vraiment une nouvelle méthode, mais permet de se rendre compte de l'incidence qu'à la taille de l'image sur les résultats.

Ces résultats étonnants n'exposent qu'une très faible baisse de la précision des calculs. En effet, quand les images de contours sont utilisées comme dans la méthode originale, la précision atteint 90.22% (Tableau 6.2), mais avec les images de gradients, cette méthode atteint les 92.66%. De plus, cette méthode accélère les performances comme le montre le tableau 6.1. Avec seulement cette réduction de la taille de l'image, on obtient une vitesse de calcul de 0.366 image par seconde soit six fois plus que la méthode originale. Les résultats obtenus avec cette méthode classent cette méthode parmi les meilleures. De fait, l'utilisation des images de gradients à la place des images de contours permet d'obtenir des résultats en précision quasi identiques à la méthode originale avec 92.66% pour une vitesse d'exécution de 0.53 image par seconde.

Méthode IV : Dépendance + Réduction + Réinitialisation (Figure 6.6.IV) La méthode IV est la concaténation des méthodes II et III. En effet, les très bons résultats obtenus avec la méthode III nous ont poussé à poursuivre dans cette voie. Dans cette méthode nous avons utilisé cette réduction de la taille couplée à la dépendance spatiale et pour limiter la propagation de l'erreur, le calcul sur toute l'image, réduite, est appliqué toutes les dix images.

Les résultats obtenus par cette méthode se sont avérés très intéressants. Effectivement, la vitesse est grandement améliorée avec seulement une légère perte de précision par rapport à la méthode originale. Quand les images de contours sont utilisées, la vitesse atteint les 1.66 images par seconde (Tableau 6.1), et quand l'usage est fait des images de gradients, l'exécution s'effectue à 1.96 images par seconde (Tableau 6.1) soit près de 32 fois plus que la méthode originale. Ces résultats sont obtenus respectivement pour une précision de 84.80% et 91.12% (Tableau 6.2).

Il apparaît que cette méthode est la plus performante des optimisations développées. Comme l'illustrent les résultats, une telle augmentation de la vitesse d'exécution sans perte de précision importante est très favorable.

Méthode V : Réduction de l'échelle (Figure 6.6.V) Comme l'illustrent les résultats de la méthode III, une perte de précision apparaît lors de l'utilisation d'image plus petite qu'originalement. Afin de répondre à ce défaut, nous avons évalué une méthode où la réduction de la taille ne

sert qu'à une estimation préliminaire. Une fois cette première estimation réalisée sur l'image réduite, une seconde estimation est réalisée afin d'affiner ce calcul.

Les résultats en précision obtenus par cette méthode sont très bons. En effet, avec les images de gradients, la précision de cette méthode dépasse la précision originale pour atteindre une valeur de 94.92% (contre 90.28% avec les images de contours, Tableau 6.2). Ces résultats en font la méthode la plus performante sur le plan de la précision. Par contre, comme le montre le tableau 6.1, la vitesse d'exécution n'est pas à la hauteur de la méthode IV. En effet, avec seulement 0.367 image traitée par seconde pour les méthodes utilisant les gradients et 0.27 pour celles utilisant les contours, cette méthode se retrouve parmi les moins performantes. Pour cette raison, nous estimons que le gain de précision n'est pas à la hauteur du gain en vitesse de la méthode IV et cette méthode ne nous semble donc pas la meilleure des méthodes testées.

Par ailleurs, cette optimisation ne faisant pas appel à la dépendance entre plusieurs images nous l'avons évaluée sur la base de données de Kong et al. Cette base de données bien que non séquencée est plus proche de notre application finale, car elle prend en compte des images dans des environnements non urbains.

Les résultats obtenus sont inférieurs aux résultats sur routes, mais restent très bons avec 88.01% de précision pour la méthode V contre 88.46% pour la méthode originale (Tableau 6.3).

Méthode VI : Dépendance + Réduction de l'échelle + Réinitialisation (Figure 6.6.VI)
Dans une recherche d'accélération de cette méthode V, nous avons évalué une méthode étant la concaténation des méthodes II et V.

Cette méthode s'est avérée étonnamment mauvaise sur le plan de la précision avec des résultats à peine supérieurs à la méthode II, 79.74% pour la méthode avec les images de contours et 83.26% avec les images de gradients (Tableau 6.2). En théorie, cette méthode aurait pu offrir les avantages des deux méthodes précédentes, mais la perte de performance peut s'expliquer par une propagation des mauvaises estimations augmentée par un trop grand nombre de dépendances. Cette méthode est par contre performante sur le plan de la vitesse, mais pas autant que la méthode IV avec seulement 0.66 et 0.71 image par secondes respectivement pour les images de contours et les images de gradients (Tableau 6.1).

6.2.3.4 Conclusion

De cette étude menée dans ce chapitre, nous pouvons extraire deux méthodes principales. La méthode IV qui selon les précédents résultats est la meilleure optimisation évaluée et la méthode V qui, bien que moins rapide, offre des résultats en précision très performants.

Aussi, nous avons pu prouver que les images de gradients utilisées à la place des images de contours pour la sélection du point de fuite offrent de bien meilleurs résultats. Ces résultats sont liés à la plus grande quantité d'informations contenues dans ces images par rapport aux images de contours.

Enfin, par les résultats de la méthode IV, nous offrons une optimisation permettant d'améliorer par un facteur 32 l'algorithme original en terme de temps de calcul. Reste à implémenter cet algorithme et ces deux optimisations dans un environnement prévu pour réaliser des traitements temps réel afin d'analyser si cette méthode est compatible avec une possible utilisation en tant que méthode préliminaire de segmentation. En effet, les résultats obtenus ne permettent pas de conclure si cet algorithme optimisé peut répondre aux contraintes temps réel étant donné qu'il ne s'exécute à ce stade qu'à 2Hz. Il offre, en revanche, une piste pour un développement optimisé.

6.3 Conclusion de la segmentation des données acquises

Ce chapitre présente des travaux préliminaires d'analyse et de segmentation des données acquises par le système présenté dans le chapitre 5. Nous avons présenté une méthode de segmentation visuelle de chemin améliorée essentiellement en vitesse d'exécution qui permettrait de définir de manière précise où se trouve la zone roulable et une méthode de segmentation du nuage de points par des méthodes de l'état de l'art.

La mise en correspondance de ces deux méthodes pourrait permettre d'analyser les données acquisises en limitant la zone de recherche dans le nuage de points et donc en réduisant l'impact calculatoire sur le système. Par cela, nous exposons des moyens qui permettront d'atteindre l'objectif du projet dans lequel cette thèse s'inscrit, l'aide au pilotage pour la détection de dangers.

6.4 Conclusion sur le système de perception d'un environnement naturel

Le chapitre 5 - *Reconstruction par données LIDAR et odométrie visuelle* - évalue le concept sur le plan algorithmique et le chapitre 6 - *Segmentation des données acquises* - développe des pistes pour l'utiliser. Ainsi, le chapitre 5 expose les résultats en comparaison avec une vérité terrain qui prouvent que d'une part, l'algorithme d'optimisation développé apporte un gain significatif à cette méthode et que d'autre part, la méthode ainsi optimisée est cohérente globalement. Cette cohérence globale se traduit par une reconstruction de l'environnement, proche de la vérité terrain. Par contre, ils illustrent aussi son manque de cohérence locale ne permettant pas de répondre pleinement à la problématique. En effet, les scans LIDAR ne sont pas positionnés de manière très précise et la reconstruction n'est par conséquent pas exploitable pour la détection des irrégularités du sol, signes des obstacles négatifs. Cependant, des pistes de réflexion pour améliorer la méthode de telle sorte qu'elle réponde pleinement à notre problématique sont indiquées.

Par ailleurs, le chapitre 6 page 129 présente des méthodes utiles à l'analyse de la reconstruction créée. La segmentation du nuage de points et l'extraction d'obstacles positifs montrent les posibilités offertes par cette reconstruction quant à l'analyse de l'environnement. De plus, l'optimisation d'une méthode de segmentation visuelle de chemin offre une brique supplémentaire dans la construction d'un système complet de détection des dangers en environnement naturel.

Nous pouvons conclure que bien que ce concept ne réponde pas pleinement à la problématique à ce stade, il offre une vision intéressante d'une reconstruction d'environnement basée uniquement sur la perception sans utilisation de GPS. Cette reconstruction, est exploitable pour la segmentation d'obstacles positifs en environnement naturel.

Ainsi, à l'instar de l'état de l'art, nous ne répondons pas complètement à notre problématique. En revanche, les résultats prouvent que le concept est cohérent avec cette dernière, même si les limites techniques ne permettent pas d'obtenir des résultats pleinement satisfaisants à ce jour. D'une part, son optimisation peut être matérielle par l'amélioration des capteurs existants et par l'ajout par exemple d'une centrale inertielle pour améliorer la cohérence locale de la reconstruction. D'autre part, l'amélioration algorithmique du calcul d'odométrie et de la construction du nuage de points laisse envisager qu'à terme ce concept y réponde pleinement.

Troisième partie

Restitution à l'opérateur

Introduction à la partie III Le chapitre 1 présente la problématique *sécuriser le pilotage des véhicules militaires avec un système de perception d'environnement.* Comme la première et la deuxième partie l'ont exposé, nous apportons à cette problématique une réponse par la reconstruction de l'environnement frontal au véhicule.

Cette dernière partie s'intéresse, elle, à l'amélioration de la perception de l'environnement par le biais des moyens de restitution. En effet, actuellement dans les véhicules militaires sont utilisés des écrans de relativement petites tailles, et il apparaît que ce moyen de restitution n'est pas le plus adapté à la conduite.

Ainsi, nous exposons dans ce chapitre les résultats d'une étude menée afin d'évaluer un possible moyen de restitution alternatif : « le casque de réalité virtuelle » aussi appelé *Head Mounted Display, visiocasque ou casque de visualisation.* Ce casque restitue de manière stéréoscopique un flux vidéo et a la particularité d'enregistrer les mouvements réalisés par la tête de l'utilisateur qui peut servir d'actionneur. Dans notre étude nous cherchons à évaluer l'efficacité d'un tel moyen de restitution pour le pilotage sur écran. Au travers d'une étude comparative réalisée dans une série d'exercices spécifiques sur un simulateur de pilotage de camion, nous exposons des résultats justifiant la poursuite de travaux sur ce type d'affichage.

Chapitre 7

Pilotage à l'aide d'un casque de réalité virtuelle

Sommaire

7.1 Moyens de restitutions **151**
7.1.1 La situation actuelle dans les véhicules 152
7.1.2 Le casque de réalité virtuelle 153
7.2 Méthodologie de l'expérience **155**
7.2.1 Hypothèse 155
7.2.2 Variables 155
7.2.3 Protocole expérimental 155
7.2.4 Tâches 157
7.3 Résultats **160**
7.3.1 Résultats et analyse 160
7.4 Conclusion **161**

L'encyclopédie Larousse définit la perception comme *[un] événement cognitif dans lequel un stimulus ou un objet, présent dans l'environnement immédiat d'un individu,* ***lui est représenté*** *dans son activité psychologique interne, en principe de façon consciente.* Il apparaît de manière évidente que la représentation tient une place importante dans le processus de perception d'un individu. Or, quand le pilotage est réalisé sur écran, dans les véhicules blindés, les moyens de restitutions sont de petits écrans mal définis.

Le moyen par lequel la restitution est faite au pilote est aussi une question cruciale pour les aider dans leurs opérations. Ainsi, une question se pose : le remplacement de ce moyen de restitution peut-il être source d'amélioration du pilotage des véhicules blindés ?

Pour répondre à cette question, nous réalisons une étude offrant une réponse préliminaire comparant dans un environnement simulé, les performances de pilotage entre une restitution sur un écran standard et par un moyen de restitution où la tête et le regard sont mis à profit. Ce moyen de restitution est le casque de réalité virtuelle.

Ce chapitre expose l'étude réalisée dans le cadre de cette comparaison. Premièrement, nous présentons la situation dans les véhicules militaires avec l'exemple du véhicule *Titus* de Nexter et le matériel de restitution utilisé dans notre expérience, le casque de réalité virtuelle *Oculus Rift*. Deuxièmement, nous détaillons l'expérience mise en place. Troisièmement, les résultats obtenus sont exposés et nous analysons de leur signification.

7.1 Moyens de restitutions

Intéressons-nous tout d'abord à la situation actuelle dans les véhicules militaires quant aux moyens de restitution et de l'alternative que nous avons choisi d'appliquer, le casque de réalité virtuelle.

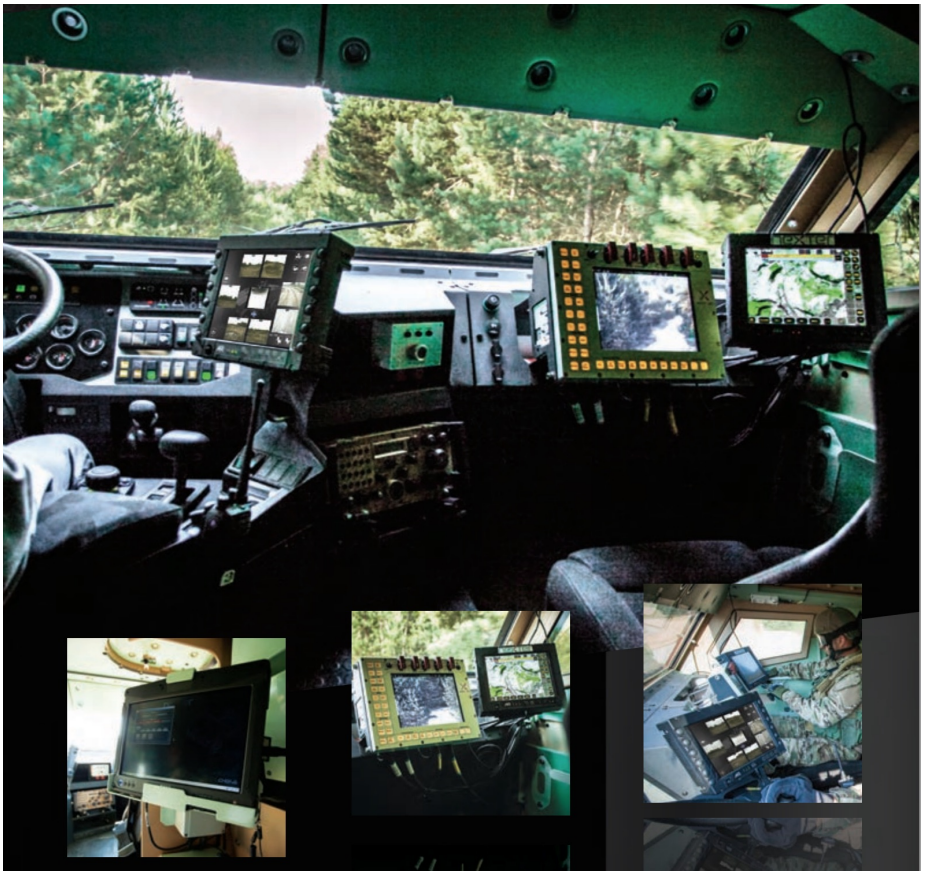


FIGURE 7.1 – L'habitacle dans le *Titus*. Vue générale des postes avants, à droite le poste tireur et à gauche le poste pilote. En détail, dans l'imagette de gauche, le poste-chef est exposé. Au milieu, la photographie représente le poste tireur et à droite la photographie est prise du poste pilote.

7.1.1 La situation actuelle dans les véhicules

Nous choisissons le *Titus* comme exemple de véhicule militaire, car il est représentatif de la situation actuelle. En effet, ce véhicule a été présenté en 2014 au salon Euro Satory [Défense-Blog, 2014]. C'est un véhicule muni d'un pare-brise, mais la configuration de ses moyens de restitutions est représentative des véhicules modernes qu'ils soient munis de pare-brise ou entièrement blindés. De fait, les véhicules modernes sont équipés d'un ensemble d'écrans sur lesquels des fonctions mineures et majeures sont installées et entre lesquelles les opérateurs doivent continuellement alterner.

L'habitacle Dans un véhicule militaire, l'espace est très restreint (Figure 7.1). L'épaisseur des blindages et la prolifération d'équipements réduisent de plus en plus la place disponible, car a contrario la taille des véhicules est critique. Un chef de véhicule possède un écran, le pilote en a un et le tireur en a deux. Comme expliqué en introduction, la place étant limitée les interfaces sont regroupées sur le moins de moniteurs possibles. Les opérateurs doivent donc continuellement naviguer entre les fonctions pour avoir des informations. De plus, les écrans sont habituellement de petite taille (de 9 à 14 pouces), ce qui rend difficile la lisibilité des informations. Enfin, ces mêmes écrans servent à piloter le véhicule ou les robots quand la vue directe (pare-brise et épiscopes) n'est pas disponible.

Les interfaces Sur ces moniteurs un grand nombre d'informations est affichable. Les opérateurs peuvent visualiser, entres autres, les images acquises par l'ensemble des caméras positionnées autour du véhicule (6 sur le *Titus*), les reproductions numériques des vues de leurs épiscopes, des cartes tactiques, les images acquises par les caméras des robots (*Nerva*), ou encore des informations sur l'état du véhicule. La plupart de ces fonctions nécessitant d'être accessibles à tout moment, les moniteurs sont surchargés d'informations ce qui rend le pilotage difficile quand il doit être réalisé en utilisant ces moniteurs.

Le choix d'un casque de Réalité Virtuelle Afin d'aider ces soldats dans leur pilotage sur caméra, nous utilisons une technologie ancienne, mais qui connaît un intérêt croissant, le casque de réalité virtuelle. La décision d'utiliser cette technologie a été réalisée pour plusieurs raisons :

1. Le prix (300$ pour l'*Oculus Rift* en 2014)
2. L'encombrement réduit
3. L'acquisition des mouvements de la tête peut être utilisée comme actionneur et est intégré au casque
4. Une technologie poussée par le grand public [1]

Ces casques de réalité virtuelle (Figure 7.2) affichent des informations directement devant les yeux de l'utilisateur. Le but est d'immerger le sujet dans l'environnement virtuel afin qu'il réalise des tâches. Les casques actuels sont équipés d'écrans haute définition et de capteurs permettant de capturer les mouvements réalisés par la tête du sujet afin de savoir où il souhaite regarder dans la scène virtuelle. Leurs avantages sont multiples. Ils permettent d'afficher beaucoup d'informations, car l'œil est situé très proche de l'écran et le champ de vision est de 110°. Ils permettent à l'opérateur de regarder partout sans avoir à utiliser de contrôleur. En effet, couplé à un système d'acquisition d'images 3D panoramique, comme il en existe de plus en plus [2], ces casques semblent apporter un grand avantage par rapport à de simples écrans.

Ainsi, cette étude s'intéresse particulièrement à montrer l'amélioration éventuelle par rapport aux écrans standards. Elle compare l'efficacité du pilotage sur un certain nombre de sujets conduisant en environnement simulé avec un casque de réalité virtuelle et un écran standard. Nos tests ont été réalisés dans un environnement simulé afin de bénéficier de la flexibilité qu'il offre dans les transformations possibles de l'environnement.

7.1.2 Le casque de réalité virtuelle

En développant l'*Oculus Rift*, la société *Oculus VR* a amélioré le système de casque de réalité virtuelle et grandement augmenté l'intérêt envers ces systèmes. Des recherches de toutes sortes sont apparues en 2014, de l'amélioration du suivi du mouvement casque [Rabet, 2014] au développement de jeux médicaux [Blaha and Gupta, 2014]. De plus, l'armée norvégienne a récemment montré son intérêt pour cette technologie pour ses recherches d'aide à la conduite sur le « blindage transparent » [Hsu, 2014]. Par ailleurs, plusieurs recherches s'intéressent aux pilotages à distance [Bug, 2014], mais aucune ne compare les solutions actuelles, les écrans, aux casques de réalité virtuelle.

L'Oculus Rift L'*Oculus Rift* est développé par la société *Oculus VR*, filiale de *Facebook* depuis 2014. Le projet fait suite à un financement collaboratif sur le site *Kickstarter.com* et la sortie de la première version commerciale (CM1) est prévue pour l'année 2016. Le système se présente sous

1. Contrairement au siècle dernier, la technologie militaire tire désormais beaucoup de la technologie du grand public. En effet, les matériels militaires, comme les matériels du grand public deviennent des consommables. La technologie grand public évoluant beaucoup plus vite que la technologie militaire, cette dernière est vite dépassée.
2. Exemple de caméras 3D panoramiques, ou toutes directions : https://ozo.nokia.com/, http://www.panocam3d.com/camera.html, http://shop.360heros.com/

FIGURE 7.2 – Vue frontale d'un Oculus Rift DK2. Cette photographie illustre le casque dans lequel sont installés un écran, la centrale inertielle et la caméra infrarouge permettant le suivi en translation du casque.

la forme d'un masque recouvrant les yeux et attaché au visage par une sangle à l'arrière du crâne [Desai et al., 2014]. Un écran plat OLED est placé à quelques centimètres en face des yeux. L'écran affiche une image stéréoscopique déformée numériquement afin de contrer la distorsion optique créée par les lentilles qui servent à augmenter le champ de vision [3]. L'écran est placé sur le plan focal de ces lentilles, de telle sorte que l'image virtuelle ainsi créée se trouve projetée à l'infini. Plusieurs capteurs détectent les mouvements de tête du sujet pour modifier son point de vue dans la scène afin de créer l'immersion.

Les différents modèles d'Oculus Rift En 2015, l'*Oculus Rift* est décliné en trois versions : le DK1, le DK2 et le futur CM1. Les DK1 et DK2 sont les deux premiers kits de développement réservés à un usage professionnel. Le CM1 est la première version vendue au grand public. Le DK1 est sorti en 2013, le DK2 en 2014 et le CM1 devrait sortir en 2016. Notre étude est réalisée avec le kit de développement DK2.

Ce modèle [4] est équipé d'un écran 5.7 pouces de résolution à 1920 x 1080 pixels. La reconnaissance des mouvements est cadencée à 1 MHz et se fait sur 6 degrés de liberté grâce à un gyroscope, un accéléromètre et un magnétomètre pour les rotations et un suivi de leds infrarouges (40) pour les translations. Le casque et la caméra infrarouge sont munis de câbles au format USB3 et l'ordinateur, le casque et la caméra sont reliés entre eux. Le casque offre un angle de vue vertical d'environ 90 degrés pour un champ de vision horizontal d'environ 110 degrés. Il pèse près de 400g.

Afin de fonctionner avec l'*Oculus Rift*, les jeux et applications doivent être développés dans cette optique. Pour cela, le constructeur fournit un kit de développement logiciel « SDK » gratuit permettant d'interfacer le casque et le programme informatique développé.

Les effets secondaires Les casques de réalité virtuelle, comme la plupart des technologies de réalité virtuelle, peuvent provoquer la cinétose [5], aussi appelée « mal du simulateur ». Ce malaise est lié à la perte de repère spatial lorsqu'on lit dans un engin en mouvement. Notre étude ne tente

3. Explication du système de lentilles de l'Oculus Rift : http://www.gatinel.com/recherche-formation/casque-de-realite-virtuelle-oculus-rift/

4. Spécifications de l'*Oculus Rift* DK2 : https://vrwiki.wikispaces.com/Oculus+Rift+Development+kit+2

5. Cinétose : *Elle débute par un état de malaise avec vertige, baisse de vigilance, état nauséeux conduisant à des vomissements.* http://dictionnaire.academie-medecine.fr/?q=cin%C3%A9tose

pas d'expliquer ou de quantifier ce malaise, mais notons qu'un grand nombre de nos utilisateurs a été victime de ces effets. De fait, la cinétose conduisant à des malaises et des vomissements, les utilisateurs en ayant été victimes n'ont pas terminé l'expérience et ne sont donc pas pris en compte dans nos résultats.

7.2 Méthodologie de l'expérience

Nous définissons ici le **point de vue** (PDV) qui dans cette étude représente l'endroit où l'utilisateur regarde dans son habitacle virtuel. Ainsi, un changement de point de vue représente le fait de déplacer son regard à l'aide des actionneurs prévus à cet effet.

7.2.1 Hypothèse

Le remplacement du moyen de restitution standard qu'est l'écran de petite ou moyenne taille par un casque de réalité virtuelle peut être une source d'amélioration du pilotage des véhicules blindés. Afin d'apporter un élément de réponse, nous démontrons en partie cette hypothèse par la mise en place d'une expérience sur une simulation de camion civil dans des épreuves proches des missions réalisées avec des véhicules militaires.

Ainsi, nous démontrons l'hypothèse : **Le remplacement du moyen de restitution standard qu'est l'écran de moyenne taille par un casque de réalité virtuelle *Oculus Rift* DK2 peut être une source d'amélioration du pilotage de camion en environnement non routier simulé.**

7.2.2 Variables

Suivant la méthode rappelée par [Légal, 2015] : *L'expérimentation teste, en terme de causalité, l'effet (l'impact) d'une ou plusieurs variable(s) indépendante(s) sur une ou plusieurs mesure(s) ou variable(s) dépendante(s).* Ainsi, nous définissons nos variables indépendantes et dépendantes.

Variable indépendante Dans notre expérience, la variable indépendante est le moyen de restitution. Cette variable est soit un écran 17 pouces, soit un casque de réalité virtuelle. Cette variable permet d'établir deux échantillons à comparer.

Variable dépendante La variable dépendante est la performance de pilotage. Cette performance est mesurée suivant deux métriques : le temps nécessaire à la complétion des différentes tâches et les erreurs réalisées dans ces tâches. Cette variable quantifie et qualifie la réponse apportée par nos sujets à l'expérience lors de la modification de la variable indépendante.

Les utilisateurs L'expérience est réalisée avec 44 personnes. Le groupe d'utilisateurs est hétérogène afin de limiter le biais qu'aurait pu apporter l'utilisation d'une population homogène. Il se compose de personnes âgées de 13 à 60 ans, de femmes, d'hommes et d'enfants, d'étudiants, de professeurs, de techniciens et de conducteurs de bus. Pour chacun de ces utilisateurs, nous avons appliqué le changement de la variable indépendante.

7.2.3 Protocole expérimental

Exposons maintenant le protocole expérimental au travers du matériel, du changement des variables dépendantes et des mesures réalisées.

FIGURE 7.3 – L'installation utilisée par les sujets de l'expérience, avec l'Oculus Rift (à gauche) et avec l'écran (à droite). Sur ces images on peut voir le volant. Sur l'image de gauche, on aperçoit le pédalier, l'écran est présent comme moyen de surveillance par l'opérateur en charge de l'expérience. Sur l'image de droite, on voit le joystick.

Matériel Dans notre expérience nous avons utilisé un certain nombre de matériels que nous présentons ici.

Les sujets devaient utiliser deux types de restitutions (Figure 7.3) : l'Oculus Rift pour tester les compétences avec les casques de réalité virtuelle et un écran 17 pouces plus grand que ceux installés dans les véhicules militaires, mais représentatif de la configuration de ceux-ci.

L'*Oculus Rift* possède les caractéristiques suivantes :
- Résolution 960×1080 par œil
- Affichage stéréoscopique
- Rafraîchissement des images : 95Hz
- A un éloignement standard (50cm) le champs de vision est d'environ 60°

L'écran possède les caractéristiques suivantes :
- Résolution 1920×1080 pour les deux yeux
- Affichage monoculaire
- Rafraîchissement des images : 60Hz
- Champs de vision d'environ 110°

Pour changer le point de vue, l'utilisateur devait utiliser un joystick avec l'écran et sa tête avec l'oculus Rift.

Pour conduire le véhicule, un volant *Logitech G27* était utilisé. Ce volant est un volant de taille similaire à un volant de voiture. Il est équipé d'un retour de force offrant un retour haptique simulant la direction assistée. Couplé à ce volant un système de deux pédales permet l'accélération et le freinage. Sur le volant, un ensemble de boutons est installé ainsi que deux palonniers situés derrière le volant, mais nous ne les utilisons pas.

L'environnement simulé était créé grâce à l'éditeur de niveau présent dans le jeu *Euro Truck Simulator 2* (ETS2) (Figure 7.4). Ce jeu offre la possibilité grâce à un éditeur d'environnement de créer différents types de cartes et d'épreuves. De plus, il supporte à l'utilisation de l'*Oculus Rift*.

FIGURE 7.4 – L'habitacle simulé d'un camion dans le jeu EUro Truck Simulator 2.

Phases L'expérience est divisée en deux phases de 5 minutes chacune qui consistent à l'exécution des tâches avec l'un ou l'autre des moyens de restitution. Afin d'éviter un biais possible causé par la connaissance préalable du circuit nous alternons pour chaque sujet le moyen de restitution par lequel l'expérience commence.

De plus, chaque phase commence par une phase de tutoriel expliquant les tâches que le sujet devra réaliser pendant l'expérience.

Les métriques Les performances des sujets sont évaluées suivant différents critères :

(1) **Le temps nécessaire pour compléter les tâches :** Le temps nécessaire à la complétion de l'expérience est mesuré en secondes par une personne autre que l'utilisateur. Cette métrique est connue des utilisateurs afin qu'ils tentent de faire au plus vite.

(2) **Le nombre de pénalités :** Les tâches possèdent des obstacles (mur ou panneau) ou des chutes à éviter. Quand le camion entre en collision avec l'un de ces obstacles ou est victime d'une chute, une pénalité est comptabilisée. Cette mesure est connue des utilisateurs afin de les encourager à réaliser les tâches de manière précise.

(3) **Le taux de changement de point de vue :** L'environnement 3D permet à l'utilisateur de regarder tout autour de lui. Nous mesurons le taux auquel l'utilisateur change son point de vue afin de récupérer de l'information sur son environnement. Ce taux est représenté par les mouvements de tête réalisés avec l'*Oculus Rift* et par l'utilisation du joystick avec l'écran. Cette métrique est évaluée sur une échelle de 0 à 5 par un opérateur autre que l'utilisateur. 0 correspond à l'absence de changement de point de vue qui se traduit par un utilisateur gardant le regard fixé sur un point devant lui durant tout l'exercice. 5 signifie que le sujet observe tout autour de lui pour récupérer de l'information. Cette métrique n'est pas connue des utilisateurs pour ne pas biaiser leur façon d'appréhender l'expérience. En effet, nous souhaitions qu'ils utilisent les moyens de restitutions selon leur besoin.

7.2.4 Tâches

L'expérience compare les performances obtenues par l'utilisateur avec les deux moyens de restitutions différents. Cette comparaison est réalisée sur un ensemble de quatre tâches. Ces tâches sont des exercices à réaliser dans le simulateur de pilotage de camion dans un environnement non routier. Nous définissons ces quatre tâches ici.

(1) Le premier exercice (ou tâche) (Figure 7.5) évalue les performances en pilotage dans un environnement à la visibilité restreinte et où les manœuvres nécessitent une certaine précision. Il a pour but d'évaluer les performances dans des manœuvres difficiles sans connaissance des prochains virages. De fait, l'exercice est conçu avec plusieurs virages très serrés (entre 90 et 180 degrés) et de hauts murs entourant la piste empêchent l'utilisateur de voir la suite de

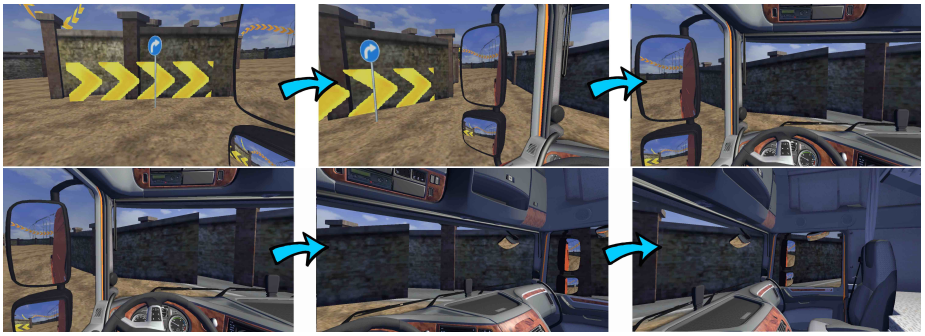


FIGURE 7.5 – Six images représentant des points de vue de gauche de la fenêtre gauche de l'habitacle jusqu'à la fenêtre droite. Cette image illustre les possibilités de changement de point de vue de l'utilisateur au cours de l'expérience. De plus, cette image illustre la situation de la première tâche : on visualise un virage à 90° entouré de grands murs.

FIGURE 7.6 – Image illustrant l'exercice 2. On visualise la plateforme sur laquelle l'utilisateur doit faire progresser son véhicule.

l'épreuve. Ainsi, le manque d'espace et de visibilité force l'utilisateur à prendre de l'information pour réaliser ses manœuvres en évitant les collisions avec les murs. Cet exercice est représentatif du pilotage dans des environnements urbains étroits non adaptés au gabarit des véhicules militaires.

(2) Le second exercice (ou tâche) (Figure 7.6) évalue les performances en pilotage dans un environnement où la visibilité est bonne, mais où la précision de pilotage doit être très forte. Contrairement au premier exercice l'intégralité de l'itinéraire à suivre est visible, mais le pilotage se fait à environ 5 mètres du sol sur une plateforme qui n'est pas plus large que le châssis du véhicule. Ainsi, les performances des moyens de restitution sont évaluées par la nécessité de la prise d'information antérieurement au virage. Cet exercice est représentatif du pilotage dans des environnements dangereux et où il est difficile de garder toutes les roues sur le chemin.

(3) Le troisième exercice (ou tâche) (Figure 7.7) évalue les performances dans le cadre de la recherche d'informations dans un environnement dense. Cette tâche demande aux utilisateurs de chercher des panneaux de signalisation particuliers, cachés dans un environnement à la

végétation dense gênant la visibilité. Quand le sujet trouve le panneau, il doit rouler dessus en évitant les panneaux interdits positionnés juste à côté de la cible. Dans cet exercice, une pénalité est attribuée si l'utilisateur percute un panneau interdit. Cette tâche est représentative de la nécessité des pilotes à rechercher de l'information pour amener leur véhicule à un endroit voulu en suivant des points de passages définis. De plus, cette mission est aussi représentative du pilotage de robots souvent utilisés pour détecter ou désamorcer des explosifs ou pour faire de la reconnaissance afin de récupérer des informations.

(4) Le quatrième et dernier exercice (ou tâche) (Figure 7.8) est un slalom à haute vitesse. Cet exercice évalue les gains apportés par la possibilité donnée à l'utilisateur de préparer ses virages en regardant là où il souhaite amener son véhicule. Dans cet exercice les pénalités sont comptabilisées lors de collisions avec les portes de passage.

FIGURE 7.7 – Exercice 3 où les sujets doit percuter le panneau parking sans percuter les panneaux stationnement interdit. Comme on peut le voir dans le rétroviseur gauche, le sujet a dû traverser une zone de végétation dense pour trouver les panneaux.

FIGURE 7.8 – Exercice 4, le slalom, où l'utilisateur doit traverser les portes le plus vite possible.

7.3 Résultats

Il est important de noter que pour la plupart des sujets, cette expérience était leur première avec un *Oculus Rift* ou avec un casque de réalité virtuelle en général. Malheureusement seuls 51% des utilisateurs ont terminé l'expérience, le reste ayant dû arrêter pour cause de malaise. Comme expliqué dans la section précédente, le nombre d'utilisateurs s'élevait à 44 personnes, mais seuls 23 ont terminé l'expérience. Les résultats exploités ne traitent que les performances des utilisateurs ayant terminé les deux phases de l'expérience.

Pour rappel, l'hypothèse de notre expérience est la suivante : *Le remplacement du moyen de restitution standard qu'est l'écran de petite ou moyenne taille par un casque de réalité virtuelle peut être une source d'amélioration du pilotage des véhicules blindés.*

7.3.1 Résultats et analyse

Les résultats sont mesurés suivant les trois métriques détaillées précédemment : le temps de complétion, le taux de changement de point de vue, et le nombre de pénalités.

Le temps de complétion Les résultats sur l'ensemble des tâches (Figure 7.9) montre un temps de complétion moyen quasiment identique. En effet, avec l'*Oculus Rift*, les utilisateurs réalisent un temps moyen de 4min et 43sec alors qu'avec l'écran il leur faut 4min 44sec pour terminer l'ensemble des tâches. De plus, la taille des boites dans la figure 7.9 illustre le fait que les résultats sont moins homogènes avec le casque de réalité virtuelle qu'avec l'écran.

En détail sur chaque tâche (Figure 7.10 page 162), on constate que les résultats sur les tâches (1), (3) et (4) donnent de meilleurs résultats avec le casque, mais que l'exercice (2) donne l'avantage à l'écran. Par ailleurs, la taille des boîtes illustre des dispersions similaires dans les différents exercices.

Ces résultats illustrent une faible amélioration au vu de notre hypothèse. Néanmoins, on constate que sur trois tâches sur quatre, l'utilisation du casque de réalité virtuelle se traduit par une nette amélioration des performances.

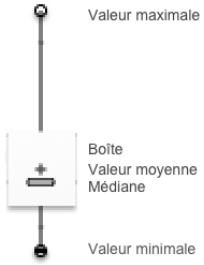
Le nombre de pénalités La figure 7.11 page 163 expose les résultats des pénalités réalisées par les utilisateurs dans les différentes tâches. On constate que l'utilisation du casque de réalité virtuelle réduit les pénalités sur les tâches (1), (3) et (4).

Ces résultats exposent que l'utilisation du casque de réalité virtuelle augmente la précision du pilotage de manière générale. Surtout, sur les (1) et (4), on constate une franche amélioration de la précision qui fait écho aux résultats en temps de complétion.

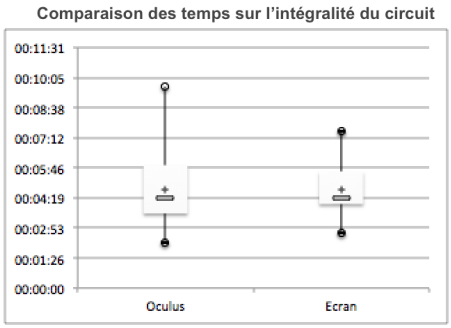
Ainsi, il apparaît que pour le pilotage à haute vitesse et les manœuvres difficiles, le casque de réalité virtuelle offre un avantage.

Le taux de changement de point de vue La figure 7.12 page 164 présente le taux avec lequel les utilisateurs ont changé leur point de vue afin de récupérer de l'information dans leur environnement. Les résultats exposent une nette supériorité de l'*Oculus Rift* sur l'écran. En effet, bien que les valeurs soient très dispersées, les moyennes traduisent ce résultat.

Ces résultats exposent une utilisation plus importante du changement de point de vue avec le casque de réalité virtuelle. Particulièrement, sur les exercices (1) et (4), cette amélioration est notable et permet de supposer que les bons résultats obtenus avec le casque de réalité virtuelle sont liés à cette forte utilisation des changements de points de vue. L'affirmation n'est pas possible au vu des exercices (2) et (3) où une prise d'information plus forte des utilisateurs ne conduit pas à une amélioration des performances.



(a) Détail d'une « boîte à moustache ». La boite représente la dispersion des valeurs. La croix représente la valeur moyenne. La barre représente la médiane. Les deux points extrêmes représentent les valeurs extrêmes.



(b) La comparaison des temps pour la complétion de toutes les tâches. Plus la valeur est basse, meilleur est le résultat.

FIGURE 7.9 – Explication de la représentation graphique des résultats et la comparaison des temps pour la complétion de toutes les tâches.

Cette étude laisse apparaître une amélioration des performances suite à l'utilisation du casque de réalité virtuelle, mais la liberté de changement de point de vue semble ne pas être le seul facteur permettant de définir la source de ce gain. De fait, le casque de réalité virtuelle apporte d'autres avantages que la seule liberté de changement de point de vue : la stéréoscopie, la fréquence de rafraîchissement, le champ de vision sont aussi des paramètres qu'il faudrait étudier indépendamment afin d'extraire l'impact de chacun sur le gain en pilotage.

En revanche, bien que les sources de cette amélioration ne soient pas définies, le casque de réalité virtuelle apporte un gain certain dans cette expérience en comparaison avec les écrans. Ainsi, dans notre expérience, notre hypothèse est validée en partie et confirme la nécessité de travailler sur des moyens de restitution plus adaptés au pilotage.

7.4 Conclusion

L'expérience menée dans ce chapitre expose des résultats en faveur de l'utilisation des casques de réalité virtuelle. De fait, les résultats illustrent une amélioration du pilotage dans le cadre de notre simulation de camion. Par cette expérience nous ne pouvons pas conclure que les casques de réalité virtuelle peuvent améliorer le pilotage des véhicules blindés, par contre, cette expérience donne une tendance positive à leur utilisation et invite à poursuivre les recherches sur l'utilisation de ce type de technologie.

En revanche, le taux d'abandons élevé alerte sur les limites de cette technologie et interroge sur son avenir, car sans l'amélioration de la tolérance des utilisateurs vis-à-vis de celle-ci, elle risque de ne pas être en mesure d'être utilisée dans des applications innovantes et ne pourra pas remplacer l'utilisation de moyen de restitutions conventionnels.

Ainsi, cette partie expose un moyen de restitution qui au vu des résultats obtenus pourrait devenir un moyen de restitution alternatif. En effet, si cette technologie s'améliore de manière à surpasser ces défauts initiaux, les avantages qu'elle offre (mains libres, liberté de point de vue, stéréoscopie et champ de vision large) la rendront très attractive pour des secteurs d'activités où la perception d'informations est cruciale.

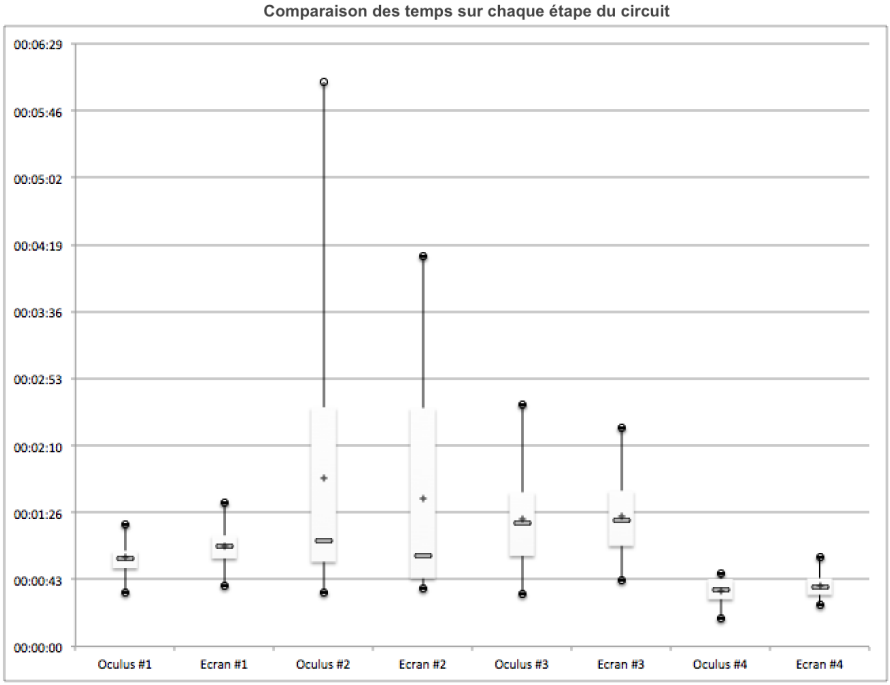


FIGURE 7.10 – La comparaison des temps pour la complétion de chaque tâche. Plus la valeur est basse, meilleur est le résultat. La boite représente la dispersion des valeurs. La croix représente la valeur moyenne. La barre représente la médiane. Les deux points extrêmes représentent les valeurs extrêmes.

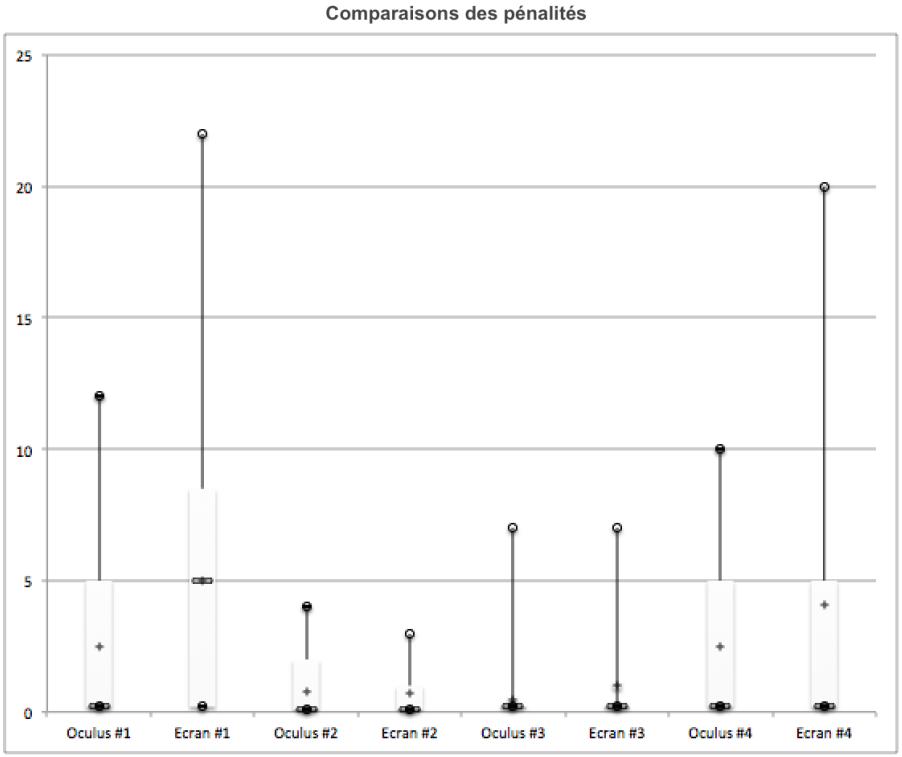


FIGURE 7.11 – La comparaison des pénalités sur chaque exercice. Plus la valeur est basse, meilleur est le résultat. La boite représente la dispersion des valeurs. La croix représente la valeur moyenne. La barre représente la médiane. Les deux points extrêmes représentent les valeurs extrêmes.

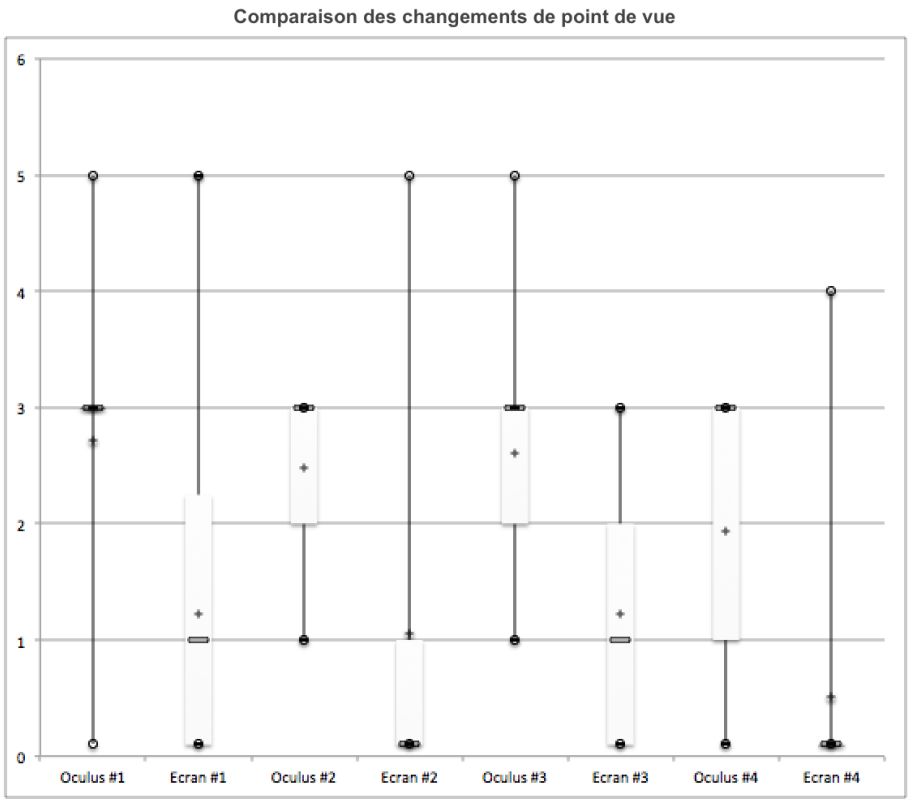


FIGURE 7.12 – La comparaison du taux de changement de point de vue sur chaque exercice. Noté entre 0 et 5, plus la valeur est haute meilleur est le résultat. La boite représente la dispersion des valeurs. La croix représente la valeur moyenne. La barre représente la médiane. Les deux points extrêmes représentent les valeurs extrêmes.

Chapitre 8

Conclusion et perspectives

8.1 Conclusion

La mortalité liée au transport motorisé est au cœur de nombreuses recherches. En effet, que les véhicules mis en cause soient des voitures, des engins de travaux ou de défense, les accidents sont inacceptables. Par conséquent, un grand effort est mené sur cette question de sécurisation du pilotage terrestre.

Cette question est adressée sur le plan de la protection des habitants, la mobilité, mais aussi de la perception permettant d'éviter des accidents en cas de situations difficiles. L'état de l'art présenté dans ce manuscrit présente la diversité des aides au pilotage qui ont permis de réduire grandement le nombre d'accidents sur les routes dans le domaine civil. En revanche, les conditions différentes des milieux dans lesquels évoluent les matériels de la défense ne permettent pas d'utiliser les mêmes techniques que celles employées dans le domaine civil. Bien que des aides au pilotage existent, comme l'a montré l'étude de l'état de l'art, aucune ne permet de sécuriser le pilotage des engins militaires dans les environnements naturels. Ainsi, nous avons défini la nécessité de *sécuriser le pilotage des véhicules militaires avec un système de perception d'environnement* en prenant en compte les contraintes liées à l'application militaire.

Nous répondons à cette problématique en réalisant la reconstruction de l'environnement proche du véhicule. L'analyse des situations d'intervention de ces systèmes a permis la définition d'un ensemble de contraintes orientant la réponse apportée. En effet, la partie d'état de l'art de ce manuscrit définit la réponse que nous apportons à cette problématique dans le respect des contraintes associées. Ainsi, la nécessité du système à être robuste à tous les environnements et à ne pas demander de calibration nous a orientés vers des choix de capteurs utilisables dans le plus de situations possibles.

Le choix du capteur LIDAR a été effectué comme une concession entre la discrétion et la nécessité de capturer l'environnement de manière très précise. Par conséquent, le choix s'est orienté vers un LIDAR dont la portée relativement faible ne met pas en péril la discrétion des opérateurs. De plus, la nécessité d'indépendance vis–à–vis des systèmes de positionnement globaux a orienté notre choix vers une solution de positionnement relatif passif. De fait, le positionnement se fait à l'aide d'une caméra stéréoscopique dont le traitement des images permet l'estimation du mouvement réalisé par le véhicule. De plus, la définition précise des obstacles que les véhicules peuvent rencontrer en environnement naturel offre une base de paramétrage des futurs algorithmes de détection d'obstacles.

Comme présenté dans la partie II, afin de valider notre concept et compte tenu de l'impossibilité de réaliser des tests préliminaires sur une plateforme militaire, nous avons construit une plateforme robotique. Cette plateforme robotique, bien que non représentative de notre application, permet d'évaluer le fonctionnement d'un tel système en environnement naturel. Cette plateforme équipée de l'ensemble des capteurs possède des capacités d'acquisition orientées vers l'avant du véhicule afin de reconstruire au mieux le futur environnement traversé par le véhicule.

La suite présente une solution algorithmique permettant de coupler les données des caméras stéréoscopiques et du LIDAR. Cette solution reconstruit l'environnement suivant la trajectoire réalisée par le véhicule et a été présentée dans l'article [RICAUD et al., 2015b]. Par ailleurs, la mise

en place d'une vérité terrain par l'acquisition de notre environnement de test a permis d'évaluer cette méthode. Suivant les métriques définies, cet algorithme s'est avéré relativement cohérent avec la vérité terrain, mais pas suffisamment pour répondre à notre application. Ainsi, l'analyse des résultats a conclu sur la nécessité d'une optimisation de la méthode pour en accélérer l'exécution. Pour optimiser cet algorithme, nous avons développé une solution d'estimation du mouvement de la trajectoire par utilisation du recalage d'image monoculaire. L'exécution rapide de cet algorithme permet d'initialiser l'algorithme précédent à une valeur proche de sa solution finale et accélère son exécution. La comparaison avec les données de la vérité terrain démontre une forte amélioration de la méthode. En effet, la reconstruction engendrée est cohérente globalement. En revanche, la reconstruction locale de la cartographie n'est pas assez précise pour obtenir des résultats suffisamment fins permettant la détection d'obstacles négatifs. En réponse à ces résultats, nous proposons d'utiliser les données d'une centrale inertielle pour positionner les scans LIDAR les uns par rapport aux autres. Quand bien même, ce système n'est pas encore fonctionnel pour une utilisation sur des véhicules roulant à haute vitesse. De fait, les résultats démontrent que la trajectoire et donc, la reconstruction, ne sont pas représentatives de la vérité terrain quand le véhicule roule à une vitesse supérieure à 1 m/s.

Par conséquent, notre expérience montre que ce système peut réaliser une reconstruction 3D à faible vitesse de l'environnement, cohérente globalement, en se basant uniquement sur des méthodes de perception de l'environnement local.

Par ailleurs, afin d'utiliser cette reconstruction pour la détection de dangers sur la trajectoire du véhicule, nous présentons ensuite des développements réalisés pour segmenter les données récoltées. En effet, nous exposons une optimisation d'une méthode de segmentation visuelle de chemin qui grâce à notre travail pourrait servir de segmentation préliminaire et ainsi réduire la complexité des traitements sur le nuage de points reconstruit. Les résultats obtenus avec cette optimisation ont été présentés dans [RICAUD et al., 2014]. De surcroît, nous présentons une méthode de segmentation du nuage de points reconstruit par notre algorithme. Cette segmentation met en exergue la hiérarchisation du nuage et l'extraction des obstacles positifs. De fait, ces travaux montrent que notre reconstruction peut être utilisée pour l'extraction d'obstacles positifs. Cette méthode apporte une piste de développement pour une future aide au pilotage.

Enfin, la dernière partie de ce manuscrit présente les travaux réalisés dans le cadre de la restitution des informations à l'opérateur. Cette étude a démontré l'intérêt des casques de réalité virtuelle pour le pilotage de véhicules, face aux écrans standards. En revanche, ces matériels nécessitent de réduire le malaise qu'ils provoquent lors d'une utilisation prolongée pour pouvoir devenir une alternative envisageable aux solutions actuelles. Les résultats de cette étude sont présentés dans l'article [RICAUD et al., 2015a].

Ainsi, l'ensemble de ce manuscrit répond partiellement à la problématique de sécurisation du pilotage, par une réflexion sur les méthodes et matériels pouvant percevoir l'environnement, les algorithmes pouvant le percevoir et l'analyser et enfin une solution de restitution qui permettra à l'opérateur d'appréhender son environnement.

Contrairement aux méthodes de l'état de l'art, nous n'utilisons pas de système de positionnement absolu, lui offrant ainsi la possibilité de fonctionner seul. Cependant, ne pas utiliser de système GPS ou Gallileo rend difficile l'obtention d'un positionnement précis. Certes, notre solution ne répond pas pleinement à la problématique, mais ces limites sont relatives aux capteurs et à la non-optimisation des algorithmes. L'intérêt de cette recherche réside dans le concept mis en place et les preuves qu'il peut fonctionner pour détecter des obstacles positifs.

Ainsi, notre travail offre des perspectives intéressantes pour de futures applications dans le domaine d'application militaire.

8.2 Perspectives

Capteurs Un des problèmes évidents est que le système est conçu pour fonctionner de jour. Or c'est de nuit que les opérateurs ont le plus besoin de solution pour gagner en visibilité sur l'environnement. Notre concept peut fonctionner dans ce cas par le remplacement ou l'adaptation de capteurs pouvant « voir » la nuit. En effet, les véhicules militaires sont actuellement équipés de systèmes visioniques thermiques offrant une visibilité qui pourrait être suffisante pour l'exécution d'algorithmes d'odométrie visuelle. De plus, les systèmes LIDAR ne sont pas gênés par la luminosité et la reconstruction pourrait être envisagée.

Par ailleurs, la baisse des prix de la technologie LIDAR rend envisageable l'utilisation de capteurs multicouches qui permettrait de gagner en précision sur la reconstruction locale. En effet, cette reconstruction pourrait gagner en résolution et ainsi mieux décrire l'environnement. En revanche, la nécéssité de l'équipement LIDAR à posséder une portée relativement faible reste importante pour les problématiques de discrétion.

Notre choix s'est porté sur l'élaboration d'une solution n'utilisant pas le positionnement par GPS, car une stratégie de véhicule indépendant était appliquée. Dans les faits, il s'avère que les opérateurs sont désormais tous équipés de smartphone dont ils se servent pour se localiser. Notre démarche est donc certainement trop contrainte vis-à-vis de la situation réelle. Elle n'en demeure pas moins intéressante, car un changement de mentalité n'est pas inenvisageable au vu de la criticité des technologies sur le champ de bataille. Cependant, l'amélioration de la précision globale pourrait être envisagée par l'utilisation d'un GPS si le critère de discrétion n'est plus prédominant.

Applications industrielles Sur le plan des applications possibles du concept développé dans ce manuscrit, nous avons exposé les possibilités de ce système à détecter les obstacles positifs relativement imposants. En effet, les travaux préliminaires sur la segmentation exposent la difficulté de segmenter finement le sol et donc de détecter les petits obstacles pouvant être assimilés au sol. Par contre, cette reconstruction est utilisable pour une application sur des véhicules de grandes tailles.

Malgré tout, il est à noter que cette application se fait sous réserve d'une amélioration de la méthode afin de fonctionner à des vitesses proches de notre cadre d'application, soit $40km/h$. Pour cela, il est nécessaire d'apporter plusieurs altérations au système.

1. Le remplacement de la caméra stéréoscopique par une caméra dont la fréquence d'acquisition dépasse $30Hz$ permettrait d'acquérir des images de manière quasi temps réel et ne limiterait pas des traitements très rapides.
2. Ensuite, comme évoqué en conclusion dans le chapitre 5.1, le remplacement du servomoteur par une roue codeuse permettrait d'optimiser l'acquisition des angles de rotation et d'accélérer la vitesse de rotation sur l'axe de roulis. Cette accélération serait bénéfique en cas d'augmentation de la vitesse qui nécessiterait des révolutions plus courtes.
3. Puis, il est aussi possible de réduire l'amplitude des mouvements du servomoteur pour gagner en résolution.
4. Enfin, un travail important devra être réalisé sur les algorithmes d'odométrie pour les accélérer. Pour obtenir ces accélérations, des optimisations de la méthode pourront être appliquées comme dans le chapitre 6. Par ailleurs, il est aussi envisageable de porter ces développements sur une solution embarquée dédiée afin de limiter l'impact du système sur les traitements et optimiser leur exécution à bas niveau.

Enfin, aucune expérimentation n'a été menée afin de connaître les gains apportés par notre concept. Une étude intéressante serait de valider le concept en simulateur avec des performances idéales et dégradées, pour connaître le niveau de détail nécessaire de la reconstruction dans les futures aides au pilotage.

Une autre application de nos travaux est la caractérisation des obstacles. En effet, notre algorithme est directement exploitable pour effectuer une reconstruction à faible vitesse sur de courtes distances. Ainsi, une des applications envisagées est d'utiliser cette reconstruction non pas pour détecter les obstacles, mais pour les caractériser une fois qu'ils sont repérés et ainsi transmettre l'information aux véhicules alliés. De fait, tous les véhicules n'ont pas les mêmes capacités de franchissement et un obstacle pour un véhicule ne l'est pas forcément pour un autre. Cette application servirait donc d'aide à la décision quant au choix de l'itinéraire emprunté par les différents véhicules disponibles.

Problèmes applicatifs Des problèmes particuliers liés à l'emploi d'une méthode d'aide au pilotage nécessitent encore des solutions inexistantes. En effet, nous avons défini en début de manuscrit que les obstacles négatifs sont un danger important pour les véhicules et nous avons évoqué des solutions pour les détecter. En revanche, comment faire quand ces obstacles négatifs sont remplis d'eau, où encore plus compliqué, de boue liquide ? Dans ce cas, bien que des solutions existent pour scanner des surfaces sous-marines vues du ciel [Irish and Lillycrop, 1999], les propriétés de la boue rendent ce problème extrêmement complexe. Ceci nous amène à un autre problème, la nature du sol. Effectivement, les obstacles sont une source de dangers « visible », mais le sol peut être dangereux tout en étant plat. Comme évoqué précédemment, des trous d'eau ou de boue peuvent être catastrophiques pour le véhicule et son personnel, mais un véhicule peut aussi se bloquer dans un banc de sable ou tout simplement un piège recouvert d'une fine planche de bois.

Il est donc nécessaire de poursuivre l'étude de cette problématique. La poursuite suivant le même concept est pertinente avec un travail d'optimisation important à réaliser. Cependant, à ce concept devront d'ajouter des moyens de détermination de la nature du sol, et des méthodes lui permettant de fonctionner avec une faible visibilité afin de répondre complétement à cette problématique.

Enfin, pour pouvoir être pleinement efficace, le système complet devra, en plus de résoudre tous les problèmes de détection, fonctionner dans toutes les situations dégradées et difficiles telles qu'une visibilité proche du néant (brouillard, tempête de sable, de neige, brouillage) et des températures extrêmes tout en résistant à la rigueur des aléas du combat.

Annexe A

Publications

Conférences internationales avec comité de relecture

1. ***General Road Detection Algorithm - A Computational Improvement.***, Bruno Ricaud, Bogdan Stanciulescu, Amaury Breheret, ICPRAM 2014 *5th International Conference on Pattern Recognition Applications and Methods*
2. ***Are Virtual Reality headsets efficient for remote driving ?***, Bruno Ricaud, Cyril Joly, Arnaud de la Fortelle, RSS 2015 *Road Safety and Simulation International Conference*
3. ***Non urban driver assistance with 2D tilting laser reconstruction***, Bruno Ricaud, Cyril Joly, Arnaud de la Fortelle, MMT 2015 *The 9th International Symposium on Mobile Mapping Technology*

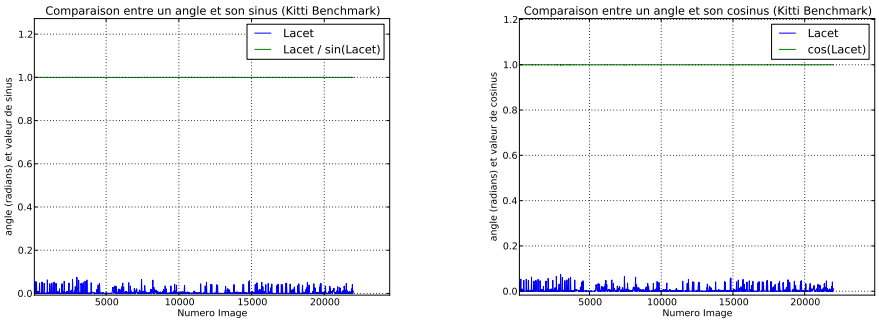
Annexe B

Justification des approximations de l'estimation monoculaire de la rotation

Dans le cadre du calcul d'odométrie visuelle nous réalisons l'optimisation de la méthode stéréoscopique par l'ajout d'une estimation visuelle de la rotation. Dans cette estimation nous appliquons trois hypothèses d'approximation dont nous justifions la véracité ici.

A.1 Approximations des petits angles

Dans notre méthode nous réalisons une approximation sur la valeur des angles estimés. En effet, nous faisons l'approximation que les angles que nous cherchons à calculer sont très petits et qu'ils leur valeur est équivalente à leur sinus et que leur cosinus est égale à 1. Comme l'illustrent les figures B.1 et B.3 page suivante, pour chaque angle de tangage et de lacet dans les 25000 images du jeu de données *Kitti* le ratio $angle/\sin(angle)$ est équivalent à 1 et le $\cos(angle)$ est aussi équivalent à 1. En effet, la figure B.2 page suivante expose que pour des angles assez petits ces cas sont vérifiés et la figure B.4 page 174 nous montre que les angles de la vérité terrain sur les jeux de données *Kitti* ont des valeurs moyennes inférieures à 0.01 radian. L'utilisation des petits angles et donc pleinement justifiée et notre approximation semble est cohérente avec la réalité.



(a) Comparaison entre l'angle de lacet et son sinus.

(b) Comparaison entre l'angle de lacet et son cosinus.

Figure B.1 – Comparaisons entre l'angle de lacet et son sinus et son cosinus.

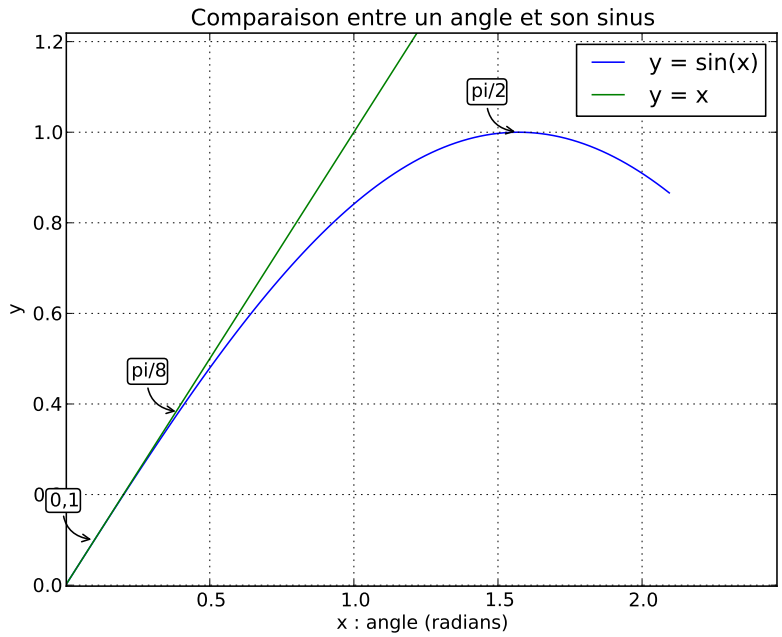
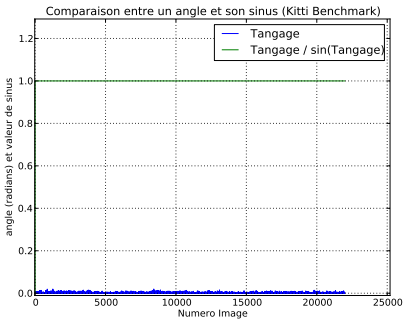
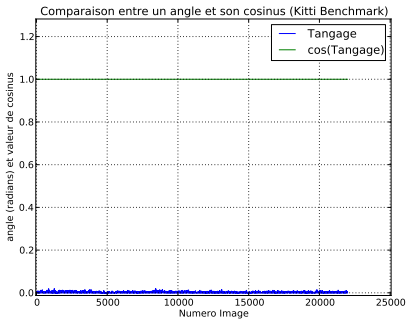


FIGURE B.2 – Schéma illustrant l'approximation aux petits angles pour le cas du sinus.



(a) Comparaison entre l'angle de tangage et son sinus.



(b) Comparaison entre l'angle de tangage et son cosinus.

FIGURE B.3 – Comparaisons entre l'angle de tangage et son sinus et son cosinus.

B.2 Les translations sont dépendantes de la distance au points

Dans notre algorithme nous faisons l'hypothèse que les translations sont négligeables. Cette approximation est liée au fait que nous considérons les points comme étant à l'infini. Nous expliquons ici pourquoi considérer des angles à l'infini nous permet de négliger les translations.

Si on considère les translations dans notre équation 5.16 page 116, on a :

$$P' = RP + t \tag{B.1}$$

où P' est le nouveau point image, P est le point d'origine, R est la rotation et t la translation qui permettent de passer du point P au point P'.

On a alors :

$$\begin{aligned} R^{-1}(P' - t) &= P \\ R^{-1}P' - R^{-1}t &= P \\ R^{T}P' - R^{T}t &= P \end{aligned} \tag{B.2}$$

donc :

$$\begin{aligned} x &= x' + e_{\theta y}z' - t_x - e_{\theta y}t_z \\ y &= y' + e_{\theta x}z' - t_y - e_{\theta x}t_z \\ z &= z' + e_{\theta y}x' + e_{\theta x}y' - e_{\theta y}t_x - e_{\theta x}t_y - t_z \end{aligned} \tag{B.3}$$

En appliquant la projection sur le plan image avec $X' = x'/z'$, $Y' = y'/z'$, $X = x/z$ et $Y = y/z$ on obtient :

$$\begin{aligned} X &= \frac{X' + e_{\theta y} - \frac{t_x}{z'} - e_{\theta y}\frac{t_z}{z'}}{1 + e_{\theta y}X' + e_{\theta x}Y' - e_{\theta y}\frac{t_x}{z'} - e_{\theta x}\frac{t_y}{z'} - \frac{t_z}{z'}} \\ Y &= \frac{Y' + e_{\theta x} - \frac{t_y}{z'} - e_{\theta x}\frac{t_z}{z'}}{1 + e_{\theta y}X' + e_{\theta x}Y' - e_{\theta y}\frac{t_x}{z'} - e_{\theta x}\frac{t_y}{z'} - \frac{t_z}{z'}} \end{aligned} \tag{B.4}$$

Il apparaît clairement que tous les termes de translations sont facteur de $1/z'$ qui représente la distance frontal au point. Dans notre expérience, le véhicule se déplace entre 0 et $5m/s$, et la caméra acquière des images à une fréquence de 15 Hz. Entre chaque image, le véhicule se déplace donc d'une valeur comprise entre 0 et 33 cm. Ainsi, pour un point situé à 10 m devant le véhicule, le ratio t/z sera compris entre 0 et 0,0033. Bien que ce rapport ne soit pas infinitésimal, il est suffisamment faible pour que notre approximation soir cohérente.

C.3 Le roulis négligé

Dans notre méthode nous négligeons le roulis pour le calcul des rotations. Comme le montre la figure B.4 page suivante, le roulis n'est pas réellement négligeable. Malgré tout, son approximation comme une valeur nulle nous donne une bonne approximation de la rotation générale.

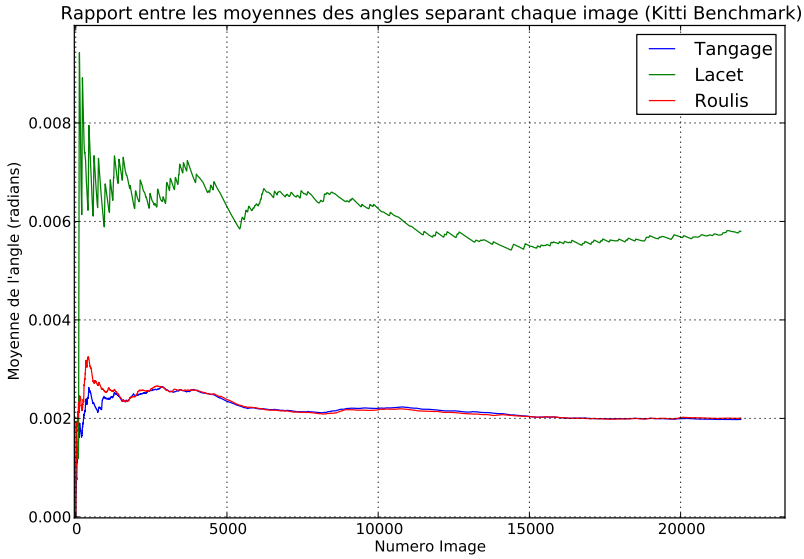


FIGURE B.4 – Comparaison des valeurs des angles de lacet, de tangage et de roulis sur l'ensemble du benchmark *Kitti*.

Bibliographie

135, R. S. C. (2011). Environmental Conditions and Test Procedures for Airborne Equipment. *Rtca/Do-160F*. (Cité en page 6.)

A. de La Fortelle (2012). Cooperative ITS : technologies and societal impacts. In *Workshop on intelligent vehicules Conf. (Mines ParisTech)*. (Cité en page 7.)

Abbeel, P. (2015). Nonlinear Dynamical Systems. pages 1–17. http://www.cs.berkeley.edu/~pabbeel/cs287-fa11/slides/EKF_UKF--v2.pdf. (Cité en page 36.)

Abdulmasih, D. and Oikonomidis, P. (2011). Operational integrity monitoring for military vehicle's integrated vetronics architecture. *Safety, 2011 6th*, pages 1–6. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6136919. (Cité en pages 5 et 21.)

Abuhadrous, I., Ammoun, S., Nashashibi, F., Goulette, F., and Laurgeau, C. (2004). Digitizing and 3D modeling of urban environments and roads using vehicle-borne laser scanner system. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, 1. (Cité en page 24.)

Alcantarilla, P. F., Yebes, J. J., Almazán, J., and Bergasa, L. M. (2012). On combining visual SLAM and dense scene flow to increase the robustness of localization and mapping in dynamic environments. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1290–1297, Department of Electronics, University of Alcalá, Alcalá de Henares, Madrid, Spain. http://www.scopus.com/inward/record.url?eid=2-s2.0-84864479912&partnerID=40&md5=aece98f7bb826a2b1e494da1038c3398. (Cité en page 27.)

Aldebaran (2014). Romeo. http://www.industrie-techno.com/sous-le-capot-de-romeo-le-dernier-prototype-humanoide-d-aldebaran.33090. (Cité en page 27.)

Amar, F. B., Jarrault, P., Bidaud, P., and Grand, C. (2009). Analysis and optimization of obstacle clearance of articulated rovers. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pages 4128–4133. (Cité en page 42.)

Anguelov, D., Dulong, C., Filip, D., Frueh, C., Lafon, S., Lyon, R., Ogale, A., Vincent, L., and Weaver, J. (2010). GooGle CapturinG Street leVel. *Most*. (Cité en page 24.)

ASE-GNSS (2008). Les programmes europeens de navigation par satellite Galileo. Technical report. http://www.gsa.europa.eu/sites/default/files/eu_gnss_prog_leaflet4_fr.pdf. (Cité en page 25.)

AutonomousParking (2014). autonomous parking. http://www.autocar.co.uk/car-news/industry/bmw-reveals-new-self-parking-autonomous-technology. (Cité en page 19.)

Baker, H. H. and Binford, T. (1981). Depth From Edge and Intensity Based Stereo. *The 7th international joint conference on Artificial intelligence*, pages 631–636. (Cité en page 61.)

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). SURF : Speeded up robust features. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3951 LNCS :404–417. (Cité en page 29.)

Beauchemin, S. S. and Barron, J. L. (1995). The computation of optical flow. (Cité en page 27.)

Berg, M. d., Cheong, O., Kreveld, M. v., and Overmars, M. (2008). *Computational Geometry : Algorithms and Applications*. Springer-Verlag TELOS, Santa Clara, CA, USA, 3rd ed. edition. (Cité en page 130.)

Bertozzi, M., Bombini, L., and Broggi, A. (2011). VIAC : An out of ordinary experiment. *(IV), 2011 IEEE*, (Iv) :175–180. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5940531. (Cité en page 7.)

Besl, J. (1988). Segmentation Through Variable-O rder Surface Fitting. *Analysis*, I(2) :167–192. (Cité en page 132.)

Blaha, J. and Gupta, M. (2014). Diplopia : A virtual reality game designed to help amblyopics. *Proceedings - IEEE Virtual Reality*, pages 163–164. (Cité en page 153.)

BMW (2015). BMW DST. http://www.bmw.com/com/en/insights/technology/technology_guide/articles/electric_power_steering.html. (Cité en page 19.)

Bobruk, J. and Austin, D. (2002). Laser Motion Detection and Hypothesis Tracking from a Mobile Platform. *Acra*. (Cité en page 52.)

Boland, E. (2002). The Pinhole Camera. *The Yale Review*, pages 1–6. http://onlinelibrary.wiley.com/doi/10.1111/0044-0124.00570/abstract. (Cité en page 56.)

Bosch (2015a). Bosch ABS. http://www.bosch-motorcycle.com/en/de/fahrsicherheit_fuer_zweiraeder/motorrad_abs/base_seite_14/description_benefits.html. (Cité en page 18.)

Bosch (2015b). ESP Bosch. http://www.bosch-mobility-solutions.fr/fr_fr/fr/specials_9/specials_for_more_driving_safety_9/bosch_esp_10/esp__facts_20/esp_technik_10/esp_questions_and_answers_16.html. (Cité en page 18.)

Bosse, M. and Zlot, R. (2008). Map matching and data association for large-scale two-dimensional laser scan-based SLAM. *International Journal of Robotics Research*, 27(6) :667–691. http://www.scopus.com/inward/record.url?eid=2-s2.0-44649175597&partnerID=40&md5=0e987d8d7c392ade7150c18d84f3d33a. (Cité en page 31.)

Bosse, M., Zlot, R., and Flick, P. (2012). Zebedee Design of a Spring-Mounted 3-D Range. 28(5) :1–15. (Cité en page 71.)

Bracewell, R. (1965). Pentagram Notation for Cross Correlation. *The Fourier Transform and Its Applications*, New York : :pp. 46 and 243. (Cité en page 30.)

Braman, T. and Grossman, O. (2006). Designing vibration and shock isolation systems for micro electrical machined based inertial measurement units. *Record - IEEE PLANS, Position Location and Navigation Symposium*, 2006 :400–404. (Cité en page 27.)

Broggi, A., Cappalunga, A., and Caraffi, C. (2010). Terramax vision at the urban challenge 2007. *Intelligent*, 11(1) :194–205. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5415547. (Cité en pages 6 et 7.)

Broggi, A., Caraffi, C., and Fedriga, R. (2005). Obstacle detection with stereo vision for off-road vehicle navigation. *Computer Vision and*. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1565369. (Cité en pages 49, 51 et 59.)

Bug, D. (2014). Oculus Rift Control of a Mobile Robot. (Cité en page 153.)

Calonder, M. (2006). EKF SLAM vs. FastSLAM {A Comparison}. *Cvlab-Report-2010-001*, pages 1–5. http://infoscience.epfl.ch/record/146805/files/ekf_fastslam_comp.pdf. (Cité en page 39.)

Camera, D. D. (2015). SLS-2. http://www.david-3d.com/fr/products/sls-2. (Cité en page 65.)

Caraffi, C., Cattani, S., and Grisleri, P. (2007). Off-Road Path and Obstacle Detection Using Decision Networks and Stereo Vision. *IEEE Transactions on Intelligent Transportation Systems*, 8(4) :607–618. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4382936. (Cité en page 7.)

Caspi, D., Kiryati, N., and Shamir, J. (1998). Range Imaging with Adaptive Color Structured light. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5) :470–480. (Cité en page 66.)

CETMF (2008). Le GPS différentiel (DGPS) et temps réel (GPS RTK). pages 1–6. http://www.eau-mer-fleuves.cerema.fr/IMG/pdf/DGPS_et_RTK_V2_cle598ba6.pdf. (Cité en page 25.)

Chang, C. and Chatterjee, S. (1992). Quantization error analysis in stereo vision. *[1992] Conference Record of the Twenty-Sixth Asilomar Conference on Signals, Systems & Computers*, pages 1037–1041. (Cité en page 58.)

Chen, S. Y., Li, Y. F., and Zhang, J. (2008). Vision processing for realtime 3-D data acquisition based on coded structured light. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 17(2) :167–176. (Cité en page 66.)

Chung, S. Y. and Huang, H. P. (2008). Simultaneous topological map prediction and moving object trajectory prediction in unknown environments. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1594–1599. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4650945. (Cité en page 52.)

Civera, J., Grasa, O. G., Davison, A. J., and Montiel, J. M. M. (2009). 1-point RANSAC for EKF-based structure from motion. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pages 3498–3504. (Cité en page 97.)

Cole, D. and Newman, P. (2006). Using laser range data for 3D SLAM in outdoor environments. *Robotics and Automation, 2006. ICRA ...*, 2006 :1556–1563. http://www.scopus.com/inward/record.url?eid=2-s2.0-33845621601&partnerID=40&md5=dfd0c60097e696ad8de621d4a0ec9347http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1641929. (Cité en page 55.)

Combelles-Siegel, P. and Géré, F. (2001). Les mythes et les réalités du concept de "zéro mort" : comparaison franco américaine. (Cité en page 1.)

DARPA (2015). DARPA web site. http://www.darpa.mil/. (Cité en page 7.)

Defense.gouv.fr (2012). Naviguer/localiser : GPS et Galileo. http://www.defense.gouv.fr/actualites/dossiers/l-espace-au-profit-des-operations-militaires/les-fonctions-spatiales-au-service-des-operations/naviguer-localiser-gps-et-galileo. (Cité en page 26.)

Defense.gouv.fr (2015a). Aravis. http://www.defense.gouv.fr/dga/equipement/terrestre/le-vbhp-aravis-vehicule-blinde-hautement-protege-aravis. (Cité en page 22.)

Defense.gouv.fr (2015b). Equipement Felin. http://www.defense.gouv.fr/terre/equipements/armement-individuel-et-collectif/felin-fantassin-a-equipement-et-liaisons-integres. (Cité en page 4.)

Defense.gouv.fr (2015c). VBCI. http://www.defense.gouv.fr/terre/equipements/vehicules/vbci-vehicule-blinde-de-combat-de-l-infanterie. (Cité en page 22.)

Desai, P. R., Desai, P. N., Ajmera, K. D., and Mehta, K. (2014). A Review Paper on Oculus Rift-A Virtual. *International Journal of Engineering Trends and Technology (IJETT)*, 13(4) :175–179. (Cité en page 154.)

Devernay, F., Rhône-alpes, I. G., and De, L. (2011). Cinéma 3D et géométrie. (Cité en page 60.)

DFRRobot (2013). Devastator Tank Mobile Platform. http://www.dfrobot.com/index.php?route=product/product&product_id=1219&search=deva&description=true#.ViCqphDhAqJ. (Cité en page 80.)

Diesel, J. (1987). Integration of GPS/INS for maximum velocity accuracy. *Institute of Navigation, National Technical Meeting, Anaheim, CA*, (January) :58–65. http://www.csa.com/partners/viewrecord.php?requester=gs&collection=TRD&recid=A8817330AH. (Cité en pages 24 et 27.)

Dubbelman, G. (2006). Vision based Obstacle Detection for both Day and Night Conditions. (Cité en pages 48 et 50.)

Dubbelman, G., van der Mark, W., van den Heuvel, J. C., and Groen, F. C. a. (2007). Obstacle detection during day and night conditions using stereo vision. *International Conference on Intelligent Robots and Systems*, pages 109–116. (Cité en page 51.)

Dunn, D., Higgins, W. E., and Member, S. (1995). Optimal Gabor Filters for Texture Segmentation. 4(7). (Cité en page 134.)

Dupire, G. (2012). Il faut achever le "zéro mort" ! *Le Monde*. http://www.lemonde.fr/idees/article/2012/04/13/il-faut-achever-le-zero-mort_1685470_3232.html. (Cité en page 1.)

Durrant-Whyte, H. and Bailey, T. (2006a). Simultaneous localization and mapping : part I. *Robotics & Automation Magazine, ...* http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1638022. (Cité en pages 31, 32 et 39.)

Durrant-Whyte, H. and Bailey, T. (2006b). Simultaneous localization and mapping : part II. *Robotics & Automation Magazine, ...*, (September). http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1638022. (Cité en pages 32 et 55.)

Défense-Blog (2014). Titus 6x6 multi-role high-mobility protected vehicle of nexter systems will be displayed at eurosatory 2014. http ://defence-blog.com/army/titus-6x6-multi-role-high-mobility-protected-vehicle-of-nexter-systems-will-be- displayed-at-eurosatory-2014.html. (Cité en page 152.)

EGNOS (2015). SBAS. http://egnos-portal.gsa.europa.eu/discover-egnos/about-egnos/what-sbas. (Cité en page 25.)

Espino-Corsino, J., Steux, B., and ElHamzaoui, O. (2011). Safe navigating system for indoor environments. *The 5th International Conference on Automation, Robotics and Applications*, pages 419–423. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6144920. (Cité en pages 7, 55 et 86.)

Estrada, C., Neira, J., and Tardos, J. (2005). Hierarchical slam : Real-time accurate mapping of large environments. *Robotics, IEEE Transactions on*, 21(4) :588–596. (Cité en page 31.)

Fioraio, N. and Konolige, K. (2011). Realtime Visual and Point Cloud SLAM. *Computer*, pages 1–3. (Cité en page 31.)

Foix, S. and Aleny, G. (2011). Lock-in Time-of-Flight (ToF) Cameras : A Survey. 11(3) :1–11. (Cité en page 65.)

Fritz, H., Gern, A., and Schiemenz, H. (2004). CHAUFFEUR Assistant : a driver assistance system for commercial vehicles based on fusion of advanced ACC and lane keeping. *Intelligent Vehicles*, (Vc) :495–500. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1336433. (Cité en pages 6, 18 et 134.)

Gallup, D., Frahm, J.-M., Mordohai, P., Pollefeys, M., and Hill, C. (2008). Variable Baseline / Resolution Stereo. *IEEE Conference on Computer Vision and Pattern Recognition (2008)*, pages 1–8. (Cité en pages 58 et 59.)

Garage, W. (2011). Opencv. *Open Source Computer Vision Library*. (Cité en page 82.)

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving ? the KITTI vision benchmark suite. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. (Cité en pages 27, 98, 101 et 138.)

Geiger, A., Roser, M., and Urtasun, R. (2011a). Efficient Large-Scale Stereo Matching. In *Computer Vision – ACCV 2010*, number 1, pages 25–38. http://link.springer.com/10.1007/978-3-642-19315-6_3. (Cité en pages 56 et 62.)

Geiger, A., Ziegler, J., and Stiller, C. (2011b). StereoScan : Dense 3d reconstruction in real-time. *IEEE Intelligent Vehicles Symposium, Proceedings*, (Iv) :963–968. (Cité en pages 27, 56 et 62.)

Gidel, S., Checchin, P., Blanc, C., Chateau, T., and Trassoudaine, L. (2010). Pedestrian Detection and Tracking in an Urban Environment Using a Multilayer Laser Scanner. *IEEE Transactions on Intelligent Transportation Systems*, 11(3) :579–588. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5444908. (Cité en page 55.)

Gizzare, D. (2004). Distinctive Image Features from Scale-invariant Keypoints. pages 1–28. (Cité en page 29.)

Glonass (2015). Www.glonass-iac.ru. https://www.glonass-iac.ru/en/guide/. (Cité en page 25.)

Gluckman, J. and Nayar, S. (1998). Ego-Motion and Omnidirectional Cameras. *Computer Vision, 1998. Sixth . . .*, pages 999–1005. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=710838. (Cité en page 27.)

gov.cn (2003). Le programme de positionnement par satellite chinois Beidu . Technical report. en.beidu.gov.cn. (Cité en page 25.)

Grégoire, J. (2014). *Coordination de robots mobiles par affectation de priorites Priority-based coordination of mobile robots*. PhD thesis. (Cité en pages 6 et 18.)

Griffin, P. M., Narasimhan, L. S., and Yee, S. R. (1992). Generation of uniquely encoded light patterns for range data acquisition. *Pattern Recognition*, 25(6) :609 – 616. (Cité en page 66.)

Grimson, W. E. L. (1981). From images to surfaces : a computational study of the human early visual system. *Cambridge, MA*. (Cité en page 61.)

Grisetti, G. and Kuummerle, R. (2010). A tutorial on graph-based SLAM. *Intelligent . . .*, 2(4) :31–43. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5681215http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5681215. (Cité en page 40.)

Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23(1) :34–46. (Cité en page 31.)

Grussenmeyer, P., Droulez, J., Recherches, D. D., Merminod, B., Bennequin, D., Chaperon, T., France, T., Thibault, G., and Senior, C. (2013). École Doctorale Mathématiques , Sciences de l ' Information et de l ' Ingénieur Consolidation de relevés laser d ' intérieurs construits : pour une approche probabiliste initialisée par géolocalisation. (Cité en page 69.)

Guibert, N. (2011). Avec les robots guerriers, la guerre va changer de visage. *Le Monde*. http://www.lemonde.fr/societe/article/2011/11/12/avec-les-robots-guerriers-la-guerre-va-changer-de-visage_1602870_3224.html#8IR62Q24vxquD1eA.99. (Cité en page 6.)

Hadsell, R., Sermanet, P., and Ben, J. (2009). Learning long-range vision for autonomous off-road driving. *Journal of Field . . .*, 26(2) :120–144. http://doi.wiley.com/10.1002/rob.20276http://onlinelibrary.wiley.com/doi/10.1002/rob.20276/abstract. (Cité en page 7.)

Hager, G. D. and Durrant-Whyte, H. F. (1986). Information and multi-sensor coordination. In *UAI '86 : Proceedings of the Second Annual Conference on Uncertainty in Artificial Intelligence, University of Pennsylvania, Philadelphia, PA, USA, August 8-10, 1986*, pages 381–394. (Cité en page 32.)

Harris, C. and Stephens, M. (1988). A Combined Corner and Edge Detector. *Procedings of the Alvey Vision Conference 1988*, pages 147–151. http://www.bmva.org/bmvc/1988/avc-88-023.html. (Cité en page 29.)

Hartley, R. I. and Sturm, P. (1997). Triangulation. *Computer Vision and Image Understanding*, 68(2) :146–157. (Cité en page 60.)

Heckman, N., Lalonde, J. F., Vandapel, N., and Hebert, M. (2007). Potential negative obstacle detection by occlusion labeling. *IEEE International Conference on Intelligent Robots and Systems*, pages 2168–2173. (Cité en page 51.)

Hervier, T. (2012). Accurate 3D maps from depth images and motion sensors. (3) :5291–5297. (Cité en page 24.)

Hofmann-Wellenhof, B, L. H. W. E. (2008). Gnss–global navigation satellite systems : Gps, glonass, galileo, and more. (Cité en page 25.)

Holmstrom, S. T. S., Baran, U., and Urey, H. (2014). MEMS Laser Scanners : A Review. *Journal of Microelectromechanical Systems*, 23(2) :259–275. (Cité en page 69.)

Homm, F., Kaempchen, N., Ota, J., and Burschka, D. (2010). Efficient occupancy grid computation on the GPU with lidar and radar for road boundary detection. *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 1006–1013. (Cité en page 52.)

Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. (Cité en page 27.)

Houguet, E. (2010). Detecteur de fatigue. *La tribune*. http://www.latribuneauto.com/reportages-67-3711-comment-fonctionne-le-detecteur-de-fatigue-de-la-volkswagen-passat-.html. (Cité en page 19.)

Howard, a. (2008). Real-time stereo visual odometry for autonomous ground vehicles. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3946–3952. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4651147. (Cité en page 26.)

Hsu, J. (2014). Norwegian army drives tanks using oculus rift vr goggle. http://spectrum.ieee.org/tech-talk/computing/hardware/norwegian-army-oculus-rift-vr-goggle. (Cité en page 153.)

Icasualties (2015). icasulaties.com. http://icasualties.org/OEF/Fatalities.aspx. (Cité en page 1.)

Institute, D. T. (2015). New possibilities in the industrial world with robot coworkers and ros. http://www.dti.dk/services/new-possibilities-in-the-industrial-world-with-robot-coworkers-and-ros/33062. (Cité en page 81.)

Intensificateur (2015). intensificateur de lumière. http://www.astrosurf.com/luxorion/photoamplificateur-vision-nuit.htm. (Cité en page 22.)

Irish, J. L. and Lillycrop, W. (1999). Scanning laser mapping of the coastal zone : the {SHOALS} system. *{ISPRS} Journal of Photogrammetry and Remote Sensing*, 54(2–3) :123 – 129. (Cité en page 168.)

Jakubowski, M. B. (2008). Lidar Review for the Sierra Nevada Adaptative Management Project (SNAMP). Technical report. (Cité en pages 27 et 68.)

Jarrault, P. (2013). Optimisation des capacites de franchissement des robots mobiles hybrides " roues-pattes ". (Cité en pages 42 et 43.)

Joly, C. (2010). Contributions aux methodes de localisation et cartographie simultanees par vision omnidirectionnelle. (Cité en page 35.)

Joly, C. and Rives, P. (2009). Building consistent local submaps with omnidirectional SLAM. *. . .), 2009 IEEE 12th International Conference on*, pages 2180–2187. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5457550. (Cité en page 40.)

Julier, S., Uhlmann, J., and Durrant-Whyte, H. F. (2001). A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators. 46(12) :1908–1913. (Cité en page 35.)

Julier, S. J. and Uhlmann, J. K. (1997). A New Extension of the Kalman Filter to Nonlinear Systems. *Int. symp. aerospace/defense sensing, simul. and controls*. https://www.cs.unc.edu/~welch/kalman/media/pdf/Julier1997_SPIE_KF.pdf. (Cité en page 35.)

Jung, H. G., Cho, Y. H., Yoon, P. J., and Kim, J. (2008). Scanning laser radar-based target position designation for parking aid system. *IEEE Transactions on Intelligent Transportation Systems*, 9(3) :406–424. (Cité en pages 19 et 26.)

Katramados, I., Crumpler, S., Breckon, T. P., and Road, S. (2008). space fusion and temporal analysis. (2007). (Cité en page 7.)

Kazik, T., Kneip, L., Nikolic, J., Pollefeys, M., and Siegwart, R. (2012). Real-time 6D stereo Visual Odometry with non-overlapping fields of view. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1529–1536. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6247843. (Cité en pages 26 et 27.)

Kilpela, A. (2004). *Pulsed time-of-flight laser range finder techniques for fast, high precision measurement applications*. (Cité en page 64.)

Kitt, B., Geiger, A., and Lategahn, H. (2010). Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme. *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 486–492. (Cité en pages 27, 96, 97 et 98.)

Klein, G. and Murray, D. (2008). Improving the agility of keyframe-based SLAM. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5303 LNCS, pages 802–815. (Cité en page 29.)

Kolb, a., Barth, E., Koch, R., and Larsen, R. (2009). Time-of-Flight Sensors in Computer Graphics. *Eurographics 2009 - State of the Art Reports*, pages 119–134. http://www.cg.informatik.uni-siegen.de/data/Publications/2009/kolb09tof-star.pdf. (Cité en page 64.)

Kong, H., Audibert, J.-Y., and Ponce, J. (2010). General road detection from a single image. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 19(8) :2211–20. http://www.ncbi.nlm.nih.gov/pubmed/20371404. (Cité en pages 133, 134 et 137.)

Kramm, S. (2011). Stéréovision : Généralités & géométrie épipolaire Introduction Deux types de productions Généralisation Contraintes & difficultés de l'appariement. (Cité en page 61.)

Kuhnert, K.-D. (2008). Concept and implementation of a software system on the Autonomous Mobile Outdoor Robot AMOR. *2008 IEEE International Conference on Industrial Technology*, pages 889–894. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4608567http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4621234. (Cité en page 7.)

Kümmerle, R., Steder, B., and Dornhege, C. (2011). Large scale graph-based SLAM using aerial images as prior information. *Autonomous ...*, 30(1) :25–39. http://www.scopus.com/inward/record.url?eid=2-s2.0-79951552967&partnerID=40&md5=1e9e7e30e724db86e881c9adc1a38683http://www.springerlink.com/index/E61282664L64Q768.pdf. (Cité en page 40.)

Labayrade, R. (2003). A single framework for vehicle roll, pitch, yaw estimation and obstacles detection by stereovision. *Intelligent Vehicles Symposium, 2003*. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1212878. (Cité en page 49.)

Labayrade, R., Aubert, D., and Tarel, J.-P. (2002). Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation. *Intelligent Vehicle Symposium, 2002. IEEE*, 2. (Cité en page 61.)

Lange, R. (2000). Time-of-flight distance measurement with with custom custom solid-state image sensors in CMOS / CCD-technology. *Sensors Peterborough NH*, 222 :223. http://dokumentix.ub.uni-siegen.de/opus/volltexte/2006/178/. (Cité en page 63.)

Langley, R. B. (1998). Rtk Gps. *GPS World*, pages 70–75. http://www2.unb.ca/gge/Resources/gpsworld.september98.pdf. (Cité en page 25.)

Larson, J. and Trivedi, M. (2011a). Lidar based off-road negative obstacle detection and analysis. *Intelligent Transportation Systems (ITSC), ...* http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6083105. (Cité en page 6.)

Larson, J. and Trivedi, M. (2011b). Lidar based off-road negative obstacle detection and analysis. *Intelligent Transportation Systems (ITSC), ...* http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6083105. (Cité en pages 50, 51, 55 et 85.)

Larson, J., Trivedi, M., and Bruch, M. (2011). Off-road terrain traversability analysis and hazard avoidance for UGVs. pages 1–7. http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA536652. (Cité en pages 51, 55, 71 et 85.)

Lasserre, I. (2012). Intervention en Libye : les limites d'une stratégie. *Figaro*. http://www.lefigaro.fr/international/2012/09/12/01003-20120912ARTFIG00648-intervention-en-libye-les-limites-d-une-strategie.php. (Cité en page 1.)

Lazaros, N., Sirakoulis, G. C., and Gasteratos, A. (2007). Review of stereo matching algorithms for 3d vision. (Cité en page 59.)

Lefaudeux, B. (2013). l ' École nationale supérieure des mines de Paris Détection , localisation et suivi des obstacles et objets mobiles à partir d ' une plate forme de stéréo-vision. (Cité en pages 35, 52, 56 et 59.)

Lewis, J. P. I. L. . M. (1995). Fast Normalized Cross-Correlation Template Matching by Cross-. *Vision Interface*, 1995(1) :1–7. (Cité en page 111.)

Li, R. (1997). Mobile Mapping : An Emerging Technology for Spatial Data Acquisition. *Photogrammetric Engineering and Remote Sensing*, 63(9) :1085–1092. (Cité en page 24.)

Lin, K.-h. and Wang, C.-c. (2010). Stereo-based simultaneous localization, mapping and moving object tracking. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3975–3980. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5649653. (Cité en page 52.)

Lin, L.-H., Lawrence, P. D., and Hall, R. (2011). Robust outdoor stereo vision SLAM for heavy machine rotation sensing. *Machine Vision and Applications*, 24(1) :205–226. http://www.scopus.com/inward/record.url?eid=2-s2.0-81555232044&partnerID=40&md5=5bfbc1ba5b309ed2c99a8cf2a862f61dhttp://www.springerlink.com/index/10.1007/s00138-011-0380-6. (Cité en page 39.)

Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2) :91–110. http://link.springer.com/10.1023/B:VISI.0000029664.99615.94. (Cité en page 29.)

Lu, Z., Hu, Z., and Uchimura, K. (2009). SLAM estimation in dynamic outdoor environments : A review. (Cité en pages 7 et 27.)

Lu, Z., Hu, Z., and Uchimura, K. (2010). Slam estimation in dynamic outdoor environments. *International Journal of Humanoid Robotics*, 7(2) :315–330. http://www.scopus.com/inward/record.url?eid=2-s2.0-77953269394&partnerID=40&md5=63ed8e2843a9dd034e8c6e2f96299b57. (Cité en page 27.)

Luca, F., Federici, M., Porta, P. P., and Broggi, A. (2012). Atelier sur les véhicules intelligents 14 mars 2012 , Mines ParisTech. pages 1–9. (Cité en page 7.)

Légal, J. (2015). Quelques rappels concernant la méthode expérimentale. http://j.b.legal.free.fr/Blog/share/Dynamiques/Methodo.pdf. (Cité en page 155.)

Maini, R. and Aggarwal, H. (2009). Study and comparison of various image edge detection techniques. *International Journal of Image Processing ...*, 147002(3) :1–12. http://wwwmath.tau.ac.il/~turkel/notes/Maini.pdf. (Cité en page 140.)

Maitre, H. (1985). Un panorama de la transformation de Hough. 2 :305–317. (Cité en page 134.)

Mallet, A. (2001). Localisation d'un robot mobile autonome en environnements naturels. http://hal.archives-ouvertes.fr/tel-00131779/. (Cité en page 6.)

Manduchi, R., Castano, A., Talukder, A., and Matthies, L. (2015). Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation. pages 81–102. (Cité en page 50.)

Marcotegui, B. and Deschaud, J.-e. (2012). Paris-rue-Madame database : a 3D mobile laser scanner dataset for benchmarking urban detection , segmentation and classification methods. (Cité en pages 7 et 24.)

Marquez, D. A. (2012). Towards Visual Navigation in Dynamic and Unknown Environment : Trajectory Learning and Following , with Detection and Tracking of Moving Objects. *PhD Thesis*. (Cité en page 52.)

Marr, D. and Poggio, T. (1979). A computational theory of human stereo vision. *Proceedings of the Royal Society of London B : Biological Sciences*, 204(1156) :301–328. (Cité en page 61.)

Martínez, M., Martínez, J., and Morales, J. (2015). Motion Detection from Mobile Robots with Fuzzy Threshold Selection in Consecutive 2D Laser Scans. *Electronics*, 4(1) :82–93. http://www.mdpi.com/2079-9292/4/1/82/. (Cité en page 24.)

Masumoto (1993). Global Positionning System. (Cité en pages 19 et 25.)

Matthies, L. and Rankin, A. (2003). Negative obstacle detection by thermal signature. *...Robots and Systems, 2003.(IROS 2003)...*, (818). http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1250744. (Cité en pages 48 et 51.)

Milanes, V., Alonso, J., and Bouraoui, L. (2011). Cooperative maneuvering in close environments among cybercars and dual-mode cars. *, IEEE Transactions on*, 12(1) :15–24. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5713839. (Cité en page 7.)

Moeslund, T. B. and Granum, E. (2001). A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3) :231 – 268. (Cité en page 26.)

Monnin, M. (2007). Approche unifiée défaillance-dommage dans la sûreté de fonctionnement pour la régénération des matériels au combat : Application aux systèmes d'armes terrestres. (Cité en page 7.)

Montemerlo, M. and Thrun, S. (2003). Simultaneous localization and mapping with unknown data association using FastSLAM. *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, 2 :1985–1991. (Cité en page 35.)

Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. (1999). Fastslam : A factored solution to the simultaneous localization and mapping problem. *Proceedings AAAI National Conference on Artificial Intelligence*. (Cité en page 39.)

Moulines, E. (2012). Méthodes de Monte Carlo par Chaînes de Markov. http://perso.telecom-paristech.fr/~gfort/Slides/EM-slides12.pdf. (Cité en page 37.)

Moutarde, F. and Stanciulescu, B. (2008). Real-time visual detection of vehicles and pedestrians with new efficient adaBoost features. pages 7–12. http://hal.inria.fr/inria-00320888/. (Cité en page 7.)

Narayana, K., Goulette, F., and Steux, B. (2010). Planar landmark detection using a specific arrangement of LIDAR scanners. *IEEE/ION Position, Location and Navigation Symposium*, pages 1057–1069. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5507336. (Cité en page 52.)

Natour, G. E., Aider, O. A., Rouveure, R., and Faure, P. (2015). Range and vision sensors fusion for outdoor 3D reconstruction. pages 2–7. (Cité en page 67.)

Nazarian, B. (2008). Imagerie Numérique, Chapitre V : Segmentation. I. http://bnazarian.free.fr/MyUploads/IN_GBM_05_SEGMENTATION.PDF. (Cité en page 132.)

Nexter (2013). Nexter-story.fr. http://nexter-story.com/home.php. (Cité en page 3.)

Nexter (2015a). *Nexter Catalogue*. http://www.nexter-group.fr/nexter/Flipping_Book/Export_FR/. (Cité en page 3.)

Nexter (2015b). Titus. http://www.nexter-group.fr/fr/component/content/article/70-group-focus/609-titusr-le-vehicule-6x6-blinde-polyvalent-du-xxieme-siecle. (Cité en page 22.)

NSA (2012). STANAG 4569. Technical report. https://www.unops.org/ApplyBO/File.aspx/4569eed02.pdf?AttachmentID=52d5a7b6-37ad-49bc-b18c-c468ea81787a. (Cité en page 3.)

Oggier, T., Büttgen, B., Lustenberger, F., Becker, G., Rüegg, B., and Hodac, A. (2006). SwissRanger SR3000 and First Experiences based on Miniaturized 3D-TOF Cameras. *PIT'06 Proceedings of the 2006 international tutorial and research conference on Perception and Interactive Technologies*, pages 212–216. (Cité en page 63.)

Olivieira, Marco antonio de (Federal University of Santa Catarina, U. (2005). Linear complexity stereo matching based on region indexing. *IEEE XVIII Brazilian Conference on COmputer Graphics and Image Processing (SIBGRAPI'05)*, (1530-1834/05) :–. (Cité en page 59.)

Orteu, J.-J. (2008). Calibrage géométrique d'une caméra ou d'un capteur de vision stéréoscopique. (Cité en page 59.)

Parent, M. (2004). From drivers assistance to full automation for improved efficiency and better safety. *Technology Conference, 2004. VTC 2004-Spring.*, pages 2931–2934. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1391461. (Cité en pages 6 et 18.)

Pauly, M., Gross, M., and Kobbelt, L. (2002). Efficient simplification of point-sampled surfaces. *13th IEEE Visualization conference.*, (Section 4) :163–170. (Cité en page 131.)

Paz, L. M., Guivant, J., Tardós, J. D., and Neira, J. (2007). Data Association in O(n) for Divide and Conquer SLAM. *Robotics : Science and Systems*, 3 :281 – 288. (Cité en page 35.)

Peugeot (2015). Peugeot Distance Alert. http://www.peugeot.fr/decouvrir/5008/monospace/#!galerie-multimedia/zoom-sur/securite/distance-alert. (Cité en page 19.)

Piatti, D. (2010). Time-of-Flight cameras : tests, calibration and multi-frame registration for automatic 3D object reconstruction. pages 2008–2010. (Cité en pages 64 et 65.)

Pini, P., Member, S., and Tard, J. D. (2008). Independent Local Maps : Application to Monocular Vision. 24(5) :1–13. (Cité en page 40.)

Poreba, M. (2014). Qualification et amelioration de la precision de systemes de balayage laser mobiles par extraction d aretes. (Cité en page 24.)

Poreba, M. and Goulette, F. (2015). A robust linear feature-based procedure for automated registration of point clouds. *Sensors (Basel, Switzerland)*, 15(1) :1435–57. http://www.mdpi.com/1424-8220/15/1/1435/htm. (Cité en pages 7 et 24.)

PrismTech (2015). OpenSplice DDS. http://www.prismtech.com/dds-community. (Cité en page 5.)

Proxel (2015). Radar de recul Proxel - Official Webpage. http://www.proxel.com/fr/radar-de-recul-caracteristiques.html. (Cité en page 67.)

Pujades, S., V, V. C. A., Woods, A. J., Holliman, N. S., and Sd, G. E. F. (2012). Focus Mismatch Detection in Stereoscopic Content Fr To cite this version : Focus mismatch detection in stereoscopic content. (Cité en page 64.)

Quigley, M., Conley, K., Gerkey, B., FAust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., and Mg, A. (2009). ROS : an open-source Robot Operating System. *Icra*, 3(Figure 1) :5. (Cité en page 81.)

Rabet, J. (2014). Localisation d'un Casque de Visualisation a l'aide de Marqueurs. (Cité en page 153.)

Rao, Y. R., Prathapani, N., and Nagabhooshanam, E. (2014). APPLICATION OF NORMALIZED CROSS CORRELATION TO IMAGE REGISTRATION. pages 12–16. (Cité en page 29.)

Ray, W. (1997). Non linear filters, estimation and applications. *International Journal of Forecasting*, 2(13) :301. (Cité en page 35.)

RAYNAL, J. (2015). Le Lidar : talon d'Achille des prototypes de voitures autonomes ? http://www.industrie-techno.com/le-lidar-talon-d-achille-des-prototypes-de-voitures-autonomes.39708. (Cité en page 69.)

Renault (2015). Renault ADAS. https ://group.renault.com/passion/innovation/renault-laudace-dans-les-genes/adas-des-technologies-au-service-de-la-facilite-de-conduite-et-de-la-securite/. (Cité en page 18.)

République-Française (2013). Le livre blanc de la Défense. Technical report. http://fr.calameo.com/read/000331627d6f04ea4fe0e. (Cité en page 4.)

RICAUD, B., Joly, C., and de La Fortelle, A. (2015a). Are Virtual Reality headsets efficient for remote driving ? In *Road Safety & Simulation, Orlando*. (Cité en pages 10 et 166.)

RICAUD, B., Joly, C., and de La Fortelle, A. (2015b). Non urban driver assistance with 2D tilting laser reconstruction. In *MMT 2015*. (Cité en pages 9 et 165.)

RICAUD, B. M. P., STANCIULESCU, B. M. P., and ABREHERET, A. M. P. (2014). General Road Detection Algorithm : Computation Improvements. *ICPRAM*, pages 825–830. (Cité en pages 9 et 166.)

Risack, R., Mohler, N., and Enkelmann, W. (2000). A video-based lane keeping assistant. *Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No.00TH8511)*, (Mi) :356–361. (Cité en pages 19 et 134.)

Robosoft (2013). Robucar. http://www.robosoft.com/products/outdoor-mobile-robots/robucar/index.html. (Cité en page 80.)

Rone, W. and Ben-Tzvi, P. (2013). Mapping, localization and motion planning in mobile multi-robotic systems. *Robotica*, 31(1) :1–23. http://www.scopus.com/inward/record.url?eid=2-s2.0-84870613502&partnerID=40&md5=ecb2d102a24b1d7702841b1548a579cbhttp://journals.cambridge.org/production/action/cjoGetFulltext?fulltextid=8487954. (Cité en page 24.)

Roß ler, R., Kadiofsky, T., and Zinner, C. (2012). Real time mapping from 3D sensor data using a Velodyne HDL-32E laser scanner. pages 1–2. (Cité en page 56.)

Rublee, E. and Bradski, G. (2011). ORB : an efficient alternative to SIFT or SURF. http://www.willowgarage.com/sites/default/files/orb_final.pdf. (Cité en page 29.)

Rusu, R. B. et al. (2010). Point cloud library. *Willow Garage, Menlo Park*. (Cité en page 82.)

S. Thrun M. Montemerlo, D. K. B. W. J. N. E. N. (2004). FastSLAM : An Efficent Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association. *Journal of Machine Learning Research*, (July). (Cité en page 34.)

Salvi, J., Pagès, J., and Batlle, J. (2004). Pattern codification strategies in structured light systems. *Pattern Recognition*, 37(4) :827 – 849. Agent Based Computer Vision. (Cité en pages 65 et 66.)

Scharstein, D. and Szeliski, R. (2003). High-accuracy stereo depth maps using structured light. *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, 1 :I–195 – I–202. (Cité en page 65.)

Segal, A., Haehnel, D., and Thrun, S. (2009). Generalized-icp. *Proc. of Robotics : Science and ...*. http://www.robots.ox.ac.uk/~avsegal/resources/papers/Generalized_ICP.pdf. (Cité en page 31.)

Shen, J., Tick, D., and Gans, N. (2011). Localization through fusion of discrete and continuous epipolar geometry with wheel and IMU odometry. *Proceedings of the 2011 American Control Conference*, pages 1292–1298. (Cité en page 27.)

Sibley, G., Sukhatme, G. S., and Matthies, L. H. (2006). The Iterated Sigma Point Kalman Filter with Applications to Long Range Stereo. *Rss*. (Cité en pages 96 et 98.)

Siciliano, B. and Khatib, O. (2008). Springer handbook of RObotics 21. Sonar Sensing. *Springer*, Part C :21. (Cité en page 68.)

Steux, B. and ElHamzaoui, O. (2010). CoreSLAM : a SLAM Algorithm in less than 200 lines of C code. *International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 1 :1975–1979. (Cité en pages 7, 26 et 31.)

Stic, D. E. S., Des, E. T., and Corebots, P. (2011). Des stic et des hommes. page 2009. (Cité en page 7.)

Surmann, H., Lingemann, K., Andreas, N., Hertzberg, J., and Augustin, S. (2001). A 3D laser range finder for autonomous mobile robots. (April) :153–158. (Cité en page 55.)

Swank, A. J. (2012). Localization Using Visual Odometry and a Single Downward-Pointing Camera. (September). (Cité en page 26.)

Systems, G.-N. U. G. (2009). Proprietary Information of G-NIUS Unmanned Ground Systems (UGS) Ltd / February 2009. (February). (Cité en page 7.)

Tanizaki, H. and Mariano, R. S. (1996). Nonlinear filters based on taylor series expansions. *Communications in Statistics - Theory and Methods*, 25(6) :1261–1282. (Cité en page 35.)

Tardif, J.-P. (2009). Non-iterative approach for fast and accurate vanishing point detection. *2009 IEEE 12th International Conference on Computer Vision*, (Iccv) :1250–1257. (Cité en page 134.)

Taylor, Z. and Nieto, J. (2012). A mutual information approach to automatic calibration of camera and lidar in natural environments. *Australasian Conference on Robotics and Automation, ACRA*, (August 2015). (Cité en page 134.)

Teleflow (2015). VPG. http://www.teleflow.net/gamme.php. (Cité en page 22.)

Teria (2015). Le reseau Teria. http://www.reseau-teria.com/dossiers_carto_teria.aspx. (Cité en page 25.)

Terrien, G., Fong, T., Thorpe, C. E., and Baur, C. (2000). Remote Driving With a Multisensor User Interface. *SAE 30th International Conference on Environmental Systems*. http://www.ri.cmu.edu/publication_view.html?pub_id=3291$\delimiter"026E30F$nhttp://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.16.754. (Cité en page 26.)

Thorel, S. (2014). *Conception et réalisation d'un drone hybride sol/air autonome*. PhD thesis. Thèse de doctorat dirigée par Andréa-Novel, Brigitte d' Informatique temps réel, robotique et automatique Paris, ENMP 2014. (Cité en page 26.)

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-e., Koelen, C., Markey, C., Rummel, C., Niekerk, J. V., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., Clara, S., and Mahoney, P. (2006). Stanley : The Robot that Won the DARPA Grand Challenge. 23(April) :661–692. (Cité en page 7.)

Tian, T., Tomasi, C., and Heeger, D. (1996). Comparison of Approaches to Egomotion Computation. *Computer Vision and Pattern ....* http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=517091. (Cité en page 27.)

Triggs, B., McLauchlan, P., Hartley, R., and Fitzgibbon, A. (2000). Bundle Adjustment - A Modern Synthesis. *Vision Algorithms : Theory and Practice*, 1883 :298–372. http://dx.doi.org/10.1007/3-540-44480-7_21. (Cité en page 40.)

UJF-Grenoble (2015). Endomorphismes symétriques d'un espace vectoriel euclidien. Applications. https://www-fourier.ujf-grenoble.fr/~rombaldi/AgregInterne/Oral1/120.pdf. (Cité en page 36.)

Ulas, C. and Temeltas, H. (2012). A robust feature extraction method and semantic data association for 6D SLAM. In *World Automation Congress Proceedings*, 2012 World Automation Congress, WAC 2012, Department of Control Engineering, Istanbul Technical University, Istanbul, Turkey. http://www.scopus.com/inward/record.url?eid=2-s2.0-84870854300&partnerID=40&md5=8ecfeaae0563cfbe6513a109cd7973d0. (Cité en page 37.)

Vegt, W. V. D., Boswinkel, D., and Witmer, R. (1987). Utilisation of GPS in large scale Photogrammetry. (Cité en page 24.)

Vivet, D. (2011). *Perception de l'environnement par radar hyperfréquence. Application à la localisation et la cartographie simultanées, à la détection et au suivi d'objets mobiles en milieu extérieur.* PhD thesis. (Cité en page 52.)

Vivet, D., Checchin, P., and Chapuis, R. (2013). Localization and Mapping Using Only a Rotating FMCW Radar Sensor. *Sensors (Basel, Switzerland)*, 13(4) :4527–52. http://www.ncbi.nlm.nih.gov/pubmed/23567523. (Cité en page 67.)

VR, O. (2015). Oculus Rift. https://www.oculus.com/en-us/. (Cité en page 11.)

Wan, E. a. and Van Der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. *Technology*, v :153–158. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=882463. (Cité en page 36.)

Wan, E. A. and Van Der Merwe, R. (2002). The unscented Kalman filter for nonlinear estimation. *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. papers3 ://publication/uuid/94787C0B-195F-4748-9277-D391BB80EB69. (Cité en pages 35 et 37.)

Wendel, A., Maurer, M., Graber, G., Pock, T., and Bischof, H. (2012). Dense reconstruction on-the-fly. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2012, pages 1450–1457, Institute for Computer Graphics and Vision, Graz University of Technology, Austria. http://www.scopus.com/inward/record.url?eid=2-s2.0-84866719352&partnerID=40&md5=70dc741616045c9d8394b2566e0cda1e. (Cité en page 26.)

Wifibot (2013). Wifibot Lab v4 + Lidar. http://www.wifibot.com/wifibot-wifibotlab_bldc_lidar.html. (Cité en page 81.)

Winslow, L. (2007). Unmanned Vehicle Robotic Warfare hide and seek strategies. pages 1–93. (Cité en pages 6 et 51.)

Wirbel, E. (2014). Localisation et navigation d ' un robot humanoïde en environnement domestique. (Cité en pages 55 et 110.)

Wirbel, E., Steux, B., Bonnabel, S., and De La Fortelle, A. (2013). Humanoid robot navigation : From a visual SLAM to a visual compass. *2013 10th IEEE International Conference on Networking, Sensing and Control, ICNSC 2013*, pages 678–683. (Cité en pages 7 et 110.)

Wirth, S., Carrasco, P. L. N., and Codina, G. O. (2013). Visual odometry for autonomous underwater vehicles. *2013 MTS/IEEE OCEANS - Bergen*, pages 1–6. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6608094. (Cité en page 26.)

Witt, J. and Weltin, U. (2012). Sparse Stereo by Edge-Based Search Using Dynamic Programming. *Proc. of ICPR*, pages 3631–3635. (Cité en page 59.)

Wu, E.-Y., Li, G.-Y., Xiang, Z.-Y., and Liu, J.-L. (2008). Stereo vision based SLAM using Rao-Blackwellised particle filter. *Journal of Zhejiang University : Science A*, 9(4) :500–509. http://www.scopus.com/inward/record.url?eid=2-s2.0-41549142813&partnerID=40&md5=acb9aff068c88fe0af5deb630221dfe1. (Cité en page 37.)

Wu, M. and Sun, J. Y. (2010). IMM-IPDA based maneuvering target tracking with Mobile Robot in unknown clutter environments. *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, pages 180–184. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5452047. (Cité en page 52.)

Xu, Z., Schwarte, R., Heinol, H.-G., Buxbaum, B., and Ringbeck, T. (1998). Smart pixel : photonic mixer device (PMD) ; new system concept of a 3D-imaging camera-on-a-chip. In Cheung, E. H. M., editor, *Proceedings : M2VIP '98 ; Nanjing, China, 10th - 12th September 1998*, pages 259–264. (Cité en page 63.)

Zaklouta, F. (2011). Real-time traffic sign recognition using spatially weighted HOG trees. *Advanced Robotics (ICAR), 2011*. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6088571. (Cité en page 7.)

Zhang, J. and Singh, S. (2014a). LOAM : Lidar Odometry and Mapping in Real-time. (Cité en pages 26, 31 et 55.)

Zhang, J. and Singh, S. (2014b). Real-time Depth Enhanced Monocular Odometry. *frc.ri.cmu.edu*. (Cité en page 27.)

Zhao, H., Chiba, M., Shibasaki, R., Shao, X., Cui, J., and Zha, H. (2008). SLAM in a dynamic large outdoor environment using a laser scanner. In *Proceedings - IEEE International Conference on Robotics and Automation*, 2008 IEEE International Conference on Robotics and Automation, ICRA 2008, pages 1455–1462, State Key Lab of Machine Perception, Peking University. http://www.scopus.com/inward/record.url?eid=2-s2.0-51649091746&partnerID=40&md5=d2208ec2a45c6a6818b7173ebe1e0d50. (Cité en page 7.)

Zhao, Y., Chen, X., and Han, J. (2010). Scan matching based SLAM in outdoor environment. *Jiqiren/Robot*, 32(5) :655–660+665. (Cité en page 31.)

Résumé

Le pilotage de véhicule blindé est rendu difficile par la faible visibilité offerte aux pilotes face aux environnements et aux situations complexes qu'ils doivent traverser.
La protection des opérateurs de véhicules militaires et l'intégrité de ces véhicules sont des besoins primordiaux pour l'armée de terre.
Afin de répondre à la problématique : *sécuriser le pilotage des véhicules militaires avec comme périmètre la définition d'un système de perception d'environnement*, nous avons procédé à l'étude au sens large de l'aide au pilotage dans le contexte militaire en environnement naturel et semi-structuré afin de mettre en exergue les moyens et les capteurs utilisables pour réaliser un système d'aide au pilotage.
Ainsi, nous offrons une réponse technique pour la réalisation d'un tel système au travers premièrement d'une étude des méthodes et algorithmes existants applicables à notre cas d'application. Ensuite nous définissons les capteurs utilisables avec de telles méthodes. De cet état de l'art, nous définissons un système répondant à notre problématique et nous expliquons sa mise en pratique au travers de la création d'une plateforme d'expérimentation.
Cette plateforme se compose des solutions présentées et permet de valider le concept par l'évaluation des solutions d'acquisition de l'environnement afin d'offrir les données nécessaires à une aide au pilotage.
Puis, l'étude des moyens d'analyse de cet environnement offre des pistes de réflexion sur le futur système d'aide au pilotage.
Enfin, une étude d'un moyen alternatif de restitution de l'information à l'opérateur complète la solution présentée en offrant une piste de réflexion sur l'impact de la restitution dans les performances des opérateurs.

Mots Clés

Reconstruction d'environnement, fusion de capteurs, détection d'obstacles, LIDAR à balancier, stéréo-vision, monovision, moyen de restitution, casque VR, Oculus Rift

Abstract

Armored vehicle driving is difficult due to low visibility experienced by pilots in tough environmental conditions and complex situations they have to manage.
Soldiers' safety and vehicle integrity are part of main topics for French "Armée de Terre". To resolve the problem, *Make the driving of military vehicles safer by improving environmental perception through driver assistance systems*, we study driving assistance in unstructured environment by looking for sensors and methods, which are suitable to make such a system.
First, we study existing methods and algorithms, which fit our application case. Conclusion of this study is the definition of our system.
Second, thanks to the previous study we explain the creation of an experimentation platform allowing evaluation of our concept. Data obtained from reconstruction is then exploited through environment analysis to bring obstacle extraction methods.
Third, study of an alternative display solution is exposed and completes this work in explaining impact of restitution in an operating cycle.

Keywords

Environment Reconstruction, sensors fusion, obstacles detection, tilting LIDAR, stereo-vision, monovision, display technologies, VR headset, Oculus Rift