Affinement de relevés laser mobiles issus de LIDARs multi-couches

Houssem Nouira

THÈSE DE DOCTORAT

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

Préparée à MINES ParisTech

Affinement de relevés laser mobiles issus de LIDARs multi-couches

Ecole doctorale n°432

Sciences des Métiers de l'ingénieur

Spécialité "Informatique temps-réel, robotique et automatique"

COMPOSITION DU JURY :

M. David FILLIAT
U2IS - ENSTA ParisTech,
Président du jury

M. Paul CHECCHIN
IUT d'Allier,
Rapporteur

M. Simon LACROIX
LAAS/CNRS, Robotics and InteractionS,
Rapporteur

M. Bruno Vallet
IGN - Institut National de L'information Géographique et Forestière,
Examinateur

M. François Goulette
Mines ParisTech,
Examinateur

M. Jean-Emmanuel Deschaud
Mines ParisTech,
Examinateur

Soutenue par Houssem **NOUIRA**
le 20 avril 2017

Dirigée par **François GOULETTE**
Encadrée par **Jean-Emmanuel DESCHAUD**

MINES ParisTech | PSL RESEARCH UNIVERSITY PARIS

Remerciements

Cette thèse a été une expérience unique et très enrichissante, qui a duré près de 4 ans et qui s'est déroulé avec plus ou moins d'aisance. J'ai fait beaucoup de rencontres durant cette thèse, aussi bien professionnelle que plus personnelle, et il y a beaucoup de monde à qui je voudrais adresser mes remerciements.

Tout d'abord, les personnes sans qui ces remerciements n'auraient pas lieu d'exister, les membres de mon jury qui ont validé ma thèse, Mr Paul Checchin, Mr David Filliat, Mr Simon Lacroix et Mr Bruno Vallet. Un remerciement plus spécial est adressé à Bruno Vallet, qui a été mon encadrant de stage de fin d'études d'ingénieurs : le laboratoire Matis de l'IGN où j'ai fait mon stage m'a directement permis d'effectuer cette thèse, notamment du fait que le Matis et le Centre de Robotique étaient partenaires d'un projet ANR, le même qui a financé ma thèse. Aussi, cette collaboration entre les deux laboratoires m'a permis de découvrir le Centre de Robotique, et vous vous doutez de la suite des événements.

Bien entendu, j'adresse un très grand remerciement à mes encadrants de thèse, François Goulette et Jean-Emmanuel Deschaud, qui ont fait un très bon travail d'encadrement, le résultat étant l'aboutissement de mes travaux de thèse. Malgré des hauts et des bas durant cette thèse, avec l'aide et le soutien de François et Jean-emmanuel, j'ai pu mener à bien mes travaux de recherches.

La thèse s'est déroulé sur un peu plus de 3 ans, mais je n'ai été présent au Centre de Robotique que jusqu'en octobre 2016 : j'ai commencé à travailler depuis le mois d'octobre 2016 dans une société qui se nomme Mensi, et qui appartient au groupe Trimble. Aussi, je tiens à remercier l'ensemble de mes collégues qui m'ont réservé un accueil des plus chaleureux, avec une attention toute particulière pour Thomas Chaperon et Olivier Sens avec qui je travaille étroitement, et Raouf Ben Jemaa sans qui je n'aurai pas eu l'opportunité de rejoindre Mensi.

Je tiens aussi à remercier l'ensemble de mes collégues du centre de Robotique, que j'ai cotoyé de près ou de loin pendant les trois ans que j'ai passé dans ce laboratoire. J'ai été plus proche de certaines personnes durant ces trois ans, et je m'en suis fait des amis plus que des collégues. Ce sont ces personnes que je vais essayer de remercier plus personnellement : Olivier, Jun, Florent, David, Martyna, Fernando, Li, Marie-Anne, Ravi. La liste est bien plus longue, et il y a de très nombreuses personnes que je voudrais remercier, mais celles-ci sont les personnes avec qui j'ai passé le plus de temps pendant cette thèse, en dehors du laboratoire notamment. Les trois ans sont passés très vite grâce à eux, et j'espère que nous resterons en contact après cette thèse.

Enfin, un remerciement plus intime, puisqu'il concerne ma famille qui a été très présente et d'un grand soutien au cours de cette thèse, et même avant la thèse : ce soutien m'a notamment permis d'obtenir mon diplôme d'ingénieur, qui a été un grand accomplissement pour ma famille, et par la suite d'obtenir mon diplôme de doctorat, qui n'était pas sur la "liste" des études prévues. Je tiens à remercier autant que possible mes parents et mon petit frère, qui m'ont apporté un soutien moral conséquent durant l'ensemble de mes études supérieurs : le 20 avril 2017 a été un très grand soulagement pour tous le monde. Durant les nombreux moments de doutes, ma famille était là ; durant les nombreux moments de joies, ma famille était présente ; lors de mes décisions les plus importantes concernant l'orientation de mes études, j'ai pu compter sur mes parents qui m'ont conseillé du mieux possible. Je dois beaucoup à mes parents et mon petit frèse, et bien sûr au reste de ma famille qui est en Tunisie, mon pays d'origine, et qui ont été les premiers à me soutenir et à être fier de mon parcours. Pour toutes ces raisons, et bien plus encore, merci du fond du coeur à ma famille.

Table des matières

Chapitre 1

Introduction

Sommaire

1.1 Contexte de la thèse

L'environnement est défini au sens premier du terme comme « ce qui entoure de tous côtés ». Sa connaissance peut être très utile pour de nombreuses applications : en effet, l'homme cherche à cartographier son environnement depuis plusieurs siècles, dans le but de mieux le connaître, mieux le comprendre, et planifier des actions. Historiquement, de l'Antiquité jusqu'au milieu du 16éme siècle, les relevés cartographique sont essentiellement issus de témoignages. [Histoire de la cartographie, 2017] présente l'Histoire de la cartographie, de l'Antiquité aux temps modernes : à partir du 16éme siècle, les relevés sont réalisés de façon plus rigoureuse, par des experts qui se servent de différents outils de plus en plus développés. Toutefois, la cartographie moderne telle que nous la connaissons n'apparaît qu'au 20éme siècle, notamment avec le développement du domaine aéronautique, qui a permis de mettre au point la cartographie aérienne, premier pas vers la cartographie moderne. Avec l'utilisation des satellites et des ondes, la cartographie à plus grande échelle est alors possible.

Depuis quelques décennies, un nouveau type de cartographie s'est beaucoup développé : la **Cartographie Mobile**. Ce type de cartographie est principalement terrestre, et pour être réalisée, un véhicule mobile est utilisé. De nombreuses applications découlent de la cartographie terrestre :

- la cartographie de zones inaccessibles à l'homme par exemple, comme des Mines aux parois instables.
- la cartographie d'environnements urbains à grande échelle, pour lesquels la cartographie aérienne ne convient pas, notamment à cause des occlusions générées par la végétation ou les bâtiments.
- les véhicules autonomes, qui se servent de la cartographie plus ou moins précise pour connaître l'environnement et se déplacer dedans.

De nombreuses applications existent, plus ou moins spécifiques à un type d'application donné, mais la cartographie occupe de nos jours une place centrale dans de nombreux travaux de recherches et développements.

Le sujet de thèse qui va être présenté dans ce manuscrit s'inscrit dans un contexte très actuel et très suivi par la communauté scientifique qui est la cartographie mobile. Cette thèse s'est aussi inscrite dans le cadre d'un projet français de recherche et développement, Terra Mobilita [Projet Terra Mobilita, 2017], qui s'est déroulé sur un peu plus de quatre ans, de 2011 à 2015.

1.1.1 Projet Terra Mobilita

L'objectif du projet Terra Mobilita était de mettre au point de nouveaux processus automatisés de création et de mise à jour de cartes 3D de voirie urbaine, avec une précision centimétrique, en utilisant des méthodes de relevé laser mobile, et de développer de nouveaux services et applications pour les collectivités territoriales utilisant ces modèles 3D de l'espace public.

FIGURE 1.1 – Les cinq axes de recherches principaux du projet Terra Mobilita

Le projet comportait cinq axes de développement, présentés en figure 1.1. Les principales innovations attendues étaient :

- l'intégration de l'utilisation de données 3D (images et lasers) issues d'un véhicule de cartographie mobile dans le processus actuel de création ou de révision de cartes de voirie 2D. Les avantages attendus étaient la réduction du délai et des coûts, mais aussi la possibilité de mise à jour plus fréquente (mensuelle par exemple) des cartes. Aussi, une fourniture systématique d'informations (revêtement de surface par exemple) non disponibles sur les cartes ou dans les processus de relevés actuels étaient espérés.
- la construction de modèles 3D complets de l'espace public, intégrant l'état de surface de la voirie et l'encombrement de l'espace public (stationnement, poubelles) afin de développer des applications et services pour l'aménagement. Ces modèles 3D ont aussi été utilisés pour la gestion et l'entretien des voiries, la production de Plans de Déplacement Urbains (PDU), en particulier pour les circulations douces (piéton, vélo, rollers, poussette, chaise roulante...), la production à la demande d'itinéraires de déplacement, et l'automatisation des diagnostics d'accessibilité de la voirie pour Personnes à Mobilité Réduite (PMR).

Le projet a réuni 8 partenaires :

1. une PME, 1Spatial, chef-de-file du projet, éditeur de logiciel dans le domaine des systèmes d'information géographique (SIG) et des technologies géospatiales, et qui apporte sa connaissance des SIG 3D et du marché des collectivités territoriales.
2. trois grandes entreprises : Thales Training System, qui apporte une capacité à produire des bases de données urbaines détaillées de grande dimension et une capacité à fournir des composants d'immersion dans l'espace public virtuel.
3. Mensi-Trimble, qui est leader dans le secteur des systèmes de numérisation 3D par relevé laser et de modélisation 3D, et qui se charge de la recherche et du développement de nouvelles technologies de relevé 3D des voiries et de modélisation 3D de l'espace public
4. Citiway (Transdev-Veolia), qui apporte une parfaite connaissance du métier de l'information voyageur et sa maîtrise métier et technique dans la réalisation de calculateurs de recherche d'itinéraires multimodaux.
5. L'Institut National de l'Information Géographique et Forestière (IGN), qui contribue à fournir les données existantes et à acquérir des données sur les sites d'expérimentation prévus par le

projet, mais aussi à développer de nouveaux outils permettant de rendre interopérables des bases de données de différentes sources, de reconnaître et qualifier de manière automatisée des objets thématiques de voirie, de développer un service en ligne de saisie collaborative de données 3D de voiries.

6. Armines (représenté par les laboratoires Centre de Robotique -CAOR- et Centre de Morphologie Mathématique -CMM-), qui met à disposition de nombreux travaux sur les systèmes de numérisation 3D mobiles, mais aussi de traitement d'images et modélisation 3D.
7. la fondation Sciences-Po (représentée par le Master d'Urbanisme), qui apporte dans le projet ses compétences d'ingénierie de la mobilité et une connaissance fine du jeu d'acteurs de la gestion des voiries et du domaine public routier.
8. une association, le Centre de Ressources et d'Innovations Mobilité Handicap (CEREMH), qui est impliqué sur la recherche et le développement de produits et services favorisant la mobilité des personnes en situation de handicap, en particulier les personnes à mobilité réduite.

1.1.2 Objectifs de la thèse

La problématique abordée est l'affinement de relevés laser issus d'acquisitions mobiles terrestres : ce que l'on cherche à faire est **d'optimiser la qualité des relevés lasers mobiles** en post-traitement, et ceci dans le but d'améliorer la qualité de cartes 3D issues d'acquisitions effectuées par un véhicule mobile de cartographie équipé d'un LIDAR multi-couches. Les Systèmes Mobiles de Cartographie basés LIDAR permettent d'obtenir des cartes 3D de l'environnement, qui sont géo-référencées grâce à d'autres capteurs embarqués sur le véhicule : GPS, centrale inertielle, ou encore odomètre sont de tels capteurs qui permettent de localiser le véhicule mobile pendant la campagne d'acquisition. Toutefois, ces cartes manquent de précision et un affinage des cartes est essentiel dans de nombreux cas d'applications où une précision fine est requise sur les cartes 3D, comme pour des applications de classification par exemple. C'était aussi le cas du projet Terra Mobilita par exemple : une précision fine des cartes 3D était requise pour les diverses applications et résultats attendus.

Dans cette thèse, nous nous sommes intéressés à l'affinement de relevés lasers mobiles issus de LIDARs multi-couches. Nous avons choisi de travailler avec ce type de capteurs pour plusieurs raisons : tout d'abord, dans le cadre du projet Terra Mobilita, les systèmes mobiles principalement utilisés pour effectuer les relevés lasers et produire les données utiles étaient équipés de capteurs multi-couches. Aussi, ces capteurs sont apparus récemment par rapport à leur homologues mono-couches, et une conséquence est que peu de travaux traitent de l'affinement de relevés lasers provenant de tels capteurs. Nous cherchons à mettre au point des méthodes d'affinements de relevés laser **rapides et robustes** à des erreurs initiales importantes ; aussi, nous voulons proposer **un ou plusieurs critères de validations** qui permettent d'évaluer la qualité des affinements qui vont être proposés.

1.2 Contributions apportées par la thèse et organisation du manuscrit

Pour atteindre l'objectif que l'on s'est fixé, nous avons cherché ce qu'il était possible d'améliorer dans le processus de cartographie mobile pour atteindre une précision plus fine des relevés lasers. L'orientation que nos travaux ont pris a été de directement « jouer » sur le processus de création des cartes à partir des données brutes acquises par le capteur LIDAR. Ainsi, nous avons proposé plusieurs solutions pour affiner les cartes, entre l'étape où les données sont acquises par le capteur, et l'étape où les données sont référencées dans un repère lié à la Terre pour avoir les cartes 3D utilisées. Lors de la création de cartes 3D géo-référencées, les données sont tout d'abord acquises par le capteur LIDAR et référencées dans le repère cartésien du laser à l'aide d'un **calibrage intrinsèque** du capteur d'acquisition. Ensuite, un **calibrage extrinsèque** du capteur permet d'avoir la transformation

entre le capteur et le véhicule, et permet de référencer les données dans le repère « body », lié au véhicule d'acquisition ; enfin, avec la **trajectoire du véhicule** obtenue en fusionnant les données issues des GPS, centrale inertielle et odomètre, il est possible de géo-référencer les données lasers. Les contributions principales de la thèse sont au nombre de **trois** :

- Dans un premier temps, nous nous sommes intéressés au **calibrage extrinsèque** du capteur LIDAR, qui avec un mauvais étalonnage des paramètres, peut entraîner un mauvais référencement des données dans le repère lié au véhicule. Notre contribution a été de proposer une optimisation des paramètres liés au calibrage extrinsèque : ces paramètres sont au nombre de six, trois translations et trois rotations, et caractérisent la transformation entre le repère laser et le repère véhicule. Une optimisation itérative de ces paramètres est proposée : pour cela, nous utilisons la structure des Lidars multi-couches, et corrigeons les paramètres de l'étalonnage extrinsèque en effectuant de l'ajustement de faisceaux entre les nuages issus des différentes couches du capteur Lidar. Pour l'ajustement de faisceaux, nous avons minimisé une énergie construite sur une distance point-à-plan entre nuages, et pour robustifier l'optimisation, nous avons introduits l'utilisation d'attributs de planarité (indication d'appartenance plus ou moins probable d'un point à une surface planaire) dans les poids liés à chaque distance. Enfin, nous proposons aussi de donner des valeurs de « précision » des résultats d'optimisation obtenus, permettant d'évaluer la confiance à accorder aux différents paramètres optimisés.
- Une deuxième contribution a été de travailler sur l'optimisation des paramètres liés au **calibrage intrinsèque** : la méthode d'optimisation proposée est la même que précédemment, toujours en effectuant de l'ajustement de faisceaux. Une seconde approche est ensuite introduite, où l'on cherche à **conjointement optimiser** les paramètres extrinsèques et intrinsèques.
- Pour cette troisième contribution, on s'intéresse à l'optimisation des paramètres de la trajectoire. La trajectoire est subdivisée en plusieurs points de contrôles, et pour chacun de ces points, **six paramètres** sont donnés : trois translations et trois rotations, qui donnent la position du véhicule par rapport à une référence. Pour cette approche d'optimisation, seuls les **paramètres de translations sont corrigés** car, pour la trajectoire, ce sont les paramètres qui possèdent la plus grande incertitude. L'approche d'optimisation proposée reste la même que précédemment : nous effectuons à nouveau de l'ajustement de faisceaux entre les nuages issus des différentes couches du capteur LIDAR.

Une contribution additionnelle de la thèse est que les algorithmes que l'on propose peuvent être appliqués à la plupart des systèmes d'acquisitions qui équipent un capteur LIDAR multi-couches. En effet, pour chacune des contributions présentées, nous utilisons des types de données similaires ; pour appliquer nos algorithmes, les jeux de données doivent être composés des paramètres suivants :

- Les mesures brutes du capteur multi-couches, c'est-à-dire la distance pour chaque point acquis au laser, l'angle par rapport au plan horizontal du capteur et l'angle induit par la rotation du capteur.
- Un modèle de calibrage intrinsèque, qui dépend de l'utilisateur et du capteur utilisé, et qui permet de transformer les données brutes du capteur laser en coordonnées cartésiennes liées au repère du capteur laser.
- Les paramètres de calibrage extrinsèques, pour référencer les données dans le repère « body » lié au véhicule.
- La position du véhicule, géo-référencée ou dans un repère local, qui peut provenir de la fusion de données issues de plusieurs sources : GPS, centrale inertielle ou encore odomètre par exemple. La position est donnée en coordonnées cartésiennes : dans le cas du GPS, les latitudes, longitudes et élévations doivent être transformées en coordonnées cartésiennes en appliquant la projection des coordonnées géodésiques qui correspond à la zone d'acquisition.

Avec ces différents paramètres pour les jeux de données utilisées, il est possible d'appliquer les solutions d'affinement que l'on présente à des nuages de points obtenus avec un scanner LIDAR multi-couches quelconque.

Le manuscrit s'organise de la façon suivante :

- le chapitre 2 présente de façon générale les systèmes d'acquisitions de données 3D, ainsi que le géoréférencement des données acquises, et une revue des sources d'erreurs possibles lors d'une cartographie mobile. Le véhicule d'acquisition que nous avons utilisé pour acquérir les jeux de données utilisés dans les différentes optimisations est également présenté, et nous présentons aussi la structure des jeux de données que nous utilisons pour tester nos optimisations.
- le chapitre 3 présente notre approche d'affinement par optimisation des paramètres de calibrage extrinsèque. Nous y présentons la fonctionnelle que l'on minimise pour optimiser les paramètres de calibrage, et un détail de notre approche d'optimisation est présentée. Nous détaillons aussi l'ensemble des paramètres utilisés dans notre résolution, et présentons les différents jeux de données utilisés. Des résultats d'optimisations sur des jeux de données simulés et réels sont présentés. Enfin, nous traitons d'une problématique qui est limitante pour notre approche d'optimisation, et qui est l'observabilité des paramètres de calibrage.
- le chapitre 4 présente notre approche d'affinement par optimisation des paramètres de calibrage intrinsèque. Nous y présentons notre approche d'optimisation, où nous nous intéressons à un ensemble de paramètres de calibrage différents de ce qui a été présenté au chapitre 3 : les paramètres de calibrage extrinsèques ne sont pas modifiés dans un premier temps. Des résultats d'optimisations sur les mêmes jeux de données que le chapitre précédent sont présentés. Ensuite, une amélioration de cet affinement est introduite, où l'on optimise conjointement les paramètres extrinsèques et intrinsèques, et les améliorations sont montrées avec des tests sur les mêmes jeux de données que précédemment.
- le chapitre 5 présente notre approche d'affinement par optimisation des paramètres de translations de la trajectoire du véhicule. La méthode d'optimisation est également détaillée, et on cherche dans ce chapitre à optimiser les positions géo-référencées du véhicule pendant l'acquisition. Les paramètres de calibrage des chapitres 3 et 4 ne sont pas modifiés : ils sont supposés corrects, ou déjà optimisés. Des résultats d'optimisation sont ensuite donnés sur deux jeux de données, un simulé et un réel.
- enfin, le chapitre 6 apporte une conclusion à ce manuscrit, ainsi qu'une présentation des voies d'améliorations possibles.

Chapitre 2

Systèmes d'acquisitions et sources de bruits

Sommaire

Introduction

La cartographie est la discipline qui permet de produire des cartes, afin d'avoir une représentation de l'environnement que l'on peut interpréter et dans laquelle il est possible de se repérer, de s'orienter. Pour obtenir des cartes précises de son environnement, il est nécessaire d'utiliser un ensemble de capteurs dont les données sont fusionnées ou mises en commun : les différents capteurs possèdent des erreurs qui peuvent fausser les cartes produites, et il convient alors de les corriger pour avoir une cartographie correcte. Nous allons traiter de ces différents éléments dans ce chapitre : dans un premier temps, nous allons présenter quelques systèmes d'acquisitions de données 3D ; ensuite, nous parlerons des différents capteurs de mesures utiles à la cartographie, ainsi que des différentes erreurs de mesures introduites par l'utilisation de ces capteurs ; enfin, nous évoquerons quelques méthodes de corrections de données utilisées pour corriger des erreurs qui peuvent apparaître dans les données 3D acquises.
Aussi, nous présentons le véhicule d'acquisition que nous avons utilisé pour acquérir les jeux de données utilisés dans les différentes optimisations, le L3D2, véhicule d'acquisition du Centre de Robotique(CAOR), et nous présentons également la structure des jeux de données que nous utilisons dans nos optimisations.

2.1 Systèmes d'acquisitions de données 3D

Pour acquérir des données et cartographier notre environnement, plusieurs types de systèmes différents existent. Lorsque l'on parle de système, on englobe l'ensemble des capteurs qui interviennent dans la chaîne de production des cartes 3D, allant des scanners LIDAR pour acquérir les données aux GPS et centrales inertielles pour référencer les données. On peut séparer les systèmes d'acquisitions en deux catégories principales : les systèmes d'acquisitions dits « fixes », utilisés principalement pour avoir des cartes représentant très précisément l'environnement, et les systèmes d'acquisitions mobiles, qui permettent d'acquérir un environnement de taille très importante, mais bien plus rapidement que les systèmes fixes. Chaque type de système possède des avantages et des inconvénients, que nous allons essayer de présenter dans cette partie. Ce qu'il faut savoir, c'est que l'on parle bien de système d'acquisition, et non de capteur d'acquisition LIDAR : en effet, un même capteur LIDAR peut très bien être utilisé dans un système fixe ou mobile. Nous verrons en revanche dans la section 2.2 que les capteurs LIDARs sont adaptés à l'un ou l'autre des types d'acquisitions.

2.1.1 Les systèmes d'acquisition fixes

Un système d'acquisition fixe permet de cartographier un environnement de façon très précise : comme le nom l'indique, un système d'acquisition fixe permet de cartographier à l'arrêt, ce qui enlève de nombreuses sources de bruits, notamment provenant de l'utilisation d'autres capteurs lors de l'acquisition mobile. Il est alors possible de cartographier une zone donnée très précisément, puisque le système ne bouge pas lors de l'acquisition, et cela permet aussi de produire des données plus ou moins denses selon le besoin. Les systèmes fixes sont très utilisés dès qu'il est nécessaire d'avoir un niveau de détail fin dans les cartes 3D.

Figure 2.1 – Exemple de système d'acquisition fixe

La figure 2.1 présente un tel système fixe, où le capteur LIDAR qui permet de faire les acquisitions

est monté sur un trépied : on peut remarquer la présence de sphères au sol, qui sont utilisées pour consolider les données. Le scanner qui est présenté est un capteur LIDAR de la gamme FARO [Faro LIDAR sensor, 2017]. Pour faire une acquisition d'une large zone avec un système fixe, il faut placer le scanner en différentes stations d'acquisitions, ce qui a pour effet de produire plusieurs « morceaux » de cartes 3D, que l'on va consolider entre elles avec un recalage par exemple. Le problème est que cela prend beaucoup de temps, même si les acquisitions sont faites avec une faible résolution ; un protocole d'acquisition simplifié serait le suivant :

1. Placer le scanner sur une station d'acquisition.
2. Effectuer l'acquisition.
3. Déplacer le scanner sur une autre station.
4. Répéter les étapes 2 et 3 jusqu'à la fin de l'acquisition.

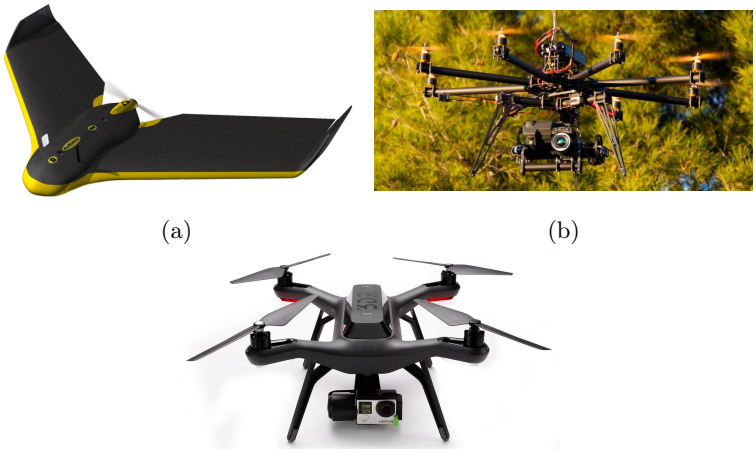
Les stations d'acquisitions sont marquées très précisément : ce type d'acquisition se fait généralement sans capteur tiers, à savoir un GPS ou des odomètres, et pour avoir des cartes 3D correctement référencées (ou géoréférencées), il faut s'appuyer sur le référencement des stations. Le scanner est déplacé à la main, ce qui contribue à allonger la durée de l'acquisition. De plus, l'immobilité du scanner permet d'effectuer des relevés avec une résolution d'acquisition très fine, ce qui augmente encore plus la durée de l'acquisition.

En résumé, les systèmes d'acquisition fixes permettent d'avoir une cartographie très précise, avec un niveau de détail fin, mais le temps pris par l'acquisition est très important : dans certains cas, ce temps est trop important, presque limitant en fonction de l'application de cartographie visée, notamment parce que les zones numérisées sont très importantes, de l'ordre de plusieurs km^2. C'est pour cela que des systèmes d'acquisitions mobiles sont utilisés.

2.1.2 Les systèmes d'acquisition mobiles

Un système d'acquisition mobile doit, comme son nom l'indique, pouvoir se déplacer. On trouve plusieurs types de systèmes mobiles, allant du drone au véhicule utilitaire, qui permettent de créer des cartographies différentes selon le cas d'application. Un des systèmes mobiles les plus connus et anciens est le système d'acquisition aéroporté : les premières acquisitions datent des années 60, avec notamment un des premiers travaux en 1967 de Patrick McCornick, où un appareil aéroporté de la NASA volait à altitude constante pour faire des acquisitions avec un LIDAR. Aujourd'hui, la miniaturisation des technologies d'acquisition ont permis le développement de drones, qui sont aussi utilisés pour faire de la cartographie et qui peuvent être équipés de l'ensemble des capteurs utiles à une telle tâche : centrale inertielle, caméras, capteurs LIDAR, GPS. De nombreuses entreprises comme Parrot [Parrot ebee site web, 2017], Flying Eye [Flying eye site web, 2017] ou 3DRobotics [3DR site web, 2017] développent des drones pour la cartographie : l'avantage de ces solutions est qu'elles sont peu coûteuses et simple d'utilisation puisqu'il suffit de savoir piloter un drone. Elles sont aussi faciles à mettre en œuvre, de nombreuses entreprises proposent des drones complets, avec maintenance comprise, et pour lesquels il suffit d'ajouter certains capteurs comme les caméras ou les scanners LIDAR.

Du côté des véhicules terrestres, [Ellum et El-Sheimy, 2002] présentent un historique des véhicules d'acquisitions mobiles, qui sont apparus au début des années 1990, notamment avec un véhicule développé par « le centre de cartographie de l'université d'état d'Ohio » [Goad et Novak, 1991]. Les véhicules d'acquisitions mobiles terrestres sont très utilisés aujourd'hui, pour pallier à l'utilisation des systèmes d'acquisitions fixes en environnement extérieur : ils permettent d'acquérir des grands volumes de données en un temps assez court, en comparaison aux systèmes fixes. Pour avoir la meilleure cartographie possible, ces véhicules sont équipés de plusieurs capteurs : des capteurs d'acquisitions pour la cartographie, mais aussi des capteurs permettant d'estimer la position du véhicule d'acquisition au cours de la cartographie. Comme pour les systèmes aéroportés, des systèmes mobiles de plus petites tailles ont été développés afin d'accéder à des zones dangereuses ou



(a) (b)

(c)

FIGURE 2.2 – Quelques exemples de drones : a) Drône ebee de Parrot ; b) Drône Flying eye ; c) Drône 3DRobotics

hors d'accès aux véhicules pour les véhicules routiers : Andreas Nüchter présente un petit robot dans [Nüchter *et al.*, 2007] qui permet de cartographier aussi bien des environnements en intérieur et en extérieur ; dans [Nüchter *et al.*, 2004], un autre robot est présenté pour de la cartographie de mines, qui sont dangereuses et difficiles d'accès à l'homme. Un autre système du même genre est présenté dans [Montemerlo et Thrun, 2006], une plateforme Segway qui accueille différents capteurs. Ces « petits » systèmes, bien que plus lents que les véhicules routiers, restent plus pratiques à utiliser que les systèmes fixes pour cartographier des zones de tailles importantes ; ils sont mêmes indispensables dans certains cas pratiques, comme par exemple lorsque la cartographie concerne une zone accidentée ou dangereuse pour l'homme. Enfin, on peut aussi évoquer l'existence de systèmes portables « humains » : [Liao *et al.*, 2013] et [Nüchter *et al.*, 2015] présentent des systèmes de cartographie portés par un homme, sans centrale inertielle, et avec différents scanners 2D ou 3D. Un des avantages est le coût très faible du système en comparaison à des robots, ou encore la praticité pour des zones en intérieur, où on peut rencontrer de nombreux obstacles et des escaliers par exemple.

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

FIGURE 2.3 – Quelques exemples de systèmes mobiles : a) robot pour acquisition en extérieur [Nüchter *et al.*, 2007] ; b) plateforme Segway [Montemerlo et Thrun, 2006] ; c) système portatif [Liao *et al.*, 2013] ; d) système portatif [Nüchter *et al.*, 2015]

2.1.3 Les véhicules mobiles d'acquisitions terrestres

Les véhicules mobiles d'acquisitions, ou MMS (Mobile Mapping System), sont assez complexes de par leur structure, mais aussi de par leur fonctionnement. En effet, ils intègrent de nombreux capteurs, qui doivent être synchronisés entre eux, afin de fournir des données cohérentes qui peuvent être fusionnées ou mises en commun. Le Centre de Robotique (CAOR) de Mines ParisTech possède un véhicule de cartographie, le L3D2. Il s'agit d'une nouvelle version du véhicule LARA 3D (LA Route Automatisée), présenté dans [Goulette *et al.*, 2007]. La figure 2.4 présente le véhicule d'acquisition du Centre de Robotique dans sa dernière version : le véhicule est équipé de plusieurs capteurs qui permettent de cartographier l'environnement, mais aussi de géoréférencer les données de façon précise. En effet, le prototype possède une caméra à l'avant qui permet d'avoir une vue de la trajectoire du véhicule, un capteur LIDAR multi-fibres Velodyne (LIDAR 3D) qui permet d'avoir un modèle en 3D de l'environnement, un odomètre de la marque BEI DHO5S, une centrale inertielle de la marque iXBlue LANDINS et un GPS de la marque Novatel FlexPak 6.

FIGURE 2.4 – Véhicule d'acquisition du Centre de Robotique

La trajectoire du véhicule est estimée avec un filtre de Kalman ; l'odomètre et la centrale inertielle donnent une estimation de sa trajectoire, et le GPS, lorsqu'il reçoit un signal, permet d'avoir la localisation du véhicule, avec une certaine marge d'erreur selon le modèle du GPS. Le filtre de Kalman met en commun ces trois données et donne une estimation assez précise du positionnement du véhicule, généralement à la fréquence de la centrale inertielle. Le reste des positions, notamment lorsque le GPS ne reçoit pas de signal, est estimé par intégration en boucle ouverte.

D'autres plateformes existent : de nombreux laboratoires de recherches, comme le Centre de Robotique, possèdent leur propre système d'acquisition, qui est d'une grande utilité dès que des problématiques de cartographie ou de modélisation sont abordées. [Huang et Barth, 2009] présentent un véhicule d'acquisition équipé d'une caméra et de deux capteurs LIDAR 3D ; [Mazzei *et al.*, 2012] présentent un autre véhicule d'acquisition, équipé de plusieurs caméras (dont une paire stéréoscopique) et de plusieurs capteurs LIDAR, 2D et 3D ; [Maddern *et al.*, 2012] présentent eux aussi un

système d'acquisition, équipé d'une centrale inertielle, d'un GPS, et de trois capteurs LIDAR - 2 sont des capteurs mono-couches et le dernier est un capteur multi-couches, composé lui-même de 3 capteurs mono-couches -. L'IGN, un autre centre de recherche français, possède lui aussi quelques véhicules d'acquisitions. La figure 2.5 présente deux des systèmes d'acquisitions utilisés par l'IGN : le véhicule Stéréopolis II, et un véhicule plus petit, le prototype Viapolis. Le véhicule mobile Stéréopolis II est un véhicule du même genre que le L3D2, doté des mêmes types de capteurs, et qui sert aussi à faire des acquisitions de l'environnement. Le véhicule Viapolis est un véhicule plus petit, de la classe des fauteuils électriques : il est du coup habilité à rouler sur les trottoirs et autres espaces piétons, ce qui est très utile pour la cartographie. En effet, il peut atteindre et cartographier des zones inaccessibles aux véhicules mobiles de plus grande taille. Comme le montre la figure 2.5 b), il est équipé d'un GPS, d'une centrale inertielle, d'une caméra omnidirectionnelle pour avoir des panoramas, et d'un capteur LIDAR de la marque Trimble [Trimble LIDAR sensor, 2017] à l'arrière. Le véhicule effectue des acquisitions en se déplaçant d'une station à une autre, et en s'arrêtant le temps d'effectuer une acquisition, ce qui donne une cartographie à mi-chemin entre la cartographie statique et la cartographie mobile.

(a) (b)

FIGURE 2.5 – Véhicules de l'IGN : a) Véhicule mobile Stereopolis II ; b) Véhicule Viapolis

2.2 Géoréférencement et capteurs de mesures utilisés en cartographie mobile 3D

Lors d'une acquisition de données avec un véhicule mobile, de nombreux capteurs peuvent être utilisés : il n'y a pas de normes ou de règles à suivre pour concevoir un véhicule de cartographie mobile. En théorie, il suffit que le véhicule soit équipé d'un capteur d'acquisition pour cartographier l'environnement : il s'agit le plus souvent d'un système LIDAR composé de plusieurs lasers et/ou de caméras, et qui permet de produire des cartes 3D ou 2D de l'environnement selon les besoins. Le seul problème avec un système mobile qui ne comprend qu'un capteur d'acquisition est que les données ne peuvent pas réellement être exploitées : le capteur fournit des données à un certain intervalle de temps qui dépend de la fréquence du capteur, mais comme le véhicule se déplace, il n'y a aucune information sur la position du véhicule à l'instant d'acquisition, et donc sur le référencement

des données acquises. C'est pour cela que lors d'une cartographie, les données sont généralement référencées.

Dans cette section, nous allons présenter dans un premier temps les différentes étapes qui permettent d'obtenir une cartographie 3D de l'environnement, puis nous présenterons un peu dans le détail les capteurs qu'il est possible d'utiliser pour faire de la cartographie 3D, et nous rentrerons plus dans le détail au sujet des capteurs LIDARs.

2.2.1 Le référencement des données

Pour une cartographie avec un véhicule mobile, nous allons parler de système d'acquisition lorsque l'on parle de ce qui sert à acquérir les données. En effet, nous faisons la différence entre un capteur d'acquisition, qui dans le cas de la cartographie 3D, peut être une caméra ou un LIDAR, voire d'autres types de capteurs aussi, et un système d'acquisition, qui est un ensemble de capteurs d'acquisitions : on peut très bien n'avoir qu'un seul capteur, un LIDAR multi-couches, ou plusieurs capteurs, comme par exemple plusieurs LIDARs mono-couches.

Lorsqu'une carte est produite, il est utile de la localiser, c'est-à-dire de la référencer par rapport à une origine connue. Il existe deux types de localisation :

- une localisation « locale », c'est-à-dire par rapport à une référence choisie arbitrairement, généralement le point de départ de l'acquisition. Ce type de référencement est utilisé pour des cartographies ponctuelles, où les données produites n'ont pas besoin d'être mises en commun avec d'autres données acquises au préalable.
- une localisation globale, ou géoréférencement, qui permet de référencer les cartes par rapport à une origine connue. Cela permet par exemple de recaler entre elles différentes cartes obtenues lors d'acquisitions séparées, ou encore de mettre en commun des cartes obtenues avec des systèmes différents.

Ces deux types de référencement fonctionnent en pratique de la même manière, ce qui change est l'origine par rapport à laquelle on localise les données.

La figure 2.6 présente les différents changements de repère des données acquises qui conduisent à leur géoréférencement dans le repère lié à la Terre, en partant du système d'acquisition qui permet d'obtenir des données sur l'environnement. Le système d'acquisition illustré par cette image est composé de un ou plusieurs capteurs LIDARs. On peut relever 3 étapes importantes pour le référencement d'un point, comme le montre la figure 2.7 dans le cas d'un géoréférencement ; dans le cas d'un référencement non global, une origine est choisie et le référencement des données se fait par rapport à cette origine, la différence étant au niveau des capteurs utilisés pour le référencement du véhicule :

1. La première étape est l'acquisition des données. Quel que soit le type de système utilisé pour effectuer l'acquisition dans de la cartographie 3D, dans un premier temps, les données doivent être référencées dans le repère du système, cartésien de préférence puisqu'on vise le géoréférencement des données dans le repère monde qui est lui-même cartésien. Avec un système LIDAR, il est possible d'obtenir ce résultat avec le **calibrage intrinsèque** du système, qui permet de décrire le changement de repère permettant de référencer les données acquises brutes par l'ensemble des capteurs LIDARs dans un même repère cartésien.
2. La deuxième étape est le **calibrage extrinsèque** : un véhicule d'acquisition, pour le référencement, utilise le plus souvent une centrale inertielle qui permet d'avoir la position du véhicule selon une référence choisie au début de l'acquisition, et le calibrage extrinsèque du système mobile permet de caractériser la transformation qui décrit au mieux le changement de repère précis permettant de correctement référencer les données dans le repère lié à la centrale inertielle. Plus généralement, cela permet de référencer les données dans le repère « body », qui est un repère lié au véhicule.

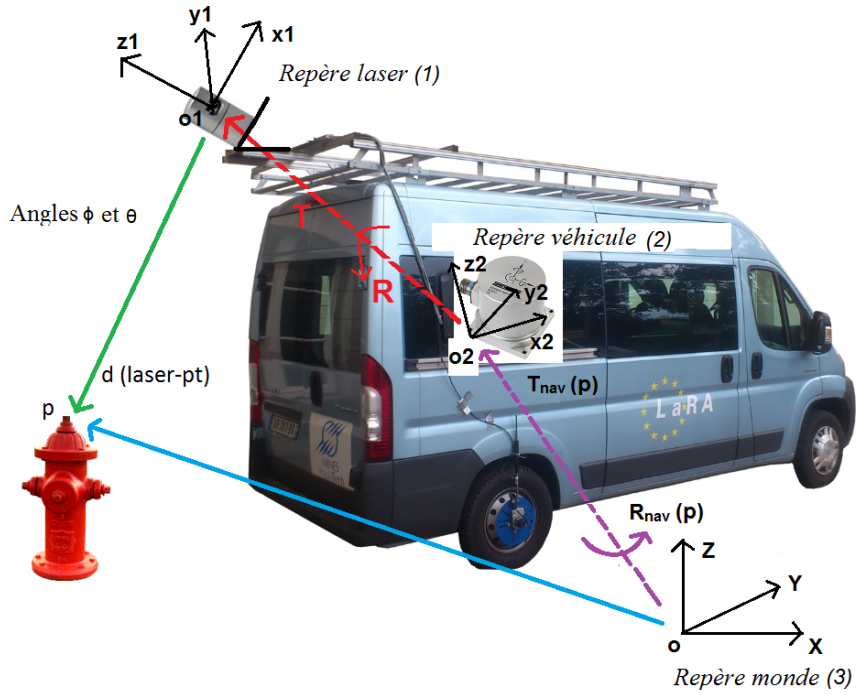


FIGURE 2.6 – Illustration présentant les transformations entre les différents repères d'un véhicule mobile d'acquisition

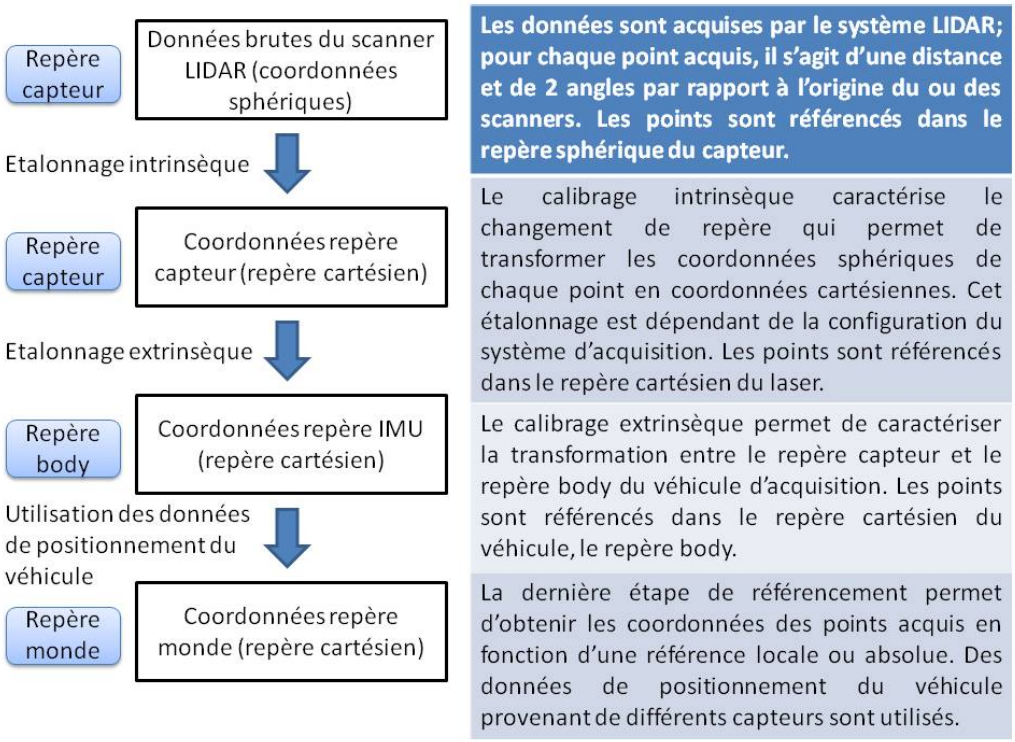


FIGURE 2.7 – Description des différentes représentations d'un point lors d'une cartographie

3. La dernière étape est l'étape de **référencement**. Les données sont avant cette étape référencées dans le repère body, du véhicule, et avec différents capteurs, il est possible de connaître de manière précise la position du véhicule par rapport à une origine précise, à une fréquence donnée qui dépend des capteurs. Une dernière transformation des données permet alors d'avoir des données référencées : la fréquence à laquelle on connaît la position du véhicule est en règle générale plus faible que la fréquence d'acquisition des données par le système LIDAR ; une interpolation est alors effectuée pour compléter les positions du véhicule dans le repère choisi pour l'ensemble des points composant la carte 3D.

$$p'(t) = \begin{pmatrix} x'(t) \\ y'(t) \\ z'(t) \end{pmatrix} = \begin{pmatrix} \rho * cos(\theta(t)) * cos(\phi) \\ -\rho * sin(\theta(t)) * cos(\phi) \\ \rho * sin(\phi) \end{pmatrix} \tag{2.1}$$

$$p(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = R_{nav}(p'_t) * (R_{ext} * p'_t + T_{ext}) + T_{nav}(p'_t) \tag{2.2}$$

Les équations 2.1 et 2.2 présentent le processus de géoréférencement d'un point acquis par un capteur LIDAR : le capteur récupère trois informations de position par point, qui sont la distance ρ du capteur au point acquis, et les angles horizontaux et verticaux θ et ϕ décrivant l'orientation du capteur lors de l'acquisition. L'équation 2.1 présente le passage des coordonnées sphériques d'un point à ses coordonnées cartésiennes dans le repère capteur, en supposant que le laser qui a acquis le point est centré par rapport au capteur : c'est le calibrage **intrinsèque**.
Ensuite, dans l'équation 2.2, on a la projection des données du repère du capteur au repère body du véhicule, avec la matrice de rotation R_{ext} et le vecteur de translation T_{ext} : ces paramètres sont estimés précisèment avec l'étalonnage des paramètres de calibrage **extrinsèque**. Enfin, la matrice de rotation R_{nav} et le vecteur de translation T_{nav} décrivent le **géoréférencement** des données : ils dépendent de l'instant d'acquisition (et changent donc pour chaque point) car le véhicule est en déplacement, et permettent d'exprimer le point dans le repère global pris comme référence.

Ce que l'on appelle repère monde dans le cadre d'un géoréférencement est un repère lié à la Terre, qui permet de mettre en commun un ensemble de données provenant de différentes sources et ayant une origine commune. Les données sont positionnées à l'aide de leur coordonnées géographiques avec l'aide d'un terminal capable d'être localisé dans le référentiel géographique, et il existe différentes techniques pour géolocaliser des données, présenté en détails dans [Geolocation Technologies, 2017] et [Wikipedia Géolocalisation, 2017] :

- **La géolocalisation GSM**, où l'on se sert des positions des antennes relais de téléphonie mobile pour localiser le terminal concerné. La précision de la localisation varie entre quelques dizaines de mètres et plusieurs kilomètres selon la couverture des zones. Il existe plusieurs types de localisation par GSM, comme par exemple la technique « time difference of arrival » (TDOA), où le terminal envoie un signal à l'antenne la plus proche, qui le lui renvoie, et qui permet ensuite d'estimer la position de l'appareil. La technique la plus utilisée car peu coûteuse est « l'identification de cellule » (Cell-id), qui consiste à récupérer les identifiants des antennes GSM ; en s'aidant de la position géographique des-dîtes antennes, le terminal peut estimer sa position.
- **La géolocalisation par wi-fi**, où le terminal équipé d'une technologie wi-fi peut récupérer les identifiants des bornes wi-fi détectées, et de la même manière que pour la technique Cell-id avec la géolocalisation GSM, le terminal peut estimer sa position à partir de celle des bornes wi-fi.
- **La géolocalisation par satellites**, qui est la technique la plus utilisée en cartographie mobile. Le terminal utilisé pour déterminer la position géographique du véhicule est un récepteur GPS, qui reçoit des signaux d'une constellation de satellites, en orbite autour de la Terre

comme le montre la figure 2.8 a). Les figures 2.8 b) et c) sont des exemples de terminaux GPS qui permettent de recevoir les signaux émis par les satellites, et qui convertissent ces données en positions géographiques pour le terminal GPS. En général, le récepteur GPS reçoit des signaux d'au moins quatre satellites : en effet, il doit déterminer 4 données qui sont les 3 positions du récepteur dans un référentiel de référence, et le décalage de son horloge par rapport au temps GPS des satellites. Cette technique est très populaire, notamment parce qu'il est possible d'atteindre une précision fine avec.

(a) (b) (c)

FIGURE 2.8 – Exemple d'outils utilisés pour la géolocalisation avec GPS : a) Exemple de constellation de satellites ; b) GNSS Leica [Leica GNSS sensor, 2017] ; c) GNSS Trimble [Trimble GNSS sensor, 2017]

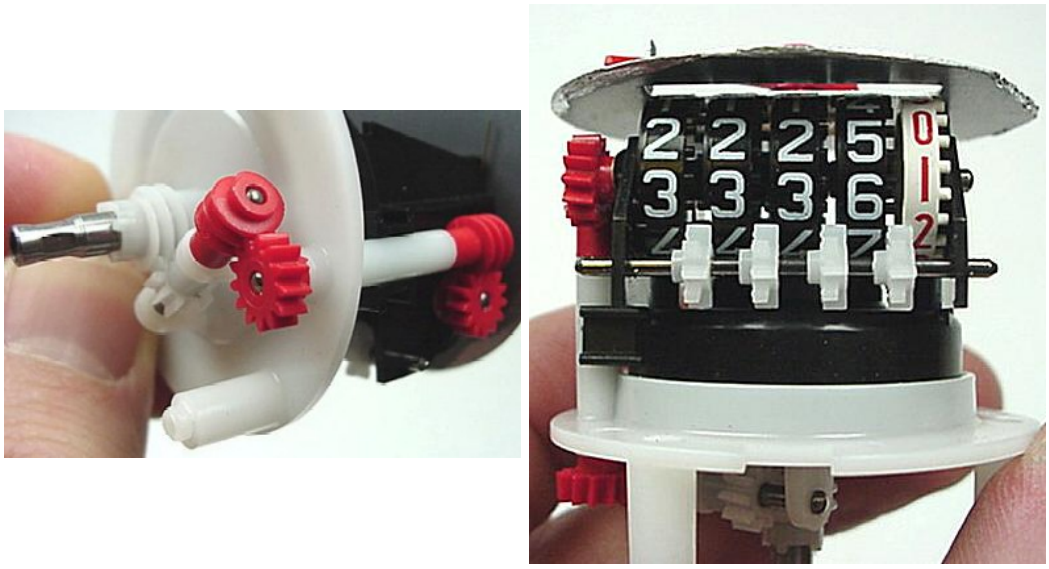
En plus des capteurs de type GPS qui sont utilisés pour localiser le véhicule, de nombreux capteurs sont utilisés en cartographie mobile, pour avoir un positionnement plus fin du véhicule ou des données plus complètes lors de la cartographie.

2.2.2 Capteurs proprioceptifs

Dans un premier temps, nous allons nous intéresser aux capteurs dits **proprioceptifs** utilisés en robotique mobile. De manière générale, on peut définir un capteur proprioceptif par le fait qu'il permet de renseigner sur l'état d'un élément « intrinsèque » au robot. En effet, ces capteurs n'utilisent pas de référence absolue pour fournir des mesures, et permettent de renseigner sur l'évolution relative de certaines données utiles lors du fonctionnement du robot. On peut par exemple trouver des capteurs de température, qui informent sur le niveau de chauffe d'un élément spécifique, ou des capteurs de tension pour surveiller l'utilisation des batteries.

En cartographie mobile, ou plus généralement en robotique mobile, de nombreux capteurs proprioceptifs peuvent être utilisés. Un des capteurs proprioceptifs les plus utiles est l'**odomètre**, qui permet d'avoir un déplacement relatif du véhicule. L'odomètre mesure une distance parcourue par le robot mobile en se basant sur les rotations d'une des roues non motrices en général. Il existe deux types d'odomètres : les premiers odomètres qui ont été utilisés, et qui le sont toujours au jour d'aujourd'hui, sont les odomètres **mécaniques**. Il s'agit d'un ensemble d'engrenages qui captent et réduisent la rotation de la roue proportionnellement à la taille de l'odomètre, et donnent approximativement la distance parcourue en fonction de la rotation résultante des engrenages. L'affichage de la distance parcourue se fait alors sur un ensemble de roues, qui tournent avec la rotation résultante des engrenages. La figure 2.9 illustre ces éléments : en a), on voit une partie du mécanisme de réduction de la rotation entrante, et en b), on a une vue d'ensemble des roues servant à l'affichage de la distance parcourue.

Le deuxième type d'odomètre que l'on peut trouver en robotique mobile est l'odomètre **électronique**. Il en existe deux versions : la première qui peut être utilisée ressemble à l'odomètre



(a) (b)

FIGURE 2.9 – Exemple d'odomètre mécanique : a) Engrenages de l'odomètre pour la réduction de la rotation entrante ; b) Roues servant à l'affichage de la distance

mécanique, la seule différence étant au niveau de l'affichage ; un écran remplace les roues et la rotation des engrenages génère des impulsions électriques qui sont utilisées pour l'affichage. L'autre type d'odomètre électronique se sépare en deux parties, un dispositif circulaire denté qui est monté sur une roue du véhicule, et un capteur magnétique qui mesure les rotations du dispositif, avec le passage des dents devant le capteur. Une impulsion électrique est alors envoyée à l'ordinateur embarqué, qui peut alors évaluer la distance parcourue par le véhicule. D'autres versions existent, avec un capteur optique à la place du capteur magnétique, et qui envoie des impulsions électriques de la même manière. On peut voir un exemple d'odomètre électronique sur la figure 2.4, sur la roue arrière du véhicule.

Un autre type de capteur utile en cartographie mobile est le **gyromètre** : il s'agit d'un appareil permettant de mesurer des vitesses angulaires autour d'un axe donné, en général un des trois axes d'un repère cartésien défini. Comme il s'agit d'évaluer des déplacements du véhicule de cartographie, le repère en question est principalement le repère monde, et le gyromètre permet d'avoir les vitesses angulaires du véhicule autour d'un des trois axes de ce repère. On trouve deux types de gyromètres principalement utilisés en cartographie mobile :

- Les gyromètres **mécaniques** : dans cette classe de gyromètre, on trouve aussi les gyromètres **avec éléments rotatifs**, comme présenté avec la figure 2.10. Une toupie est en rotation et est maintenue en équilibre grâce à la conservation du moment angulaire, et la rotation du véhicule induit une inclinaison du gyromètre, qui peut du coup aussi être utilisé en tant que gyroscope, puisqu'il peut directement mesurer la variation angulaire du véhicule. La théorie de son principe de fonctionnement est expliquée sur [Gyroscope et gyromètres à éléments rotatifs, 2017].

- Les gyromètres **optiques**, qui sont plus utilisés car il n'y a plus de frottements comme pour les gyromètres mécaniques, ce qui implique une durée de vie plus élevée et moins de pannes.

On trouve 2 types de gyromètres optiques : le **gyromètre laser** ou **gyrolaser** et le **gyromètre à fibre optique** ou **gyrofibre**. Ces deux gyromètres optiques fonctionnent sur le même principe, et permettent de mesurer des vitesses angulaires en se basant sur le principe de l'**effet Sagnac**.

Le gyromètre optique consiste en l'envoi de 2 rayons lasers dans des sens opposés, qui ont une trajectoire circulaire de rayon **R** et des fréquences différentes induites par l'amplitude de la rotation.

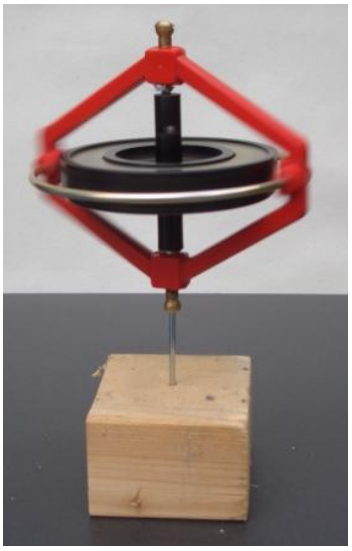


FIGURE 2.10 – Exemple de gyromètre avec éléments rotatifs

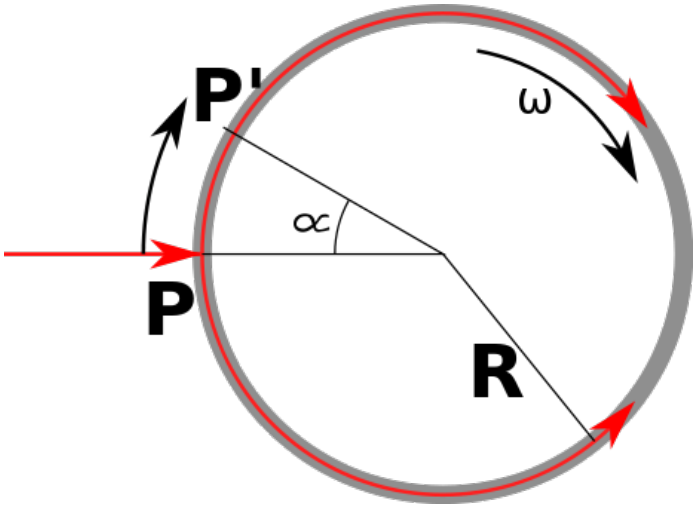


FIGURE 2.11 – Exemple de gyromètre optique (source : [Wikiwand site web, 2017])

La figure 2.11 montre le principe de fonctionnement du gyrolaser ; l'objet circulaire (le gyromètre) tourne à une vitesse angulaire ω constante, et après un temps de rotation Δt, le point P se trouve à la position du point P'. On montre alors que la différence de chemin optique Δp entre les deux rayons vaut :

$$\Delta p = \frac{4 * S}{c} * \omega \tag{2.3}$$

où **S** représente la surface du disque entre les points P et P', et **c** la vitesse de la lumière dans le vide.
A l'heure actuelle, le gyrolaser est très utilisé, notamment grâce à son coût faible. Cependant, le gyrofibre est plus précis et génère moins de bruit, ce qui rend son application très utile. De plus, la technologie s'améliore continŭment et le coût diminue d'année en année, alors que la technologie du gyrolaser ne s'améliore pas aussi vite.
Enfin, une dernière catégorie de gyromètres très répandue sont les « MEMS » (Micro Electro Mechanical Systems, ou Micro-Système Electromécanique en français), présenté dans [Woodman, 2007].

Ces capteurs ont la particularité de ne nécessiter que peu de composants en comparaison des gyromètres optiques et mécaniques, et sont aussi bien moins cher à produire. Ces capteurs effectuent leur mesure en se basant sur la force de **Coriolis** lors d'un virage, qui est mesurée à l'aide d'un élément vibrant dans le capteur. Les avantages des MEMS sont nombreux : petite taille ; très léger ; faible coût de production (surtout à grande échelle) ; faible consommation électrique. Cependant, ils restent moins précis que les gyrofibres et le choix du gyromètres dépend alors des besoins et de la précision voulue.

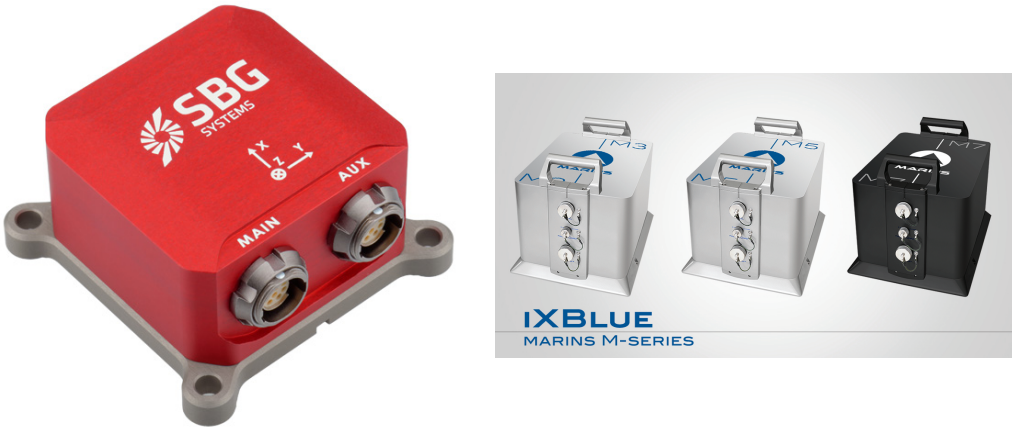
Tout comme les gyromètres, les **accéléromètres** sont des capteurs aussi très utiles en cartographie mobile, toujours dans une optique de suivi des déplacements du véhicule d'acquisition. Un accéléromètre permet de mesurer une accélération dans une direction donnée, et avec une intégration, permet de renseigner sur la vitesse du véhicule dans cette direction. Son fonctionnement peut être schématisé par un système masse-ressort, qui sort de sa position de repos lors d'une accélération du véhicule, et en négligeant l'action du poids qui est faible au vu de la masse du ressort, il est possible d'avoir le déplacement de la masse par rapport à un référentiel Galiléen, déplacement qui est corrélé à l'accélération du véhicule. On trouve de nombreuses sortes d'accéléromètres, allant du simple système masse-ressort à des capteurs équipés d'éléments piézoélectriques ou vibrants ; pour ces derniers, une accélération du véhicule entraîne un changement d'état interne du capteur, ce qui permet de mesurer l'accélération.
Il y a 2 types d'accéléromètres différents, avec plusieurs technologies différentes :

- Les accéléromètres **non asservis**, qui donnent une indication de l'accélération du véhicule directement en fonction du déplacement de la masse de l'accéléromètre.
- Les accéléromètres à **asservissement**, qui sont préférés pour les véhicules mobiles car leurs centres de gravité varie légèrement au cours du déplacement. L'accélération effective est mesurée à la sortie d'une boucle de correction, qui corrige l'accélération fournie en fonction de l'énergie nécessaire au ressort pour que la masse retrouve sa position d'origine. La précision des mesures est bien meilleure, mais ces accéléromètres coûtent aussi plus cher, et sont surtout utilisés dans des application sensibles où une précision très fine est requise.

Aussi, comme pour les gyromètres, on trouve une autre catégorie d'accéléromètres, les MEMS. Ils ont les mêmes avantages et inconvénients que les MEMS gyromètres.

Enfin, un capteur proprioceptif très populaire est la **centrale inertielle**, qui est en réalité composé de plusieurs gyromètres et accéléromètres : en effet, elle est généralement constituée de 3 gyromètres et de 3 accéléromètres pour avoir les vitesses angulaires autour de chacun des 3 axes du repère de référence de l'acquisition et les vitesses de déplacements selon ces 3 axes. En général, les véhicules mobiles d'acquisition ne sont équipés que d'une centrale inertielle et d'un odomètre, puisque les accéléromètres et gyromètres sont déjà intégrés à la centrale. L'appareil se charge d'intégrer les données des différents capteurs pour avoir la vitesse de déplacement et les angles lors des rotations du véhicule. La figure 2.12 présente 2 exemples de centrales inertielles commerciales, utilisées aussi bien pour des systèmes mobiles marins que routiers, voire aussi pour les drones, notamment avec la petite taille de certains modèles.

La centrale inertielle est le capteur proprioceptif le plus indispensable quand on parle de cartographie mobile, et dès que le géoréférencement précis des données est requis. En effet, le principal capteur utilisé pour déterminer la position du véhicule mobile est le GPS, qui fournit un positionnement absolu du véhicule dans un repère monde et par rapport à une origine de référence en se basant sur des données reçues de différents satellites. Cependant, ces données ont une certaine précision qui peut ne pas être suffisante, et en milieu urbain, de nombreux problèmes viennent fausser les données : canyons urbains ; réflexion sur les bâtiments... C'est pourquoi la centrale inertielle est très utilisée (avec l'association d'autres capteurs, comme l'odomètre) pour corriger le positionnement du véhicule : le déplacement du véhicule est estimé à partir d'une origine en général géoréférencée, avec une fréquence élevée de l'ordre d'une centaine de Hertz en moyenne. Les données des capteurs proprioceptifs sont fusionnées avec les données du GPS, en utilisant un filtre de Kalman par exemple, pour avoir une estimation plus précise de la position du véhicule mobile.



(a) (b)

FIGURE 2.12 – Exemples de centrales inertielles : a) Centrale inertielle Xsens ; b) Centrales inertielles Ixblue

2.2.3 Capteurs extéroceptifs

Les capteurs **extéroceptifs**, au même titre que les capteurs proprioceptifs, font partie intégrante de la cartographie mobile. Une définition du capteur extéroceptif que l'on peut donner est qu'un tel capteur renseigne sur l'environnement, l'état **externe** du véhicule mobile, ou plus généralement du robot. Ainsi, un des capteurs extéroceptifs les plus connus et l'un des plus utilisé est le GPS, que l'on a présenté dans la section 2.2.1 : en effet, le GPS permet d'avoir un positionnement du véhicule, comme des capteurs proprioceptifs tels que l'odomètre ou la centrale inertielle, mais la différence est que ce positionnement est absolu, par rapport à une référence externe non choisie par l'« utilisateur » du GPS, alors que la référence pour la centrale inertielle par exemple est fixée à chaque début d'acquisition.
D'autres capteurs extéroceptifs existent, et on peut notamment ajouter dans cette catégorie l'ensemble des capteurs d'acquisitions tels que les caméras, les LIDARs ou encore les Radars. Ces capteurs permettent de scanner l'environnement, et renseignent du coup sur ce qui est externe au véhicule : où se trouve le véhicule, où peut-il aller, comment peut-il interagir avec son environnement.

Le **Radar**, de l'anglais « RAdio Detection And Ranging », utilise la réflexion des ondes électromagnétiques qu'il émet pour cartographier l'environnement. Il est possible de déterminer la distance des cibles au radar avec le temps de retour de l'onde, la direction de la cible par rapport au radar en fonction de la direction d'envoi de l'onde, voire même la forme et la nature de la cible en fonction de l'intensité et de la forme de l'onde de retour.
Les radars ont commencé à être utilisés dans les environs de la seconde guerre mondiale : les fondements théoriques étaient mis en place depuis le début des années 1900, notamment avec le dépôt d'un brevet sur le « Telemobiliskop » par l'allemand **Christian Hülsmeyer**, mais c'est en 1934 que les premiers radars à ondes décimétriques équipent des navires français. Un peu plus tard, dans les années 1950, le **radar à synthèse d'ouverture** (RSO) voit le jour, qui est un radar imageur et qui permet d'avoir une meilleure résolution verticale que les précédents modèles, sans pour autant devoir modifier la taille de l'antenne émettrice.
On trouve actuellement 2 types de radar principalement utilisés : les radars « monostatiques » et les radars « multistatiques ». La majorité des radars actuellement utilisés sont monostatiques, car pour ces radars, la partie électronique et l'antenne sont partagés entre l'émission et la réception des ondes radios, ce qui réduit l'encombrement et le coût des radars, mais ce qui pose un problème dans le cas de pannes par exemple puisque l'émission et la réception sont liés. Les radars multistatiques,

au contraire, voient le récepteur et l'émetteur séparés. Il faut toutefois gérer la synchronisation entre l'émetteur et le récepteur puisque ceux-ci sont indépendants. Un système radar est principalement composé des éléments suivants :

- Un émetteur qui produit l'onde radio.
- Un duplexeur, qui dans le cas d'un radar monostatique se charge de rediriger correctement les ondes émises et reçues par le radar.
- Une antenne qui diffuse l'onde et qui selon le modèle de radar, reçoit aussi l'onde radar dans le cas des radars monostatiques.
- Dans le cas d'un radar multistatique, la réception peut se faire avec une antenne différente de l'émission.

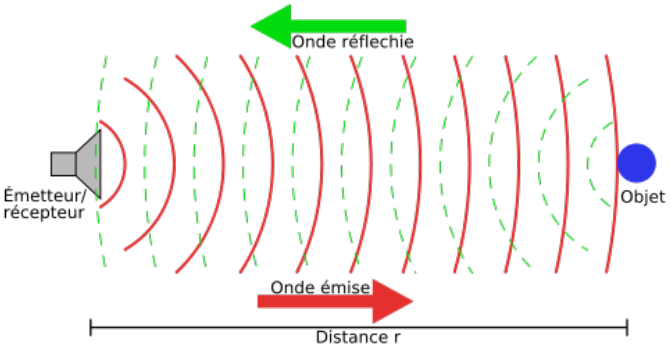


FIGURE 2.13 – Principe de l'écholocation utilisée par le radar

La figure 2.13 présente schématiquement le fonctionnement du radar. Dans [Gérossier, 2012] et [Vivet, 2011], des travaux de thèse sur la cartographie et localisation simultanée à partir de données issus de capteurs radars sont présentés. Ils soulèvent et proposent des solutions à des problématiques de localisation d'un véhicule mobile, avec pour contrainte l'utilisation d'un radar comme capteur extéroceptif.

La **caméra** est un autre type de capteur exterocéptif très utilisé en cartographie mobile. Le précurseur de la caméra est la **chambre noire**, dont le principe remonterait au 5éme siècle selon des écrits de **Mozi**, un philosophe chinois. Il aurait correctement supposé que dans une chambre noire et à travers un petit « trou » comme celui du modèle de la caméra sténopé, les images d'un quelconque objet seraient inversés du fait que la lumière se propage en ligne droite.

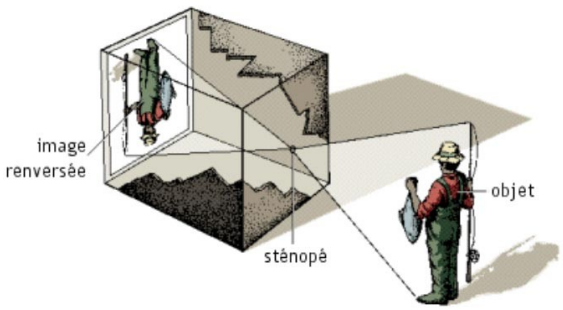
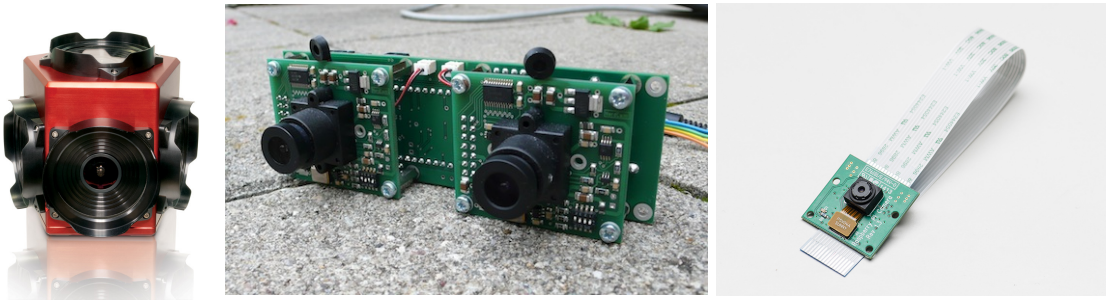


FIGURE 2.14 – Modèle sténopé

Dans l'Histoire, d'autres écrits ont été trouvés, traitant du même phénomène, notamment d'Aristote au 4ème siècle et Alhazen au 10ème siècle. La première image « permanente » d'une caméra

date de 1826, produite par Niépce avec un temps d'exposition de 8 heures sur de l'étain enduit de bitume. Un peu plus tard, c'est au tour de Daguerre de proposer le développement d'une image par exposition sur des solutions salines. Finalement, la première caméra commerciale, qui reprenait le principe de Daguerre appelé « Daguerreotype », est construite par **Alphonse Giroux** en 1839 ; le temps d'exposition était alors variable, entre 5 et 30 minutes. Avec le temps, de nouveaux modèles de caméras sont proposés, toujours utilisant le principe de Daguerre, et avec des temps d'exposition de plus en plus courts, de quelques secondes seulement dans les environs des années 1850. On arrive alors en 1885, ou la pellicule est **inventée** par George Eastman. Sa première caméra, qu'il appela « Kodak », fut mise en vente en 1888. Enfin, c'est en 1975 que les caméras numériques ont été inventées, par « Steven Sasson », un ingénieur de Kodak. Il a utilisé un capteur CCD (Charged-Couple Device) pour capturer la lumière, et qui permettait à l'époque d'avoir des résolutions de l'ordre de 0.01 MegaPixels. Les caméras numériques ont été popularisées au début des années 2000, jusqu'à finir par presque entièrement remplacer les caméras à pellicules de nos jours.



(a) (b) (c)

FIGURE 2.15 – Quelques exemples de caméras : a) Caméra ladybug ; b) exemple d'une paire stéréo ; c) Caméra pour Raspberry

Les caméras sont très utiles en cartographie, que ce soit de la cartographie statique ou mobile. En effet, on trouve tout d'abord plusieurs types de caméras différentes, qui permettent d'avoir des jeux de données spécifiques à certaines applications. La figure 2.15 présente trois exemples de systèmes de cartographie équipés de caméras :

- la ladybug est un système composé de 6 caméras, 5 couvrant un champs de vue horizontal de 360°, et la dernière caméra pointant vers le haut du système. La caméra permet d'obtenir des images 360, inclues dans une sphère autour du système.
- la paire stéréoscopique permet de faire de la reconstruction 3D à partir de paires d'images, et donc d'obtenir des nuages de points.
- la caméra pour Raspberry est un exemple de caméra que l'on peut fixer sur une carte embarquée (Raspberry pour cette caméra). Cela permet d'équiper des petits appareils mobiles de caméras assez performantes, sans encombrer le véhicule avec une caméra trop grande et trop lourde.

On peut trouver plusieurs systèmes de cartographie mobiles qui sont équipés de caméras : c'est par exemple le cas de [Mazzei *et al.*, 2012] et [Huang et Barth, 2009] qui présentent deux systèmes de cartographies équipés aussi bien de caméras que capteurs LIDARs. Historiquement, et comme pour le LIDAR, c'est avec des systèmes aéroportés que les caméras ont été utilisés pour la première fois en cartographie mobile, pour effectuer de « l'imagerie aérienne ». C'est en 1858 que les premières images aériennes ont été prises par **Gaspard-Félix Tournachon** au-dessus de Paris, à bord d'une montgolfière ; ces images ont été perdues, et les plus anciens clichés aéroportés existants remontent à 1860, pris par **James Wallace Black** et **Samuel Archer King** au-dessus de Boston. La technologie de la cartographie aérienne s'est développée pendant le 20ème siècle, notamment avec les enjeux militaires des grandes guerres, et reste très utilisée de nos jours, dans le domaine de

la photogrammétrie par exemple parce que cette technologie permet de cartographier des grandes surfaces très rapidement par rapport à d'autres types d'acquisitions.

Un autre type de capteur extéroceptif très utilisé est le LIDAR. « LIDAR » est l'acronyme de LIght Detection And Ranging, et il s'agit d'une technologie qui utilise la lumière et sa réflexion pour faire des mesures ou scanner l'environnement. Le premier laser opérationnel est le « laser à rubis », inventé par Théodore Maiman dans les années 1960, et utilisé pour de la télémétrie fine à grande distance. Ce type de laser a notamment été utilisé en 1962 pour mesurer la distance Terre-Lune. En 1971, un altimètre laser a été utilisé pour cartographier la surface de la Lune lors de la mission Apollo 15 [Abshire, 2010]. C'est aussi cette année là que l'acronyme LIDAR commence à être utilisé, et depuis, le LIDAR est devenu une technologie très répandue dans de nombreux domaines. Parmi les domaines d'applications, on trouve par exemple la numérisation 3D pour l'archéologie ou l'architecture, ou encore la surveillance du fait que le LIDAR est une technologie très précise et qui permet d'effectuer des acquisitions d'objets très éloignés de la source du laser pour certains modèles. Aussi, la cartographie, qu'elle soit aérienne, terrienne ou maritime, est un domaine où le LIDAR est très utilisé, toujours par souci du détail et du besoin de précision.
Il existe plusieurs technologies de LIDAR selon le besoin : certains LIDARs sont par exemple utilisés pour mesurer des vitesses, et cette technologie est alors différente de celle utilisée pour la cartographie mobile. Pour la cartographie terrestre (statique ou mobile), ce sont les télémètres **laser à balayage** ou **à décalage de phase** qui sont utilisés : les télémètres à balayages, utilisés pour la cartographie mobile, fonctionnent sur le principe du temps de vol pour mesurer la distance d'un objet au laser. Ces télémètres émettent des impulsions lasers à une certaine fréquence dépendant du modèle, et mesurent l'intensité du laser retournée pour avoir une information radiométrique associé au matériau de l'objet scanné, ce qui donne généralement des nuages de points à l'intensité représentée en niveaux de gris pour la lisibilité ; [Poullain, 2013] traite par exemple de l'utilisation de l'intensité retournée par un LIDAR et des différents modèles possibles pour caractériser cette intensité retournée. Il est tout à fait possible de « coloriser » le nuage selon le paramètre voulu, qu'il soit renvoyé par le télémètre ou calculé à partir des données.

Pour la cartographie mobile, plusieurs types de capteurs LIDARs existent : la figure 2.16 présente quelques exemples de capteurs utilisés.

- Les capteurs Sick et Hokuyo sont des LIDARs 2D, qui balaient l'environnement selon un plan, à une fréquence élevée de plusieurs centaines de Hertz. Cela donne en pratique des données en 2 dimensions, sauf si le scanner est mis en mouvement pendant le déplacement du véhicule mobile ; [Lin *et al.*, 2013] présentent une méthode d'étalonnage d'un système spécifique pour avoir des données 3D à partir d'un scanner 2D monté sur une plateforme pivotante, et [Nüchter *et al.*, 2007] présentent une application de SLAM utilisant un petit robot équipé d'un capteur Sick monté sur une plateforme contrôlée par un servo-moteur. Ces capteurs sont très populaires car ils sont légers, mais aussi peu chers en comparaison d'autres capteurs LIDAR.

- Le capteur Velodyne est un capteur dit « multi-couches »(ou multi-fibres), au contraire des capteurs 2D qui sont mono-couche ; ce capteur est un LIDAR 3D, qui permet directement de scanner l'environnement avec un champ horizontal de 360° et une ouverture verticale de plusieurs degrés, différent selon le modèle et le nombre de couches. Ce type de capteur est beaucoup plus cher que les capteurs 2D, même si les prix diminuent d'année en année, mais permet d'avoir des données plus denses en comparaison des capteurs 2D, notamment lorsque l'on s'éloigne un peu du véhicule d'acquisition.

- Les capteurs LIDARs Riegl sont en pratique des capteurs 2D, mais le système présenté sur la figure 2.16 d) est un système d'acquisition 3D, qui embarque plusieurs capteurs 2d ainsi qu'une centrale inertielle, et qui se charge de reconstruire des données 3D à partir de l'ensemble des données issues des capteurs 2D.

(a) (b)

(c) (d)

FIGURE 2.16 – Quelques exemples de capteurs LIDAR utilisés en cartographie mobile : a) capteur SICK 2D [Sick LIDAR sensor, 2017] ; b) capteur Hokuyo 2D [Hokuyo LIDAR sensor, 2017] ; c) capteur Velodyne 3D [Velodyne LIDAR sensors, 2017] ; d) capteur Riegl 3D [Riegl LIDAR sensor, 2017]

2.3 Les sources d'erreurs en cartographie mobile

La cartographie mobile est un domaine d'application où les enjeux sont assez importants : le but est de rivaliser avec les systèmes d'acquisitions statiques au niveau de la précision des cartes construites. En effet, la cartographie mobile permet d'avoir un volume de données bien plus important que la cartographie statique, mais plusieurs problèmes se posent alors :

- Tout d'abord, la géolocalisation des données est plus complexe parce que le système de cartographie se déplace : il faut connaître la position du système dans un référentiel global à tout instant, mais ces informations doivent aussi être synchronisées avec les données acquises.
- Toujours pour la localisation des données, de nombreux capteurs proprioceptifs que l'on a présenté dans la partie 2.2.2 sont utilisés, ce qui amène un niveau de complexité supplémentaire à gérer, notamment par rapport à la synchronisation des données et aux erreurs induites par ces capteurs.
- Enfin, on a un système qui se déplace dans un environnement non contrôlé, ce qui amène des erreurs au niveau des acquisitions.

2.3.1 Les erreurs liées aux capteurs

Une première source d'erreurs que l'on obtient lors d'une cartographie mobile provient de l'ensemble des capteurs utilisés sur un système mobile, aussi bien extéroceptifs que proprioceptifs.

Comme dans la partie 2.2, nous allons séparer cette sous-partie en fonction des erreurs liées à chaque type de capteurs.

2.3.1.1 Les erreurs des capteurs proprioceptifs

Les capteurs proprioceptifs renseignent sur des données internes au système d'acquisition, comme sa position par rapport à une origine arbitraire ou encore son déplacement relatif par rapport à une position connue. Toutefois, ces données sont incertaines : les capteurs donnent des informations avec plus ou moins de précision selon le modèle, mais même les capteurs les plus performants induisent des erreurs soit avec le temps, soit avec chaque mesure.

Dans un premier temps, les odomètres mesurent une distance parcourue par le véhicule mobile en prenant en compte la rotation d'une des roues non motrices du véhicule mobile. L'odomètre est adapté à la dimension de la roue sur laquelle elle est fixée : la mesure de distance dépend de la rotation de la roue en question, qui elle-même dépend de sa dimension. Lors d'un changement de roue, il faut paramétrer à nouveau l'odomètre : dans le cas d'un odomètre optique comme celui présenté en figure 2.4, la circonférence de la roue liée au déplacement du véhicule (la distance parcourue) est évaluée par la formule « Etat actuel - Etat initial », qui est mesurée avec plus ou moins de précision selon le nombre de dents sur l'odomètre. Ensuite, pour avoir une distance plus correcte, cette mesure est corrigée en multipliant par la fraction « Diamètre actuel de la roue / Diamètre standard de la roue », le diamètre standard étant le diamètre de la roue lors du montage de l'odomètre ; en effet, la dimension de la roue varie en fonction de plusieurs paramètres, comme l'usure de celle-ci ou encore la température ambiante. Malgré cela, des erreurs peuvent être introduites lors de la mesure de la distance, et ces erreurs s'accumulent pendant l'acquisition, ce qui peut entraîner des dérives de plusieurs mètres pour des acquisitions de plusieurs heures.
Un autre type d'erreur que l'on peut avoir concerne la marche arrière du véhicule : sur la plupart des odomètres numériques, le sens de déplacement du véhicule ne rentre pas en compte dans le calcul de la distance parcourue ; seuls la circonférence de la roue et son diamètre sont pris en compte. De ce fait, un véhicule qui se déplace sur une distance relativement élevée en marche arrière lors d'une acquisition peut entraîner une erreur sur la distance parcourue, qui sera plus élevée qu'en réalité : dans le cadre d'une cartographie, cela peut générer des erreurs lors du géoréférencement des données, mais ce problème est généralement corrigé par l'utilisation d'autres capteurs et la fusion des données de l'ensemble de ces capteurs.

Concernant les gyromètres, une erreur possible peut provenir de l'étalonnage du capteur, comme pour tous les autres types de capteurs, et cela donne des mesures qui sont fausses. Ensuite, on trouve des erreurs spécifiques à chaque type de gyromètre :

- **Les gyromètres optiques :** cette catégorie de gyromètres est assez stable et fonctionne correctement. Il n'y a pas d'usure matérielle importante comme pour les gyromètres mécaniques vu que les éléments qui le composent ne sont pas en mouvement. Cependant, un problème intrinsèque à ce type de capteur est présenté dans [Désilles *et al.*, 2011] : il s'agit du phénomène du « lock-in », qui apparaît quand l'amplitude de la rotation du véhicule que le gyromètre doit mesurer est trop faible, inférieure à un seuil. Les deux ondes qui sont mis en rotation dans le gyromètre ont la même fréquence d'entrée, mais celles-ci sont modifiées avec l'amplitude de la rotation que l'on mesure, ce qui crée le déphasage entre les deux ondes, et lorsque l'amplitude de la rotation est trop faible, les fréquences ne sont pas assez différentes pour que le déphasage soit précisément mesuré, et le gyromètre renvoie alors une rotation d'angle 0° bien que la rotation existe. Le seuil est assez bas en général, environ $0.1^\circ/s$, ce qui n'est pas contraignant pour de la cartographie mobile, mais peut l'être pour d'autres applications.
- **Les gyromètres mécaniques :** au contraire des gyromètres optiques, les gyromètres mécaniques subissent une usure matérielle qui peut générer des erreurs avec le temps. Ensuite, les gyromètres mécaniques fonctionnent sur le principe de la conservation du moment angulaire, et des erreurs peuvent apparaître à cause des approximations qui sont faîtes.

On peut aussi intégrer dans cette catégorie les MEMS gyromètres, qui sont composés d'éléments vibrants : [Woodman, 2007] présente les sources d'erreurs pour les MEMS en détail. On trouve par exemple que ces gyromètres ont un biais linéaire en fonction du temps sur la mesure d'angle (et donc constant sur la mesure de vitesse angulaire), qui peut être corrigé en compensant cette erreur sur la vitesse angulaire avant de l'intégrer pour avoir la position angulaire. Les MEMS sont aussi soumis à deux types de bruits : un bruit blanc qui provient de sources thermomécaniques, et un bruit de scintillation, qui fait partie de la famille des bruits « rose ». Le premier bruit génère une incertitude sur les mesures d'angles, qu'il est possible de borner en fonction de l'écart-type du bruit blanc, et le deuxième bruit est plus visible sur les basses fréquences (car les bruits roses ont une courbe de puissance proportionnelles à 1/f). Enfin, une dernière source de bruit provient d'un changement de température interne ou externe au capteur, qui aura pour effet d'introduire un résidu sur la mesure de la vitesse angulaire, devenant une erreur linéaire selon le temps pour la mesure de la rotation.

Pour les accéléromètres, on trouve aussi plusieurs sources d'erreurs : l'une d'entre elles provient de la configuration du capteur, qui implique des erreurs sur l'orientation et la position de l'accéléromètre sur le véhicule. [Tan et Park, 2002] présente ces types d'erreurs, ainsi qu'un moyen de les identifier. Ces imprécisions entraînent des erreurs de mesures de l'accélération, puisque celle-ci est mesurée dans une direction bien précise par le capteur et que la position du capteur permet aussi de correctement évaluer l'amplitude de l'accélération.
Pour les MEMS accéléromètres, les erreurs ont à peu près les mêmes origines que pour les MEMS gyromètres, car les deux types de capteurs font partie de la même famille des micro-systèmes électromécaniques. Les capteurs ont aussi un biais constant sur la mesure de l'accélération, qui implique un biais sur la position évoluant quadratiquement avec le temps : il suffit de le corriger avant intégration pour ne plus avoir ces effets. On retrouve aussi un bruit blanc et un bruit de scintillation, qui suivent les mêmes modèles que pour les MEMS gyromètres : il est alors possible de connaître leur influences sur les mesures de position après une double intégration, avec une certaine précision dépendant du modèle. Enfin, on a aussi une erreur provenant des changements de températures, qu'il est possible de corriger si il y a une indication de la température au niveau du capteur.

Les centrales inertielles sont composés de gyromètres et d'accéléromètres, ce qui implique que les sources d'erreurs pour les deux derniers types de capteurs entraînent aussi des erreurs sur les mesures de positions et d'orientations données par la centrale. Une source d'erreur provient de l'amplitude des données d'entrées : les accéléromètres et gyromètres ont une certaine plage d'amplitude pour les données en entrée, et des variations d'accélérations et/ou d'orientations trop brusques peuvent saturer un ou plusieurs capteurs, et du coup saturer la centrale inertielle qui ne sera pas capable de fournir des résultats corrects. Dans le même sens, des vibrations trop importantes vont conduire à un mauvais résultat. Aussi, une autre erreur peut être générée à cause d'un mauvais alignement des accéléromètres et des gyromètres. Les gyromètres et les accéléromètres font des mesures autour ou selon une seule direction, et comme un véhicule de cartographie mobile possède six degrés de liberté concernant ses déplacements, les trois gyromètres et accéléromètres doivent être correctement placés pour avoir des mesures correctes dans le repère monde cartésien. Une erreur peut apparaître si les capteurs sont mal positionnés ; en général, les accéléromètres sont positionnés avec un angle de 90° les uns par rapport aux autres, de même pour les gyromètres. Cela induit des corrélations entre les données mesurées par chaque capteur s'ils sont mal positionnés les uns par rapport aux autres, ou des erreurs de mesures si ils sont mal alignés par rapport aux axes du repère monde.

2.3.1.2 Les erreurs des capteurs extéroceptifs

Les capteurs extéroceptifs d'un véhicule mobile d'acquisition sont utilisés principalement pour déterminer une position « absolue » du véhicule ou pour scanner l'environnement. Ces capteurs peuvent avoir des problèmes internes, par exemple au niveau des composants électroniques - sur-

chauffe ; panne ou composant électronique qui grille ; mauvaise alimentation électrique -, mais nous allons principalement nous intéresser aux erreurs de mesures, notamment dues à des bruits ou des biais affectant les mesures des capteurs.

Le GPS donne une indication de positionnement du véhicule terrestre, à l'aide d'une constellation de satellites qui orbitent autour de la Terre. Les systèmes GPS sont très précis lorsqu'ils fonctionnent correctement, c'est-à-dire lorsque la communication entre les satellites et les récepteurs se fait sans obstacles, ou encore lorsque le récepteur est correctement calibré. Si ce n'est pas le cas, des erreurs peuvent apparaître :

- Une première source d'erreur provient d'une mauvaise synchronisation de l'horloge entre le récepteur GPS et les satellites qu'il utilise pour déterminer sa position.
- Ensuite, on trouve toutes les erreurs liées à une mauvaise communication entre les satellites et le récepteur GPS. Tout d'abord, les signaux provenants des satellites sont atténués simplement en traversant l'atmosphère, comme toutes les autres ondes qui se propagent sur Terre : aussi, une présence trop importante d'obstacles entre le récepteur et les satellites peut gêner la réception par le récepteur GPS. Dans la même catégorie, on peut inclure les canyons urbains : il s'agit d'un phénomène où le récepteur GPS ne reçoit aucun signal des satellites à cause d'une trop grande présence d'obstacles, comme la végétation ou les bâtiments par exemple.
- Un autre problème, toujours dû à la présence d'un grand nombre d'objets dans l'environnement du véhicule, est la réflexion des ondes émises par les satellites sur des surfaces quelconques avant d'atteindre le récepteur GPS, aussi nommé **multi-trajets** : de ce fait, la position du GPS est mal évaluée et entraîne un mauvais géoréférencement des données acquises par le système. Ce type d'erreur est aussi difficilement évitable, car pour la localisation du véhicule, le GPS est préféré à la centrale inertielle lorsqu'il reçoit un signal, et dans le cas du multi-trajets, le GPS reçoit bien un signal, qui est alors interprété comme étant correct.

Les radars émettent des ondes électromagnétiques, et par conséquent plusieurs phénomènes qui s'appliquent à la propagation des ondes peuvent venir perturber le signal retourné au radar, et donc fausser les mesures de celui-ci. On trouve ainsi un phénomène tel que la diffraction qui va « éclairer » des zones occultées, ou encore de la réfraction : dans ce cas-là, les mesures de distances et de positions des objets sont faussées car la réfraction modifie la marche de l'onde émise par le radar (ou qui lui revient) et sa vitesse, ce qui implique des mesures fausses.
Les caméras peuvent elles aussi engendrer des erreurs lors des acquisitions : d'une part, on a des erreurs dues à des mauvaises conditions de prises de vues. C'est le cas par exemple des conditions d'éclairage qui peuvent complètement modifier le rendu des images : les caméras fonctionnent sur un principe d'exposition à la lumière, et le rendu final dépend de l'intensité et de la durée de l'exposition. Dans le cas d'une acquisition mobile, le temps d'exposition est très court, et la qualité des images finales dépend de la luminosité qui ne peut pas être contrôlée. Aussi, on trouve des problèmes similaires à la réflexion des ondes radios pour la réflexion des ondes lumineuses avant d'atteindre la caméra, ce qui pose là encore des problèmes pour les acquisitions d'images.
Une autre source de problèmes pour les acquisitions avec une caméra est un mauvais étalonnage de celle-ci. En effet, comme tout appareil électronique, la caméra a besoin d'être calibrée afin de correctement projeter la scène 3D photographiée dans un repère 2D lié à la résolution de la caméra. Pour cela, il y a 2 étalonnages distincts :

- Un premier étalonnage des paramètres de calibrage extrinsèque qui permettent de caractériser la transformation entre le repère « monde » - dans lequel se trouvent la caméra et la scène à photographier - et le repère caméra, dont l'origine est le centre de la caméra. Pour cela, six paramètres (3 de translations et 3 de rotations) sont calculés.
- Un étalonnage des paramètres de calibrage intrinsèque, qui permettent de caractériser la transformation de l'information 3D de la scène en images, avec la prise en compte des facteurs d'échelles et des éventuelles distorsions engendrées par l'acquisition. Le résultat est l'image telle qu'on la connaît, dont la résolution dépend de la qualité des capteurs CMOS ou CCD.

$$\begin{bmatrix} su \\ sv \\ 1 \end{bmatrix} = \begin{bmatrix} k_u & s_{uv} & c_u \\ 0 & k_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.4}$$

L'équation (2.4) exprime la projection de la scène 3D dans une image, en considérant un modèle simplifié de caméra. En partant de la droite, les différents éléments représentent pour un point de l'image finale :

- Le vecteur des coordonnées 3D homogènes dudit point dans l'espace, afin de prendre en compte la perspective et simplifier la projection.
- La matrice des paramètres de calibrage extrinsèque, avec les 3 translations selon chaque axe et la matrice de rotation 3x3. Cette matrice est-elle aussi en coordonnées homogènes.
- Les deux matrices suivantes sont composées des paramètres de calibrage intrinsèque, et sont généralement écrites sous forme d'une seule matrice, issue de leur produit. La première matrice prend en compte la distance focale selon les deux axes horizontaux et verticaux de la caméra ; la deuxième matrice prend en compte les transformations affines entre l'image résultante et la projection 2D des données d'entrées. c_u et c_v décrivent la projection du centre optique de la caméra dans l'image ; s_uv est un facteur prenant en compte la potentielle non-orthogonalité des lignes et colonnes des cellules photo-sensibles servant à l'acquisition, mais il est généralement négligé ; enfin, k_u et k_v sont les facteurs d'agrandissement respectifs selon les axes horizontaux et verticaux de l'image.
- Le résultat est un vecteur des coordonnées homogènes 2D du point dans l'image, qui est en correspondance directe avec son homologue 3D à la transformation inverse près.

En réalité, le modèle utilisé pour l'étalonnage d'une caméra n'est pas aussi simple, et les paramètres ne sont pas connus très précisément dans la plupart des cas ; les paramètres sont estimés à l'aide d'une mire par exemple, ou à l'aide d'autres techniques, comme présentés dans [Tsai, 1987], [Weng *et al.*, 1992], ou encore [Kannala et Brandt, 2006] et [Strecha *et al.*, 2008] pour des méthodes plus récentes. A cause d'un mauvais étalonnage, des erreurs et distorsions peuvent apparaître dans les images.

Enfin, concernant les erreurs des systèmes LIDARS, elles sont pour la plupart assez proches de ce que l'on a pu voir précédemment : le LIDAR émet des ondes lumineuses, et récupère les ondes réfléchies par l'environnement pour le cartographier. Du coup, des problèmes liés aux ondes se posent alors : dispersions ; intensités de retour trop faibles ; multi-trajets ou encore multi-écho. On est aussi assez proche d'un système caméra dans le principe d'étalonnage : en effet, le LIDAR « capture » un ensemble de points 3D, et les projette dans le repère du capteur laser ; ce sont les données brutes que l'on obtient en sortie, qui sont équivalentes aux images pour une caméra. On peut remonter aux données 3D avec les différents étalonnages extrinsèque et intrinsèque (qui dépend du modèle de LIDAR), de la même façon que pour une caméra.

2.3.2 Les erreurs d'acquisitions

Nous avons vu dans cette section de nombreuses sources d'erreurs, liées à la technologie des capteurs utilisés pour la plupart. Toutefois, on trouve aussi de nombreuses erreurs indépendantes des capteurs qui viennent s'ajouter lors de la cartographie. C'est par exemple le cas du calibrage extrinsèque d'un système mobile : la figure 2.6 présente la transformation obtenue par calibrage, qui décrit le passage du repère capteur au repère body, lié au véhicule. Le repère body est en général associé à la centrale inertielle embarquée sur le véhicule d'acquisition, car une fois que les données acquises sont référencées dans le repère body, il suffit de connaître la position du véhicule dans le repère global pour géo-référencer les données : la centrale inertielle permet d'avoir le positionnement relatif du véhicule par rapport au point de départ de l'acquisition, et à l'aide d'autres capteurs

comme le GPS par exemple, on connaît la position de la centrale inertielle dans le repère monde. L'étalonnage des paramètres de calibrage extrinsèque est assez fastidieux à effectuer : il dépend des capteurs utilisés et sera différent selon que l'on ait un ou plusieurs capteurs d'acquisitions ; aussi, il dépend du positionnement des capteurs d'acquisitions et de la centrale inertielle sur le véhicule, et il faut alors le refaire lors du déplacement d'un de ces éléments. Nous rentrerons plus en détails sur les erreurs liées à un problème d'étalonnage de ces paramètres dans un autre chapitre.

L'environnement est aussi une source d'erreur importante ; par environnement, on englobe aussi bien le côté statique que dynamique de l'environnement. Effectivement, par rapport à ce que l'on peut appeler environnement « statique », on trouve la plupart des sources d'erreurs présentées en section 2.3.1.2, comme par exemple les problèmes de réflexion sur certains matériaux rugueux, ou au contraire spéculaires, ou encore les problèmes de multi-écho ou multi-trajets des ondes émises par les capteurs, ce qui vient gêner le référencement précis des données et fausser la cartographie. L'environnement peut aussi être dynamique : cela englobe les piétons, les véhicules qui se déplacent, le feuillage des arbres qui est rarement statique, ou encore les conditions météorologiques qui changent et qui peuvent poser des problèmes pour la cartographie mobile, comme par exemple avec un temps pluvieux, qui gêne la propagation des ondes dans l'air.

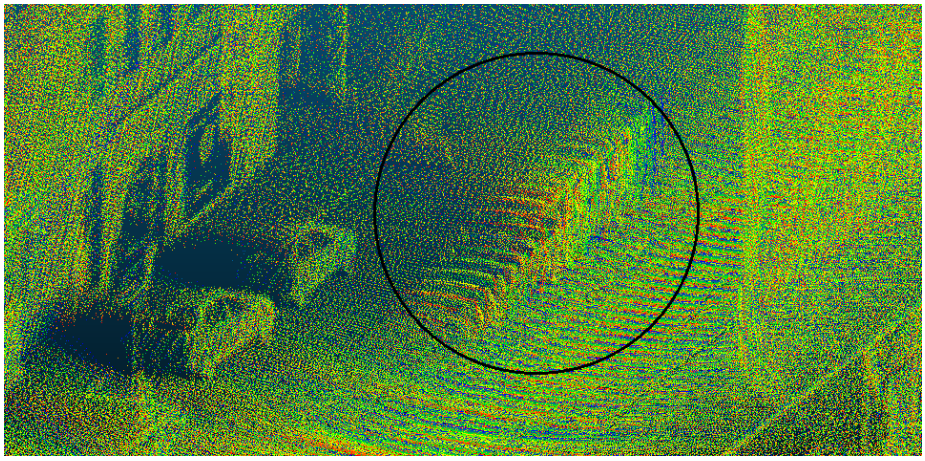


FIGURE 2.17 – Exemple d'erreurs liées au déplacement d'autres véhicules

La figure 2.17 montre un exemple de problèmes dûs aux déplacements d'autres véhicules lors de la cartographie : la zone noire montre la trace que laisse un véhicule qui arrive en sens inverse lors de la cartographie mobile. Ce genre d'élément pose problème dans la plupart des traitements appliqués à des nuages de points, que ce soit de la classification ou de l'extraction d'amers, et nécessite un traitement particulier pour le supprimer des données. Plus généralement, on peut parler de la circulation des véhicules et du trafic routier qui posent problème et ajoutent du bruit dans les cartes 3D construites.

Une autre source d'erreur provient d'une étape clé de la cartographie mobile : la fusion, ou association des données. En effet, les capteurs embarqués sur les systèmes mobiles ont chacun un rôle bien défini, et chaque capteur engendre des erreurs plus ou moins importantes qui sont traités au possible. Toutefois, pour avoir le « produit » final qui est un nuage de point correctement géo-référencé, les données provenant de chaque capteur doivent être fusionnées. Cette étape peut malheureusement être une source d'erreur assez importante, puisque les fréquences des différents capteurs doivent doivent être synchronisés, et certaines données doivent soit être sous-échantillonnées, soit sur-échantillonnées pour ne pas perdre d'informations. Une mauvaise synchronisation peut entraîner des erreurs sur le référencement des données, voire une perte d'informations.

2.3.3 Les données utilisées pour les expérimentations

Dans la section 2.1.3, nous avons présenté le véhicule de cartographie mobile utilisé par le centre de robotique pour effectuer des acquisitions. C'est avec ce véhicule que nous avons obtenu les données qui seront présentés ultérieurement pour les expérimentations : le capteur LIDAR Velodyne permet d'obtenir des cartes 3D de l'environnement, qui sont géo-référencées à l'aide des autres capteurs embarqués sur le véhicule, et cela donne des nuages similaires à celui présenté en figure 2.18.

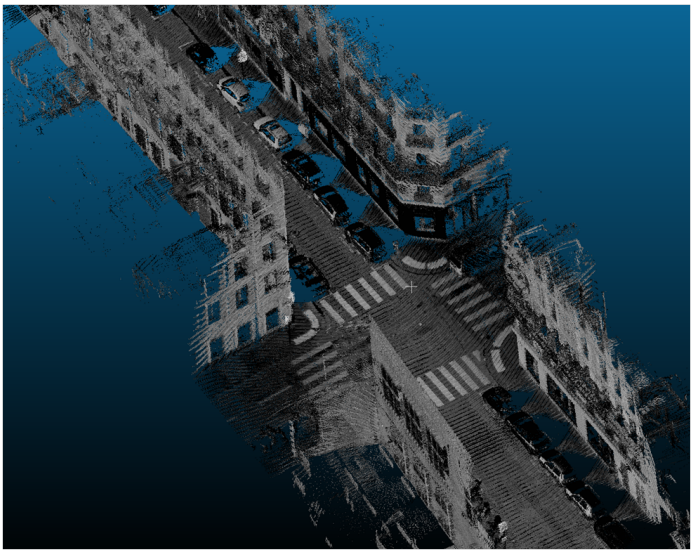


FIGURE 2.18 – Exemple de carte 3D obtenue avec le véhicule mobile du centre de robotique

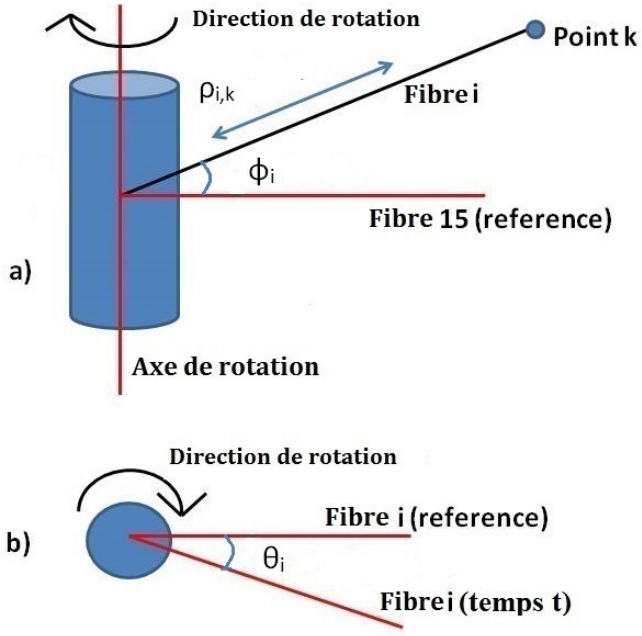


FIGURE 2.19 – Paramètres intrinsèques du Velodyne HDL-32E

Les cartes 3D, ou aussi appelées « nuages de points », sont des bases de données de millions de points qui correspondent aux acquisitions effectuées par le véhicule mobile : chaque point du nuage est issu d'un impact du capteur LIDAR qui a été retourné au capteur. Principalement, les données que le capteur LIDAR permet de mesurer pour chaque point du nuage sont la **distance entre le centre du laser et l'impact laser**, ainsi que **deux angles par rapport à l'horizontale et la verticale** du capteur LIDAR : cela donne des points référencés dans le repère sphérique du capteur. Ces différents paramètres sont présentés sur la figure 2.19. Ensuite, selon les besoins, d'autres paramètres peuvent être récupéré des acquisitions : la position du centre laser pour chaque point acquis ; l'intensité de retour des impacts lasers ; la position du véhicule estimée pour chaque point acquis.

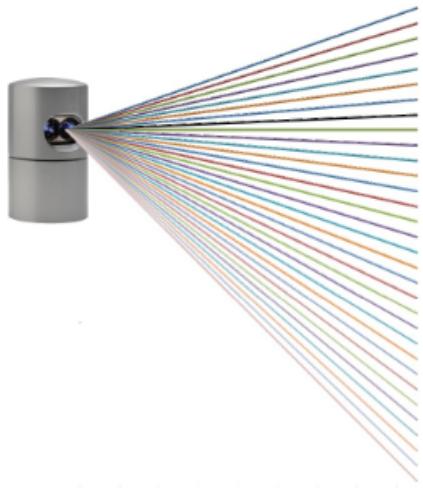


FIGURE 2.20 – Répartition des fibres du Velodyne HDL-32E

Pour nos besoins lors des tests de nos algorithmes, les nuages de points que l'on utilise sont composés des données suivantes pour chaque point :

1. La distance par rapport au centre du laser.
2. L'angle par rapport à la verticale du capteur laser.
3. La fibre qui a acquis le point : le capteur utilisé est un capteur Velodyne, qui est multi-couches, c'est-à-dire qu'il possède plusieurs lasers qui acquièrent chacun des points 3D. La figure 2.20 présente la répartition des fibres du Velodyne HDL-32E sur le capteur : les fibres sont positionnées sur l'axe de rotation vertical du capteur, et en connaissant le numéro de la fibre, il est possible de remonter à la valeur de l'angle par rapport à l'horizontale pour le point acquis par cette fibre.
4. La trajectoire d'acquisition du véhicule mobile, ce qui fait 6 paramètres additionnels qui nous permettent d'avoir les positions en translations et rotations du véhicule dans le référentiel lié à la Terre.

Avec ces 9 paramètres pour chaque point acquis, il est possible de géo-référencer le nuage acquis et d'avoir une carte 3D consistante. Pour l'affichage des nuages de points, nous utilisons un code de couleur basé sur les numéros de fibres, et non sur l'intensité retournée par les impacts lasers : en effet, nous verrons lors des expérimentations que l'intensité des impacts mesurées par le capteur LIDAR ne nous est pas utile dans les approches d'affinements que l'on se propose de traiter.

Chapitre 3

Affinement de nuages de points par optimisation des paramètres extrinsèques

Sommaire

Introduction

L'amélioration de la qualité des données en post-traitement, ou **affinement de données**, est un domaine de recherche vaste et très suivi par la communauté : le but est le plus souvent de compenser certaines faiblesses des systèmes d'acquisitions en supprimant des erreurs introduites pendant le

processus d'obtention des données, ou tout du moins en les réduisant. Le but de ces méthodes est généralement de lisser les données, ou de réduire les erreurs introduites lors de l'obtention de ces données : il s'agit de méthodes utilisées en pré-traitement d'autres applications où une bonne précision des données en entrée est requise.
Dans ce chapitre, nous présentons une méthode qui permet d'améliorer la qualité de jeux de données nuages de points : il s'agit de l'affinement de données de type nuages de points par optimisation des paramètres de calibrage extrinsèque. Ce chapitre est organisé de la façon suivante : dans un premier temps, nous montrons les effets d'un mauvais calibrage extrinsèque sur le rendu et l'exploitation des données issues d'une cartographie mobile ; ensuite, nous présentons une synthèse des travaux existants dans le domaine de l'optimisation du calibrage extrinsèque ; la section suivante traitera de la méthode d'optimisation que l'on a mise au point et de son principe, suivi d'une section présentant différents résultats d'optimisation ; enfin, nous évoquerons quelques facteurs limitants de notre algorithme. Aussi, il convient de noter que la méthode d'optimisation que l'on va présenter est applicable à des systèmes équipé de plusieurs LIDARs, ou d'un capteur LIDAR multi-fibres car nous utilisons la redondance de données entre plusieurs sources pour optimiser les paramètres de calibrage extrinsèque.

3.1 Importance des paramètres de calibrage extrinsèque

Pour avoir une cartographie correcte à l'issue d'une acquisition mobile, le calibrage extrinsèque occupe une place centrale : en effet, sur un véhicule mobile de cartographie équipé d'un capteur LIDAR, le capteur laser permet d'acquérir les données, qui sont référencées en coordonnées sphériques le plus souvent ; une fois que les données sont référencées dans le repère cartésien du capteur, il y a une transformation **extrinsèque** pour représenter les données dans le repère du véhicule, le repère « body ». La figure 2.6 illustre cette transformation, avec la rotation R et la translation T entre les deux repères. Avec un mauvais calibrage extrinsèque, après projection dans les repères body puis monde, les données ne seront pas correctement géoréférencées, faussant la cartographie et l'ensemble des traitements prévus sur les données.

3.1.1 Définition du calibrage extrinsèque pour un système mobile LIDAR

Le calibrage extrinsèque décrit un changement de repère, généralement entre le repère « capteur » - qui est lié au capteur LIDAR - et le repère « body » - qui lui est lié au véhicule mobile -. La transformation qui projette un point $p'(t)$ du repère capteur en un point $p(t)$ du repère body est la suivante :

$$p(t) = R_{ext} * p'_t + T_{ext} \tag{3.1}$$

$$\text{avec :} \begin{cases} T_{ext} = T(t_x, t_y, t_z) = [t_x \quad t_y \quad t_z]^T \\ R_{ext} = R(\alpha, \beta, \gamma) = R_{\mathbf{z}}(\gamma) * R_{\mathbf{y}}(\beta) * R_{\mathbf{x}}(\alpha) \\ \\ R_{\mathbf{x}}(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \\ R_{\mathbf{y}}(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \\ R_{\mathbf{z}}(\gamma) = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{cases}$$

$R_{\mathbf{z}}(\gamma)$, $R_{\mathbf{y}}(\beta)$, et $R_{\mathbf{x}}(\alpha)$ représentent respectivement les matrices de rotations autour des trois axes d'un repère cartésien orthonormé : dans notre cas, il s'agit des rotations autour des trois axes

du repère body permettant de projeter les données du repère capteur au repère body. Les rotations sont représentées par les 3 angles α, β et γ, que l'on appelle aussi « roulis, tangage et lacet ». Pour les translations selon chacun des trois axes, elles sont représentées par le vecteur T_{ext}. On appelle paramètres extrinsèques les 6 paramètres dont les valeurs sont obtenues par étalonnage extrinsèque du système d'acquisition, et qui caractérisent la transformation que l'on vient de décrire.

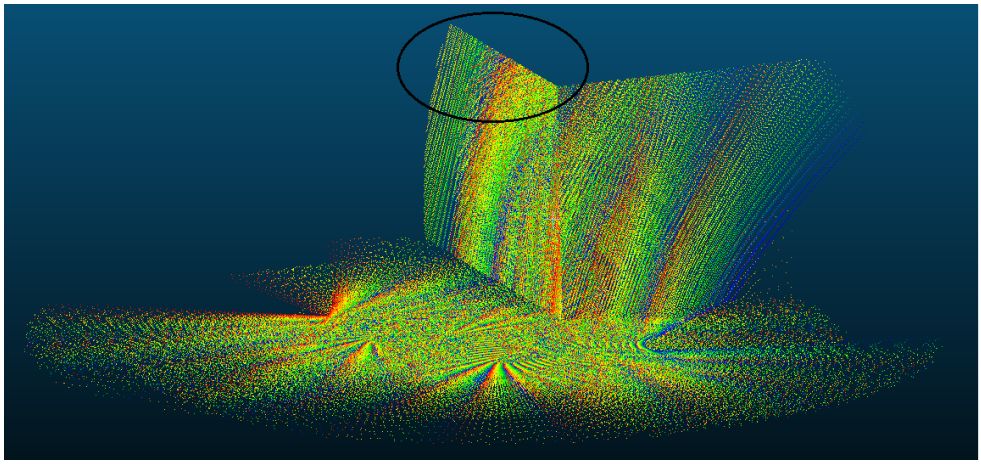
3.1.2 Effets d'un mauvais étalonnage des paramètres extrinsèques sur les données

Un mauvais étalonnage des paramètres extrinsèques implique une projection des données dans le repère body erronée. Dans un premier temps, si l'on a une erreur au niveau des paramètres de translation, le problème qui apparaît est une translation de l'ensemble des données avec un offset constant, qui est égal à la valeur de l'erreur dans le cas d'un capteur mono-fibre ; dans le cas d'un capteur multi-fibres, comme présenté en figure 3.1, on a le même résultat pour chaque fibre du capteur. La sous-figure a) présente le nuage avec un étalonnage correctement effectué, et la sous-figure b) un zoom sur un plan vertical du nuage. On voit avec la sous-figure c) les effets d'une erreur sur les paramètres de translations : le plan est déformé, mais les données de chaque fibre restent consistantes entre elles. L'erreur appliquée est de quelques dizaines de centimètres, entre 20 et 30 centimètres selon l'axe. Avec la sous-figure d), on montre les déformations introduites avec une erreur sur les paramètres de rotations extrinsèque. Chaque point du nuage est acquis dans une certaine direction du capteur LIDAR, et une erreur sur la rotation extrinsèque déforme les données de manière non consistante, au contraire d'une erreur sur la translation extrinsèque. La sous-figure d) illustre ce type d'erreur : cette fois-ci, on remarque très nettement la déformation par rapport à la sous-figure b). L'erreur que l'on illustre sur cette figure est de quelques degrés, entre 2 et 3 degrés.

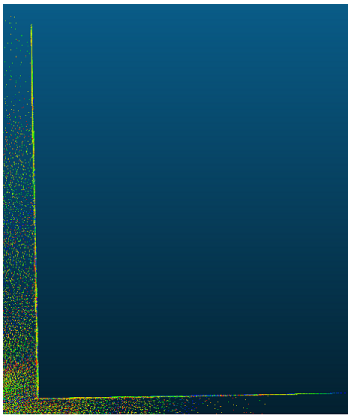
3.2 Etat de l'art

3.2.1 Calibrage automatique de différents systèmes d'acquisitions

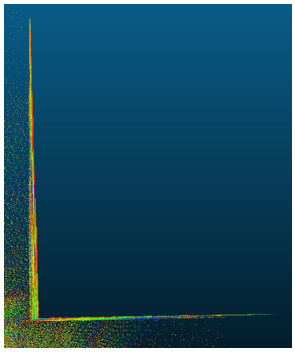
Il existe plusieurs types de capteurs LIDAR, que l'on peut séparer en deux catégories principales comme présenté en section 2.2.3. Par conséquent, on trouve aussi de nombreuses méthodes de calibrage des systèmes d'acquisitions équipés de capteur LIDARs, car la plupart des systèmes embarquent un ou plusieurs capteurs, et nécessitent un calibrage spécifique. En effet, si on s'intéresse aux systèmes comportant au moins un capteur laser, on trouve quelques configurations particulières : par exemple, Huang [Huang et Barth, 2009] présente un système utilisé pour effectuer des acquisitions, qui est composé d'une caméra et de deux capteurs lasers 3D. Pour l'étalonnage, une mire est utilisée, et les paramètres des deux types de capteurs sont estimés en même temps. Dans le même style, Mazzei [Mazzei *et al.*, 2012] propose une méthode d'étalonnage des paramètres de calibrage pour un système hybride caméras/LIDAR, composé de sept caméras et quatre capteurs lasers 2D et 3D. Leur approche utilise des marqueurs au sol dont la position est connue précisément, pour effectuer l'étalonnage des paramètres de calibrage extrinsèque du système complet. L'idée particulière est que pour les lasers, il y en a un « principal », qui est positionné à l'avant du véhicule, et il y a les autres lasers : l'étalonnage des lasers restants se fait en s'aidant des données issues du capteur laser principal, qui est calibré à l'aide d'une vérité terrain. Enfin, on peut aussi noter l'article de Maddern [Maddern *et al.*, 2012], qui présente un système composé d'un capteur laser 3D et de deux capteurs laser 2D. Selon les auteurs, leur méthode fonctionne pour des configurations composées d'un capteur laser 3D et de n capteurs laser 2D. L'étalonnage du système se fait en deux étapes : d'abord, celle du capteur 3D, qui est automatique, et qui ne nécessite une intervention humaine que pour l'initialisation des paramètres ; puis celle du/des capteur(s) 2D, qui se base sur la



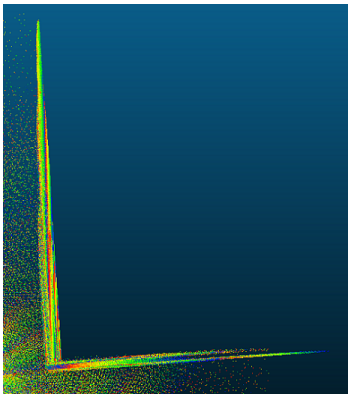
(a)



(b)



(c)



(d)

FIGURE 3.1 – Nuages de points avec différents étalonnages extrinsèques : a) Paramètres corrects ; b) Paramètres corrects, avec zoom sur la zone d'intérêt ; c) Erreurs sur les paramètres de translations ; d) Erreurs sur les paramètres de rotations

structure du nuage créé par le capteur laser 3D. Le nuage produit par les capteurs 2D est comparé avec celui produit par le capteur 3D déjà calibré.

3.2.2 L'optimisation des paramètres de calibrage extrinsèque d'un capteur LIDAR

Dans la section 3.2.1, nous avons présenté quelques méthodes d'étalonnage automatique des paramètres de calibrage de systèmes de cartographie équipés de capteurs LIDAR mono-fibres principalement : ce type de capteur est très utilisé pour plusieurs raisons, comme par exemple leur précision correcte et leur coût faible pour une grande partie des modèles. De nombreuses méthodes d'optimisation des paramètres de calibrage existent, mais ces capteurs bas coûts ne sont pas très bien adaptés à l'acquisition d'informations 3D, sauf pour les modèles RIEGL, mais dont la gamme de prix se rapproche de celle des capteurs multi-fibres. Par exemple, dans [Gao et Spletzer, 2010], les auteurs présentent un système composé de plusieurs LIDAR 2D, et proposent une optimisation des paramètres extrinsèques pour chaque capteur en se servant de la réflexion des impacts lasers sur des cibles réfléchissantes posées sur des poteaux verticaux. Dans [Lin *et al.*, 2013], les auteurs

présentent un système composé d'un capteur LIDAR mono-fibre monté sur une plateforme pivotante, ce qui permet d'avoir des données 3D : avec la rotation induite par la plateforme, le système nécessite un calibrage spécifique, qui est présenté dans l'article. Dans [Maddern *et al.*, 2012], les auteurs présentent un système composé d'un capteur LIDAR 3D et de 2 capteurs mono-fibres : la particularité est que le capteur 3D est composé de 3 capteurs mono-fibres placés sur une plateforme tournante.

D'un autre côté, on trouve aussi des capteurs permettant d'acquérir des données 3D, mais multi-fibres, c'est-à-dire qu'ils englobent plusieurs « lasers », comme par exemple les capteurs Velodyne [Velodyne LIDAR sensors, 2017] ou Quanergy [Quanergy product page, 2017], sorti plus récemment. Peu de méthodes d'optimisation des paramètres extrinsèques pour des systèmes équipés de ce genre de capteur existent dans la littérature. Dans [Zhu et Liu, 2013], les auteurs proposent une méthode d'optimisation des 3 paramètres de rotation en 2 étapes : tout d'abord, l'optimisation des angles de roulis et de tangage en estimant des paramètres planaires pour les sols, puis l'angle de lacet en mettant en correspondance des objets de forme cylindrique. La méthode est non supervisée et ne nécessite pas de mire de calibrage comme la plupart des routines de calibrage de systèmes d'acquisitions, mais il y a une limitation du fait que seul les paramètres de rotation sont estimés. Dans [Huang *et al.*, 2013], les auteurs proposent une optimisation de l'ensemble des paramètres extrinsèques d'un capteur LIDAR multi-fibres, mais pour cela, ils utilisent une mire de calibrage et une caméra infrarouge qui leur permet de récupérer une image des impacts lasers sur la mire, et l'optimisation vise à avoir une correspondance correcte entre les nuages de points et les images infrarouges. Enfin, dans [Elseberg *et al.*, 2013], les auteurs cherchent à optimiser les paramètres de calibrage induits par leur système composé de plusieurs capteurs LIDARs. Ils mesurent la compacité de leurs nuages de points avec une fonction qui est une somme de fonctions de densités de points qu'ils cherchent à minimiser. La méthode est non supervisée, n'utilise pas de mire de calibrage et est appliquée en post-traitement.

Dans la littérature, deux articles ont retenus notre attention : [Levinson et Thrun, 2010], dans lequel les auteurs présentent une méthode non supervisée d'optimisation des paramètres extrinsèques d'un système équipé d'un capteur multi-fibres, sans utilisation de mire de calibrage et en post-traitement, ainsi que [Mengwen *et al.*, 2014], où une optimisation des paramètres extrinsèques entre un capteur LIDAR et le repère body est présenté, aussi en post-traitement. Nous expliquerons plus en détails en section 3.3 pourquoi ces deux articles nous intéressent.

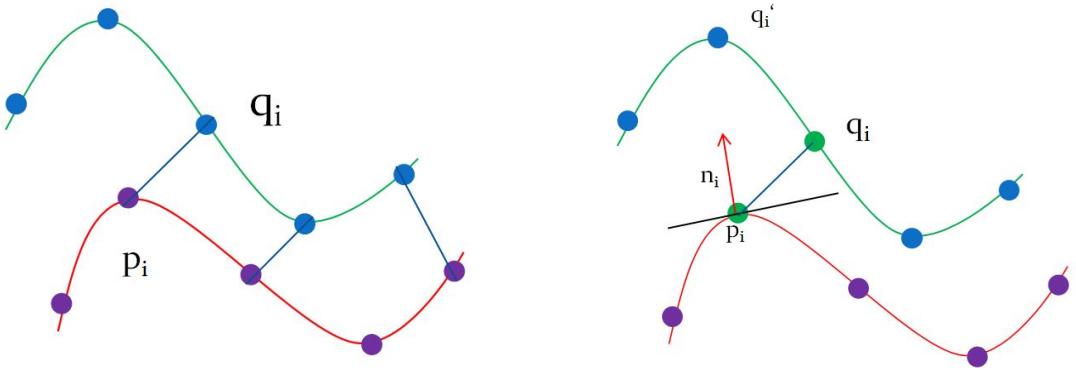
3.2.3 Le recalage de données

Le recalage de données occupe une place importante dans de nombreux domaines : la reconstruction 3D ; la cartographie - mobile ou statique - ; les méthodes de type SLAM orienté trajectoire. Dans le cas de la cartographie (et aussi dans celui du SLAM), le scan matching est une étape importante, puisqu'elle permet d'affiner le géo-référencement des scans, et par conséquent la localisation du véhicule lors de l'acquisition. Selon la fréquence des points de contrôle – c'est-à-dire des instants où l'on cherche à déterminer le vecteur d'état du véhicule -, la localisation sera plus ou moins précise, au détriment de l'application temps réel. Dans notre cas, le recalage de données est important car comme nous allons le présenter en section 3.3, l'optimisation que nous avons choisi d'appliquer s'appuie sur du recalage de données.

L'algorithme de scan matching le plus connu et le plus utilisé est l'Iterative Closest Point (ICP). Cet algorithme est apparu conjointement en 1991 et 1992, respectivement présenté par Chen et Medioni [Chen et Medioni, 1991], et par Besl et Mckay [Besl et McKay, 1992]. En effet, des algorithmes de mise en correspondance de scans (scan registration) existaient déjà, mais Chen et Medioni sont les premiers à utiliser une distance point à un plan pour recaler les deux scans, et Besl et Mckay ont démontré la convergence de l'algorithme lorsqu'une configuration initiale proche de la solution était connue.

L'algorithme d'ICP permet de mettre en correspondance des scans pris à des instants consécutifs différents, ou acquis à des positions différentes. Pour cela, les scans doivent être référencés par rapport à une origine commune et dans un repère commun : en effet, pour effectuer le recalage entre les deux scans, l'algorithme cherche à mettre en correspondance des points appariés issus de chacun des deux scans, d'où la nécessité d'avoir une référence commune, et un des deux scans est pris comme référence, le deuxième scan étant recalé par rapport à cette référence. Une fonctionnelle avec laquelle la distance moyenne entre les deux scans est minimisée itérativement ; la fonctionnelle a en général cette forme :

$$J(R,T) = \frac{1}{N} * \sum_{i=1}^{N} [d_{R,T}(p_i, q_i)]^2 \tag{3.2}$$



(a) Illustration de la distance point à point

(b) Illustration de la distance point à plan

FIGURE 3.2 – Schéma des deux types de distances principalement utilisées pour l'ICP

Dans l'équation (3.2), N représente le nombre de points appariés entre les deux scans et i itère sur l'ensemble des points p_i que l'on a sélectionné du scan de référence ; les points q_i sont les points mis en correspondances avec les points p_i. Enfin, la distance qui est utilisée suit principalement deux métriques différentes : une distance point à point (3.3), introduite par Besl et Mckay [Besl et McKay, 1992], et une distance point à plan (3.4), introduite par Chen et Medioni [Chen et Medioni, 1991].

$$d_{R,T}(p_i, q_i) = \|(R * p_i + T) - q_i\| \tag{3.3}$$

avec R la rotation et T la translation estimées entre les deux scans à mettre en correspondance

$$d_{R,T}(p_i, q_i) = \|n_i \cdot [(R * p_i + T) - q_i]\| \tag{3.4}$$

avec R la rotation et T la translation estimées entre les deux scans à mettre en correspondance ; n_i représente la normale au plan local auquel appartient le point p_i.

Sur la figure 3.3, on peut voir un exemple de recalage entre deux scans du lapin de Stanford. Dans ce cas là, on a un recouvrement total des deux scans, ce qui rend la mise en correspondance plus simple, puisque chaque point d'un scan est apparié à un point de l'autre scan. On peut tout de même noter que, bien que l'ICP soit très utilisé pour aligner des scans entre eux, d'autres approches existent. On trouve par exemple l'algorithme de **Normal Distribution Transform** (NDT), introduit par Peter Biber [Biber et Straber, 2003] : l'idée de base ressemble à celle de la carte d'occupation, puisque l'on a une carte 2D divisée en plusieurs cellules de taille réglable. Cependant, la probabilité de mesurer un échantillon dans une cellule est modélisée par une distribution normale. Le scan matching est effectué plus ou moins de la même manière que l'ICP : les deux scans sont

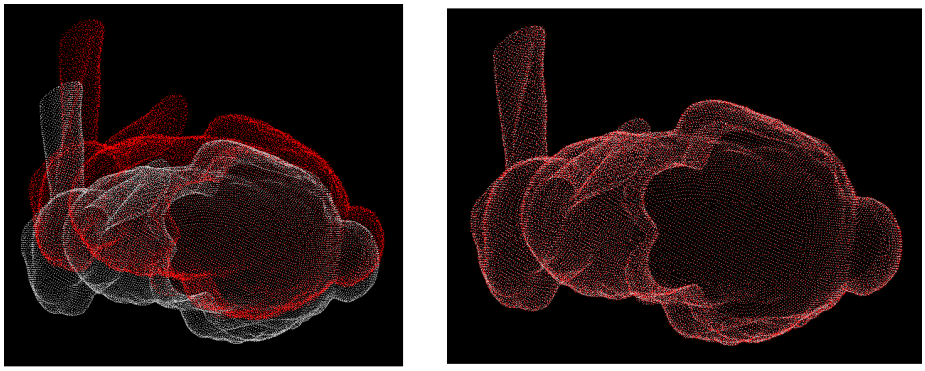


FIGURE 3.3 – Exemple de recalage entre deux scans

référencés dans un même référentiel, les NDT sont calculées pour les deux scans, puis comparées entre elles, et le processus est itéré jusqu'à convergence. Toutefois, au niveau de la comparaison, la fonctionnelle à minimiser n'est pas la même : là où pour l'ICP, la fonctionnelle à minimiser est une distance point à point ou point à plan (équations (3.3) et (3.4)), pour le NDT, on cherche à minimiser une fonctionnelle f(X), qui vaut :

$$f(R,T) = -\sum_{i=1}^{n} h(T[(R,T), x_i]) \tag{3.5}$$

où R et T sont les paramètres de calibrage à optimiser, x_i les points à recaler et $T[(R,T)x_i]$ les points « corrigés », après application de la transformation (rotation et translation) trouvée. Cette fonctionnelle est basée sur des densités de probabilités de mesures d'échantillons, comme indiqué ci-dessus. Cette approche est au départ introduite pour effectuer du 2D-SLAM, mais Magnusson [Magnusson *et al.*, 2007] proposent une extension en 3 dimensions. Magnusson propose ensuite deux articles qui traitent tous les deux de l'algorithme de NDT [Magnusson *et al.*, 2009b], [Magnusson *et al.*, 2009a]. Dans le premier, l'efficacité de l'ICP et la NDT sont comparés : il se trouve que la NDT permet de converger plus vite et a un meilleur taux de réussite que l'ICP, mais n'est pas assez précis. En effet, les problèmes de convergence ne peuvent pas réellement être prédit (il y a convergence malgré un grand écart d'alignement, mais l'algorithme échoue pour un plus petit écart) et lorsque la méthode NDT échoue à converger, l'erreur résiduelle est plus élevée qu'avec l'ICP. Dans le deuxième article, une fermeture de boucle effectuée avec un algorithme de NDT est présentée, afin de voir si les résultats sont satisfaisants, comme avec des systèmes visuels par exemple. Pour les auteurs, la réponse est positive, ce qui est assez encourageant pour de futures améliorations, comme celles qu'a eu l'algorithme d'ICP.

3.2.3.1 L'algorithme d'ICP

Comme le résument Rusinkiewicz et Levoy [Rusinkiewicz et Levoy, 2001], l'algorithme d'ICP est composé de quatre étapes (il y en a six dans leur article, mais quatre sont plus importantes que les autres, les deux autres sont généralement intégrées à l'une de ces quatre étapes) :

- Tout d'abord, il faut sélectionner un scan qui servira de référence (généralement, celui acquis en premier sur les deux considéré) et sélectionner des points pour la mise en correspondance. Effectivement, entre les deux scans, il n'y a qu'un recouvrement partiel la plupart du temps. Ainsi, au niveau de la sélection des points, on trouve comme approche l'utilisation de tous les points disponibles, qui est proposée par Besl et Mckay [Besl et McKay, 1992], ou des échantillonnages uniformes ou aléatoires. Il est aussi possible dès cette première étape de rejeter certains points, notamment ceux présents sur les bords du scan.

- Ensuite, on va chercher à mettre en correspondance les points sélectionnés avec d'autres points du deuxième scan considérés. On trouve la méthode de base proposée par Besl et Mckay, qui est de chercher le point le plus proche sur l'autre scan : en effet, comme indiqué plus haut, on suppose que les deux scans sont très proches. Une alternative est la métrique normal-shooting, proposée par Chen et Medioni [Chen et Medioni, 1991], et qui consiste à « suivre » la normale du point que l'on cherche à apparier pour avoir l'intersection avec le deuxième scan. Cette métrique fonctionne moins bien sur les structures un peu complexes.
- L'étape suivante voit le rejet de certaines paires créés. En effet, cette étape permet d'améliorer la convergence de l'algorithme : avec trop de paires sélectionnées, la convergence sera lente et l'algorithme sera plus sensible au bruit. Ainsi, une première idée est de rejeter les paires présentes sur les « bords » des scans : c'est par exemple ce que fait Turk [Turk et Levoy, 1994], afin de réduire le nombre de faux appariements. D'autres approches consistent à rejeter par exemple les paires dont la distance entre les deux points est trop élevée, mais cette condition est généralement intégrée au processus d'appariement de points entre deux scans, ou encore le rejet d'un certain pourcentage de paires. Toutefois, lorsque le déplacement entre deux scans est élevé, mis à part le rejet des paires sur les bords, il n'y a pas d'améliorations visibles.
- Enfin, la dernière étape consiste en le choix d'une métrique de correspondance entre scans à minimiser : l'idée est que plus les scans seront correctement mis en correspondance, plus cette distance sera faible. Au niveau des métriques pour les distances, on en trouve deux utilisées principalement : la distance point à point, présentée en équation (3.3), et la distance point à plan, présentée en équation (3.4).

Pour la distance point à point, des solutions analytiques existent pour la minimisation de la fonctionnelle présentée en équation 3.2, les deux plus utilisées étant la décomposition en SVD, introduite par Arun et utilisée par exemple dans [Nüchter *et al.*, 2007], et résolution avec une représentation des rotations par des quaternions, introduite par Horn [Horn, 1987] et par exemple utilisée dans [Nüchter *et al.*, 2004]. La méthode par SVD a l'avantage d'être robuste et facile à implémenter, tandis que les quaternions sont plus adaptés pour la représentation des rotations dans l'espace. Pour la distance point à plan, il n'est pas possible de la résoudre linéairement. Deux possibilités sont alors à envisager : utiliser des méthodes de descente de gradient, type Gauss-Newton ou Levenberg-Marquardt, ou sinon linéariser la fonctionnelle que l'on cherche à minimiser, notamment au niveau des matrices de rotations. Andrea Censi [Censi, 2008], dans son article, reprend cette métrique en la linéarisant, ce qui a pour effet d'accélérer l'appariement de points de manière non négligeable. Enfin, Zhu [Zhu *et al.*, 2009] proposent une amélioration de la distance point à plan, qui est selon les auteurs plus robuste pour la mise en correspondance de points entre scans. En effet, là où la métrique de Censi n'est pas robuste au large déplacement entre scans, celle-ci, tout en conservant la rapidité de la métrique point à plan, donne de meilleurs résultats. Le plus souvent, c'est la distance point à plan qui est utilisée car elle donne une meilleure convergence que la distance point à point, comme le montrent Rusinkiewicz et Levoy [Rusinkiewicz et Levoy, 2001] dans leur comparaison de différentes versions de l'algorithme d'ICP.

L'étape qui n'est pas détaillée ici est celle de l'application d'un poids aux paires créées lors de l'appariement. En effet, d'après Rusinkiewicz et Levoy, cette étape n'influe ni sur le temps de calcul, ni sur la convergence de l'algorithme. De plus, elle peut être intégrée à l'étape de mise en correspondance des points, et n'est généralement pas considérée comme une étape de l'algorithme de l'ICP à part.

3.2.3.2 Accélération de l'algorithme

Enfin, il convient de noter que d'autres améliorations ont été apportées à l'ICP, notamment au niveau du temps de calcul. Effectivement, dans l'algorithme, c'est la mise en correspondance des points qui prend le plus de temps, car, naïvement, on cherche à sélectionner la meilleure correspondance pour un point en testant toutes les correspondances possibles. Plusieurs étapes de l'ICP

qui n'influent pas (ou peu) sur la convergence ont ainsi pu être améliorées, permettant de réduire grandement les temps de calcul. On trouve entre autres :

- Le nombre de points des scans en entrée. Lorsque les scans possèdent un trop grand nombre de points, des échantillonnages sont effectués : comme expliqué dans [Nüchter *et al.*, 2004], une méthode dite de « force brute », qui prendrait en compte tous les points des deux scans et tous les appariements possibles, aurait une complexité en $O(n^2)$. C'est pourquoi dans [Nüchter *et al.*, 2004] et [Nüchter *et al.*, 2007], il utilise des filtres pour réduire le bruit au niveau de ces données et les échantillonner : comme il l'explique, pour un scan laser, lorsque la densité est élevée dans une certaine zone, les points sont remplacés par leur barycentre ; des filtres médians sont aussi appliqués pour échantillonner uniformément les scans.
- D'autres approches concernent le format de données utilisées. En effet, l'utilisation d'un kd-tree pour structurer les données et accéder aux plus proches voisins (pour l'appariement, qui est l'étape la plus lente de l'algorithme d'ICP) rapidement est devenue indispensable pour envisager des applications temps réel sur des jeux de données volumineux. La recherche des plus proches voisins se fait par approximation, à l'aide de l'algorithme de recherche **Approximate Nearest Neighbour** (ANN). En effet, l'arbre est parcouru. Ainsi, on trouve Nüchter [Nüchter *et al.*, 2007] qui utilise dans son article le **Cached kd-tree** : la nouveauté par rapport au kd-tree est qu'entre chaque itération, la recherche du plus proche voisin n'est pas entièrement faite. Des pointeurs vers les feuilles visitées sont gardés en mémoire, et constituent le point de départ de la recherche, ce qui a pour effet d'accélérer la recherche des plus proches voisins. Choi [Choi *et al.*, 2012] propose lui aussi une modification des kd-trees, en utilisant les **Approximate cached kd-trees**, qui ont les avantages des **Approximate kd-trees** et des **Cached kd-trees**, et qui selon lui seraient 24 fois plus rapide que les kd-trees standards.

3.2.3.3 Amélioration de l'ICP

A côté des modifications qui ne concernent qu'une étape de l'ICP, on trouve des modifications un peu plus importantes. Ainsi, Fabrice Monnier [Monnier *et al.*, 2013] propose une modification de l'algorithme d'ICP afin de chercher une transformation non rigide entre deux nuages de points : l'objectif donné par l'auteur est de mettre à jour des bases de données existantes, notamment en termes de précision, niveau de détails et diversité d'objets représentés. La non-rigidité de la transformation vient du fait qu'au cours du temps, l'erreur de positionnement de la centrale inertielle évolue de manière non linéaire : même lorsque le véhicule est à l'arrêt, l'erreur augmente tout de même.

D'autre part, on trouve des approches de type « n-scan matching », où le but n'est plus seulement d'aligner deux scans consécutifs entre eux, et remonter de proche en proche sur l'ensemble des scans, mais de contraindre l'ensemble des scans à être alignés en une seule passe. C'est ce qu'ont introduits Lu et Millios [Lu et Milios, 1997], dans le but de réduire l'erreur d'alignement entre scans qui s'accumule au cours de l'acquisition. Leur approche est effectuée en post-traitement. Dans le même style, on trouve un article de Nüchter plus récent, [Nüchter *et al.*, 2010], où une linéarisation de rotations est effectuée dans une approche de n-scans matching, et ceci dans le but d'accélérer la procédure, mais aussi de pouvoir appliquer une solution analytique pour résoudre l'équation du système considéré.

Une dernière modification assez importante, qui permet d'améliorer l'alignement des scans entre eux est la fermeture de boucle, ou plus généralement l'ajustement de faisceaux. En effet, la fermeture de boucle consiste à revenir à un emplacement déjà visité, à scanner l'environnement et à détecter ce bouclage sur les acquisitions pour réduire l'erreur d'alignement accumulée lors des précédents alignements. En effet, comme l'explique Gérossier dans sa thèse [Gérossier, 2012], en repérant une fermeture de boucle entre deux scans, l'ensemble des alignements effectués avant le bouclage peuvent être affinés.

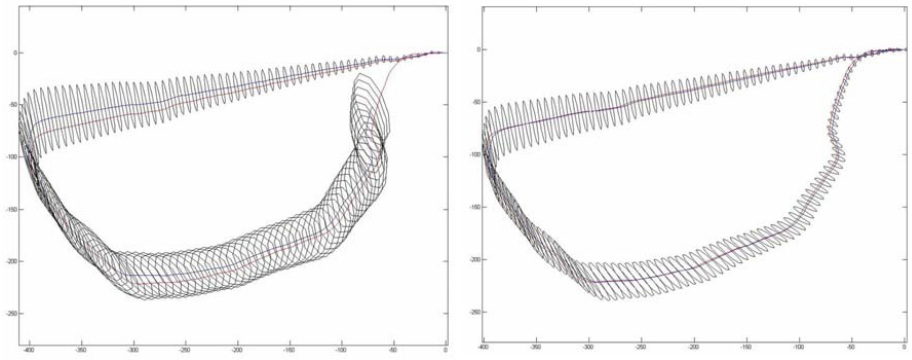


FIGURE 3.4 – Comparaison des incertitudes sur les mesures [Gérossier, 2012]

Sur la figure 3.4, les ellipses représentent les incertitudes sur les mesures. A gauche, on a les incertitudes avant la fermeture de boucle, et à droite après : on voit bien que la fermeture de boucle permet d'améliorer la précision des mesures. L'ajustement de faisceaux, qui vient du domaine de la photogrammétrie et qui a pour but d'affiner les coordonnées des points 3D obtenus par reconstruction 3D à partir des images en repérant les recouvrements de données entre images, est tout aussi applicable avec des données issues de capteur laser 2D ou 3D. On trouve ainsi Lu et Milios [Lu et Milios, 1997], qui, dans leur approche de recalage global, effectuent de l'ajustement de faisceaux.

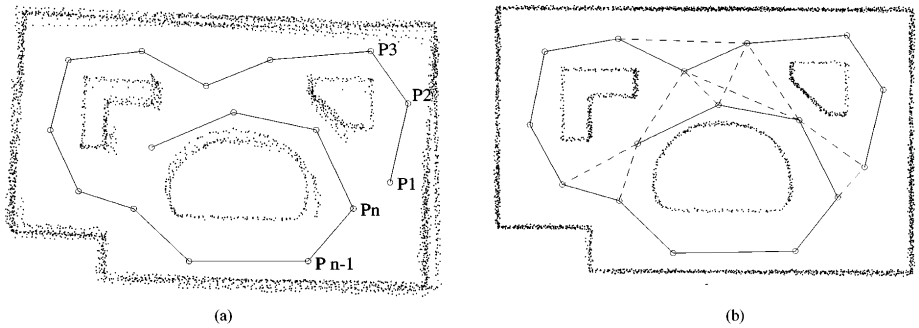


FIGURE 3.5 – Recalages successifs de scans et recalage global [Lu et Milios, 1997]

Sur la figure 3.5, la figure de gauche donne la carte obtenue par recalage successifs de scans entre deux positions consécutives, et la figure de droite présente la même carte, mais où le recalage des scans a été effectué de manière globale. On voit très bien que l'ajustement de faisceaux, au même titre que la fermeture de boucle qu'il généralise, permet d'affiner les mesures et améliore considérablement les cartes. C'est aussi ce qu'utilise Nüchter [Nüchter *et al.*, 2007], pour affiner l'alignement entre les scans, en comparant le dernier scan acquis avec les précédents, mais sous certaines conditions pour ne pas avoir à tester l'ensemble des scans disponibles, ou plus récemment [Nüchter *et al.*, 2010] avec son approche de n-scan matching. Dans cette approche de n-scan matching, l'ensemble des scans qui ont un pourcentage de recouvrement suffisant sont alignés, ce qui permet de détecter automatiquement les recouvrements lorsque deux scans se superposent en partie. Newman [Newman *et al.*, 2006] propose dans son article une méthode qui permet de détecter les recouvrements à partir d'images, approche plus intuitive qu'avec des nuages de points car les points d'intérêts sont extraits plus facilement des images que des nuages. De plus, il détecte et élimine les fausses alarmes en prenant en compte divers paramètres, comme l'écart temporel entre les deux images par exemple.

3.3 Méthode d'optimisation proposée

Dans la section 3.2.2, nous avons introduit 2 méthodes qui sont proches de nos travaux, notamment par rapport à l'orientation que nous avons pris pour l'optimisation des paramètres extrinsèques. La première, présentée dans [Levinson et Thrun, 2010], permet d'optimiser les 6 paramètres extrinsèques d'un système LIDAR multi-fibres séquentiellement et itérativement : l'optimisation se fait en post-traitement, sans mire de calibrage et en se basant sur l'observation qu'avec des paramètres corrects, les données issues de chaque fibre du capteur se superposent comme l'illustre la figure 3.6, et en utilisant une hypothèse de monde plan localement - qui est vraie au vu de la densité de points fournis par le capteur -, les auteurs cherchent à minimiser une fonctionnelle qui pénalise les points éloignés des plans locaux. La résolution est de la forme « grid search », et l'optimisation est séparée pour les paramètres de rotations et de translations : à chaque itération, une énergie est calculée en modifiant chaque paramètre dans un voisinage de l'estimation initiale et selon un pas donné, qui diminuent à chaque itération pour converger vers les paramètres « optimaux ».
Le deuxième article proche de nos travaux est [Mengwen *et al.*, 2014], où un système composé de plusieurs LIDARs est calibré en deux temps : en premier, un LIDAR de référence est étalonné en effectuant plusieurs acquisitions et en recalant les différentes données entre elles. Pour cela, des caractéristiques planaires sont extraites manuellement des nuages et mis en correspondance en jouant sur les paramètres extrinsèques. Ensuite, les autres LIDARs sont étalonnés en recalant chaque nuage de points par rapport au nuage issu du LIDAR de référence, en extrayant aussi manuellement des caractéristiques planaires : par conséquent, chaque LIDAR à son propre calibrage extrinsèque par rapport au véhicule mobile, et pour les résultats, une initialisation proche de la vérité terrain est choisie.

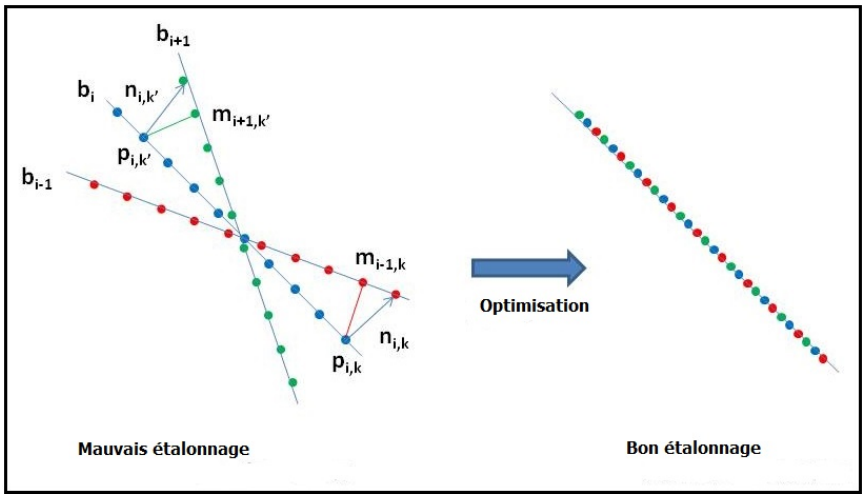


FIGURE 3.6 – Vue de côté d'une surface planaire acquise avec un LIDAR multi-fibres

L'optimisation que nous avons choisi d'effectuer et qui est différente des 2 algorithmes que l'on vient de présenter pour répondre à nos besoins. En effet, nous cherchons aussi à optimiser les 6 paramètres de calibrage extrinsèque d'un capteur LIDAR multi-couches, mais au contraire des 2 méthodes précédentes, notre méthode est automatique, même lorsque nous effectuons une extraction de caractéristiques planaires comme cela est expliqué en section 3.3.5. Notre méthode d'optimisation est rapide, et robuste à une mauvaise initialisation des paramètres de calibrage. Pour effectuer nos expérimentations, nous avons utilisé des jeux de données obtenus à l'aide d'un véhicule mobile équipé d'un capteur LIDAR Velodyne 32 fibres, présenté en figure 2.6. Le capteur LIDAR permet d'obtenir jusqu'à 700 000 points à la seconde, et couvre un champ vertical de 40°(de -8°à +32°) et un champ horizontal de 360°, puisque la base du capteur tourne à une fréquence pouvant aller jusqu'à 10 Hz. Dans la suite de ce chapitre, nous supposons que l'étalonnage intrinsèque du capteur

est effectué et donne des valeurs correctes et précises, et que l'ensemble des capteurs embarqués sur le véhicule permettent d'avoir le positionnement précis du véhicule à chaque instant de contrôle voulu.

3.3.1 Présentation de l'algorithme mis au point

Notre point de départ est le même que celui introduit dans [Levinson et Thrun, 2010] : nous avons un nuage de point obtenu après une acquisition avec notre véhicule mobile, et les valeurs des paramètres de calibrage extrinsèque qui sont utilisés pendant l'acquisition sont erronées, ce qui donne un nuage de point bruité et difficilement exploitable. Nous avons choisi de définir une fonctionnelle à minimiser, similaire à du recalage de données avec une distance point à plan : en effet, comme le montre la figure 3.6, avec un bon calibrage, les données issues de fibres voisines d'un capteur multi-fibres devraient être superposées, et un recalage de données issues de fibres voisines permet d'obtenir ce résultat en optimisant les paramètres extrinsèques. L'optimisation se fait en post-traitement, de façon non supervisée et sans mire de calibrage, et vise à corriger les paramètres de calibrage pour un nuage de points. L'énergie que l'on veut minimiser est définie par l'équation (3.6) :

$$J(R,T) = \frac{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_{k} w_{i,j,k} * d_{i,j,k}^{2}(R,T)}{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_{k} w_{i,j,k}} \tag{3.6}$$

avec :

$$\begin{cases} d_{i,j,k}(R,T) = n_{i,k}^{T} \times (p_{i,k}(R,T) - m_{j,k}(R,T)) \\ R(\alpha,\beta,\gamma) \text{ et } T(t_x,t_y,t_z) \\ p_{i,k}(R,T) = R_{nav}(p'_{i,k}) \times (R \times p'_{i,k} + T) + T_{nav}(p'_{i,k}) \\ m_{j,k}(R,T) = R_{nav}(m'_{j,k}) \times (R \times m'_{j,k} + T) + T_{nav}(m'_{j,k}) \end{cases}$$

Dans l'énergie définie en (3.6), les différents termes utilisés sont :

- B est un sous-ensemble des fibres du capteur multi-fibres, avec $B \subset [\![0;31]\!]$
- 2*N représente le nombre de fibres voisines à la fibre i que l'on prend en compte pour le recalage
- k itère sur un sous-ensemble des points acquis par la fibre i
- $w_{i,j,k}$ est un poids dont la valeur vaut 1 en fonction d'un seuil de distance entre les points $p_{i,k}$ et $m_{j,k}$.
- $n_{i,k}$ est la normale au point $p_{i,k}$ du plan tangent au même point $p_{i,k}$.
- $p_{i,k}$ et $m_{j,k}$ sont respectivement le k^{me} point de la fibre i, projeté dans le repère lié à la Terre et son plus proche voisin sur la fibre j, aussi projeté dans le même repère.
- $p'_{i,k}$ et $m'_{j,k}$ sont respectivement le k^{me} point de la fibre i, projeté dans le repère cartésien du capteur et son plus proche voisin sur la fibre j, aussi projeté dans le même référentiel.
- R_{nav} et T_{nav} sont respectivement la matrice de rotation et le vecteur de translation décrivant la transformation entre le repère body du véhicule et le repère monde. Ces transformations sont calculées pour chaque point acquis.
- L'énergie que l'on a définie a un sens physique : nous avons une somme de distances au carré, pondérée par la somme des poids pris en compte, qui dans le cas de l'optimisation des paramètres de calibrage extrinsèque est égal au nombre de points N_t pris en compte dans le calcul de l'énergie. Nous supposons que le bruit du nuage provient de plusieurs sources indépendantes, et qu'il est centré, réduit et que sa distribution suit une loi normale : avec ces hypothèses, l'énergie J suit une loi du χ^2, et lorsque N_t est suffisamment grand - ce qui est le cas avec le capteur utilisé -, l'énergie J donne une estimation de la variance σ^2 du bruit total du nuage.

3.3.2 Optimisation de l'énergie

3.3.2.1 Approximation linéaire

Pour notre optimisation, nous partons de paramètres extrinsèques qui ne sont pas corrects, ou plutôt qui donnent un nuage de « mauvaise » qualité, et que nous voulons optimiser en minimisant l'énergie définie par (3.6). Nous ne sommes pas capable de trouver la solution optimale car l'énergie est non linéaire à cause des paramètres de rotation. Nous avons changé le problème en une minimisation itérative où nous cherchons des biais à ajouter à ces paramètres extrinsèques : nous partons de paramètres connus $(t_x, t_y, t_z, \alpha, \beta, \gamma)$, et nous cherchons les variations $(\delta t_x, \delta t_y, \delta t_z, \delta\alpha, \delta\beta, \delta\gamma)$, qui donnent le résultat suivant :
$\mathrm{J}(\mathrm{R}(\alpha+\delta\alpha, \beta+\delta\beta, \gamma+\delta\gamma),\ \mathrm{T}(t_x+\delta t_x, t_y+\delta t_y, t_z+\delta t_z)) < \mathrm{J}(\mathrm{R}(\alpha,\beta,\gamma),\ \mathrm{T}(t_x,t_y,t_z))$

La matrice $\mathrm{R}(\alpha+\delta\alpha, \beta+\delta\beta, \gamma+\delta\gamma)$ est remplacée par une approximation linéaire $\mathrm{R}(\alpha,\beta,\gamma) + R_\alpha * \delta\alpha + R_\beta * \delta\beta + R_\gamma * \delta\gamma$, en supposant que les variations que l'on cherche sont faibles ; des intermédiaires de calculs sont présentés en Annexe B. Avec cette approximation dans l'équation (3.6), on peut réécrire l'énergie J sous la forme :

$$J(\delta X) = \frac{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_k w_{i,j,k} * (D_{i,j,k} + C_{i,j,k}^T \times \delta X + o(\delta X))^2}{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_k w_{i,j,k}} \tag{3.7}$$

avec :

$$\begin{cases} \delta X = (\delta t_x \quad \delta t_y \quad \delta t_z \quad \delta\alpha \quad \delta\beta \quad \delta\gamma)^T \\ D_{i,j,k} = n_{i,k}^T \times \begin{pmatrix} T_{nav}(p'_{i,k}) - T_{nav}(m'_{j,k}) \\ + \left[R_{nav}(p'_{i,k}) \times (R(\alpha,\beta,\gamma) \times p'_{i,k} + T(t_x,t_y,t_z))\right] \\ - \left[R_{nav}(m'_{j,k}) \times (R(\alpha,\beta,\gamma) \times m'_{j,k} + T(t_x,t_y,t_z))\right] \end{pmatrix} \\ C_{i,j,k} = \begin{pmatrix} n_{i,k}^T \times \left[R_{nav}(p'_{i,k}) - R_{nav}(m'_{j,k})\right] \times \begin{bmatrix}1 & 0 & 0\end{bmatrix}^T \\ n_{i,k}^T \times \left[R_{nav}(p'_{i,k}) - R_{nav}(m'_{j,k})\right] \times \begin{bmatrix}0 & 1 & 0\end{bmatrix}^T \\ n_{i,k}^T \times \left[R_{nav}(p'_{i,k}) - R_{nav}(m'_{j,k})\right] \times \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T \\ n_{i,k}^T \times \left[R_{nav}(p'_{k,i}) \times R_\alpha \times p'_{i,k} - R_{nav}(m'_{j,k}) \times R_\alpha \times m'_{j,k}\right] \\ n_{i,k}^T \times \left[R_{nav}(p'_{i,k}) \times R_\beta \times p'_{i,k} - R_{nav}(m'_{j,k}) \times R_\beta \times m'_{j,k}\right] \\ n_{i,k}^T \times \left[R_{nav}(p'_{i,k}) \times R_\gamma \times p'_{i,k} - R_{nav}(m'_{j,k}) \times R_\gamma \times m'_{j,k}\right] \end{pmatrix} \end{cases}$$

La solution qui minimise l'énergie (3.7) est la solution du système linéaire suivant :

$$C \times \delta X = -V \tag{3.8}$$

avec :

$$\begin{cases} C = \sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_k w_{i,j,k} * (C_{i,j,k} \times C_{i,j,k}^T) \\ V = \sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_k w_{i,j,k} * (D_{i,j,k} \times C_{i,j,k}) \end{cases}$$

3.3.2.2 Approche d'optimisation

Pour trouver les paramètres extrinsèques optimaux, et comme nous avons linéarisé certains paramètres, nous calculons δX de façon itérative, jusqu'à convergence des paramètres de calibrage.

A chaque nouvelle itération n+1, les paramètres extrinsèques sont définis comme étant :

$$\begin{cases} t_{x_{n+1}} = t_{x_n} + \delta t_{x_n} \\ t_{y_{n+1}} = t_{y_n} + \delta t_{y_n} \\ t_{z_{n+1}} = t_{z_n} + \delta t_{z_n} \\ \alpha_{n+1} = \alpha_n + \delta\alpha_n \\ \beta_{n+1} = \beta_n + \delta\beta_n \\ \gamma_{n+1} = \gamma_n + \delta\gamma_n \end{cases} \tag{3.9}$$

Pour démarrer l'optimisation, nous avons des paramètres initiaux $(t_{x_0}, t_{y_0}, t_{z_0}, \alpha_0, \beta_0, \gamma_0)$ plus ou moins proches des paramètres optimaux, et à chaque itération, nous résolvons la fonction objectif (3.7). Nous montrerons dans la section 3.4 que les paramètres initiaux ne doivent pas nécessairement être proches de la solution optimale, et que notre optimisation des paramètres extrinsèques est robuste à une grande erreur sur les valeurs de paramètres obtenus par étalonnage.
L'algorithme complet d'optimisation des paramètres extrinsèques est présenté dans l'algorithme 1. Le critère d'arrêt que l'on a choisi concerne les variations des paramètres δXentre deux itérations : l'optimisation s'arrête lorsque $\|\delta X\|_{max}$ est inférieur à un seuil δ, ou dans certains cas si le nombre d'itérations est trop élevé.

Algorithme 1 Optimisation linéaire itérative des paramètres extrinsèque

Data: Un nuage de point avec des paramètres extrinsèques initiaux, choisis arbitrairement, et une trajectoire du véhicule connue
Result: Un nuage pour lequel lequel les paramètres de calibrage extrinsèque sont optimisés
Lecture et sous-échantillonnage du nuage de point ;
Paramètres extrinsèques initiaux ;
repeat
  Projection des points acquis dans le repère global avec les paramètres extrinsèque actuels $(t_{x_n}, t_{y_n}, t_{z_n}, \alpha_n, \beta_n, \gamma_n)$;
  Sélection d'un ensemble de points $p_{i,k}$;
  Construction des paires de points $p_{i,k}$ appartenant à la fibre i et $m_{j,k}$, leur plus proche voisin appartenant à une des fibres voisines j ;
  Calculs des normales aux plans en chaque point $p_{i,k}$;
  Construction de l'équation (3.8), et résolution qui permet d'obtenir les variations $(\delta t_{x_n}, \delta t_{y_n}, \delta t_{z_n}, \delta\alpha_n, \delta\beta_n, \delta\gamma_n)$;
until $\|\delta X\|_{max} < \delta$, *ou* $n_{iter} \geq n_{max}$;
Enregistrement des paramètres extrinsèque obtenus par optimisation ;

3.3.3 Validation du résultat de l'optimisation

Pour valider le résultat d'optimisation, l'énergie doit diminuer et atteindre un minimum global, ce qui dans le cas du recalage de données n'est pas trivial, notamment lorsque l'initialisation est un peu éloignée du résultat global. Aussi, un des résultats attendu est que notre valeur d'énergie en fin d'optimisation soit inférieure à sa valeur avant optimisation, mais pour valider l'optimisation, la valeur de l'énergie doit aussi être inférieure à un seuil donné : comme nous l'avons expliqué dans la section 3.3.1, notre énergie suit une distribution du χ^2, et lorsque le nombre de points N_t pris en compte dans son calcul est suffisamment grand, l'énergie donne un estimateur du bruit du nuage. Un seuil de validation pour cette énergie à 97% est $3\sigma^2$, avec σ^2 la valeur de l'énergie. Par exemple, en considérant un nuage provenant d'une acquisition réelle, et si l'on veut un nuage de bonne qualité, comme le bruit provient de plusieurs sources telles que l'acquisition même, la centrale inertielle ou encore le GPS, un bruit de déviation standard inférieur à 5cm est acceptable ; cela donne un seuil

pour la valeur finale de l'énergie J d'environ 75 cm^2, qui est le seuil que l'on va prendre en compte pour la validation des résultats d'optimisation sur des jeux de données réelles.

3.3.4 Précision des paramètres de calibrage extrinsèque

Avec les tests qui vont être présentés en section 3.4, nous allons voir que l'optimisation des paramètres extrinsèques donne de bons résultats à quelques exceptions près : dans la littérature concernant l'optimisation des paramètres de calibrage extrinsèque, le résultat est en général comparé à une vérité terrain. Cela peut prendre du temps de construire correctement cette vérité terrain, notamment dans le cas d'acquisitions réelles où l'on n'est pas assuré d'avoir une vérité terrain correcte. C'est pour cela qu'en plus du critère de validation défini en section 3.3.3, nous avons aussi défini un paramètre de précision concernant les valeurs de paramètres extrinsèques optimisés ; la matrice C de l'équation (3.8) peut-être vue comme une matrice de covariance des biais δX que l'on calcule à chaque itération, comme expliqué dans [Press *et al.*, 1992] dans le cas d'un système des moindres carrés linéaires, et elle nous sert à donner la « précision » des paramètres obtenus en fin d'optimisation, que l'on peut considérer comme un indice de confiance sur les valeurs obtenues :

$$\begin{cases} \text{Pour les translations : } \sigma_x(m) = \sqrt{(C^{-1})_{1,1}} \\ \text{Pour les rotations : } \sigma_\alpha(rad) = \sqrt{(C^{-1})_{4,4}} \end{cases} \tag{3.10}$$

Les précisions restantes pour les paramètres de translations et de rotations sont définies de la même manière. Nous verrons dans les sections 3.4 et 3.5 que selon le type de résultat souhaité, la précision des paramètres finaux est importante. En effet, ce qui nous intéresse est l'affinement des nuages, et le critère qui valide ou non cet affinement est lié la valeur finale de l'énergie ; toutefois, selon le type de jeux de données et de la trajectoire du véhicule, le critère de l'énergie peut être validé, mais la précision peut être mauvaise pour un ou plusieurs paramètres, ce qui poserait problème dans le cas où le but recherché avec notre algorithme est de se rapprocher de la vérité terrain des paramètres de calibrage extrinsèque, que l'on va supposer non connue pour nos tests car comme expliqué précédemment pour des jeux de données réels, elle est difficile à obtenir et peu fiable dans le cas où les paramètres sont mesurés à « la main ».

3.3.5 Définition des poids de la fonctionnelle à minimiser

Les poids $w_{i,j,k}$ définis dans la fonctionnelle (3.6) valent par défaut 1 ou 0 selon que la distance entre les deux points $p_{i,k}$ et $m_{j,k}$ soit inférieure à un seuil ou non. Dans [Demantke *et al.*, 2011], des attributs de dimensionnalité sont présentés avec une façon particulière de les calculer. Les attributs de dimensionnalité sont trois valeurs qui peuvent s'écrire sous la forme :

$$\begin{cases} a_{1D} = \frac{\sigma_1 - \sigma_2}{\sigma_1} \\ a_{2D} = \frac{\sigma_2 - \sigma_3}{\sigma_1} \\ a_{3D} = \frac{\sigma_3}{\sigma_1} \end{cases} \tag{3.11}$$

Pour calculer ces attributs pour un point du nuage, un voisinage du point est choisi, puis une analyse en composante principale est effectuée sur ce voisinage. Les valeurs propres obtenues sont classées par ordre décroissant, et numérotées de 1 à 3 : les σ_1 à σ_3 de l'équation 3.11 représentent les racines carrés de ces valeurs propres ordonnées. Les attributs a_{iD} ensuite calculés permettent de décrire la nature de la surface à laquelle le point concerné appartient : si a_{1D} a la plus grande valeur, cela signifie que le point appartient à un objet plutôt linéaire comme un poteau, et la dimensionnalité du point est alors de 1 ; si a_{2D} est le plus élevé, cela signifie que le point appartient à une surface

planaire, comme le sol ou une façade par exemple, et la dimensionnalité du point est alors de 2 ; enfin, si c'est a_{3D} qui est le plus élevé, le point devrait appartenir à un objet volumique, comme le feuillage d'un arbre, et la dimensionnalité est alors de 3. De plus, ces paramètres sont construits de sorte que leur valeur soit comprise entre 0 et 1 : nous avons donc choisi de les utiliser comme poids dans notre optimisation, puisque notre optimisation utilise une distance point à plan pour effectuer le recalage des données, et que l'on peut donner plus ou moins de poids aux associations avec ces attributs, selon que les points appartiennent à des surfaces planaires. Nous montrerons dans la section 3.4.4 que l'utilisation des attributs de dimensionnalité permet de réduire la valeur de l'énergie en fin d'optimisation.

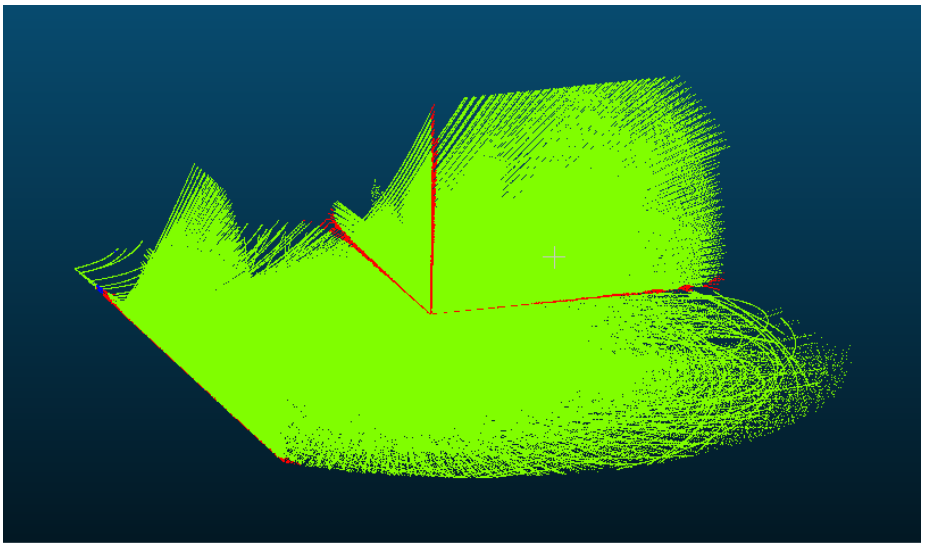


FIGURE 3.7 – Dimensionnalité calculée pour un nuage simulé : en vert, la dimensionnalité vaut 2, et en rouge elle vaut 3

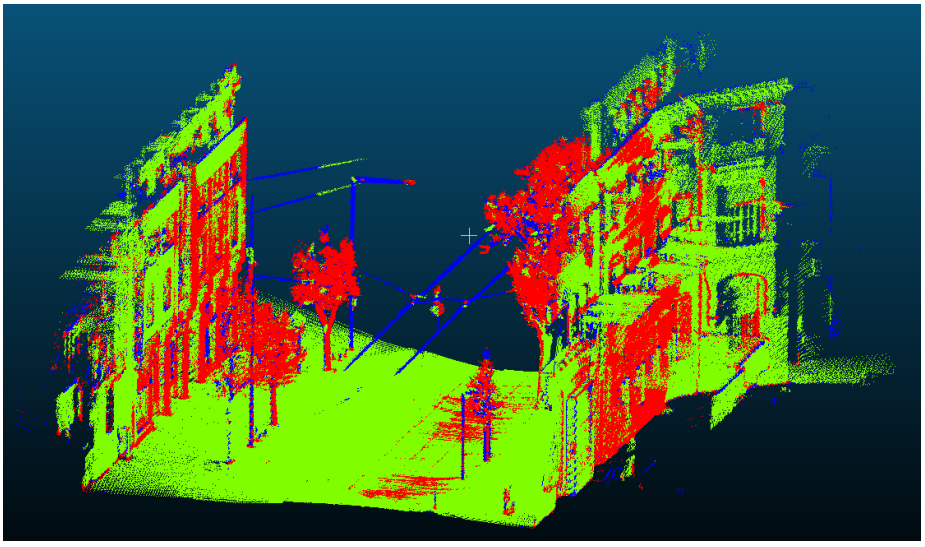


FIGURE 3.8 – Dimensionnalité calculée pour un nuage réel : en bleu, la dimensionnalité vaut 1 ; en vert, elle vaut 2, et en rouge elle vaut 3

Les figures 3.7 et 3.8 montrent la répartition des dimensionnalités des points sur un nuage simulé et un nuage issu d'une acquisition réelle. Pour les jeux de données simulés, les points ont une

dimensionnalité de 2 en majorité, sauf sur les bords, ce qui est logique car il s'agit de l'intersection de 2 plans orthogonaux. Pour les jeux de données réelles, la répartition est moins homogène : en effet, on peut voir que l'on a bien le feuillage des arbres avec une dimensionnalité de 3 en majorité, mais certains éléments du sol et des façades ont aussi une dimensionnalité de 3. Cela est dû aux erreurs provenant de différentes sources, et prendre en compte la valeur de l'attribut de dimensionnalité a_{2d} permet de donner un poids plus petit à ce genre de point. C'est ce que l'on appelle la planéité en un point, à savoir la valeur de l'attribut de dimensionnalité a_{2D} calculé en ce point : plus la valeur est proche de 1, plus le point concerné est à même d'appartenir à une surface planaire, et plus le rôle que l'on donne à ce point dans l'optimisation est important. Les poids $w_{i,j,k}$ utilisés qui valent :

$$w_{i,j,k} = \begin{cases} 1 \ si \ ||p_{i,k} - m_{j,k}|| < d_{max} \\ \\ 0 \ sinon \end{cases}$$

deviennent alors en utilisant la planéité en un point :

$$w_{i,j,k} = \begin{cases} max(a_{2D}(p), a_{2D}(m)) \ si \ ||p_{i,k} - m_{j,k}|| < d_{max} \\ \\ 0 \ sinon \end{cases}$$

Enfin, comme lors du calcul de la distance point à plan, 2 points $p_{i,k}$ et $m_{j,k}$ sont concernés, nous avons testé de prendre la valeur minimale entre les deux attributs, ou encore la moyenne, mais c'est la valeur maximale qui permettait d'avoir les meilleurs résultats de convergence.

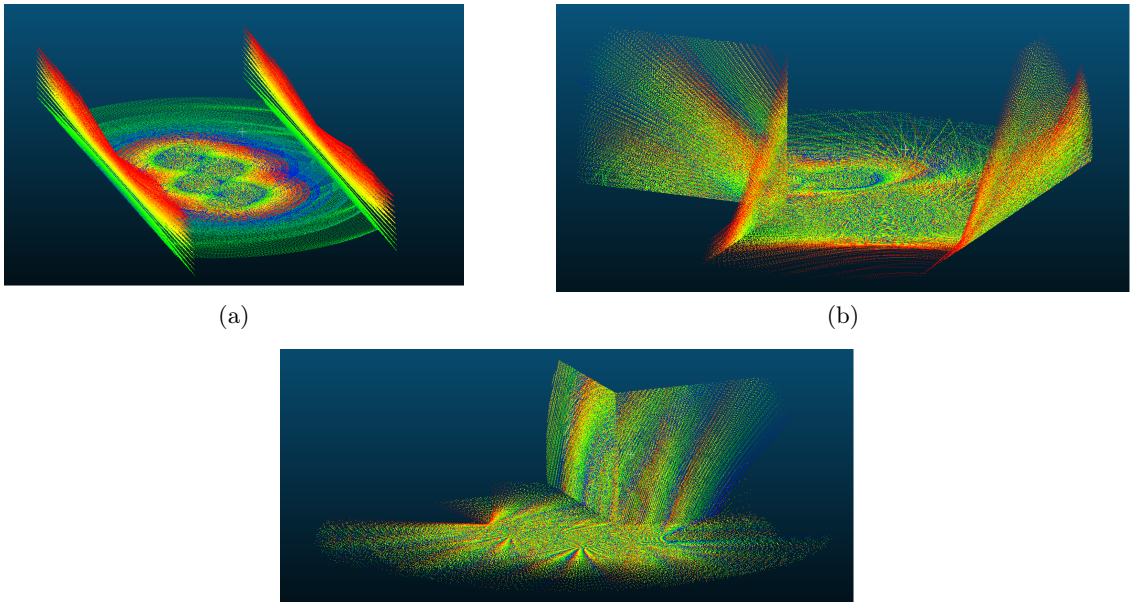
3.4 Résultats expérimentaux

L'optimisation des paramètres extrinsèques a été testée sur plusieurs jeux de données, simulés et issus d'acquisitions réelles. Dans un premier temps, nous allons présenter l'ensemble des jeux de données utilisés pour nos expérimentations, ainsi que les différents paramètres que l'on a dû fixer pour l'optimisation, puis nous présenterons plusieurs résultats d'optimisation obtenus avec les différents jeux de données.

3.4.1 Jeux de données utilisés pour les expérimentations

Les jeux de données simulés sont des nuages qui représentent une acquisition en environnement urbain. L'environnement est composé de plans verticaux (représentant des façades d'immeubles) et horizontaux (représentant le sol, ou la route dans ce cas). Les 3 nuages de points que l'on va présenter ont à peu près le même nombre de points, environ 5 millions de points, et la différence entre ces jeux de données est multiple : l'environnement change, le nombre et la position des plans est différente, mais la trajectoire du véhicule change aussi. Ces jeux de données sont utilisés pour valider notre approche car ils nous fournissent une vérité terrain avec laquelle comparer le résultat d'optimisation. En effet, les paramètres issus de calibrage extrinsèque sont précisément connus pour ces jeux de données, et pour tester notre optimisation, des erreurs ont été ajoutées aux paramètres extrinsèques, et ces erreurs devaient être corrigées par l'optimisation. Les 3 jeux de données simulées ont les propriétés suivantes :

- Le nuage #1, présenté en figure 3.9 a) est composé d'un sol et de 2 plans verticaux. Pour palier certains problèmes d'observabilité qui vont être détaillés en section 3.5, le véhicule a une trajectoire oscillante. Aussi, il n'y a pas de variation d'altitude.
- Le nuage #2 est présenté en figure 3.9 b), et est composé d'un sol, ainsi que de 3 plans verticaux. Le véhicule effectue un virage, et il y a une variation d'altitude.



(a) (b)

(c)

FIGURE 3.9 – Jeux de données simulées utilisées pour les expérimentations : a) jeu de données simulées #1 ; b) jeu de données simulées #2 ; c) jeu de données simulées #3

- Le nuage #3 est présenté en figure 3.9 c), et est assez proche du nuage # 2 : il n'y a que 2 plans verticaux cette fois, et pas de variation d'altitude par contre.

Les 2 jeux de données réelles que l'on a utilisés proviennent de 2 campagnes d'acquisitions différentes, et sont utilisées pour confirmer notre optimisation sur des jeux de données réelles où l'on ne maîtrise pas la valeur des paramètres extrinsèques de départ. Les deux nuages sont les suivants :

— Le nuage #4 est présenté en figure 3.10 a), et provient d'une acquisition effectué à Montbéliard, en France. Le nuage est composé de plusieurs façades, et le véhicule effectue quelques virages ; il y a aussi une légère variation d'altitude. Le nuage est environ composé de 10 millions de points.

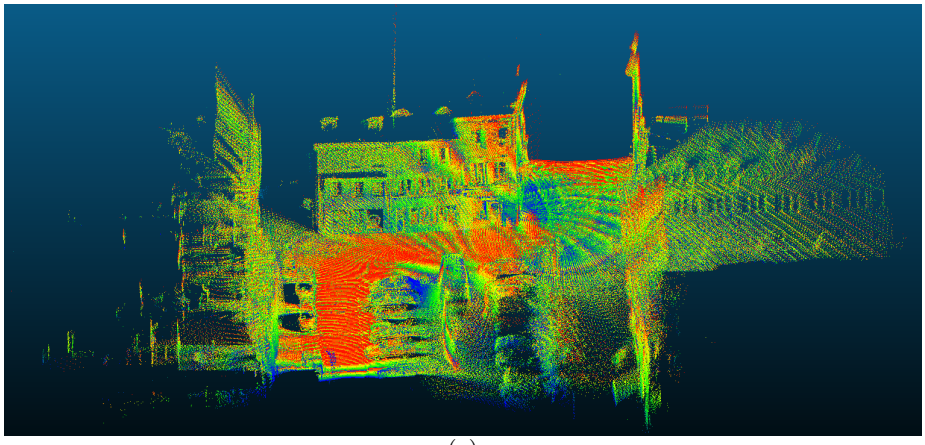
— Le nuage #5 est présenté en figure 3.10 b), et provient d'une acquisition effectué à Dijon, en France. Il n'y a que 2 façades parallèles entre elles, et une légère variation d'altitude, mais pas de virage : par contre, il y a aussi beaucoup de végétation, ainsi que quelques câbles électriques, qui sont des éléments non planaires. Le nuage de points est environ composé de 5 millions de points.

3.4.2 Choix des différents paramètres de l'optimisation

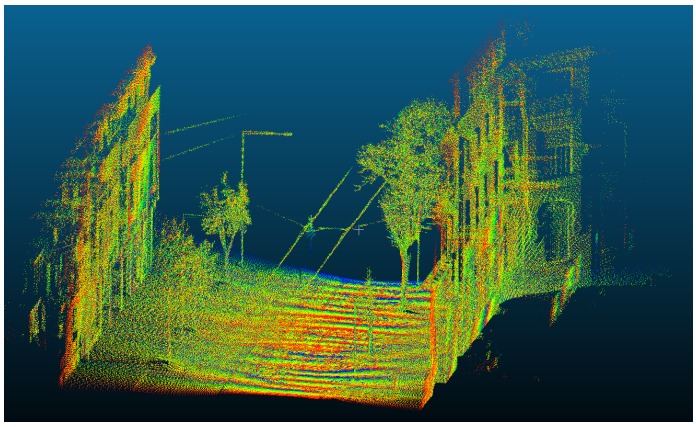
Notre optimisation a été codée en C++. La librairie EIGEN [Eigen library, 2017] a été utilisée pour toutes les opérations matricielles ou vectorielles, et la librairie FLANN [FLANN library, 2017] (Fast Library for Approximated Nearest Neighbor) a été utilisée pour la recherche des plus proches voisins, notamment lors de l'appariement des points ou le calcul des normales. L'algorithme a été exécuté sur un ordinateur avec un processeur de fréquence 3.40 GHz.

De nombreux paramètres avaient besoin d'être fixés dans notre algorithme :

- Pour commencer, nous avons sous-échantillonné les nuages de points à 1 point sur 3, car la



(a)



(b)

FIGURE 3.10 – Jeux de données réelles utilisées pour les expérimentations : a) jeu de données réelles #4 ; b) jeu de données réelles #5

densité des nuages provenant du capteur Velodyne est très élevée, et le sous-échantillonnage permet d'accélérer l'optimisation sans perdre en précision.

- Ensuite, nous avons pris une distance maximale entre deux points appariés $p_{i,k}$ et $m_{j,k}$ de 20cm : plusieurs valeurs ont été testées, et ce seuil donnait les meilleurs résultats d'optimisation. Les poids $w_{i,j,k}$ valent la valeur de l'attribut de dimensionnalité a_{2D} maximum entre les points $p_{i,k}$ et $m_{j,k}$ si le seuil est respecté, 0 sinon.
- Nous faisons du recalage de données entre fibres, et nous avons fixé le nombre de fibres voisines à la fibre b_i à 4, c'est à dire N=2. Une justification du choix de cette valeur est donnée en Annexe B.
- Aussi, il nous a fallu fixer la taille du voisinage pris en compte pour le calcul des normales. Un nombre de 150 voisins a été retenu : le choix de cette valeur est aussi justifié en Annexe B.
- Pour les attributs de dimensionnalité, il a fallu choisir une taille de voisinage pour le calcul des attributs, le nombre de voisins pris en compte influant directement sur la dimensionnalité des points. Dans [Demantke *et al.*, 2011], la taille du voisinage est automatiquement optimisée pour qu'une valeur d'entropie liée aux valeurs des attributs soit la plus petite possible : pour des soucis de temps de calcul, nous avons préféré fixer le nombre de points voisins pour le calcul des attributs. La figure 3.11 présente la répartition des dimensionnalité en fonction du nombre de voisins. On voit que la répartition est correcte à partir d'un voisinage supérieur à 100 points : le sol et les plans verticaux ont des dimensionnalités de 2 en majorité, ce qui est

ce que l'on attend pour des surfaces planaires. Aussi, le temps de calcul des attributs n'étant pas très différent entre un voisinage de 100 et de 150, nous avons fixé le nombre de voisins pris en compte pour le calcul des attributs à 100.

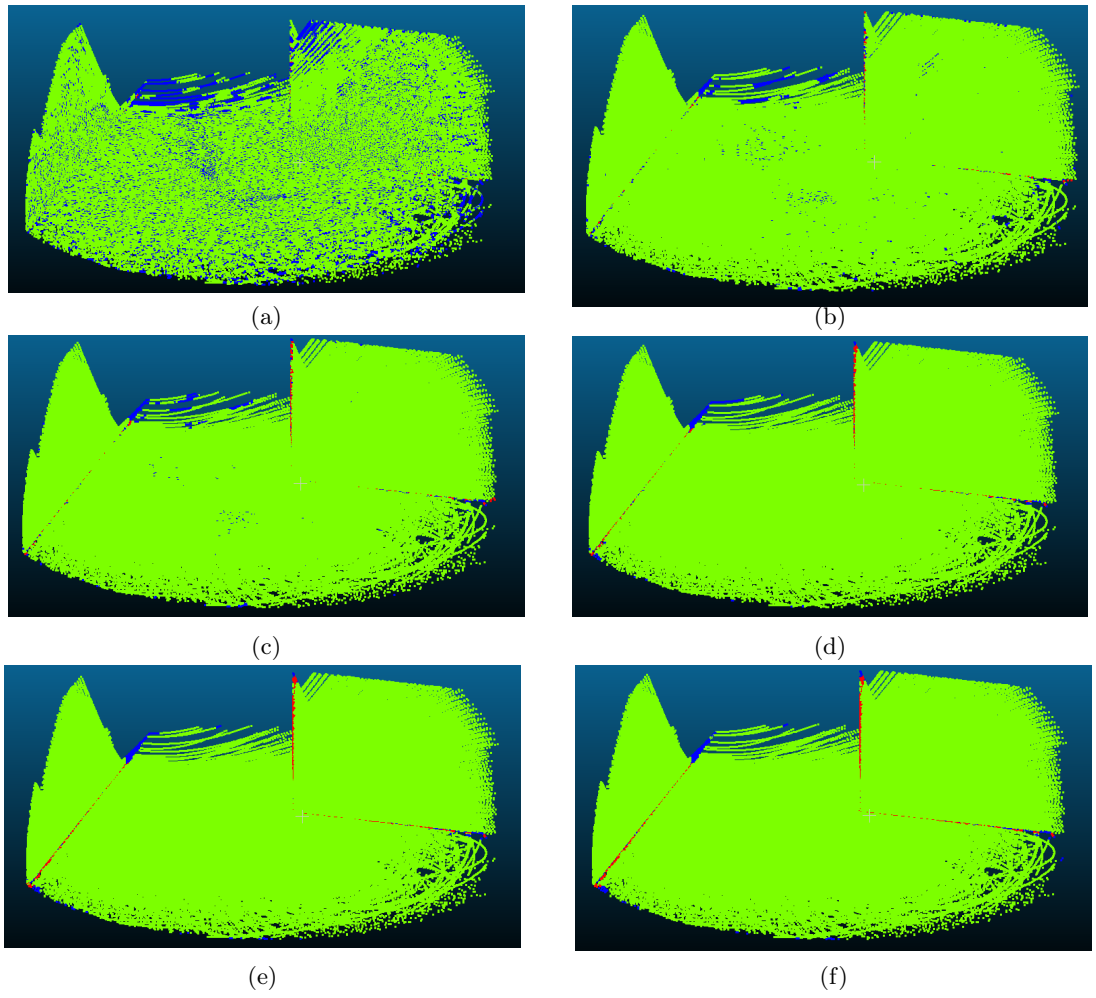


FIGURE 3.11 – Répartition des dimensionnalités en fonction du nombre de voisins pris en compte pour le calcul des attributs : a) 10 points les plus proches ; b) 30 voisins ; c) 50 voisins ; d) 100 voisins ; e) 150 voisins ; f) 200 voisins

- Pour les critères d'arrêts de l'optimisation, nous avons pris un seuil δ de 1cm pour les translations et 0.01° pour les rotations ; l'optimisation s'arrête si l'ensemble des paramètres extrinsèques sont inférieurs à ces seuils. Sinon, le nombre d'itérations maximums a été fixé à 40 ; au cours des expérimentations, nous avons remarqué que l'énergie n'évolue plus de façon significative après un certain nombre d'itérations, mais qu'à cause d'une mauvaise observabilité de certains paramètres pour certains jeux de données, les paramètres continuaient à être modifiés, sans que cela ne change le résultat d'affinement.

Enfin, au niveau des paramètres extrinsèques utilisés pour initialiser notre algorithme, nous avons ajouté des erreurs aux paramètres extrinsèques connus pour les jeux de données simulées, et pour les jeux de données réelles, nous avons pris des paramètres choisis arbitrairement pour l'initialisation de l'optimisation : d'une part, il n'est pas nécessaire d'avoir une initialisation précise des paramètres extrinsèques, et d'autre part, cela à aussi permis de tester la robustesse de notre optimisation à une initialisation qui serait éloignée de la solution optimale.

3.4.3 Comparaison de notre optimisation avec une méthode de l'état de l'art

Dans une première expérimentation, nous avons voulu comparer notre méthode d'optimisation avec une méthode tirée de l'état de l'art. Pour cela, nous avons pris l'optimisation des paramètres extrinsèques présentée dans [Levinson et Thrun, 2010], qui utilise aussi des données provenant d'un capteur Velodyne. Leur méthode d'optimisation est différente de la notre, comme nous l'avons présenté en section 3.3 ; pour comparer leur méthode à la notre, nous avons fixé plusieurs paramètres :

- La taille du voisinage pris en compte dans la recherche des paramètres optimaux a été fixée à + ou - 3 mètres autour des paramètres initiaux, car l'erreur que l'on rajoute aux paramètres de translations est de l'ordre de 3 mètres ; pour les paramètres de rotations, le voisinage était compris entre + ou - 7°, pour les même raisons.
- Le pas de discrétisation pour la recherche des valeurs des paramètres de calibrage optimaux a été fixée à 5, pour réduire le temps de calcul, et le nombre d'itérations choisi est de 10 : à chaque itération, la taille du voisinage dans lequel les recherches sont effectuées est divisée par 2, et avec les tailles de voisinages initiales, au bout de 10 itérations, cela permettait de finir avec une largeur de recherche inférieure au centimètre pour les translations, et inférieure au centième de degré pour les rotations, ce qui correspond aux critères d'arrêt que nous avons défini en section 3.4.2.

Le nuage de point utilisé pour la comparaison est le nuage simulé #1, présenté en section 3.4.1. En fin d'optimisation, on peut voir avec la figure 3.12 que visuellement, les deux méthodes d'optimisations donnent un nuage de points affiné, ce qui montre que les paramètres extrinsèques ont bien été optimisés. Le résultat est meilleur avec notre approche, où les plans verticaux sont correctement reconstruits.
La table 3.1 présente les erreurs que l'on a au niveau des paramètres extrinsèques avant optimisation, puis après optimisation avec la méthode de Levinson et la notre. Avec notre optimisation, les paramètres de calibrage extrinsèque sont très proches de la vérité terrain après optimisation, sauf pour le paramètre de translation z, mais cela n'est pas surprenant car l'observabilité est nulle dans cette direction, le nuage ne présentant pas de variation d'altitude. Avec l'optimisation comparée, les résultats vont dans le même sens, mais sont moins bons, ce qui explique la différence que l'on voit avec la figure 3.12.

La figure 3.13 présente l'évolution des énergies pour les 2 optimisations comparées. Les énergies démarrent à une valeur de 217.41 cm^2, et décroissent jusqu'à une valeur de 7.20 cm^2 pour la méthode comparée, et jusqu'à une valeur de 0.29 cm^2 pour notre optimisation. L'évolution des énergies confirme l'observation faite précédemment, qui est que notre optimisation donne un meilleur affinement que la méthode comparée.

Enfin, un dernier point que l'on peut relever est le temps de calcul : notre optimisation a mis 2 minutes pour converger et donner le résultat final, tandis que la méthode comparée à mis environ 1 heure pour le même nuage : là aussi, notre optimisation est meilleure que la méthode comparée. La figure 3.13 montre que notre optimisation nécessite un plus grand nombre d'itérations que l'optimisation de Levinson, mais le temps de calcul est bien plus faible : en effet, l'optimisation comparée est une recherche exhaustive, pour laquelle l'énergie est calculée à plusieurs reprises à chaque itération.

3.4.4 Optimisation des paramètres extrinsèques

Nous allons maintenant présenter plusieurs résultats d'optimisation des paramètres extrinsèques. Dans un premier temps, nous allons présenter deux résultats avec des jeux de données simulées, puis deux résultats avec des jeux de données réelles. Des comparaisons entre deux approches d'optimisations sont effectuées : en effet, nous cherchons aussi à montrer que l'utilisation des attributs de

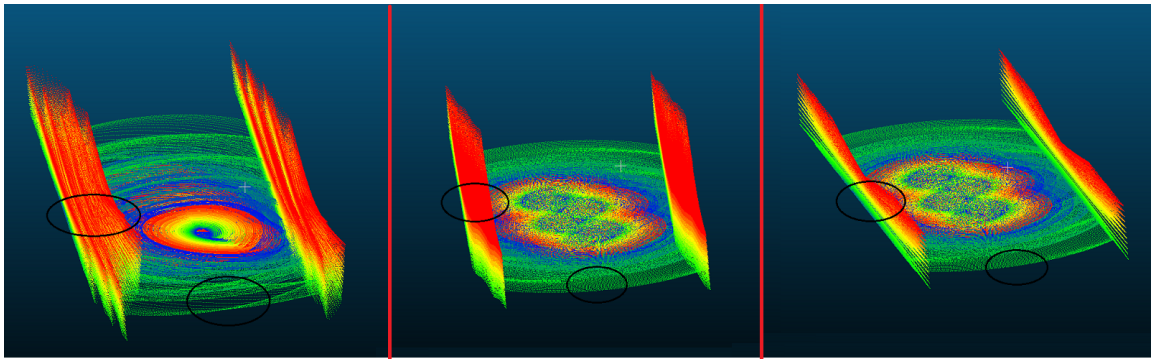


FIGURE 3.12 – Nuage de points simulé #1 : sur la gauche, on a le nuage de points avec des paramètres erronés, avant optimisation ; au milieu, le résultat de l'optimisation avec la méthode de Levinson ; à droite, le résultat d'optimisation avec notre approche. Les 3 nuages sont vus avec la même orientation

	t_x(cm)	t_y(cm)	t_z(cm)	roulis $\alpha(^\circ)$	tangage $\beta(^\circ)$	lacet $\gamma(^\circ)$
Erreur ajoutée aux paramètres extrinsèques	-150.000	250.000	-200.000	5.000	-7.000	-5.500
Différence finale entre la vérité terrain et l'optimisation comparée	6.445	8.594	-500.000	-0.004	0.000	-9.875
Différence entre la vérité terrain et notre optimisation	**0.001**	**-0.012**	-200.000	**0.000**	**0.000**	**0.000**

TABLE 3.1 – Erreurs finales des optimisations des paramètres extrinsèques pour le nuage #1

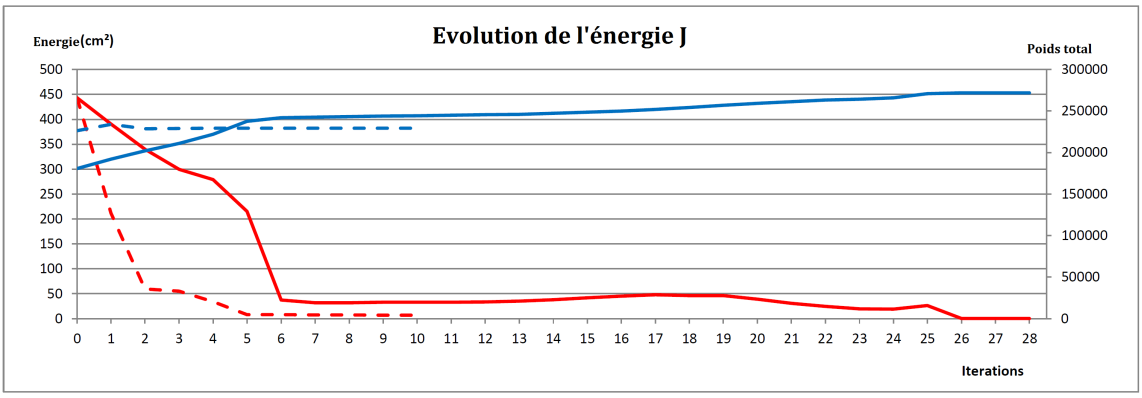


FIGURE 3.13 – Évolution des énergies pour le nuage #1 : en rouge et en trait plein, nous avons notre énergie ; en rouge et en pointillé, nous avons l'énergie de la méthode comparée ; en bleu et en trait plein, nous avons le nombre de points appariés pris en compte dans notre optimisation ; en bleu et en pointillé, le nombre de points appariés pour l'optimisation comparée

dimensionnalité dans les poids utilisés pendant l'optimisation permettent de réduire la valeur finale de l'énergie minimisée. Pour les mises à jour des valeur des attributs de planéité, elles sont effectuées toutes les 7 itérations, afin d'avoir des attributs qui évoluent avec l'affinement des données, et afin de ne pas trop ralentir l'optimisation.

3.4.4.1 Résultats sur des jeux de données simulées

Le premier nuage utilisé pour les tests est le nuage simulé #2. Il y a une variation d'altitude dans le nuage, et le véhicule effectue un virage. La figure 3.14 présente le même nuage de points avec la même orientation pour la visualisation, mais avec deux ensembles de paramètres extrinsèques différentes :

- En haut, le nuage de point est créé avec les paramètres initiaux que l'on donne en entrée à l'optimisation.
- En bas, le même nuage avec les paramètres extrinsèques optimisés

Un seul résultat d'optimisation est présenté car en fin d'optimisation, il n'y a pas de différence visuelle entre les deux nuages affinés avec ou sans l'utilisation des attributs de dimensionnalité. La table 3.2 donne les erreurs ajoutés aux paramètres extrinsèques du nuage de points simulé #2, et donne aussi la différence entre les paramètres extrinsèques optimisés pour le nuage simulé et la vérité terrain. Dans les deux cas d'optimisations, nous avons des paramètres extrinsèques très proches de la vérité terrain en fin d'optimisation, malgré l'initialisation assez éloignée de la vérité terrain : pour les translations, l'erreur finale est inférieure au millimètre, et pour les translations, l'erreur est environ de $1/1000^{\circ}$. Les erreurs sont globalement les plus petites lorsque les attributs de dimensionnalité sont pris en compte. La figure 3.15 donne l'évolution des énergies d'optimisation, avec et sans utilisation des attributs de dimensionnalité : sans les attributs, l'énergie démarre à une valeur de 243.07 cm^2, et diminue jusqu'à atteindre 0.58 cm^2 ; avec les attributs, l'énergie évolue d'une valeur de 86.63 cm^2 à une valeur de 0.46 cm^2. L'énergie est plus faible avec l'utilisation des attributs de dimensionnalité, et la convergence est aussi plus rapide en terme de nombre d'itérations. Comme l'énergie est normalisée, nous donnons aussi sur la même figure l'évolution du facteur de normalisation qui est la somme des poids pour chaque paire de points : avec l'utilisation des attributs, ce facteur augmente considérablement à chaque mise à jour, ce qui illustre le fait que le recalage des données entre fibres se fait correctement.
L'optimisation donne de bons résultats pour chacun des paramètres extrinsèques, et la table 3.3 donne des valeurs de précisions qui vont dans le même sens : en effet, pour chaque paramètre, la précision est très bonne, ce qui va dans le sens des erreurs très faibles par rapport à la vérité terrain décrites dans la table 3.2. Un dernier résultat que l'on peut présenter est le temps de calcul des optimisations : sans l'utilisation des attributs de dimensionnalité, l'optimisation prend 1 minute et 15 secondes pour converger. Avec l'utilisation des attributs de dimensionnalité, l'optimisation prend environ 5 minutes à cause du calcul des attributs de dimensionnalité, mais le temps de calcul reste assez faible par rapport au résultat fourni qui est meilleur que la première optimisation.

	t_x(cm)	t_y(cm)	t_z(cm)	roulis $\alpha(^{\circ})$	tangage $\beta(^{\circ})$	lacet $\gamma(^{\circ})$
Erreurs initiales par rapport à la vérité terrain	-150.00	250.00	-200.00	5.00	-7.00	-5.50
Différence entre la vérité terrain et notre résultat d'optimisation (sans utilisation de la dimensionnalité)	-0.003	-0.016	0.033	-0.000	0.000	0.001
Différence entre la vérité terrain et notre résultat d'optimisation (avec utilisation de la dimensionnalité)	-0.001	-0.023	0.008	-0.000	0.000	0.000

TABLE 3.2 – Erreurs sur les paramètres extrinsèques avant et après optimisation par rapport à la vérité terrain pour le nuage #2

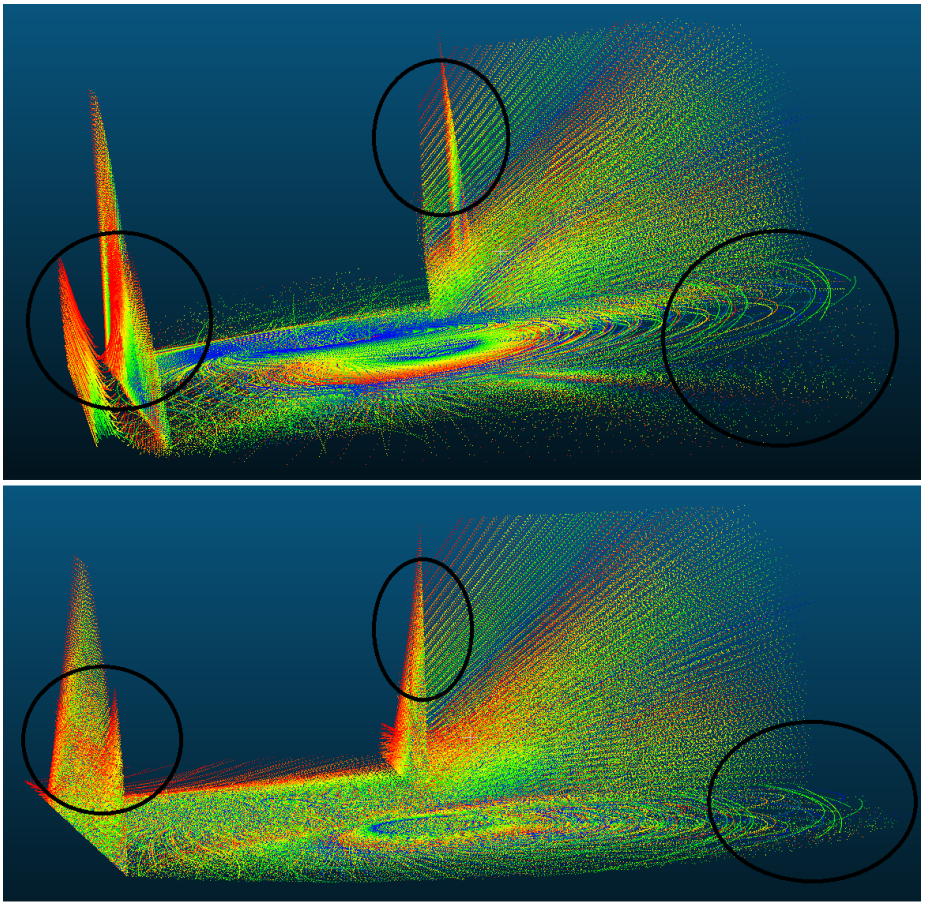


FIGURE 3.14 – Nuage de points simulé #2 : en haut, nuage avec les paramètres extrinsèques initiaux ; en dessous, le résultat de notre optimisation. Les deux images sont vues depuis le même point.

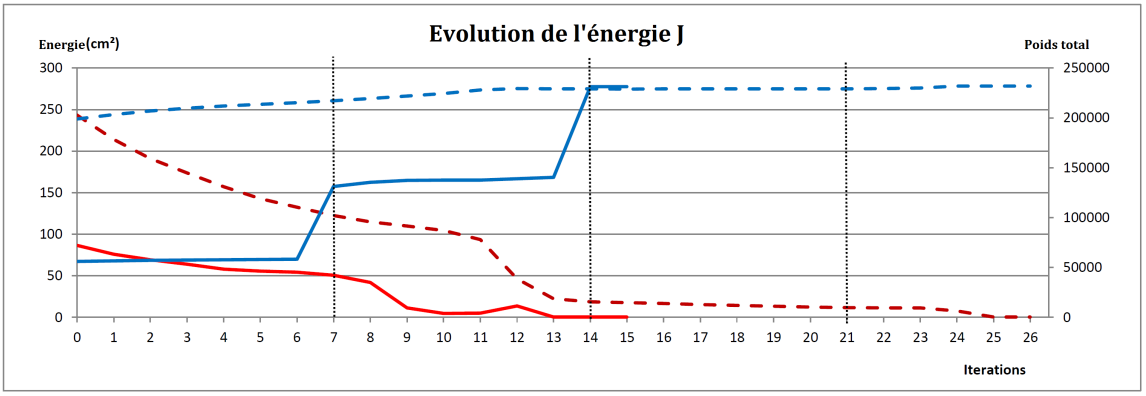


FIGURE 3.15 – Évolution de l'énergie au cours de l'optimisation pour le nuage simulé #2. La ligne rouge pleine représente l'évolution de l'énergie avec l'utilisation des attributs de dimensionnalité ; la ligne rouge pointillé représente la même énergie sans l'utilisation des attributs ; la ligne bleue pleine représente l'évolution du poids total utilisé pour normaliser notre énergie, avec l'utilisation des attributs de dimensionnalité ; la ligne bleue pointillé représente la même donnée, mais sans l'utilisation des attributs de dimensionnalité ; enfin, les lignes noires verticales marquent la mise à jour des attributs de dimensionnalité, toutes les 7 itérations.

	σ_{t_x}(cm)	σ_{t_y}(cm)	σ_{t_z}(cm)	$\sigma_\alpha(^\circ)$	$\sigma_\beta(^\circ)$	$\sigma_\gamma(^\circ)$
Nuage de points simulé #2	1.00	2.18	3.43	0.06	0.04	0.07

TABLE 3.3 – Précision des paramètres extrinsèques pour le nuage de points simulé #2

Un deuxième jeu de données simulées a été utilisé pour valider notre optimisation des paramètres extrinsèques, le nuage #3 : pour les mêmes raisons qu'avec le nuage #2, un seul résultat d'optimisation est présenté. Les résultats sont similaires à ceux présentés pour le nuage #2 ; plus de détails sur les résultats d'optimisation pour le nuage #3 sont donnés en Annexe C.

3.4.4.2 Résultats sur des jeux de données réelles

Dans cette section, nous présentons des résultats d'optimisation des paramètres extrinsèques sur des jeux de données réelles. Le premier nuage de points que l'on utilise est le nuage #4, qui provient d'une acquisition à Montbéliard. La figure 3.17 présente en haut le nuage de points avec les paramètres de calibrage initiaux donnés en entrée de l'optimisation, et sans hypothèse ou connaissance des paramètres extrinsèques comme pour les jeux de données simulées, nous présentons dans la sous-figure du bas le même nuage de points avec des paramètres extrinsèques optimisés par notre méthode.
Comme pour les jeux de données simulées, nous présentons aussi une comparaison entre deux optimisations, une sans l'utilisation des attributs de dimensionnalité et une autre avec leur utilisation. La figure 3.16 présente l'évolution des énergies pour les deux optimisations : l'énergie évolue d'une valeur de 224.81 cm^2 à une valeur de 59.26 cm^2 sans l'utilisation des attributs, et d'une valeur de 145.11 cm^2 à une valeur de 43.26 cm^2, qui est plus faible. Si l'on considère le critère de validation défini en section 3.3.3, avec une déviation standard de 5 cm pour le bruit du nuage, cela nous donne un seuil de 75 cm^2 pour la valeur finale de l'énergie, et l'optimisation respecte ce critère dans les deux cas présentés. Comme pour les jeux de données simulées, nous mettons à jour les attributs de dimensionnalité toutes les 7 itérations, et on peut voir que les effets de ces mises à jour sont similaires : cela se traduit par une diminution importante de l'énergie et une augmentation non négligeable du poids total pris en compte. Cette fois-ci, et contrairement aux jeux de données simulées, le poids total avec utilisation des attributs de dimensionnalité reste toujours plus petit que le poids total sans, ce qui n'est pas surprenant vu que l'on considère un jeu de données réelles et que de nombreux éléments non plans sont présents dans le nuage : pour ces éléments non plans, les valeurs des attributs de planéité sont faibles. Aussi, l'énergie converge assez rapidement vers la valeur optimale : le nombre d'itérations continue d'augmenter parce que les paramètres extrinsèques sont toujours modifiés. La précision des paramètres est présenté dans la table 3.5, et on peut voir que globalement, les valeurs sont correctes pour l'ensemble des paramètres, sauf pour la translation en z, mais cela est normal vu que la variation d'altitude est assez faible. Aussi, les valeurs de précisions sont plus mauvaises que pour les jeux de données simulées, ce qui justifie le fait que les énergies présentées avec la figure 3.16 ne diminuent plus après un certain nombre d'itérations : l'optimisation a convergé, mais à cause des précisions, une variation des paramètres extrinsèques ne modifie pas l'énergie, et c'est aussi pour cela que l'on a limité le nombre d'itérations à 40.
La table 3.4 présente les paramètres extrinsèques utilisés en entrée de l'optimisation, et les valeurs optimisées pour le nuage #4. Les lignes donnent :

— les paramètres initiaux, choisis arbitrairement afin de ne pas avoir besoin d'effectuer une initialisation précise.

— les paramètres extrinsèques, mesurés à la « main » sur notre véhicule d'acquisition mobile lors d'un premier étalonnage, afin d'avoir une idée du voisinage dans lequel les paramètres optimisés devraient se trouver si l'observabilité est bonne dans toutes les directions. Ces paramètres ne sont pas optimaux, et c'est pour cela que l'on a une initialisation arbitraire : le résultat d'optimisation ne doit pas nécessairement être proche de ces valeurs, puisque le but de notre

optimisation est d'affiner le nuage de point et de trouver les paramètres extrinsèques qui permettent d'avoir le meilleur affinement possible.

— les résultats d'optimisation pour les deux approches, avec et sans utilisation des attributs de dimensionnalité

Les paramètres extrinsèques optimisés sont assez proches pour les deux approches, et sont aussi assez proches des paramètres mesurés sur le véhicule : c'est le genre de résultats que l'on veut avoir avec notre optimisation, et c'est le cas ici parce que les précisions sont globalement assez bonnes pour l'ensemble des paramètres. Pour le paramètre de translation en z, la différence est très élevée, mais c'est le résultat que l'on pouvait attendre au vu de la mauvaise précision dans cette direction : l'erreur finale est de plusieurs mètres par rapport aux paramètres mesurés, alors que la précision dans cette direction est de l'ordre du mètre. Enfin, pour les temps de calculs, il est de 2 minutes environ sans les attributs de dimensionnalité et de 7 minutes environ avec, qui sont des temps de calculs acceptables au vu de la taille du nuage considéré.

	t_x(m)	t_y(m)	t_z(m)	roulis $\alpha(^\circ)$	tangage $\beta(^\circ)$	lacet $\gamma(^\circ)$
Valeurs des paramètres extrinsèques initiale	0.00	0.00	0.00	0.00	-45.00	90.00
Paramètres mesurés à la main sur notre véhicule d'acquisition	-0.21	-1.22	0.95	0.00	-60.00	90.00
Paramètres obtenus après optimisation (sans dimensionnalité)	-0.46	-1.56	-7.21	4.49	-60.06	89.59
Paramètres obtenus après optimisation (avec dimensionnalité)	-0.48	-1.37	-7.60	-0.27	-59.84	93.74

TABLE 3.4 – Erreurs sur les paramètres extrinsèques après optimisation pour le nuage #4

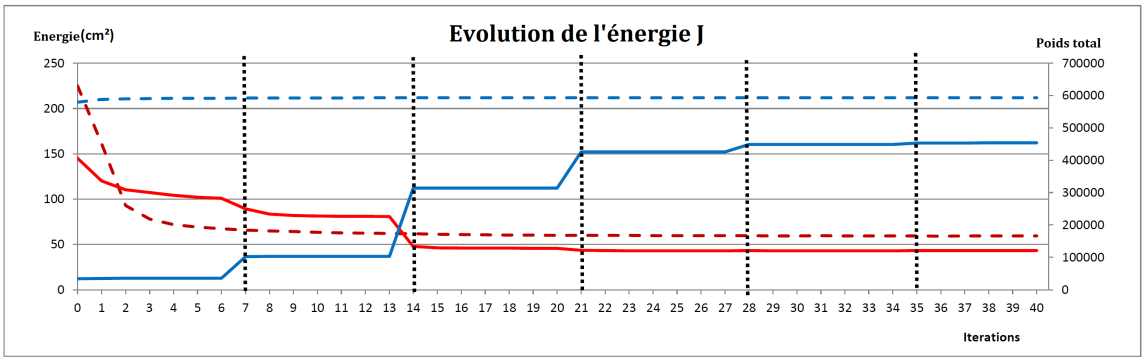


FIGURE 3.16 – Évolution de l'énergie au cours de l'optimisation pour le nuage réel #4. La ligne rouge pleine représente l'évolution de l'énergie avec l'utilisation des attributs de dimensionnalité ; la ligne rouge pointillé représente la même énergie sans l'utilisation des attributs ; la ligne bleue pleine représente l'évolution du poids total utilisé pour normaliser notre énergie, avec l'utilisation des attributs de dimensionnalité ; la ligne bleue pointillé représente la même donnée, mais sans l'utilisation des attributs de dimensionnalité ; enfin, les lignes noires verticales marquent la mise à jour des attributs de dimensionnalité, toutes les 7 itérations.

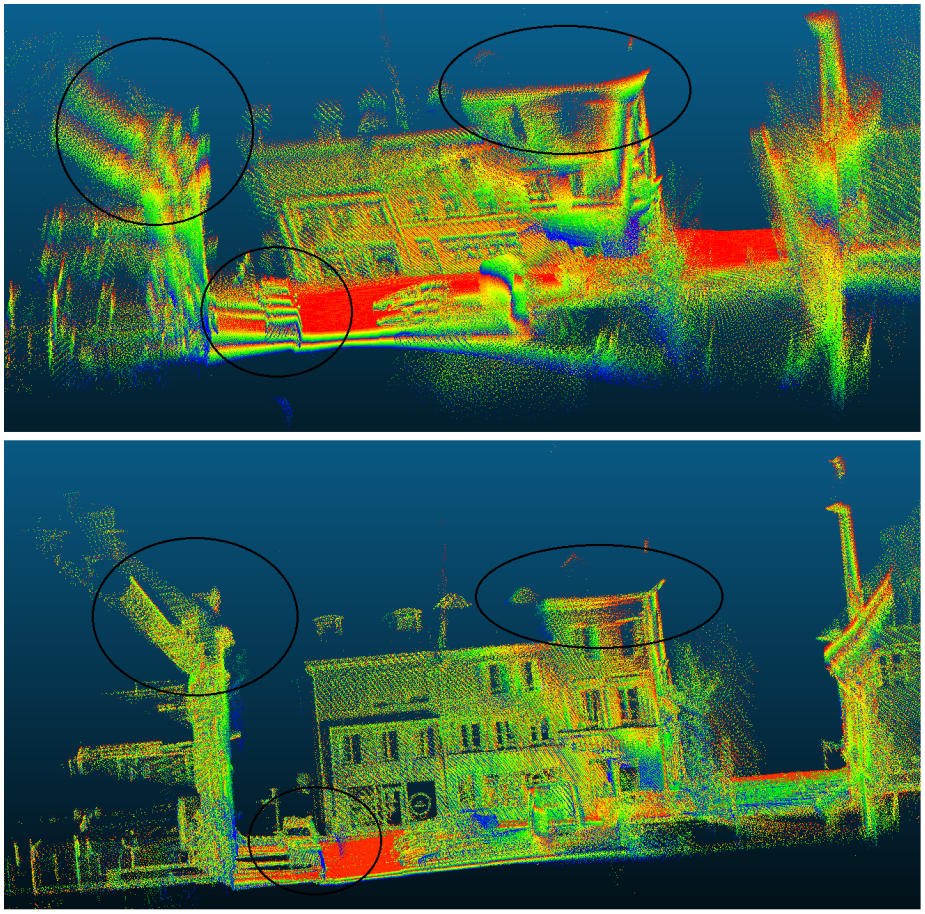
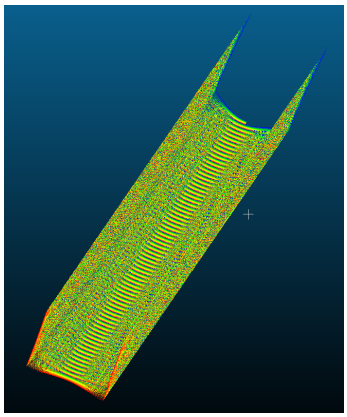


FIGURE 3.17 – Nuage de points réel #4 : en haut, nuage avec les paramètres de calibrage extrinsèque initiaux ; en dessous, le résultat de notre optimisation. Les deux images sont vues depuis le même point

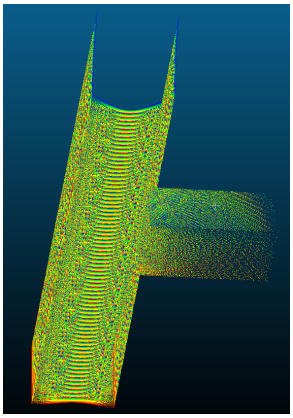
	σ_{tx}(cm)	σ_{ty}(cm)	σ_{tz}(cm)	$\sigma_{\alpha}(^{\circ})$	$\sigma_{\beta}(^{\circ})$	$\sigma_{\gamma}(^{\circ})$
Nuage réel #4	7.77	3.10	157.66	0.50	0.33	0.47

TABLE 3.5 – Précision des paramètres extrinsèques pour le nuage de points réel #4

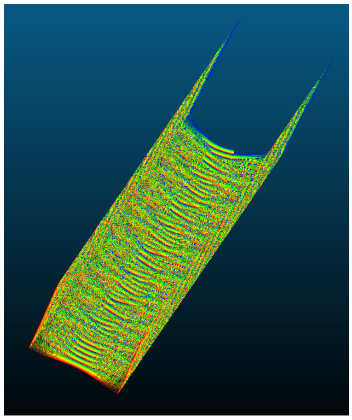
Un deuxième jeu de données réelles est utilisé, le nuage #5. Les résultats sont similaires à ceux présentés pour le nuage #4, et sont détaillés en Annexe C.



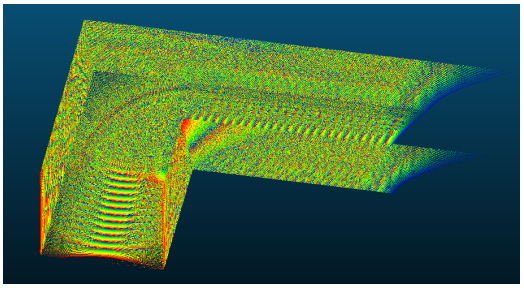
(a) Nuage simulé #a



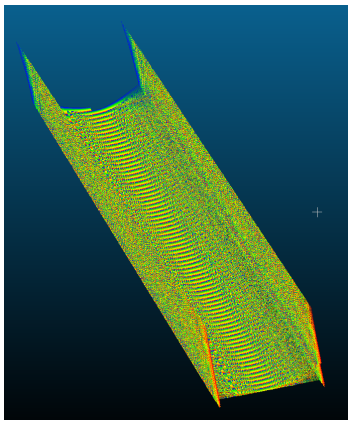
(b) Nuage simulé #b



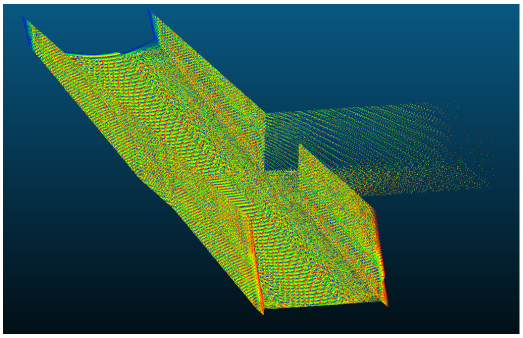
(c) Nuage simulé #c



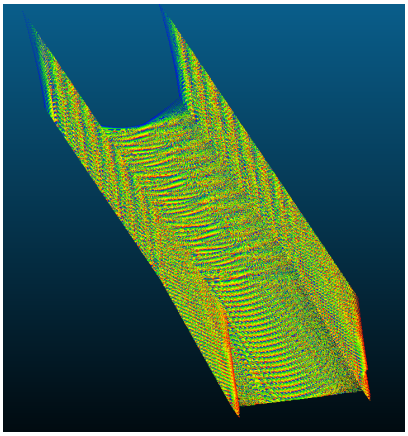
(d) Nuage simulé #d



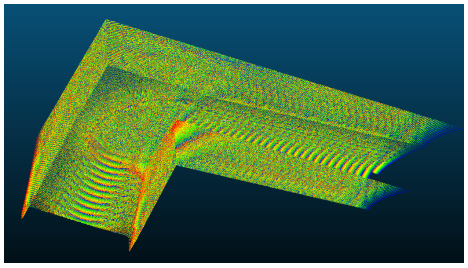
(e) Nuage simulé #e



(f) Nuage simulé #f



(g) Nuage simulé #g



(h) Nuage simulé #h

FIGURE 3.18 – Nuages simulés utilisés pour l'étude de l'observabilité lors de l'optimisation des paramètres extrinsèques

3.5 Observabilité des paramètres extrinsèques

Dans les sections 3.3 et 3.4, nous avons vu une méthode d'optimisation automatique des paramètres de calibrage extrinsèque d'un système LIDAR multi-couches, avec des résultats simulés et réels où le système concerné est un capteur LIDAR multi-fibres, le Velodyne HDL-32E. Nous avons montré des résultats satisfaisants pour nos objectifs, mais on a tout de même pu relever certains problèmes, notamment par rapport à ce que l'on a appelé « l'observabilité » des paramètres.

L'observabilité d'un paramètre permet de traduire la « capacité » à détecter les variations de ce paramètre, et plus un paramètre est observable, plus il est par exemple facilement optimisable ou plus il aura d'influence sur un système donné. Si on reprend les résultats présentés en section 3.4.4.2, avec le jeu de données réelles #4, les paramètres de précision présentés dans la table 3.5 sont inversement liés à l'observabilité : plus la précision est faible, plus un paramètre est observable, et inversement. Du coup, comme nous l'avons expliqué, les 3 paramètres de rotations extrinsèques sont les paramètres les plus observables, et pour les translations, ce sont les translations selon les axes x et y qui sont correctement observables ; la translation selon l'axe z n'est pas du tout observable. L'effet de l'observabilité est visible sur les résultats d'optimisation des paramètres extrinsèques : si l'observabilité d'un paramètre est bonne, le paramètre optimisé sera proche de la vérité terrain, ou tout du moins proche de la valeur réelle ; à l'inverse, si un paramètre n'est pas observable, la valeur optimisée par notre algorithme ne modifiera pas la structure du nuage, mais sera éloignée de la valeur réelle.
L'observabilité d'un paramètre dépend de plusieurs choses, des données mais aussi du système numérique dans lequel le paramètre intervient.

- La structure des données influe directement sur l'observabilité des paramètres. Si on reprend le nuage réel #4 utilisé pour les tests de notre algorithme, ce nuage comporte plusieurs bâtiments, ainsi que certains objets non linéaires et quelques véhicules ; il y a aussi une légère variation d'altitude durant l'acquisition. Le véhicule mobile d'acquisition effectue quelques virages. Ces différentes caractéristiques influent donc directement sur l'observabilité des paramètres : les virages permettent d'avoir une bonne observabilité dans les directions de translations x et y ; les bâtiments et leur structure non uniforme, ainsi que les déplacements du véhicule permettent d'avoir une bonne observabilité pour les trois paramètres de rotations ; par contre, pour le paramètre de translation selon l'axe z, l'observabilité est mauvaise car il y a peu de changements dans la direction verticale et l'altitude ne varie pas assez. Avec le nuage réel #5, dont les résultats d'affinement sont présentés en Annexe C, l'observabilité est différente car les caractéristiques du nuage ont changé : la trajectoire du véhicule est une ligne droite, et cela suffit à rendre l'observabilité dans plusieurs directions mauvaises.
- Le système numérique dans lequel les paramètres interviennent est aussi important : dans notre cas, nous avons une fonctionnelle à minimiser et une résolution qui nous permet d'obtenir les valeurs de précisions comme présenté dans la section 3.3.4 ; un autre choix de résolution où des paramètres à optimiser différents n'auraient pas conduits à la même précision puisque la matrice de covariance que l'on utilise aurait été différente.

Avec notre résolution numérique, nous avons testé plusieurs configurations d'acquisitions pour voir comment évolue l'observabilité des paramètres extrinsèques en fonction des caractéristiques des nuages de points et de la trajectoire du véhicule. Pour cela, nous avons simulé 8 acquisitions, avec différentes trajectoires du véhicule et caractéristiques ; ces nuages sont présentés dans la figure 3.18 :

- Le jeu de données #a est composé d'un sol et de 2 plans verticaux. Il n'y a pas de variation d'altitude, et le véhicule a une trajectoire rectiligne entre les 2 murs.
- Le jeu de données #b est composé d'un sol et de 5 plans verticaux. Le véhicule mobile a aussi une trajectoire rectiligne, et il n'y a pas de variation d'altitude lors de l'acquisition non plus.
- Le jeu de données #c est le même que le jeu de données #a, à ceci près que le véhicule n'a pas une trajectoire rectiligne cette fois-ci.

- Le jeu de données #d est composé d'un sol et de 4 plans verticaux représentants des façades. Il n'y a pas de variations d'altitude, et le véhicule mobile effectue un virage pendant l'acquisition.
- Les jeux de données #e à #h sont les mêmes que les jeux de données #a à #d, mais avec une variation de l'altitude cette fois-ci.

Les paramètres extrinsèques utilisés pour initialiser l'optimisation sont les mêmes pour l'ensemble des jeux de données simulées : des biais présentés sur la première ligne du tableau 3.6 ont été ajoutés à la vérité terrain, qui est la même pour tous les nuages. Sur le même tableau, les erreurs après optimisation par rapport aux paramètres extrinsèques de la vérité terrain sont données pour chaque nuage. Si l'on s'intéresse au nuage #a, les paramètres de translation ne devraient pas être observables, et l'observabilité pour les paramètres de rotations devrait être correcte : en effet, la structure de l'environnement est du type couloir et le véhicule a une trajectoire rectiligne selon l'axe des x entre les 2 « murs » parallèles ; de plus, il n'y a pas de variation d'altitude, ce qui enlève toute observabilité dans les directions des trois translations. Pour les rotations, comme le capteur Velodyne tourne et que celui-ci est penché sur le toit du véhicule mobile, comme le présente la figure 2.4, on a une observabilité correcte pour chacune des trois rotations. C'est ce que l'on peut voir avec la table 3.6 qui présente les erreurs des paramètres extrinsèques après optimisation par rapport à la vérité terrain, et la table 3.7 qui présente la précision des paramètres extrinsèques avec notre optimisation : l'erreur après optimisation est la même qu'avant optimisation pour les paramètres de translations, et elle est inférieure au degré pour les rotations sauf pour le tangage, qui est la rotation autour de l'axe des y, transverse au déplacement du véhicule. Pour les précisions liées aux paramètres extrinsèques, elles sont présentées dans la table 3.7 et suivent les observations faites plus tôt : pour les translations, les précisions sont mauvaises, alors qu'elles sont correctes pour les rotations. Le jeu de données #b est très proche du jeu de données #a, ce qu'il y a de différent étant une variation de structure pour une des deux façades. La variation de structure devrait ajouter de l'observabilité pour certaines translations : les tableaux d'erreurs et de précisions donnent des résultats similaires et proches de ceux obtenus pour le jeu de données #a. Cela s'explique du fait qu'à l'échelle de l'acquisition, la variation de structure de la façade est assez négligeable.
Le jeu de données #c présente le même environnement que celui présenté avec le jeu de données #a, mais cette fois-ci, le véhicule mobile a une trajectoire non rectiligne. Cela apporte de l'observabilité selon les directions horizontales de translations x et y : la table 3.6 confirme ce résultat, avec des erreurs par rapport à la vérité terrain inférieure au centimètre pour les translations selon les axes x et y ; la translation selon l'axe z n'est toujours pas observable puisque il n'y a pas de variation d'altitude. Pour les rotations, les résultats sont sensiblement meilleurs pour le roulis et le tangage grâce à la variation d'orientation du véhicule pendant l'acquisition. Les précisions pour chacun des paramètres confirment ces observations. Finalement, le jeu de données #d présente une acquisition où le véhicule a une trajectoire non rectiligne, et où l'environnement à une variation de structure bien marquée ; nous ne sommes plus dans une configuration de type couloir, et les résultats d'optimisation sont meilleurs. Les erreurs des paramètres optimisés par rapport à la vérité terrain sont globalement meilleurs que pour le jeu de données #c, toujours sauf pour la translation selon l'axe des z car il n'y a pas de variation d'altitude. Les précision pour le jeu de données #d vont aussi dans le sens de cette observation, avec en général des meilleures précisions que pour le jeu de données #c.

Le jeu de données #e est le même que le jeu de données #a, avec une variation d'altitude en plus. Le résultat attendu est que l'on ajoute de l'observabilité dans la direction de la translation en z avec la variation d'altitude : or, d'après les tableaux d'erreurs et de précisions, ce n'est pas le cas et les résultats sont similaires à ceux obtenus pour le jeu de données #a. Au contraire, pour le jeu de données #f qui est le même que le jeu de données #b avec une variation d'altitude en plus, les 2 tableaux montrent que l'on a ajouté de l'observabilité dans les directions x et z ; il semblerait qu'une structure différente d'un couloir pour le nuage couplée à une variation d'altitude permette d'apporter de l'observabilité dans la direction de la marche du véhicule, ainsi que dans la direction verticale. Par contre, la translation selon l'axe des y n'est toujours pas observable car la trajectoire du véhicule est rectiligne selon l'axe des x : il n'y a pas de variations selon l'axe des y, et

l'environnement est assez proche d'un couloir malgré la variation de structure d'une des 2 façades parallèles. Pour le jeu de données #g, nous avons les mêmes caractéristiques que pour le jeu de données #c, avec une variation d'altitude en plus. L'environnement est toujours de type couloir, mais comme nous avons des variations de trajectoire dans chacune des directions de translations, on s'attend à avoir une observabilité correcte pour les paramètres de translations : c'est le cas, comme présenté avec les tableaux d'erreurs et de précisions. Pour les erreurs, elles sont assez faibles pour chacun des paramètres optimisés, et pour les précisions, nous avons une très bonne valeur pour la translation selon l'axe des x et pour les rotations ; pour les translations selon les axes y et z, la précision est moins bonne, mais les résultats d'optimisations sont corrects avec des erreurs de l'ordre de quelques millimètres après optimisation par rapport à la vérité terrain : cela est normal, parce que l'on travaille avec des données simulées « parfaites », et où seuls des plans sont présents. Sur des données réelles, avec ces mêmes valeurs de précisions, les erreurs pour les paramètres de translations par rapport à une vérité terrain seraient plus élevées. Enfin, le jeu de données #h est le même que le jeu de données #d, avec une variation d'altitude en plus. Comme pour les données sans variation d'altitude, les erreurs sont globalement les plus faibles avec cette optimisation, pour tous les paramètres cette fois-ci. En effet, comme pour le jeu de données #g, et puisque nous avons une variation de la direction d'acquisition dans toutes les directions de translation, aussi parce que la structure de l'environnement n'est pas uniforme comme pour un couloir, les erreurs après optimisation pour le jeu de données #h sont très faibles pour tous les paramètres extrinsèques. Aussi, on peut voir que les précisions sont très bonnes pour l'ensemble des paramètres, ce qui va dans le sens des résultats obtenus sur l'erreur entre les paramètres extrinsèques optimisés et la vérité terrain.

Ce que l'on peut voir en comparant les tables 3.6 3.7 est qu'il y a une forte corrélation entre les résultats d'optimisation pour un nuage et la précision des paramètres extrinsèques que l'on mesure pour ce même nuage. En effet, pour l'ensemble des nuages simulés qui ont servis à effectuer cette étude, lorsque l'erreur d'un paramètre extrinsèque optimisé avec la vérité terrain est très élevée, on remarque que la précision du paramètre associée est très mauvaise. D'un autre côté, lorsque l'erreur entre un paramètre optimisé et la vérité terrain est faible, la précision associée est très bonne. On peut ainsi s'appuyer sur les valeurs de précisions pour valider ou non les valeurs de paramètres optimisés, puisque une valeur de précision faible signifie que l'on est capable avec notre optimisation de retrouver des paramètres correctements optimisés.

En résumé, pour que les paramètres de translations extrinsèques que l'on optimise soient correctement observables, il est nécessaire que :

- le véhicule ne doit pas avoir une trajectoire rectiligne pour qu'il y ait des changements dans les directions de translations horizontales, dans l'idéal avec un ou plusieurs virages, ce qui permet d'apporter de l'observabilité dans ces directions de translations.
- l'altitude varie au cours de l'acquisition ; plus l'altitude varie fortement, plus le paramètre de translations en z sera observable.

Pour les paramètres de rotations, la configuration du système d'acquisition donne déjà une bonne observabilité pour chacun des paramètres. Une combinaison de ces caractéristiques permet d'avoir des paramètres extrinsèques correctement optimisés et proches de la vérité terrain. Ces observations sont confirmées avec les résultats d'optimisation sur des jeux de données réelles présentés en section 3.4.4.2 : pour le nuage #4, qui comporte une variation d'altitude et pour lequel le véhicule mobile a une trajectoire non rectiligne, les précisions sont correctes pour les rotations, et pour les translations, les précisions sont aussi correctes sauf pour la translation en z car la variation d'altitude. Pour le nuage #5, les précisions pour les rotations sont correctes, et pour les translations, les précisions sont plus mauvaises car l'environnement est de type couloir et la variation d'altitude est assez faible.

	t_x(cm)	t_y(cm)	t_z(cm)	roulis $\alpha(^\circ)$	tangage $\beta(^\circ)$	lacet $\gamma(^\circ)$
Erreurs initiales ajoutées à la vérité terrain	-150.00	250.00	-200.00	5.00	-7.00	-5.50
Différence entre la vérité terrain et le résultat d'optimisation pour le nuage #a	**-150.00**	**250.00**	**-200.00**	-0.0465	-4.41	-0.0001
Différence entre la vérité terrain et le résultat d'optimisation pour le nuage #b	**-150.00**	**250.00**	**-200.00**	-0.031	-3.60	-0.0013
Différence entre la vérité terrain et le résultat d'optimisation pour le nuage #c	-0.0374	0.19	**-200.00**	-0.0077	-0.011	0.0004
Différence entre la vérité terrain et le résultat d'optimisation pour le nuage #d	-0.0015	0.025	**-200.00**	-0.0038	0.0027	-0.0016
Différence entre la vérité terrain et le résultat d'optimisation pour le nuage #e	**-150.00**	**250.00**	**-200.00**	-0.0068	-4.49	-0.4515
Différence entre la vérité terrain et le résultat d'optimisation pour le nuage #f	-2.0063	**250.00**	4.2441	-0.0427	0.188	0.0359
Différence entre la vérité terrain et le résultat d'optimisation pour le nuage #g	-0.06	0.0058	0.46	-0.0186	0.015	-0.0014
Différence entre la vérité terrain et le résultat d'optimisation pour le nuage #h	-0.0362	-0.02	0.43	-0.0034	0.0012	0.

TABLE 3.6 – Erreurs sur les paramètres extrinsèques ajoutés à l'ensemble des nuages simulés pour l'étude de l'observabilité

	σ_{t_x}(cm)	σ_{t_y}(cm)	σ_{t_z}(cm)	$\sigma_\alpha(^\circ)$	$\sigma_\beta(^\circ)$	$\sigma_\gamma(^\circ)$
Nuage de points simulé #a	$\mathbf{1.00 * 10^{20}}$	$\mathbf{1.00 * 10^{20}}$	$\mathbf{1.00 * 10^{20}}$	1.56	2.47	0.72
Nuage de points simulé #b	$\mathbf{1.00 * 10^{20}}$	$\mathbf{1.00 * 10^{20}}$	$\mathbf{1.00 * 10^{20}}$	1.48	2.51	0.75
Nuage de points simulé #c	1.99	45.46	$\mathbf{1.00 * 10^{20}}$	0.41	1.08	0.13
Nuage de points simulé #d	2.90	1.50	$\mathbf{1.00 * 10^{20}}$	0.25	0.13	0.15
Nuage de points simulé #e	$\mathbf{1.00 * 10^{20}}$	$\mathbf{1.00 * 10^{20}}$	$\mathbf{1.00 * 10^{20}}$	1.47	2.12	0.67
Nuage de points simulé #f	118.27	$\mathbf{1.00 * 10^{20}}$	91.05	1.34	1.89	0.72
Nuage de points simulé #g	2.27	53.44	345.49	0.62	0.99	0.16
Nuage de points simulé #h	2.88	1.59	16.02	0.22	0.12	0.13

TABLE 3.7 – Précision des paramètres extrinsèques pour les nuages simulés utilisés pour l'étude de l'observabilité

3.6 Conclusion

Dans ce chapitre, nous avons présenté une méthode d'affinement de nuages de points par optimisation des paramètres de calibrage extrinsèque d'un système mobile d'acquisition. La méthode que l'on propose est automatique et n'utilise que des informations extraites des données : nous utilisons la structure des LIDARs multi-couches et la redondance de données entre les couches du capteur pour réaliser notre optimisation. Des résultats d'optimisation sur des jeux de données simulées et réelles ont été présentés, et les affinements ont tous été validés. Nous avons aussi montré que l'utilisation des attributs de dimensionnalité apporte un plus à l'optimisation proposée, notamment du fait que nous effectuons un recalage entre couches du capteur basée sur une distance point-à-plan. L'optimisation que l'on propose est rapide : en effet, les temps de calculs que l'on présente englobent l'ensemble des opérations qui sont effectuées sur les données (lecture, écriture, échantillonnage, optimisation...), et sont de l'ordre de quelques minutes pour plusieurs millions de points. Aussi, notre optimisation est robuste à une mauvaise initialisation des paramètres que l'on corrige : nos tests ont montré que nous n'avons pas besoin d'avoir une initialisation précise des paramètres pour que notre optimisation converge et permette de correctement affiner le nuage de points. Enfin, nous avons aussi traité d'un problème récurrent dans l'optimisation de données qui est l'observabilité des données : dans le cas des paramètres de calibrage extrinsèque, nous avons pu définir les caractéristiques « idéales » (structure du nuage de points et trajectoire du véhicule) qui permettent d'avoir des paramètres extrinsèques correctement optimisés.

Chapitre 4

Affinement de nuages de points par optimisation des paramètres intrinsèques

Sommaire

Introduction

Dans le chapitre 3, nous avons présenté un algorithme d'optimisation des paramètres de calibrage extrinsèque dans le but d'affiner des nuages de points issus de cartographies mobiles. Si on se réfère à la figure 2.6, il s'agit d'une des trois étapes principales au géoréférencement des données lors d'une cartographie avec un véhicule mobile. Dans ce cas, nous avions supposé que les paramètres intrinsèques du système LIDAR étaient corrects, et que la trajectoire du véhicule à tout instant était connue. Cette hypothèse n'est pas forcément vérifiée, notamment concernant le calibrage intrinsèque d'un système LIDAR. En effet, là où le calibrage extrinsèque dépend du système mobile utilisé et de la configuration choisie par « l'utilisateur », le calibrage intrinsèque est plus spécifique, et dépend de la configuration du capteur utilisé pour l'acquisition : il s'agit de correctement récupérer les données brutes, généralement en coordonnées sphériques, et de référencer ces données dans un repère laser

cartésien.
Dans ce chapitre, nous allons présenter un algorithme permettant aussi d'affiner des nuages de points, mais en jouant sur les paramètres de calibrage intrinsèque cette fois-ci ; l'idée est d'aller dans la continuité de ce qui a été présenté au chapitre 3, voire de proposer une amélioration en optimisant un plus grand nombre de paramètres de calibrage pour un système d'acquisition. La solution que l'on présente dans ce chapitre est applicable dans sa méthodologie à l'optimisation des paramètres de calibrage d'un système mobile équipé soit d'un LIDAR multi-couches, soit de plusieurs LIDARs. Nous allons présenter un modèle d'optimisation pour un LIDAR du type Velodyne 32 fibres, mais en adaptant les équations de calibrage intrinsèque au type de système utilisé, l'optimisation est applicable aux mêmes types de systèmes que ceux présentés dans l'introduction du chapitre 3.

4.1 Importance des paramètres de calibrage intrinsèque

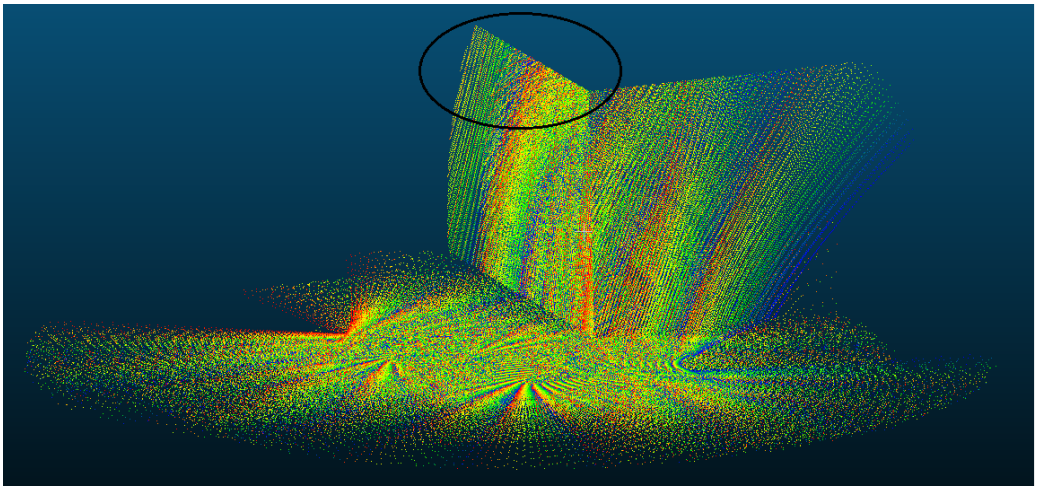
Le calibrage intrinsèque d'un système LIDAR dépend du modèle que le constructeur a choisi pour construire son ou ses capteurs LIDAR. En effet, la configuration « intrinsèque » d'un capteur LIDAR est fixe, et un modèle de calibrage est alors choisi pour représenter au mieux la transformation qui permet d'obtenir des données référencées dans le repère cartésien du capteur à partir des données brutes que le capteur acquiert. Le modèle peut prendre en compte des erreurs potentielles, ou être le plus simple possible : le modèle de calibrage est choisi par l'utilisateur en fonction de la configuration du scanner. Tout comme pour l'optimisation des paramètres de calibrage extrinsèque, l'optimisation des paramètres de calibrage intrinsèque a pour but d'affiner le nuage de point.

4.1.1 Définition du calibrage intrinsèque pour un capteur LIDAR

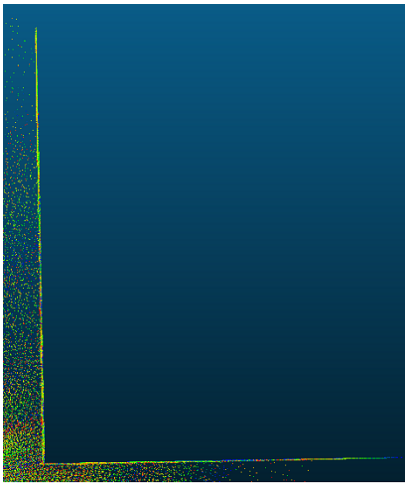
Ce que l'on appelle **calibrage intrinsèque** pour un capteur LIDAR définit la transformation qui permet de référencer des données brutes qui sont en coordonnées sphériques en coordonnées cartésiennes dans le repère lié au capteur. Dans le cas d'un LIDAR mono-fibre, on trouve trois équations qui permettent d'effectuer le calibrage intrinsèque qui sont celles données par l'équation (2.1). Dans le cas d'un capteur multi-fibres, le modèle peut-être le même, que l'on applique à chaque laser du capteur. Ce modèle peut être utilisé par les systèmes LIDARs composés de plusieurs lasers centrés sur un axe vertical et tournants autour de ce même axe, comme le Velodyne 32 fibres par exemple ; pour certains capteurs comme le Velodyne modèle 64 fibres, les lasers sont excentrés et le modèle de calibrage donné par le constructeur est différent de celui-ci, plus complexe car il prend notamment en compte les offsets verticaux et horizontaux de chaque laser, ainsi que les offsets au niveau des rotations. Dans la suite de ce chapitre, pour la construction du modèle corrigé de calibrage intrinsèque que l'on va chercher à optimiser, nous allons détailler les calculs dans le cas du modèle présenté avec l'équation (2.1).

4.1.2 Effets d'un mauvais étalonnage des paramètres intrinsèques sur les données

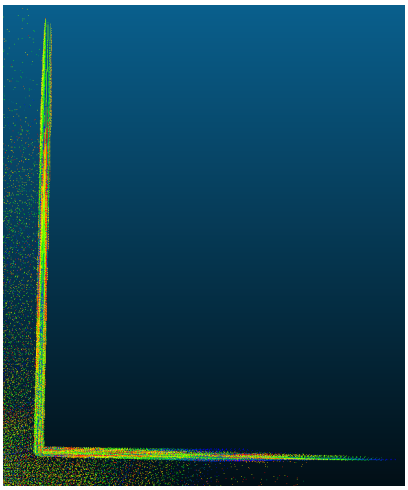
Tout comme pour les paramètres de calibrage extrinsèques, des paramètres de calibrage intrinsèques erronés vont aussi fausser la projection des données dans le repère capteur, ce qui a pour effet d'engendrer un mauvais référencement et une déformation des données dans le repère monde. La figure 4.1 présente le type de déformation que l'on peut obtenir lorsque les paramètres intrinsèques sont erronés pour des fibres d'un capteur multi-fibres : l'erreur est de l'ordre du degré pour les angles et de quelques centimètres pour les distances. La sous-figure a) présente le nuage avec des paramètres de calibrage corrects, et la sous-figure b) un zoom sur un plan vertical du nuage. On voit avec la sous-figure c) les effets d'une erreur sur les paramètres de calibrage, où les plans verticaux sont déformés.



(a)



(b)



(c)

FIGURE 4.1 – Nuages de point avec plusieurs calibrages intrinsèques : a) Paramètres correctement étalonnés ; b) Paramètres corrects, avec zoom sur la zone d'intérêt ; c) Paramètres erronés

Les erreurs sur les paramètres de calibrage intrinsèque proviennent généralement d'un modèle simplifié induit par la structure du système LIDAR. Dans le cas de capteurs multi-fibres comme le Velodyne 32 ou le Quanergy, on a un faisceau de fibres réparties en éventail sur un axe vertical tournant, et un modèle de calibrage simplifié est celui que l'on donne avec l'équation (2.1). Une correction du modèle permet de corriger une bonne partie des erreurs induites par ce modèle : c'est ce que nous allons présenter dans la suite de ce chapitre, tout d'abord en présentant un état de l'art de différentes méthodes d'optimisation de calibrage intrinsèques, puis le modèle de calibrage corrigé avec lequel on cherche à affiner les données issues d'une acquisition mobile, ainsi que différents résultats expérimentaux permettant de valider le modèle corrigé et l'optimisation effectuée, et nous terminerons par présenter certaines limites que nous avons rencontrés au cours des expérimentations effectuées.

4.2 Etat de l'art sur l'optimisation des paramètres de calibrage intrinsèque

Pour le calibrage intrinsèque de systèmes LIDAR, plusieurs algorithmes d'optimisation des paramètres existent dans la littérature. Ils traitent en majorité de l'optimisation des paramètres intrinsèques d'un capteur LIDAR multi-fibres tournant, et le Velodyne 32 ou 64 fibres est à chaque fois utilisé pour les expérimentations et la validation de leur méthode. Les capteurs Velodynes sont apparus en 2007, et ont connu une popularité croissante, notamment grâce à leur introduction dans le Defense Advanced Research Projects Agency (DARPA) Urban Challenge en 2007, où de nombreux véhicules autonomes qui ont réussi à finir la course étaient équipés d'un capteur Velodyne pour analyser l'environnement. Depuis, de nombreuses méthodes d'optimisation des paramètres intrinsèques ont vu le jour ; en 2010, [Glennie et Lichti, 2010] et [Muhammad et Lacroix, 2010] proposent des optimisations des paramètres intrinsèques pour le modèle 64 fibres du Velodyne, en utilisant un environnement de calibrage particulier : de nombreux plans sont présents dans l'environnement, et ces plans sont extraits des nuages de points acquis pour que leurs structures soient contraintes, ce qui a pour effet de corriger les paramètres intrinsèques et de les optimiser. [Levinson et Thrun, 2010] présente aussi une méthode d'optimisation des paramètres intrinsèques, qui est similaire à leur méthode d'optimisation pour les paramètres extrinsèques : l'optimisation est effectuée en post-traitement, sans mire de calibrage, et n'utilise que les données acquises. Ils optimisent les paramètres intrinsèques de leur LIDAR en ajoutant un paramètre qui permet de diminuer la valeur de leur fonctionnelle.
[Atanacio-jiménez *et al.*, 2011] présente une optimisation du calibrage intrinsèque du Velodyne 64 fibres, où différents offsets sont ajoutés au modèle fourni par le constructeur pour le corriger et améliorer la qualité des nuages qui proviendront d'acquisitions futures. L'optimisation est effectuée à l'aide d'un environnement composé de plusieurs plans, dont la structure est optimisée en ajoutant différents offsets aux paramètres intrinsèques. [Mirzaei *et al.*, 2012] présente une méthode similaire pour optimiser les paramètres intrinsèques d'un LIDAR multi-fibres ; un plan d'étalonnage est utilisé, et les impacts du LIDAR sur ce plan selon plusieurs configurations sont mesurés : une mise en correspondance des différentes acquisitions permet de réévaluer les paramètres intrinsèques du LIDAR. Toujours la même année, dans [Chen *et al.*, 2012], une autre méthode d'optimisation des paramètres intrinsèques est proposée, là aussi en cherchant à extraire des plans des données acquises et à les mettre en correspondance d'un point de vue à l'autre pour optimiser le calibrage intrinsèque. La nouveauté est que les auteurs cherchent aussi à avoir une cohérence temporelle pour l'optimisation, et appliquent donc un recalage spatio-temporel sur les plans, ce qui donne de la consistance temporelle aux paramètres intrinsèques optimisés.
En 2013, plusieurs méthodes d'optimisation des paramètres intrinsèques d'un capteur Velodyne ont vu le jour : [Chan et Lichti, 2013] et [Chan *et al.*, 2013] présentent deux méthodes d'optimisation des paramètres intrinsèques du Velodyne 32 fibres, le premier article en extrayant des plans et des cylindres des données, puis en optimisant les paramètres de calibrage pour que ces éléments soient correctement recalés entre eux, et le deuxième article en n'utilisant que des éléments cylindriques des données. Dans [Huang *et al.*, 2013], une optimisation partielle des paramètres intrinsèques d'un Velodyne 64 fibres est proposée, avec l'utilisation d'une caméra infrarouge pour visualiser les impacts lasers sur un plan.
La majorité de ces méthodes d'optimisation sont appliquées en « mode **statique** », et dans une extension des travaux présentés dans [Chan et Lichti, 2013] est proposée dans [Chan et Lichti, 2015], avec l'utilisation de données acquises en mode **cinématique**, en recalant des plans et cylindres extraits des données.

4.3 Méthode d'optimisation proposée

Nous avons déjà traité de l'optimisation des paramètres extrinsèques en supposant que les paramètres intrinsèques du système LIDAR étaient correctement étalonnés, et aussi que la fusion de données GPS + centrale inertielle nous permettait d'avoir un positionnement précis du véhicule à tout instant. Toutefois, cette hypothèse n'est pas tout à fait vérifiée : il est vrai que l'étalonnage intrinsèque donne des valeurs assez précises, mais le modèle donné par le constructeur peut être amélioré, et c'est ce que nous avons voulu proposer. Dans cette section, nous allons traiter de l'optimisation des paramètres intrinsèques liés au Velodyne HDL-32E, mais l'optimisation se veut « généralisable » : en effet, comme nous allons l'expliquer, le modèle d'optimisation corrigé est spécifique à un capteur ou un système LIDAR ; cependant, en adaptant le modèle corrigé au type de capteur, l'optimisation reste la même.
Pour le Velodyne HDL-32E, nous avons choisi un modèle à corriger qui prenait en compte selon nous la plupart des biais qui pouvaient être introduits par le modèle du constructeur. Le modèle corrigé que l'on a choisi d'utiliser va être présenté dans la section suivante 4.3.1.

4.3.1 Algorithme proposé et optimisation

L'idée est d'aller dans la continuité de l'optimisation des paramètres extrinsèques. Pour cela, nous avons repris l'énergie J(R, T) de l'équation (3.6) avec laquelle nous avons optimisé les paramètres extrinsèques. Dans un premier temps, nous supposons que les paramètres extrinsèques sont connus et précis, et que seuls les paramètres de calibrage intrinsèques ont besoin d'être optimisés.

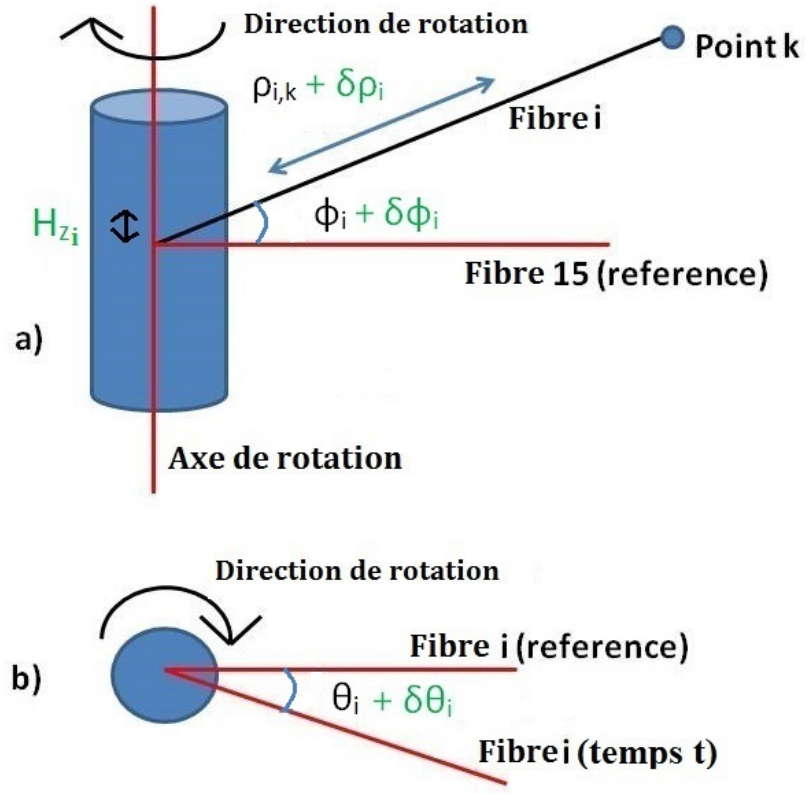


FIGURE 4.2 – Représentation des paramètres intrinsèques pour le capteur Velodyne HDL-32E

La figure 4.2 présente schématiquement la façon dont le capteur Velodyne acquiert les données : le capteur est équipé de 32 fibres placées en éventail sur un plan vertical, mais nous n'en avons représenté que 2 sur la figure a). Aussi, nous y reviendrons plus tard, mais nous considérons la fibre numérotée 15 par le constructeur comme étant une fibre de « référence ». Pour chaque acquisition, le capteur fournit les informations suivantes :

- L'angle vertical ϕ_i de chaque fibre i, par rapport à l'horizontale du capteur.
- La distance $\rho_{i,k}$ entre l'origine de la fibre i et le point acquis k.
- L'angle horizontal θ_i, dû à la rotation du capteur.

Le modèle de calibrage intrinsèque du Velodyne 32 fibres a déjà été présenté dans le chapitre 2 : il s'agit d'un modèle sphérique, puisque l'ensemble des fibres du capteur sont positionnées sur un axe tournant. Les trois équations sont les suivantes :

$$p'_i(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \tag{4.1}$$

Ce modèle est proche de celui présenté par les auteurs dans [Chan et Lichti, 2013] : les auteurs ne corrigent que la distance mesurée et l'angle vertical de chaque fibre, alors que l'on a ajouté une correction sur l'angle horizontal induit par la rotation du scanner et une correction de positionnement vertical de la fibre au sein du scanner. Il s'agit du modèle que l'on veut corriger, pour plusieurs raisons :

- Le modèle suppose qu'entre chaque fibre du capteur, il y a le même écart angulaire vertical, ce qui n'est pas certain à cause de quelques défauts de construction par exemple. Nous ajoutons un offset $\delta\phi_i$ pour chaque angle vertical ϕ_i afin de corriger des écarts angulaires non uniformes.
- Toutes les fibres sont placées sur le même axe vertical, mais le modèle suppose que les fibres sont toutes orientés dans la même direction. Pour les mêmes raisons que précédemment, nous avons ajouté un offset $\delta\theta_i$ pour chaque angle horizontal θ_i.
- Aussi,pour palier à quelques petites erreurs de mesures de distances, notamment lorsqu'elles sont élevées, nous avons un ajouté offset $\delta\rho_i$ par fibre i (et identique pour tous les points acquis par cette fibre) sur les distances $\rho_i(k)$ mesurés entre le centre de la fibre i et le point k.
- Enfin, les fibres sont supposés concentriques, et nous avons ajouté un offset vertical H_z de position pour chaque fibre, afin de prendre en compte des erreurs dues à une origine différente des fibres.

Avec ces différents offsets, les équations (4.1) de calibrage intrinsèque deviennent :

$$p'_i(t) = \begin{pmatrix} (\rho_i(k) + \delta\rho_i) * cos(\theta_i(t) + \delta\theta_i) * cos(\phi_i + \delta\phi_i) \\ -(\rho_i(k) + \delta\rho_i) * sin(\theta_i(t) + \delta\theta_i) * cos(\phi_i + \delta\phi_i) \\ (\rho_i(k) + \delta\rho_i) * sin(\phi_i + \delta\phi_i) + H_{z,i} \end{pmatrix} \tag{4.2}$$

4.3.1.1 Approximation linéaire

Les équations de calibrage intrinsèque ne sont pas linéaires, et pour simplifier notre problème d'optimisation, nous effectuons une linéarisation au 1^{er} ordre, ce qui donne des équations de calibrage linéarisées pour le point k acquis par la fibre i :

$$\begin{aligned} p'_{i,k}(t) = p'_{1,i,k}(t) + p'_{2,i,k}(t) * \delta\rho_i + p'_{3,i,k}(t) * \delta\theta_i \\ + p'_{4,i,k}(t) * \delta\phi_i + p'_{5,i} \end{aligned} \tag{4.3}$$

avec :

$$\begin{cases} p'_{1,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \\ p'_{2,i,k}(t) = \begin{pmatrix} cos(\theta_i(t)) * cos(\phi_i) \\ -sin(\theta_i(t)) * cos(\phi_i) \\ sin(\phi_i) \end{pmatrix} \\ p'_{3,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t) + 90) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t) + 90) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \\ p'_{4,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i + 90) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i + 90) \\ \rho_i(k) * sin(\phi_i + 90) \end{pmatrix} \\ p'_{5,i} = \begin{pmatrix} 0 \\ 0 \\ H_{z,i} \end{pmatrix} \end{cases}$$

$p'_{i,k}(t)$ défini dans l'équation (4.3) est ensuite réinjecté dans la fonctionnelle J définie dans (3.6), et il ne reste plus qu'à minimiser la fonctionnelle pour trouver les différents offsets optimaux.

4.3.1.2 Approche d'optimisation

Pour l'optimisation des offsets que nous avons ajoutés au modèle de calibrage, nous avons choisi de prendre la fibre numérotée 15 comme référence pour plusieurs raisons :

- Tout d'abord, le problème d'optimisation que l'on a est sous-contraint, et une optimisation de l'ensemble des offsets ajoutés aux fibres du Velodyne ne convergerait pas vers une solution optimale puisque ces mêmes offsets peuvent se compenser d'une fibre à l'autre.
- D'autre part, lorsque l'on regarde la datasheet du Velodyne 32 fibres, la structure du Velodyne y est détaillée : les fibres sont numérotées de 0 à 31 en fonction de l'ordre de tir lors d'une acquisition, et l'angle vertical de la fibre 15 est de 0°. Par conséquent, cette fibre est la plus à même d'être la « plus correctement » placée dans le capteur, et le problème d'optimisation devient alors un recalage des nuages issus des autres fibres les unes par rapport aux autres, tout en supposant que le nuage issu de la fibre 15 est optimal.

Au total, il nous reste $4 * 31 = 124$ paramètres à optimiser. Dans l'équation (3.6), ces offsets apparaissent aussi bien avec les termes $p'_{i,k}$ que $m'_{j,k}$. Comme pour l'optimisation des paramètres extrinsèques, nous pouvons réécrire l'énergie J de la façon suivante :

$$J(\delta X_{int}) = \frac{\sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_k w_{i,j,k} * (B_{i,j,k} + A^T_{i,j,k} \times \delta X_{int} + o(\delta X_{int}))^2}{\sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_k w_{i,j,k}} \tag{4.4}$$

avec :

$$\begin{cases} \delta X_{int} = ([\delta\rho_i \quad \delta\theta_i \quad \delta\phi_i \quad H_{z,i}]_{i\in[\![0,31]\!]\setminus 15})^T \\ \\ B_{i,j,k} = n_{i,k}^T \times \begin{pmatrix} T_{nav}(p'_{i,k}) - T_{nav}(m'_{j,k}) \\ + \left(R_{nav}(p'_{i,k}) \times (R(\alpha,\beta,\gamma) \times p'_{1,i,k} + T(t_x,t_y,t_z))\right) \\ - \left(R_{nav}(m'_{j,k}) \times (R(\alpha,\beta,\gamma) \times m'_{1,j,k} + T(t_x,t_y,t_z))\right) \end{pmatrix} \\ \\ A_{i,j,k} = \begin{pmatrix} \dots \\ n_{i,k}^T \times \left(R_{nav}(p'_{i,k}) \times R(\alpha,\beta,\gamma) \times p'_{2,i,k}\right) \\ n_{i,k}^T \times \left(R_{nav}(p'_{i,k}) \times R(\alpha,\beta,\gamma) \times p'_{3,i,k}\right) \\ n_{i,k}^T \times \left(R_{nav}(p'_{i,k}) \times R(\alpha,\beta,\gamma) \times p'_{4,i,k}\right) \\ n_{i,k}^T \times \left(R_{nav}(p'_{i,k}) \times R(\alpha,\beta,\gamma) \times \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T\right) \\ \dots \\ n_{i,k}^T \times \left(R_{nav}(m'_{j,k}) \times R(\alpha,\beta,\gamma) \times m'_{2,j,k}\right) \\ n_{i,k}^T \times \left(R_{nav}(m'_{j,k}) \times R(\alpha,\beta,\gamma) \times m'_{3,j,k}\right) \\ n_{i,k}^T \times \left(R_{nav}(m'_{j,k}) \times R(\alpha,\beta,\gamma) \times m'_{4,j,k}\right) \\ n_{i,k}^T \times \left(R_{nav}(m'_{j,k}) \times R(\alpha,\beta,\gamma) \times \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T\right) \\ \dots \end{pmatrix} \\ \\ i \text{ et } j \neq 15 \end{cases}$$

$A_{i,j,k}$ est un vecteur rempli de 0, sauf pour les quelques éléments présentés en équation 4.4. Le vecteur des offsets qui minimise la fonction objectif (4.4) est la solution du système linéaire :

$$C_{int} \times \delta X_{int} = -V_{int} \tag{4.5}$$

avec :

$$\begin{cases} C_{int} = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_k w_{i,j,k} * (A_{i,j,k} \times A_{i,j,k}^T) \\ V_{int} = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_k w_{i,j,k} * (B_{i,j,k} \times A_{i,j,k}) \end{cases}$$

Pour trouver les offsets de calibrage optimaux, et comme nous avons linéarisé certains paramètres, nous calculons δX de façon itérative, jusqu'à convergence des offsets. A chaque nouvelle itération n+1, les paramètres de calibrage intrinsèque sont définis comme étant corrigés par les offsets calculés à l'itération précédente :

$$X_{n+1} = X_n + \delta X_n$$

Pour démarrer l'optimisation, nous partons d'offsets nuls car le but de l'optimisation est de calculer et ajouter des offsets aux paramètres de calibrage intrinsèque pour affiner le nuage de points : ces offsets n'existent pas avant l'optimisation. Il est toutefois possible de rajouter quelques offsets pour « déformer » le nuage de points et sortir d'un minimum local : en effet, bien qu'il soit possible d'améliorer le modèle donné par le constructeur, celui-ci est assez précis et certains tests ont montré qu'il valait mieux ajouter des offsets peu élevés au démarrage de l'optimisation pour que la convergence soit optimale.
L'algorithme complet d'optimisation des paramètres de calibrage intrinsèque est similaire à l'algorithme 1 pour l'optimisation des paramètres extrinsèques : ce qui change entre ces deux optimisations sont les paramètres que l'on optimise. Le critère d'arrêt est le même : l'optimisation s'arrête lorsque $\|\delta X\|_{max}$ est inférieur à un seuil δ, ou dans certains cas si le nombre d'itérations est trop élevé. Ces paramètres δ et n_max sont les mêmes que pour l'optimisation des paramètres extrinsèques.

4.3.2 Validation du résultat de l'optimisation

La validation du résultat d'optimisation est la même que celle présentée dans la section 3.3.3, puisque nous utilisons la même fonctionnelle pour l'optimisation, et que la procédure d'optimisation est aussi similaire.

Aussi, nous avons défini une métrique d'erreur pour chaque type d'offset de paramètres intrinsèques :

$$\begin{cases} \Delta\rho = \sqrt{\frac{1}{31} * \sum_{i=0/i\neq 15}^{31} (\delta\rho_i)^2} \\ \Delta\theta = \sqrt{\frac{1}{31} * \sum_{i=0/i\neq 15}^{31} (\delta\theta_i)^2} \\ \Delta\phi = \sqrt{\frac{1}{31} * \sum_{i=0/i\neq 15}^{31} (\delta\phi_i)^2} \\ \Delta H_z = \sqrt{\frac{1}{31} * \sum_{i=0/i\neq 15}^{31} (\delta H_{z,i})^2} \end{cases} \tag{4.6}$$

Dans le cas où la vérité terrain est connue et que l'on déforme le nuage, on peut comparer ces erreurs avant et après optimisation, et le résultat espéré est une erreur faible pour chaque type de paramètre intrinsèque. Dans le cas où la vérité terrain n'est pas connue, la validation se fait d'une part avec la valeur finale de l'énergie J, mais aussi avec le fait que ces erreurs doivent être faibles : en effet, on ajoute des offsets sur les paramètres de calibrage intrinsèque, mais le modèle du constructeur étant assez proche de la réalité, les offsets ne doivent pas être très grands.

4.4 Résultats expérimentaux

Cette optimisation a été codée en C++ et utilise les mêmes librairies EIGEN et FLANN que pour l'optimisation des paramètres extrinsèques. Pour les paramètres à régler dans notre optimisation, les mêmes que ceux définis en section 3.4.2 ont été utilisés afin d'aller dans la continuité de l'optimisation des paramètres de calibrage du système mobile.

Enfin, au niveau des paramètres intrinsèques avec lesquels nous initialisons l'algorithme, il y a 2 cas de figure :

- les données sont simulées ; nous avons alors une vérité terrain des paramètres intrinsèques, et une erreur est ajoutée pour chacun des 4 paramètres à optimiser de chaque fibre, sauf pour la fibre référence pour laquelle on considère que les paramètres sont optimaux. Le but de l'optimisation est alors d'obtenir des offsets qui donnent des paramètres intrinsèques corrigés aussi proches que possible de ceux donnés par la vérité terrain.
- les données sont issues d'acquisitions réelles ; dans ce cas là, nous n'avons pas accès à la vérité terrain. Pour tester la robustesse de notre optimisation, une erreur est aussi ajoutée à tous les paramètres de chaque fibre sauf pour la fibre référence du Velodyne : en effet, pour certains tests d'optimisation, aucune différence entre le nuage avant et après optimisation n'était visible. Cela nous permettait ainsi de tester notre optimisation dans le cas où les paramètres intrinsèques étaient « trop » erronés.

Dans la suite de cette section, nous allons présenter 4 résultats d'optimisation des paramètres intrinsèques : les résultats concernent les nuages simulés #2 et #3, et les nuages réels #4 et #5 introduits en section 3.4.1.

4.4.1 Résultats sur des jeux de données simulées

Pour les tests d'optimisation sur des jeux de données simulées, nous avons une vérité terrain, qui est composée des paramètres de calibrage du nuage de point. Nous avons ajouté des erreurs

arbitraires à ces valeurs de paramètres intrinsèques, et le but de l'optimisation est d'obtenir au final des offsets les plus proches de 0. Les erreurs ajoutées aux paramètres intrinsèques sont de l'ordre de -3° à 3° pour les paramètres angulaires ϕ et θ, et entre -10 cm et 10 cm pour les paramètres de distance et de translation ρ et H_z : cela a aussi permis de tester la robustesse de notre optimisation à une initialisation trop mauvaise.

Le premier nuage simulé est le nuage #2 présenté en figure 4.3. L'évolution de l'énergie est présentée avec la figure 4.4, et on peut voir que l'énergie n'est pas strictement décroissante : ce n'est pas surprenant car nous utilisons une approche de type recalage de données ; or, dans [Besl et McKay, 1992], l'énergie mesurée à chaque itération du recalage est strictement décroissante dans le cas où le facteur de normalisation est le même à chaque itération, et cela a été prouvé. Dans notre cas, nous autorisons de plus en plus de points appariés à être pris en compte, ce qui a pour effet d'augmenter le poids total utilisé pour la normalisation à chaque itération, et ce qui justifie que l'énergie augmente à certaines itérations, tout en convergeant vers une valeur minimale en fin d'optimisation. L'énergie varie d'une valeur de 265.13 cm^2 à une valeur de 0.45 cm^2 avec l'optimisation. La table 4.1 donne les erreurs définies en section 4.3.2 pour chaque catégorie de paramètres : on peut voir que pour l'ensemble des offsets concernés, nous avons des valeurs très proche de 0, ce qui permet de valider le résultat d'optimisation.

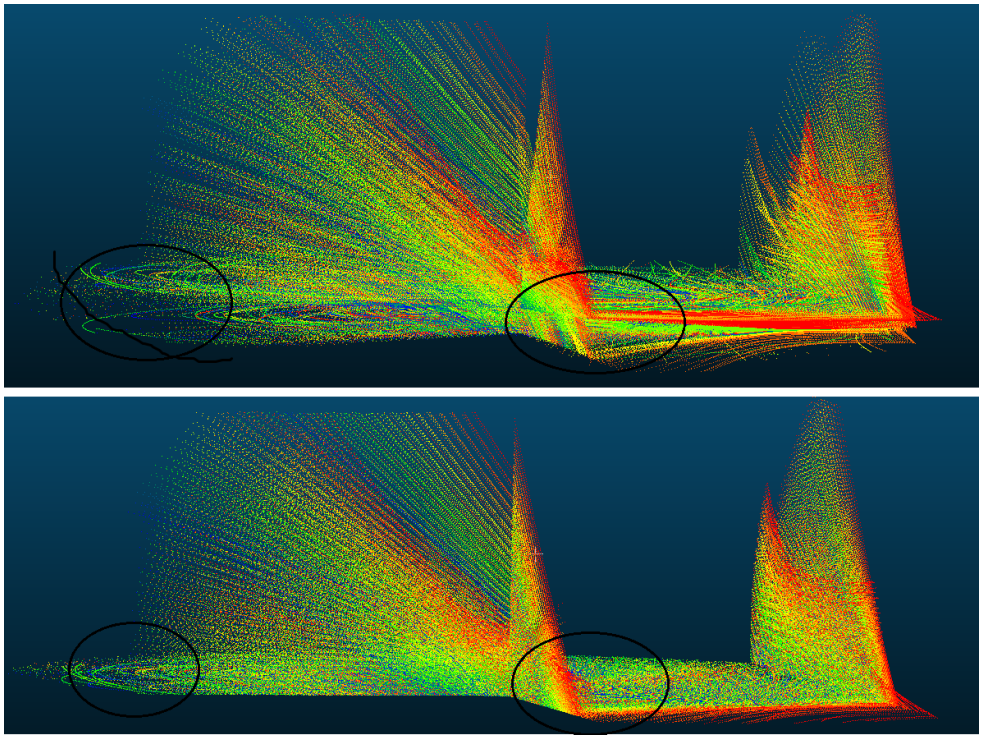


FIGURE 4.3 – Nuage simulé #2 : en haut, nuage avant optimisation des paramètres intrinsèques ; en dessous, le même nuage après optimisation des paramètres intrinsèques. Les deux images sont vues avec la même orientation.

	$\Delta\rho(cm)$	$\Delta\theta(^\circ)$	$\Delta\phi(^\circ)$	$\Delta H_z(cm)$
Erreurs initiales	10	2.5	3	10
Erreurs après notre optimisation	$1.52 * 10^{-2}$	$1.38 * 10^{-3}$	$7.81 * 10^{-4}$	$1.19 * 10^{-2}$

TABLE 4.1 – Erreurs entre les offsets et la vérité terrain pour le nuage simulé #2

Pour le nuage #3, les observations sont les mêmes : le résultat d'optimisation est similaire à

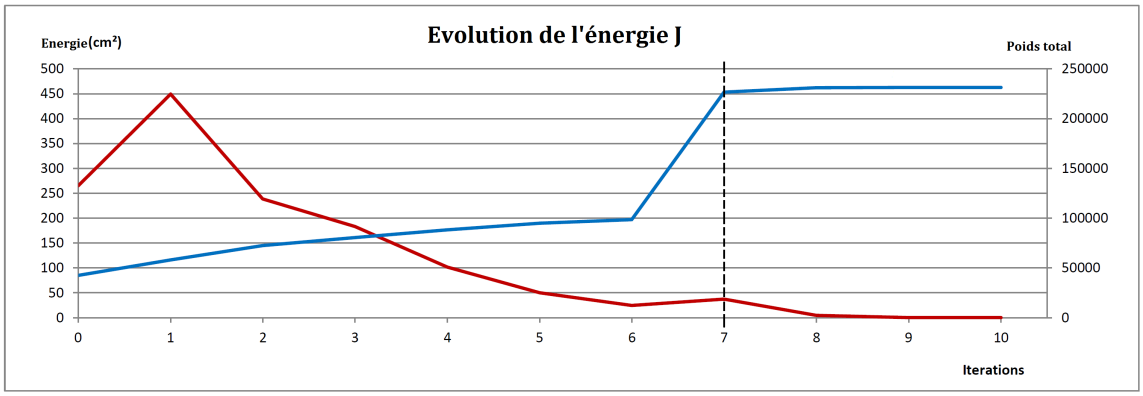


FIGURE 4.4 – Évolution de l'énergie pour le nuage simulé #2 : en rouge, nous avons la courbe d'énergie en fonction des itérations ; en bleu, nous avons la courbe représentant le poids total utilisé pour normaliser l'énergie ; les lignes noires verticales mettent en évidence les itérations où les attributs de dimensionnalité sont mis à jour.

celui présenté avec le nuage #2, et plus de détails sont donnés en Annexe D.

Enfin, concernant les temps de calculs pour les optimisations des nuages #2 et #3, ils sont respectivement de 9 minutes et 5 minutes, ce qui est acceptable au vu du nombre de paramètres optimisés.

4.4.2 Résultats sur des jeux de données réelles

Dans cette section, nous présentons des résultats d'optimisation sur 2 jeux de données réelles. Cette fois-ci, nous n'avons pas de vérité terrain comme pour les jeux de données simulées : nous partons alors du modèle donné par le constructeur en ajoutant les mêmes offsets que pour les jeux de données simulées, et les résultats attendus sont des offsets plus faibles, qui donnent une énergie plus petite et un nuage correctement affiné.

La figure 4.6 montre les améliorations après optimisation des offsets intrinsèques pour le nuage #4 : en haut, nous présentons le nuage avec des offsets intrinsèques initiaux, et la figure du bas présente le même nuage avec des offset optimisés, où le nuage est correctement affiné. La figure 4.5 présente l'évolution de l'énergie au cours de l'optimisation pour le nuage #4 : elle évolue d'une valeur de 278.47 cm^2 à une valeur de 33.40 cm^2, ce qui montre que l'on affine correctement le nuage de points ; aussi, la valeur de l'énergie est validée par notre critère défini en section 3.3.3. On peut remarquer sur cette figure que l'énergie d'optimisation commence par diminuer, puis augmente jusqu'à la mise à jour des attributs de dimensionnalité, avant de diminuer à nouveau jusqu'à convergence ; ce résultat peut-être expliqué du fait qu'entre les itérations 1 à 7, nous utilisons les mêmes valeurs d'attributs de dimensionnalité à chaque itération. Ces attibuts sont calculés à l'aide des normales estimées en chaque point du nuage de point, et à chaque itération, les paramètres de calibrages sont corrigés, ce qui modifie la structure du nuage, ce qui implique que les attributs de dimensionnalité devraient changer. Or, pour des soucis de temps de calculs, nous ne les mettons à jour que toutes les 7 itérations, ce qui explique cette augmentation de l'énergie jusqu'à la première mise à jour des attributs.

Pour le nuage réel #5, nous avons les mêmes observations que pour le nuage #4 : les résultats sont similaires à ceux présentés pour le jeu de données #4, et sont plus détaillés en Annexe D.

Pour les temps de calculs des optimisations des jeux de données réelles, nous avons respectivement 25 minutes pour le nuage #4 et 12 minutes pour le nuage #5, ce qui est acceptable au vu de la taille des nuages et de la quantité de paramètres optimisés.

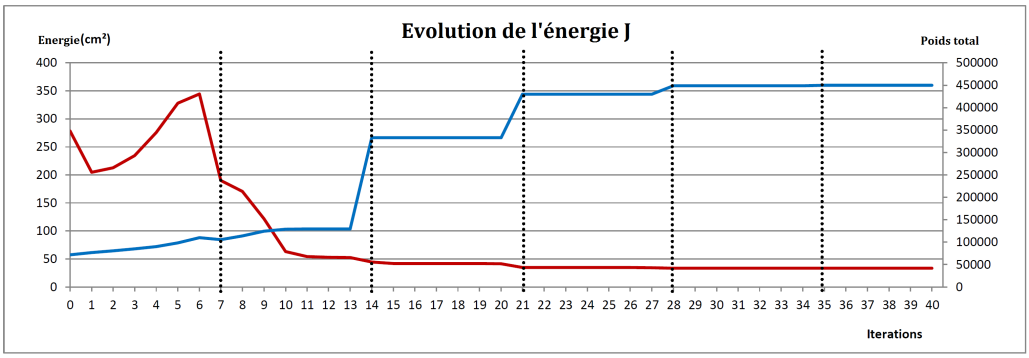


FIGURE 4.5 – Évolution de l'énergie pour le nuage réel #4 : en rouge, nous avons la courbe d'énergie en fonction des itérations ; en bleu, nous avons la courbe représentant le poids total utilisé pour normaliser l'énergie ; les lignes noires verticales mettent en évidence les itérations où les attributs de dimensionnalité sont mis à jour.

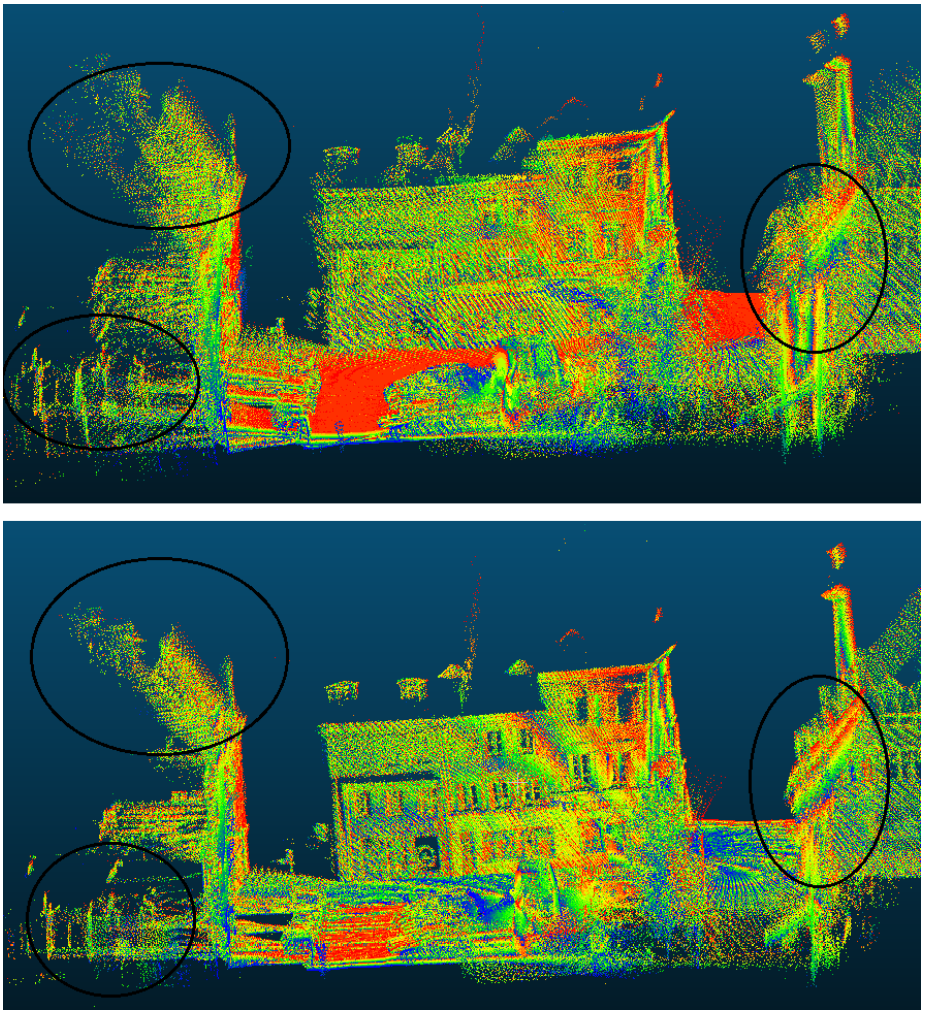


FIGURE 4.6 – Nuage réel #4 : en haut, nuage avant optimisation des paramètres intrinsèques ; en dessous, le même nuage après optimisation des paramètres intrinsèques. Les deux images sont vues depuis le même point.

4.5 Optimisation conjointe avec les paramètres extrinsèques

Dans le chapitre 3, nous avons présenté une optimisation des paramètres extrinsèques en faisant la supposition que les paramètres intrinsèques étaient correctement estimés. Dans ce chapitre, nous avons présenté l'optimisation du modèle de calibrage intrinsèque en supposant que cette fois-ci, il s'agissait des paramètres extrinsèques qui étaient correctement étalonnés. Les deux optimisations utilisent la même fonctionnelle à minimiser, et l'algorithme est le même : nous avons présenté des résultats d'optimisations avec des paramètres égaux. Toutefois, il n'est pas toujours possible de faire ces hypothèses, et en général, les deux étalonnages ne sont pas optimaux. Dans cette section, nous allons proposer une optimisation conjointe des paramètres de calibrage extrinsèque et intrinsèque.

4.5.1 Changements sur la résolution numérique

Pour l'optimisation conjointe des paramètres de calibrage, nous avons repris la fonctionnelle (3.6), et les paramètres à optimiser sont ceux définis dans les sections 3.3.1 et 4.3.1. Nous effectuons aussi une linéarisation au premier ordre, et cela conduit au système linéaire suivant à résoudre :

$$C_{tot} \times \delta X_{tot} = -V_{tot} \tag{4.7}$$

avec :

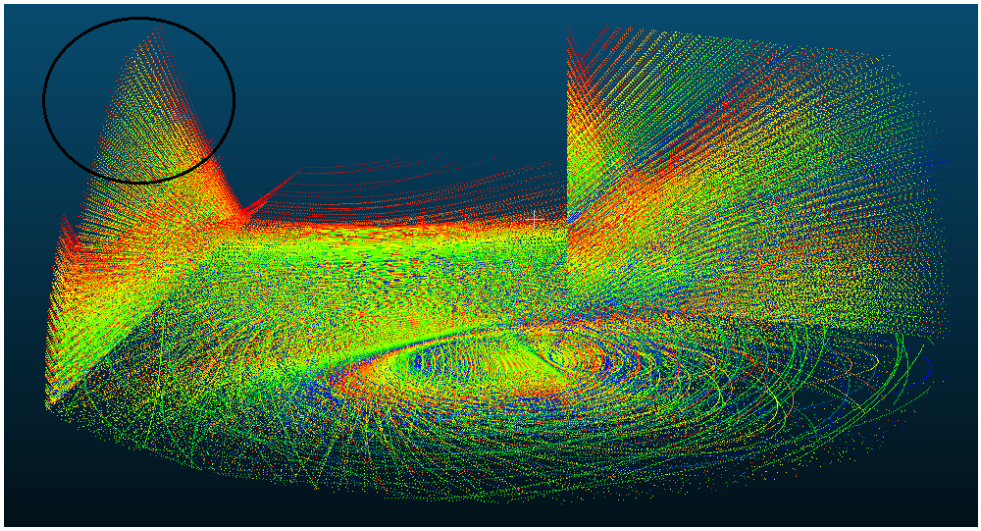
$$\begin{cases} \delta X_{tot} = [\delta X_{int} \quad \delta X_{ext}]^T \\ C_{tot} = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * (E_{i,j,k} \times E_{i,j,k}^T) \\ E_{i,j,k} = [A_{i,j,k}^T \quad C_{i,j,k}^T]^T \\ V_{tot} = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * (B_{i,j,k} \times E_{i,j,k}) \end{cases}$$

Les paramètres intrinsèques et extrinsèques à optimiser sont concaténés en un seul vecteur d'inconnues, qui est itérativement calculé comme cela a été présenté plus tôt.
$C_{i,j,k}$ est le vecteur introduit en équation 3.7. Le terme constant $B_{i,j,k}$ est égal au terme $D_{i,j,k}$ lui aussi introduit en équation 3.7.

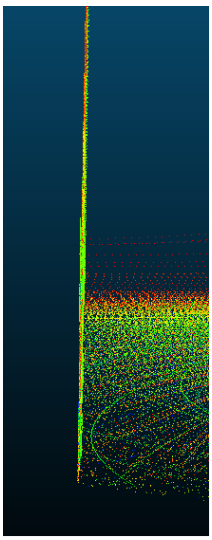
4.5.2 Comparaison des approches d'optimisations successives et d'optimisation conjointe des paramètres intrinsèques et extrinsèques

Dans un premier temps, nous avons comparé des optimisations successives des paramètres extrinsèques et intrinsèques, dans les deux sens, c'est à dire en effectuant l'optimisation extrinsèque suivie de l'optimisation intrinsèque dans un premier temps, et inversement ensuite. Pour comparer ces approches d'optimisation des paramètres de calibrage, nous avons pris les 2 jeux de données simulées #2 et #3, présentés en section 3.4.

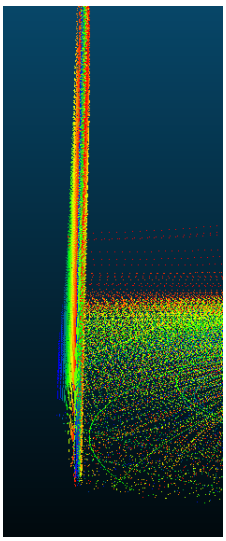
La figure 4.7 présente le résultat d'optimisation avec les 3 approches testées, lorsque au départ les paramètres extrinsèques et intrinsèques sont erronés. Les erreurs sur les paramètres de calibrage extrinsèque sont de l'ordre de 50 cm pour les translations selon les axes x et y, et de 80 cm selon l'axe z, et des erreurs de respectivement 2.5, 3 et -2° pour les rotations roulis, tangage et lacet. Pour les erreurs des paramètres intrinsèques, elles sont de l'ordre de 0.3° pour les paramètres liés aux rotations, et d'environ 3 cm pour les paramètres de distance et d'offset vertical.



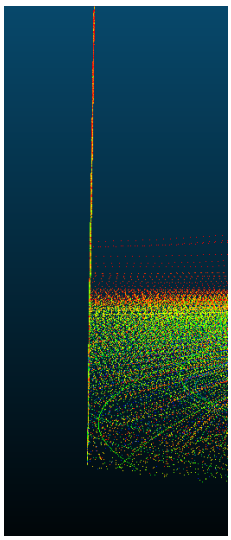
(a)



(b)



(c)



(d)

FIGURE 4.7 – Nuages de point #2 avec plusieurs optimisations des paramètres de calibrage : a) Paramètres corrects ; b) Optimisation extrinsèque puis intrinsèque ; c) Optimisation intrinsèque puis extrinsèque ; d) Optimisation conjointe des paramètres de calibrage

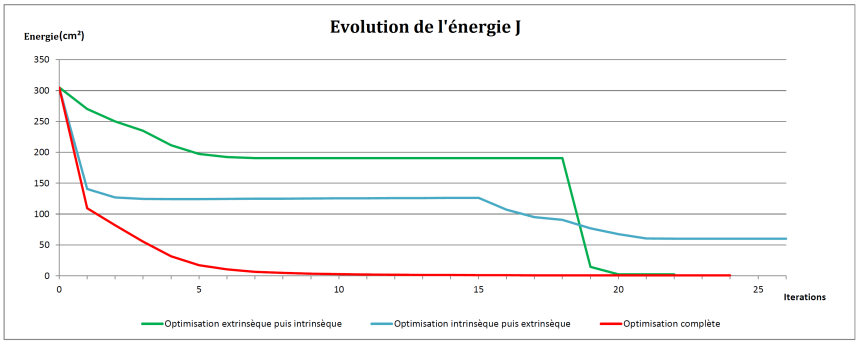
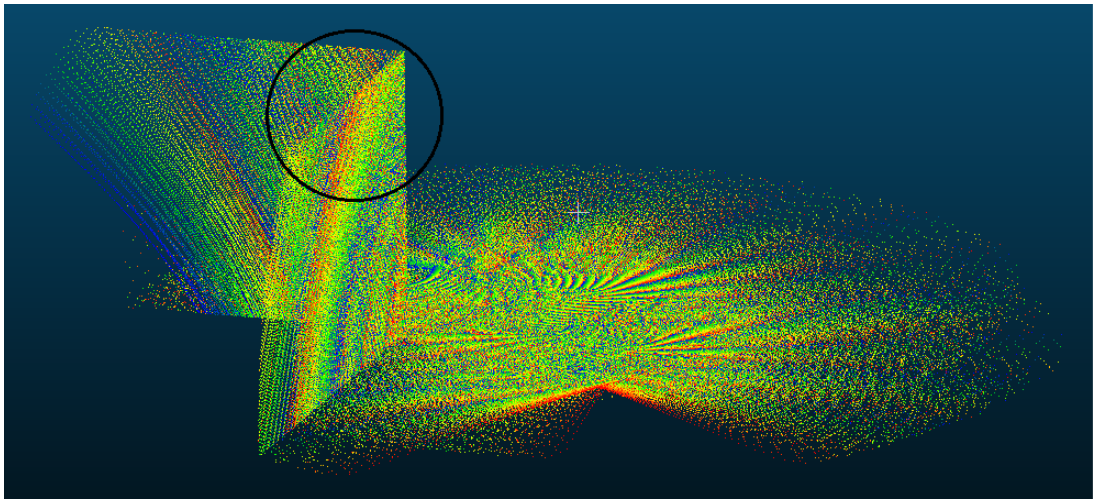
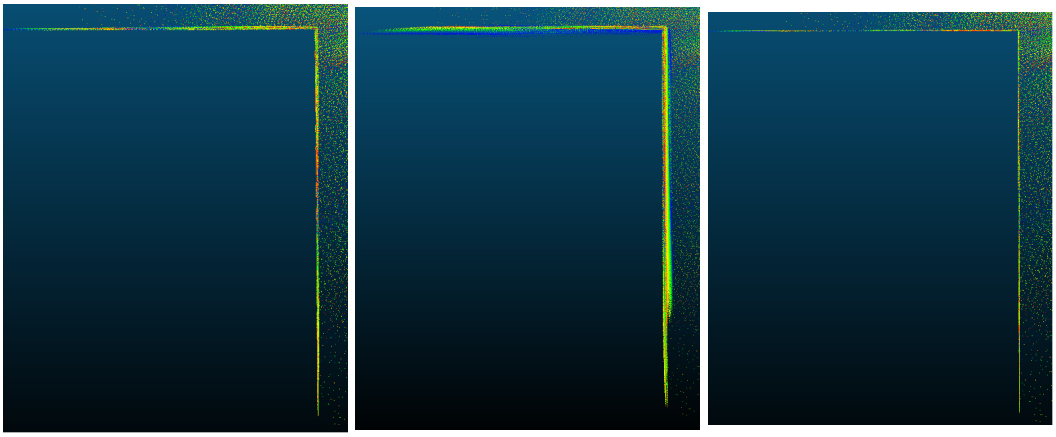


FIGURE 4.8 – Comparaison des énergies avec les 3 approches d'optimisations pour le nuage #2

Les trois images de la figure 4.7 présentent les résultats avec la même orientation des jeux de données par rapport au nuage a), et on peut voir les problèmes d'optimisation assez distinctement. Effectuer l'optimisation des paramètres extrinsèques puis intrinsèques donne un nuage mieux affiné que dans l'autre sens notamment parce que les erreurs sur les paramètres extrinsèques sont plus importantes que sur les paramètres intrinsèques. Toutefois, c'est avec l'optimisation conjointe que l'on a le meilleur affinement du nuage. La figure 4.8 confirme cette observation : on peut voir les 3 courbes d'énergies, selon l'optimisation effectuée. Pour les deux approches successives, les « cassures » représentent le changement d'optimisation. On peut remarquer que l'énergie est la plus faible pour l'optimisation conjointe des paramètres de calibrage, et que l'énergie est la plus élevée pour l'optimisation successive des paramètres intrinsèques puis extrinsèques, ce qui va dans le sens de l'observation faite plus tôt. La convergence est aussi nettement plus rapide avec l'optimisation conjointe.



(a)



(b) (c) (d)

FIGURE 4.9 – Nuages de point #3 avec plusieurs optimisations des paramètres de calibrage : a) Paramètres corrects ; b) Optimisation extrinsèque puis intrinsèque ; c) Optimisation intrinsèque puis extrinsèque ; d) Optimisation conjointe des paramètres de calibrage

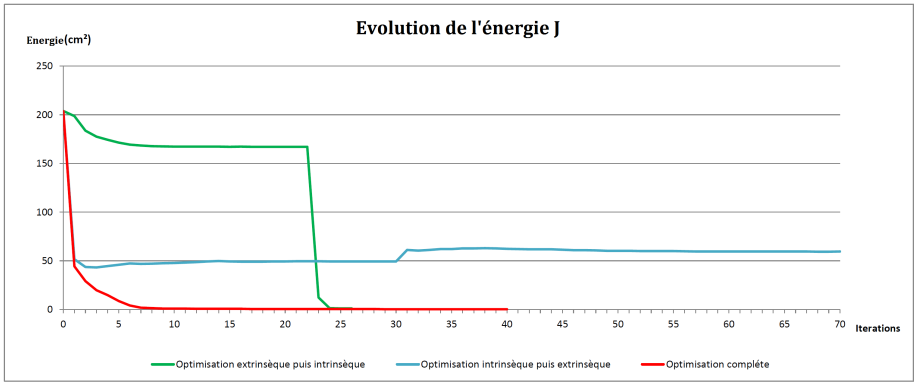


FIGURE 4.10 – Comparaison des énergies avec les 3 approches d'optimisations pour le nuage #3

Pour confirmer les observations faîtes avec le nuage #2, nous avons aussi effectué ces comparaisons avec le nuage #3. Là aussi, nous avons les mêmes erreurs sur les paramètres de calibrage que pour le nuage #2, et les observations sont exactement les mêmes : l'optimisation conjointe donne les meilleurs résultats d'optimisation, et l'optimisation successive des paramètres intrinsèques puis extrinsèques donne les plus mauvais résultats d'optimisation, comme le montre les figures 4.9 et 4.10.

4.5.3 Quelques résultats

Pour les paramètres à régler dans notre optimisation, nous avons utilisé les mêmes que ceux définis en sections 3.4.2 et 4.4. Les jeux de données utilisés sont aussi les mêmes, à savoir les nuages simulés #2 et #3, et les jeux de données réelles #4 et #5. Enfin, ce que l'on va chercher à mettre en évidence dans cette section est l'amélioration des résultats d'optimisation lorsque les paramètres intrinsèques et extrinsèques sont optimisés conjointement, par rapport à l'optimisation des paramètres extrinsèques seuls ; dans le cas des jeux de données simulées, les optimisations sont initialisées avec des paramètres extrinsèques et intrinsèques erronés ; pour les jeux de données réelles, seuls les paramètres extrinsèques sont erronés en début d'optimisation.

4.5.3.1 Résultats sur les jeux de données simulées

Pour les tests d'optimisation sur des jeux de données simulées, nous avons une vérité terrain, qui est composée des paramètres de calibrage du nuage de points. Nous avons ajouté des erreurs arbitraires aux paramètres de calibrage extrinsèque et intrinsèque, et comparé les résultats d'optimisations avec la vérité terrain, avec l'optimisation conjointe des paramètres de calibrage extrinsèque et intrinsèque, et l'optimisation des paramètres extrinsèques seuls. Les erreurs ajoutées aux paramètres intrinsèques sont de l'ordre de -0.3° à 0.3° pour les paramètres angulaires ϕ et θ, et entre -3 cm et 3 cm pour les paramètres de distance et de translation ρ et H_z. Pour les paramètres extrinsèques, les erreurs sont de l'ordre du mètre pour les paramètres de translations, et de environ 2-3° pour les paramètres de rotations.

Le premier test a été effectué sur le nuage simulé #2, et la figure 4.11 donne l'évolution des énergies pour les deux optimisations concernées. Avec l'optimisation des paramètres extrinsèques seuls, c'est à dire que les paramètres intrinsèques initiaux qui sont erronés ne sont pas corrigés, l'énergie varie d'une valeur de 261.69 cm^2 à une valeur de 178.48 cm^2, et pour l'optimisation conjointe des paramètres de calibrage intrinsèque et extrinsèque, l'énergie évolue aussi d'une valeur de 261.69 cm^2 à une valeur de 0.597 cm^2 cette fois-ci. L'énergie est considérablement plus petite avec l'optimisation conjointe des paramètres extrinsèques et intrinsèques, ce qui est le résultat auquel on s'attendait.

La figure 4.12 montre les résultats d'optimisations sur le nuage #2 : en haut, nous avons le nuage calibré avec la vérité terrain, et en bas à gauche le résultat après optimisation des paramètres extrinsèques seuls : nous voyons qu'il y a quelques problèmes d'affinement du nuage, avec notamment une surface planaire qui n'est pas complètement planaire à cause de la non-correction des paramètres intrinsèques. En bas à droite, avec l'optimisation de l'ensemble des paramètres de calibrage, nous voyons que le nuage est correctement affiné : cette observation rejoint le résultat montré par l'évolution des énergies d'optimisations. La table 4.2 donne les erreurs entre les paramètres de calibrage du nuage et la vérité terrain : concernant les paramètres extrinsèques, ce que l'on remarque est que contrairement à ce que l'on pouvait attendre, les meilleurs résultats d'optimisation sont obtenus lorsque les paramètres extrinsèques sont considérés seuls, notamment pour les paramètres de translation. Pour les paramètres de rotations, l'optimisation conjointe des paramètres de calibrage donne de meilleurs résultats. D'un autre côté, concernant les paramètres intrinsèques, la table donne les erreurs que nous avons défini en section 4.3.2 pour mesurer le résultat d'optimisation et le comparer à la vérité terrain. Ce que l'on peut voir est que pour l'optimisation des paramètres extrinsèques seuls, ces erreurs ne changent pas après optimisation, ce qui est logique vu qu'on ne les optimise pas ; au contraire, pour l'optimisation conjointe des paramètres extrinsèques et intrinsèques, ces erreurs sont très faibles après optimisation. Ces observations permettent de justifier le résultat que l'on a vu pour les paramètres extrinsèques, qui est que les paramètres de translations sont mieux optimisés lorsque les paramètres extrinsèques sont considérés seuls pour l'optimisation : lorsque l'optimisation est faite conjointement pour l'ensemble des paramètres, les paramètres extrinsèques et intrinsèques sont optimisés « en même temps », et le résultat optimal qui donne le nuage le mieux affiné possible est celui présenté dans la table 4.2 ; les valeurs d'énergies finales entre les deux optimisations confirment ce résultat.
Concernant les temps de calcul, nous avons un temps de calcul d'environ 5 minutes pour l'optimisation des paramètres extrinsèques seuls, et de 20 minutes pour l'optimisation conjointe des paramètres extrinsèques et intrinsèques : le temps de calcul est plus long, mais acceptable au vu du nombre de paramètres considéré et du résultat présenté.

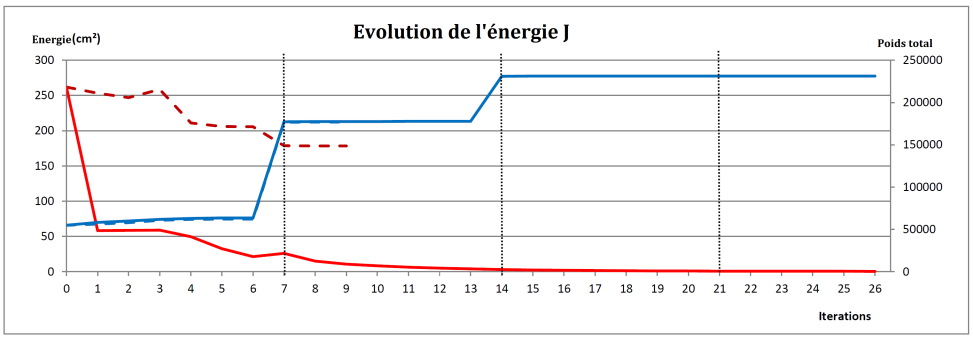


FIGURE 4.11 – Évolution des énergies pour les optimisations comparées sur le nuage simulé #2. La ligne pleine rouge représente l'évolution de l'énergie avec l'optimisation conjointe des paramètres intrinsèques et extrinsèques ; la ligne pointillé rouge représente l'évolution de l'énergie pour l'optimisation des paramètres extrinsèques seuls ; la ligne bleue pleine représente le poids total avec lequel on normalise l'énergie pour l'optimisation conjointe des paramètres ; la ligne bleue en pointillé représente le poids total avec lequel on normalise l'énergie dans le cas de l'optimisation des paramètres extrinsèques seuls ; enfin, les lignes pointillés verticales noires mettent en évidence les itérations où les attributs de dimensionnalité sont mis à jour.

	t_x(cm)	t_y(cm)	t_z(cm)	$\alpha(^\circ)$	$\beta(^\circ)$	$\gamma(^\circ)$	$\Delta\rho(cm)$	$\Delta\theta(^\circ)$	$\Delta\phi(^\circ)$	$\Delta H_z(cm)$
Erreurs initiales par rapport à la vérité terrain	-50.00	60.00	-80.00	2.50	3.00	-2.00	2	0.3	0.2	3
Différence entre la vérité terrain et notre résultat d'optimisation (Optimisation des paramètres extrinsèques seuls)	-0.181	-0.742	-0.463	-0.201	0.014	0.220	-	-	-	-
Différence entre la vérité terrain et notre résultat d'optimisation (Optimisation conjointe des paramètres extrinsèques et intrinsèques)	-0.682	-0.046	1.116	-0.008	0.006	0.039	0.11	$3.59 * 10^{-2}$	$1.80 * 10^{-2}$	0.70

TABLE 4.2 – Erreurs entre les paramètres de calibrages et la vérité terrain pour le nuage simulé #2

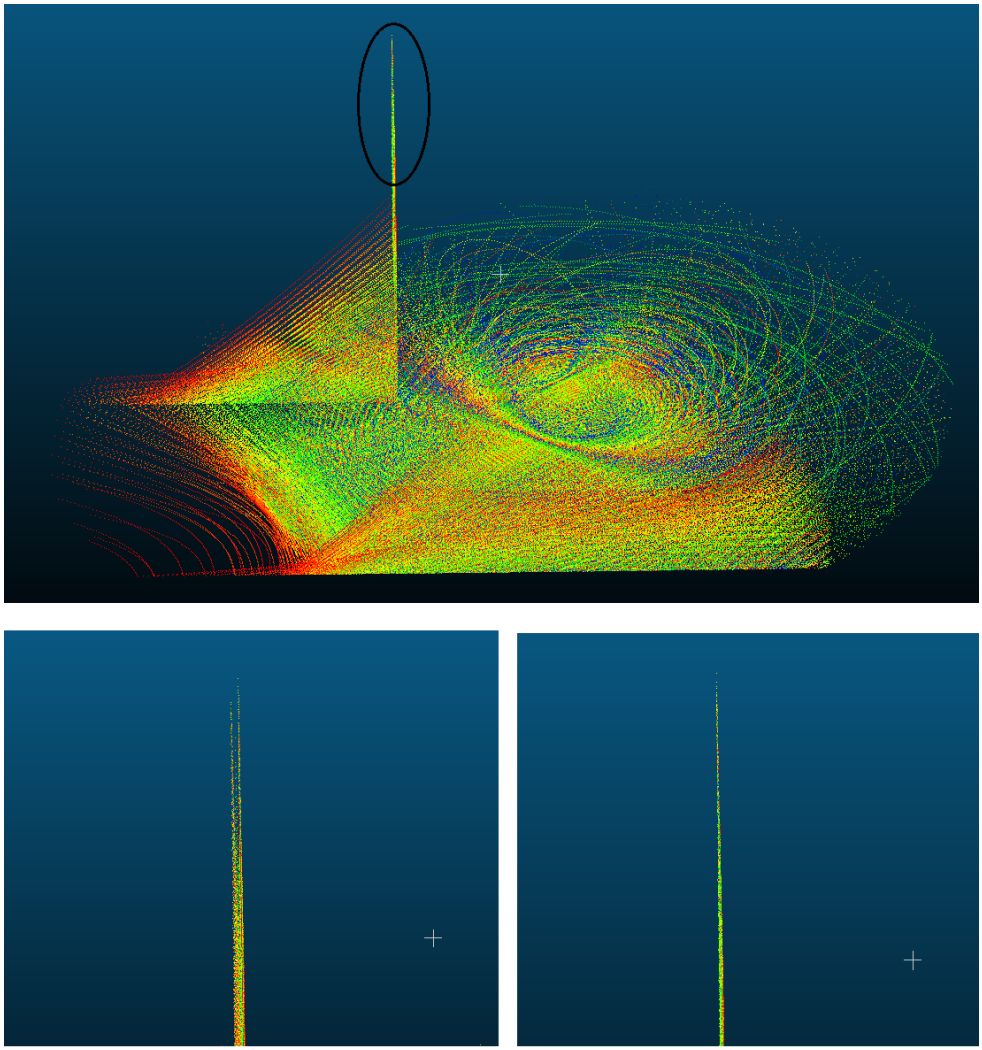


FIGURE 4.12 – Nuage simulé #2 : en haut, nous avons le nuage avec des paramètres intrinsèques et extrinsèques corrects, correspondant à la vérité terrain ; en bas à gauche, nous montrons un morceau de plan après l'optimisation des paramètres extrinsèques seuls ; en bas à droite, nous montrons le même morceau de plan, mais après l'optimisation conjointe des paramètres intrinsèques et extrinsèques.

Un deuxième jeu de données simulées a été considéré pour les tests d'optimisation, le nuage #3. Les résultats d'optimisation sont similaires à ceux donnés pour le nuage #2, et plus de détails sont données en Annexe D.

4.5.3.2 Résultats sur les jeux de données réelles

Nous avons aussi testé notre optimisation sur des jeux de données réelles, les nuages #4 et #5 introduits en section 3.4. Pour ces tests, nous comparons deux approches d'optimisations, qui sont les mêmes que pour les nuages #2 et #3. Pour l'initialisation, et comme nous n'avons pas de vérité terrain, pour les paramètres extrinsèques, nous partons de valeurs choisies arbitrairement, et pour les paramètres intrinsèques, nous partons du modèle donné par le constructeur, et le but de l'optimisation conjointe est d'avoir des offsets intrinsèques qui affinent le nuage.

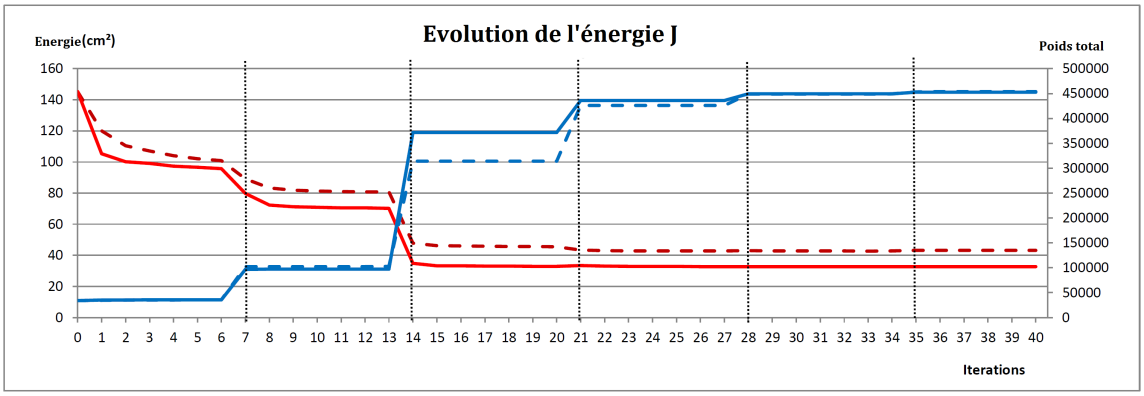


FIGURE 4.13 – Évolution des énergies pour les optimisations comparées sur le nuage réel #4. La ligne pleine rouge représente l'évolution de l'énergie avec l'optimisation conjointe des paramètres intrinsèques et extrinsèques ; la ligne pointillé rouge représente l'évolution de l'énergie pour l'optimisation des paramètres extrinsèques seuls ; la ligne bleue pleine représente le poids total avec lequel on normalise l'énergie pour l'optimisation conjointe des paramètres ; la ligne bleue en pointillé représente le poids total avec lequel on normalise l'énergie dans le cas de l'optimisation des paramètres extrinsèques seuls ; enfin, les lignes pointillés verticales noires mettent en évidence les itérations où les attributs de dimensionnalité sont mis à jour.

Pour le nuage réel #4, nous avons visuellement le même résultat que celui présenté à la figure 4.6, dans les deux cas d'optimisations. Concernant les énergies, la figure 4.13 montre l'évolution des énergies pour les deux optimisations, et pour l'optimisation des paramètres extrinsèques seuls, elle évolue d'une valeur de 145.11 cm^2 à une valeur de 43.26 cm^2 ; avec l'optimisation de l'ensemble des paramètres, l'énergie évolue d'une valeur de 145.11 cm^2 à une valeur de 32.70 cm^2. On peut voir que dans les deux cas d'optimisations, le poids total est à peu près le même en fin d'optimisation pour les deux approches : cela explique le fait qu'il n'y ait pas de différence visuelle entre les nuages avec les deux approches d'optimisations. La table 4.3 donne les différences entre les paramètres extrinsèques optimisés et ceux mesurés sur le véhicule pour le nuage #4 : les paramètres optimisés sont sensiblement les mêmes pour les deux approches, et le résultat se rapproche de ce que l'on a montré en section 3.4.4.2. Pour les paramètres intrinsèques, nous avons des offsets initiaux nuls pour chaque fibre, et après optimisation de l'ensemble des paramètres de calibrage, nous avons des offsets entre -22 cm et 20 cm pour l'offset de translation H_z : ces valeurs sont assez élevées, mais cela est dû au problème d'observabilité évoqué dans le chapitre 3, qui est plus important pour la direction verticale car c'est dans cette direction que l'on a le moins de variations. Pour les autres offsets de paramètres intrinsèques, nous avons des valeurs finales comprises entre -1 et 2 cm pour l'offset de distance ρ, et entre -0.30 ˚et 0.80 ˚pour les offsets angulaires concernant les angles ϕ and θ.
Enfin, pour les temps de calculs, ils sont de l'ordre de 8 minutes pour l'optimisation des paramètres extrinsèques seuls et de 35 minutes pour l'optimisation de l'ensemble des paramètres, en utilisant les attributs de dimensionnalités et pour un nuage de 10 millions de points environs.

Pour le nuage réel #5, les résultats visuels d'optimisation sont aussi similaires à ceux présentés en figure D.4. Les résultats sont similaires à ceux présentés pour le nuage #4, et plus de détails sont donnés en Annexe D.

	t_x(cm)	t_y(cm)	t_z(cm)	$\alpha(^\circ)$	$\beta(^\circ)$	$\gamma(^\circ)$	$\Delta\rho(cm)$	$\Delta\theta(^\circ)$	$\Delta\phi(^\circ)$	$\Delta H_z(cm)$
Paramètres initiaux	0.00	0.00	0.00	0.00	-45.00	90.00	?	?	?	?
Paramètres après optimisation (Optimisation des paramètres extrinsèques seuls)	-0.48	-1.37	-7.60	-0.27	-59.84	93.74	-	-	-	-
Paramètres après optimisation (Optimisation conjointe des paramètres extrinsèques et intrinsèques)	-0.52	-1.39	-6.26	2.74	-57.31	90.99	1.30	0.20	0.32	13.0

TABLE 4.3 – Paramètres de calibrages pour le nuage réel #4

4.6 Conclusion

Dans ce chapitre, nous avons présenté une méthode d'affinement de nuages de points par optimisation des paramètres de calibrage intrinsèque du système mobile ayant servi à acquérir les données. Dans un premier temps, nous avons optimisé les paramètres intrinsèques en reprenant la méthode d'optimisation présentée dans le chapitre 3 : nous avons changé les paramètres qui sont optimisés, mais la fonctionnelle et l'algorithme d'optimisation sont restés les mêmes. Comme pour les paramètres extrinsèques, l'optimisation que l'on propose est rapide, mais aussi robuste : pour les paramètres intrinsèques aussi, nos tests ont montré que nous n'avons pas besoin d'avoir une initialisation précise des paramètres pour que notre optimisation converge. Dans un deuxième temps, nous avons exploré la possibilité d'une optimisation conjointe des paramètres extrinsèques et intrinsèques : les résultats finaux ont montré que l'optimisation conjointe donne un affinement plus fin des nuages.

Chapitre 5

Affinement de nuages de points par correction de la trajectoire

Sommaire

Introduction

Dans les chapitres 3 et 4, nous avons présenté deux algorithmes d'optimisation des paramètres extrinsèques et intrinsèques avec comme objectif l'affinement de nuages de points par optimisation des paramètres de calibrage. Dans les deux cas, nous supposions que la trajectoire du véhicule mobile d'acquisition était parfaitement connue et correcte, ce qui n'est pas certain notamment en environnement urbain où les capteurs GNSS ne sont pas capables de mesurer en permanence la position du véhicule. Si on se réfère à la figure 2.6, nous avons traité l'optimisation des paramètres de deux calibrages différents, qui représentent les transformations des données brutes acquises par le système mobile vers le repère body dont le centre est la position du véhicule dans un repère monde à tout instant.
Dans ce chapitre, nous allons nous intéresser à l'optimisation de la dernière transformation nécessaire au géo-référencement : il s'agit de l'optimisation de la trajectoire du véhicule mobile. Le but est toujours d'aller dans le sens de ce qui a été présenté jusqu'à maintenant : nous reprenons la même approche, avec la même fonctionnelle à minimiser. La solution que l'on va proposer dans ce chapitre est applicable dans la méthodologie à l'optimisation de la trajectoire d'un système mobile équipé soit d'un LIDAR multi-couches, soit de plusieurs LIDARs ; en effet, nous allons aussi utiliser la redondance de données entre fibres du même capteur, et les résultats seront présentés pour des données acquises avec un capteur LIDAR Velodyne HDL-32E.

5.1 Etat de l'art sur l'optimisation de trajectoire

Pour la cartographie mobile, la connaissance de la trajectoire est indispensable pour correctement construire des cartes 3D de l'environnement. Nous avons vu dans le chapitre 2 que le géoréférencement des données est très important dans de nombreux cas d'application, et les systèmes mobiles

d'acquisition embarquent de nombreux capteurs pour connaître leur position par rapport à une référence de façon très précise : GPS, centrale inertielle, odomètres, et les données de ces différents capteurs sont fusionnées pour avoir le positionnement du véhicule.
Un domaine de recherche pour lequel le positionnement du véhicule est très important est le SLAM : en effet, pour de la « Localisation et Cartographie Simultanée », la localisation précise du véhicule est un des résultats espérés, en plus de la cartographie précise. Ces algorithmes peuvent être séparés en plusieurs catégories, comme le fait Gérossier dans sa thèse [Gérossier, 2012]. Nous pouvons réaliser une classification de ces algorithmes similaire :

- Le SLAM avec une représentation par grilles d'occupation. Cette approche permet d'obtenir une carte divisée en plusieurs cellules, et qui fournit une information de probabilité d'occupation des différentes cellules de la carte, permettant la planification de trajectoires par exemple. Ce type de SLAM est généralement utilisé en amont d'une autre méthode de SLAM, notamment l'EKF-SLAM, car il permet d'avoir une estimation de la trajectoire, qui est ensuite affinée.
- Le SLAM fondé sur une représentation par cartes topologiques. Cette fois-ci, l'environnement est représenté par des noeuds, qui représentent les différentes positions du système mobile, et des arcs qui illustrent les trajets entre deux nœuds. C'est une approche simple à mettre en œuvre, puisqu'il n'y a plus de problèmes d'incertitudes sur la position du robot, la navigation de celui-ci étant locale. Toutefois, cette approche n'est pas optimale et ne prend en compte aucune notion géométrique : c'est pourquoi il vaut mieux l'utiliser de pair avec un autre type d'approche.

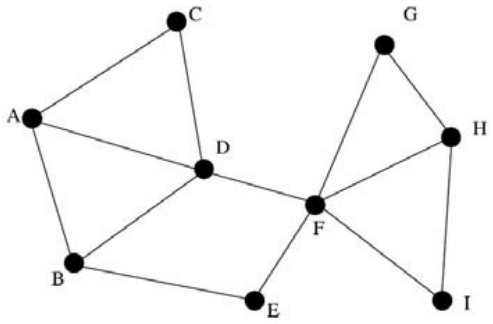


FIGURE 5.1 – Carte topologique [Bailey, 2002]

- Le Graph SLAM, qui dans l'idée est assez proche du SLAM basé sur une représentation par cartes topologiques ; les deux approches sont basées sur la construction d'un graphe avec les données acquises. Cependant, ici, les nœuds représentent aussi bien les positions successives du véhicule que les points d'intérêt extraits de l'environnement, et les arcs traduisent la présence de recouvrement entre nœuds. C'est une approche d'optimisation, car les recouvrements permettent de simplifier le Graphe construit, ce qui conduit à un problème de plus faible dimension, finalement solvable avec des techniques d'optimisations standards, comme la minimisation des moindres carrés par exemple.
- Le SLAM avec une carte d'amers. Cette approche utilise des points d'intérêts extraits des nuages de points - ou plus généralement des données acquises à l'aide des capteurs montés sur le véhicule - que l'on nomme « amers ». Le plus souvent, cette approche est utilisée dans une optique temps réel, notamment avec un filtre de Kalman, ou un filtre de Kalman étendu (EKF-SLAM), car la fusion de données est le plus souvent non linéaire. Dans une telle approche, les capteurs extéroceptifs type GPS et proprioceptifs type odomètre, centrale inertielle ont un rôle très important, car ils rentrent à part entière dans l'estimation de la localisation et la correction de celle-ci à l'aide du filtre de Kalman. Dans cette approche, le vecteur d'état

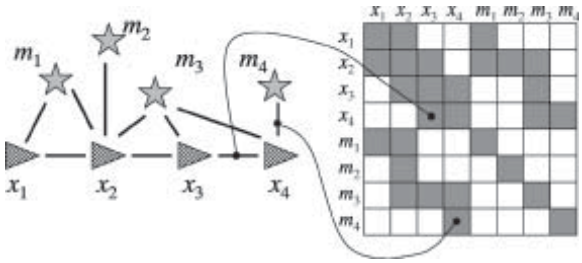


FIGURE 5.2 – Exemple de graphe utilisé dans une approche de Graph SLAM [Thrun et Montemerlo, 2005]

à chaque instant de contrôle contient les positions du véhicule et les positions des amers. C'est un SLAM très utilisé pour des applications temps réel, notamment grâce à l'utilisation de l'EKF pour la fusion de données issues des capteurs du véhicule. Tim Bailey et Hugh Durrant-Whyte, dans deux articles publiés la même année [Bailey et Durrant-Whyte, 2006a], [Bailey et Durrant-Whyte, 2006b], proposent un historique très complet sur le SLAM probabiliste, basé amers. Plusieurs solutions au problème du SLAM sont proposées, dont l'EKF, et un autre filtre, le filtre particulaire. Pour résumer la méthodologie de ce type d'approche, on peut distinguer deux temps dans l'algorithme de SLAM avec une carte d'amers : tout d'abord, le vecteur d'état du véhicule est estimé à l'aide des capteurs de mesure, puis, grâce au filtre de Kalman (par exemple), les données issues des capteurs et les amers extraits des cartes produites sont fusionnés pour affiner la localisation, et la précision des cartes dans le même temps.

- Le SLAM orienté trajectoire. Dans cette dernière approche, on cherche à estimer les positions successives du véhicule lors de la campagne d'acquisition, mais sans estimation de la trajectiore en s'appuyant sur des amers comme dans le SLAM avec une carte d'amers. Cette fois-ci, le vecteur d'état à chaque instant de contrôle ne contient que les positions du véhicule, qui sont au nombre de six : trois positions et trois rotations. Historiquement, dans ce genre d'approche, on ne cherchait à trouver un vecteur d'état composé que de trois paramètres, en supposant que le véhicule n'avait que trois degrés de libertés, qui sont les deux positions dans le plan de déplacement du véhicule et la rotation autour de l'axe vertical. Cependant, grâce à l'évolution des systèmes d'acquisitions mobiles, et surtout pour répondre à de nouvelles problématiques de cartographie comme l'explique Nüchter [Nüchter *et al.*, 2004], le vecteur d'état est maintenant composé de six paramètres : c'est ce que l'on appelle notamment le 6D-SLAM. En effet, dans son article de 2004, Nüchter présente le 6D-SLAM, qui voit deux méthodes d'optimisation distinctes : d'une part, on a le recalage en tant que problème d'optimisation. C'est le « scan matching », tel qu'il a été introduit par Besl et Mckay en 1992 [Besl et McKay, 1992], et qui consiste à trouver le vecteur d'état entre deux scans qui minimise l'énergie qu'ils définissent. D'autre part, on a aussi une optimisation orientée "scan matching", mais basée sur l'extraction d'amers : cette fois-ci, le problème à résoudre consiste à recaler les scans concernés sur la base du recouvrement de points d'intérêts extraits de chacun des scans.

A côté de ces approches de SLAM, on peut trouver d'autres approches plus spécifiques, qui dépendent des systèmes d'acquisitions utilisés, comme par exemple le visual-SLAM (V-SLAM), où une approche « image » est considérée [Lategahn *et al.*, 2011]. Effectivement, ce type de SLAM est particulier car le type de capteurs utilisé est une caméra, mais cette approche possède l'avantage de permettre l'extraction de points d'intérêts - ce que l'on appelle des amers visuels dans ce cas-là - plus facilement qu'avec un nuage de points. Un autre type de SLAM rencontré est le Cooperative-SLAM (C-SLAM) dans lequel plusieurs robots sont utilisés, comme présenté dans [Heon-Cheol *et al.*, 2011] et [Mourikis et Roumeliotis, 2006]. Ces robots vont coopérer afin de créer les cartes des environs et se localiser. Dans le cas où l'information de positionnement absolu est manquante, les robots peuvent tout de même améliorer la précision de leur positionnement en prenant en compte la position relative

par rapport aux autres robots.

5.2 Méthode d'optimisation proposée

La figure 2.6 présente les principales étapes de géoréférencement des données lors d'une cartographie mobile. Dans les chapitres 3 et 4, nous avons présenté des méthodes d'affinement de nuages de points par optimisation des paramètres de calibrage extrinsèque et intrinsèque. Ces optimisations ont été effectuées avec l'hypothèse que la position du véhicule était connue à tout instant de façon précise : cette hypothèse n'est pas complètement vérifiée, puisque la centrale inertielle et le système GPS ont des erreurs de mesures non négligeables avec le temps. C'est pour cela que nous nous sommes intéressés à l'optimisation de la trajectoire du véhicule au cours de l'acquisition. L'article qui traite de ce sujet qui nous a le plus intéressé est le suivant, [Monnier *et al.*, 2013], dans lequel une méthode pour recaler un nuage de point mobile avec une vérité terrain appartenant à une base de données est proposée. La méthode proposée est du recalage non-rigide : en effet, le caractère non-rigide du recalage est important car au cours de l'acquisition, l'erreur sur la trajectoire n'est pas constante, et il est alors préférable de ne pas appliquer la même transformation à l'ensemble des données pour avoir un meilleur recalage avec la vérité terrain.
La méthode que l'on propose pour la correction de la trajectoire va dans la continuité de ce que l'on a proposé dans les chapitres 3 et 4. En effet, nous voulons corriger les translations de la trajectoire du véhicule, et pour cela, nous supposons que les paramètres de calibrages extrinsèques et intrinsèques sont soit corrects, soit déjà corrigés.

5.2.1 Présentation de l'algorithme mis au point et optimisation

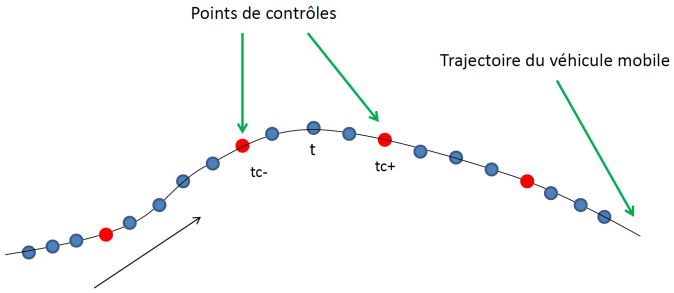


FIGURE 5.3 – Schéma d'une acquisition avec plusieurs points de contrôles

La figure 5.3 présente une acquisition avec plusieurs « points de contrôles » : un point de contrôle est un point qui sert de repère dans une acquisition. Il peut être **virtuel**, interpolé entre deux points de l'acquisition, ou **réel**. Ce point de contrôle possède un « temps de contrôle », et est en général choisi pour sa position 3D ou son temps d'acquisition : dans notre cas, et pour l'application de correction de la trajectoire, nous avons choisi des points de contrôles en fonction de leur temps d'acquisition, afin d'avoir une trajectoire subdivisée en « morceaux » de plages temporelles égales. Comme nous l'avons dit dans la section 5.2, un de nos objectifs avec l'optimisation de la trajectoire du véhicule est de proposer une méthode qui aille dans la continuité de ce qui a été fait pour l'instant. Nous cherchons à corriger la trajectoire en utilisant la structure du capteur LIDAR multi-couches et en exploitant la redondance de données entre les différentes couches. Dans ce chapitre, nous ne présentons que l'optimisation des paramètres de translation de la trajectoire : c'est une première approche pour l'affinement des nuages de points par correction de la trajectoire. Pour réduire la complexité de la correction de la trajectoire, seuls certains points sont concernés par la correction de la position du véhicule, les points de contrôles : pour les positions du véhicule mobile

sur le reste de l'acquisition, une interpolation linéaire sur les positions du véhicule aux 2 points de contrôles « encadrants » chaque point permet d'appliquer une correction de trajectoire à l'ensemble de l'acquisition.

La trajectoire que l'on cherche à corriger peut être supposée à peu près correcte : en effet, celle-ci provient de la fusion des données venant du GPS et de la centrale inertielle, qui donnent des informations précises. Certaines erreurs de positionnement par GPS et une dérive de la centrale inertielle peuvent fausser la trajectoire du véhicule, qui a besoin d'être affinée. Toutefois, ces erreurs peuvent être supposées faibles : c'est pour cela que la fonctionnelle que l'on cherche à minimiser pour la correction de la trajectoire se décompose en deux parties : une première **énergie d'attache aux données J** qui permet de corriger la trajectoire du véhicule, et une deuxième **énergie de déformation** E_{def} qui permet de contrôler l'amplitude des modifications apportées à la trajectoire, comme présenté en équation 5.1 :

$$E_{tot}(\delta X) = J(\delta X) + \lambda_{rigid} * E_{def}(\delta X) \tag{5.1}$$

La correction que l'on apporte à la translation de la trajectoire en chaque point de contrôle est la suivante : $\delta X_{t_c} = (\delta x_{t_c} \quad \delta y_{t_c} \quad \delta z_{t_c})^T$, où t_c est un temps de contrôle ; il s'agit du temps d'acquisition d'un point de contrôle. L'énergie d'attache aux données est définie en équation 5.2 :

$$J(\delta X) = \frac{\sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_k w_{i,j,k} * d_{i,j,k}^2(\delta X)}{\sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_k w_{i,j,k}} \tag{5.2}$$

avec :

$$\begin{cases} d_{i,j,k}(\delta X) = n_{i,k}^T \times (p_{i,k}(\delta X) - m_{j,k}(\delta X)) \\ p_{i,k}(\delta X) = R_{nav}(p'_{i,k}) \times (R \times p'_{i,k} + T) + Xp_{i,k} + \delta Xp_{i,k} \\ m_{j,k}(\delta X) = R_{nav}(m'_{j,k}) \times (R \times m'_{j,k} + T) + Xm_{j,k} + \delta Xm_{j,k} \\ Xp(i,k) = T_{nav}(p'_{i,k}) \\ Xm(j,k) = T_{nav}(m'_{j,k}) \\ \\ \delta Xp_{i,k} = (1 - \alpha(t_p)) * \delta X_{t_{p,c-}} + \alpha(t_p) * \delta X_{t_{p,c+}} \\ \delta Xm_{j,k} = (1 - \alpha(t_m)) * \delta X_{t_{p,c-}} + \alpha(t_m) * \delta X_{t_{p,c+}} \\ \alpha(t) = (t - t_{c-})/(t_{c+} - t_{c-}) \end{cases}$$

Les corrections sur la position du véhicule en chaque point issus de l'acquisition sont calculées à partir d'une interpolation linéaire sur les corrections des positions aux 2 points de contrôles encadrants chaque point, comme présenté dans la même équation. Pour le terme $\delta Xp_{i,k}$, t_p est le temps d'acquisition du point $p_{i,k}$, et $t_{p,c+}$ et $t_{p,c-}$ représentent respectivement les temps de contrôles supérieur et inférieur au temps t_p ; de même pour le temps t_m. Cette interpolation linéaire décrit le caractère non-rigide du recalage, puisque chaque point du nuage aura sa propre correction de trajectoire.
Comme nous l'avons évoqué, nous avons aussi ajouté une énergie de déformation contrôlant l'amplitude des corrections apportées à la trajectoire, définie en équation 5.3 :

$$E_{def}(\delta X) = \sum_{i=1}^{N_{t_c}} \|\delta X_{t_{c,i}}\|^2 \tag{5.3}$$

Dans l'équation 5.3, l'énergie E_{def} est définie comme la somme des corrections apportées aux positions du véhicule en chaque point de contrôle. N_{t_c} donne le nombre de points de contrôles gardés pour l'optimisation. Dans l'article de Monnier [Monnier *et al.*, 2013], l'énergie de déformation choisie vaut $\sum_{i=1}^{N_{t_c}-1} \|\delta X_{t_{c,i+1}} - \delta X_{t_{c,i}}\|^2$, ce qui permet de contrôler l'amplitude des corrections entre 2

positions de contrôle successives. Comme expliqué dans l'article, les auteurs cherchent à recaler une nouvelle acquisition à une référence géo-référencée, et la distance entre les deux nuages de points peut-être élevée : la seule contrainte est d'avoir une continuité sur la trajectoire du nuage de point qui est recalé, d'où une rigidité sur les corrections entre deux positions de contrôle successives. Pour notre approche d'optimisation, nous partons de l'hypothèse que la trajectoire actuelle est proche de la trajectoire réelle : nous cherchons essentiellement à corriger des inconsistances sur la trajectoire du véhicule au sein d'une même acquisition. C'est pour cela que nous avons choisi le terme de déformation présenté en équation 5.3 : entre deux positions de contrôles successives, la correction à apporter à la translation de la trajectoire peut-être différente, mais elle reste assez faible.

Pour contrôler les corrections qui sont apportées à la trajectoire, en plus de l'énergie de déformation, un terme de rigidité λ_{rigid} adimensionnel est ajouté à l'optimisation, qui permet de contrôler l'amplitude des déformations. Selon sa valeur, le résultat d'optimisation sera différent :

- si le λ_{rigid} est faible, des amplitudes de corrections plus élevées seront permises, ce qui permet de corriger les erreurs de trajectoires importantes, au risque de trop s'éloigner de la trajectoire réelle du véhicule.
- si le λ_{rigid} est élevé, les corrections apportées seront plus petites, mais le nuage affiné aura une trajectoire corrigée proche de celle d'origine, ce qui peut être préférable dans certains cas. L'optimisation est aussi plus lente puisque les corrections voulues sont faibles.

Comme nous l'avons évoqué précédemment, nous prenons un certain nombre de points de contrôle N_{t_c} pour optimiser la trajectoire, et l'équation 5.1 peut alors se réécrire sous la forme :

$$E_{tot}(\delta X) = \frac{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_k w_{i,j,k} * (D_{i,j,k} + C^T_{i,j,k} \times \delta X)^2}{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_k w_{i,j,k}} + \lambda_{rigid} * \sum_{i'=1}^{N_{t_c}} \|\delta X_{t_{c,i'}}\|^2 \qquad (5.4)$$

avec :

$$\begin{cases} \delta X = ([\delta x_l \quad \delta y_l \quad \delta z_l]_{,l\in[\![1,N_{t_c}]\!]})^T \\ D_{i,j,k} = n^T_{i,k} \times \big(T_{nav}(p'_{i,k}) - T_{nav}(m'_{j,k}) + (R_{nav}(p'_{i,k}) \times (R \times p'_{i,k} + T)) - (R_{nav}(m'_{j,k}) \times (R \times m'_{j,k} + T))\big) \\ C_{i,j,k} = \begin{bmatrix} \dots \\ (1-\alpha(tp_{i,k})) * n_{i,k} \\ \alpha(tp_{i,k}) * n_{i,k} \\ \dots \\ -(1-\alpha(tm_{j,k})) * n_{i,k} \\ -\alpha(tm_{j,k}) * n_{i,k} \\ \dots \end{bmatrix} \end{cases}$$

Dans l'équation 5.4, δX représente le vecteur des corrections que l'on apporte à la trajectoire en chaque point de contrôle sélectionné : il a une taille de $3 * N_{t_c}$, car pour chaque point de contrôle, les 3 paramètres de translations sont optimisés. $C_{i,j,k}$ est un vecteur de taille $3 * N_{t_c}$ (3 paramètres de position par temps de contrôle), avec entre 6 et 12 termes au maximum non nuls, selon le temps de contrôle considéré. Pour le point p, 6 termes sont non nuls, comme indiqué dans le vecteur $C_{i,j,k}$, et proviennent de l'interpolation linéaire effectuée entre les temps de contrôle supérieur et inférieur au point p. Les 6 termes non nuls du vecteur sont les termes $3 * t_{p,c-}$, $3 * t_{p,c-} + 1$ et $3 * t_{p,c-} + 2$, ainsi que $3 * t_{p,c+}$, $3 * t_{p,c+} + 1$ et $3 * t_{p,c+} + 2$. Nous avons la même construction pour le point m, avec les temps de contrôles $t_{m,c-}$ et $t_{m,c+}$ et où 6 termes sont aussi non nuls. Dans le cas où $t_{p,c-} = t_{m,c-}$ et $t_{p,c+} = t_{m,c+}$, seuls 6 termes sont non nuls (au lieu de 12). $n_{i,k}$ est la normale au plan passant par le point k acquis par la fibre i.

Finalement, la solution qui minimise l'équation (5.4) est la solution du système linéaire suivant :

$$C \times \delta X = -V \tag{5.5}$$

avec :

$$\begin{cases} C = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * (C_{i,j,k} \times C_{i,j,k}^{T}) + \lambda_{rigid} * I_{3n} \\ V = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * (D_{i,j,k} \times C_{i,j,k}) \end{cases}$$

5.2.2 Validation du résultat de l'optimisation

Contrairement aux chapitres 3 et 4, nous ne pouvons pas directement utiliser la valeur finale de l'énergie pour valider le résultat d'optimisation. En effet, la fonctionnelle que l'on cherche à minimiser pour l'optimisation de la trajectoire est composée de 2 termes :

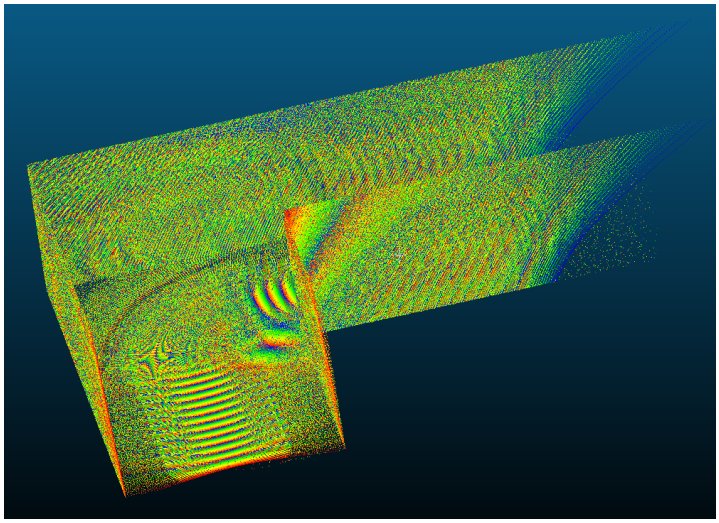
- un terme d'attache aux données, qui est similaire aux énergies minimisées dans les deux chapitres précédents.
- un terme de déformation, qui permet de contrôler l'amplitude des déformations que l'on applique à la trajectoire. Ce terme ne permet plus d'utiliser la valeur de l'énergie J après convergence comme critère de validation de l'optimisation, parce que ce terme peut être élevé en fonction de la valeur de λ_{rigid} que l'on choisit.

L'énergie que l'on a définie n'a plus le même sens physique qu'avant. Aussi, pour valider notre résultat d'optimisation, nous mesurons des distances nuage à nuage : pour le jeu de données simulées, nous comparons les distances entre la vérité terrain et le nuage avec une mauvaise trajectoire d'un côté, et entre la vérité terrain et le nuage dont la trajectoire est optimisée de l'autre ; le résultat attendu est une distance par rapport à la vérité terrain plus faible dans le cas où la trajectoire est optimisée. Pour le jeu de données réelles, la validation est visuelle, car nous n'avons pas de vérité terrain avec laquelle comparer le résultat d'optimisation.

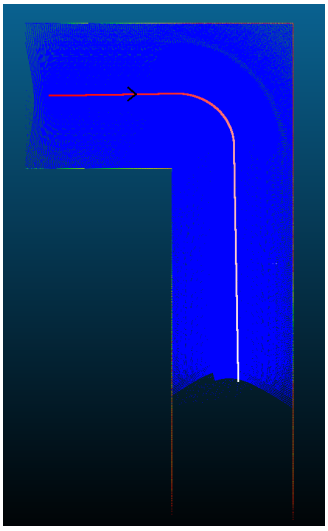
5.3 Résultats expérimentaux

Pour les paramètres à régler dans notre optimisation (nombres de fibres voisines recalées entre elles ; taille du voisinage pour le calcul des normales), nous avons utilisé les mêmes valeurs que celles retenues pour les sections 3.4.2 et 4.4. Pour l'expérimentation réelle, un sous-échantillonnage plus important a été effectué car le nuage est composé d'un très grand nombre de points et possède de nombreux recouvrements entre différents instants d'acquisitions. Deux nouveaux paramètres apparaissent dans cette optimisation : d'une part, nous avons le nombre de temps de contrôle à prendre en compte pour la correction des paramètres de translation de la trajectoire, et d'autre part le paramètre λ_{rigid} de rigidité. Différents tests ont montré qu'un espacement de l'ordre de la seconde des temps de contrôles donnait les résultats optimaux. Pour le paramètre d'attache aux données, son ordre de grandeur détermine la « liberté » laissée aux données pour se déformer ou non : plus la valeur est grande, plus faiblement les paramètres de trajectoires seront modifiés, et inversement. Dans [Monnier *et al.*, 2013], une description de son ordre de grandeur est donnée : dans notre optimisation, nous donnerons moins d'importance à l'attache aux données qu'à la correction de la trajectoire. Aussi, en fonction des caractéristiques de notre système d'acquisition et du résultat que l'on souhaite avoir, l'ordre de grandeur que l'on a retenu pour λ_{rigid} est de 10^2.

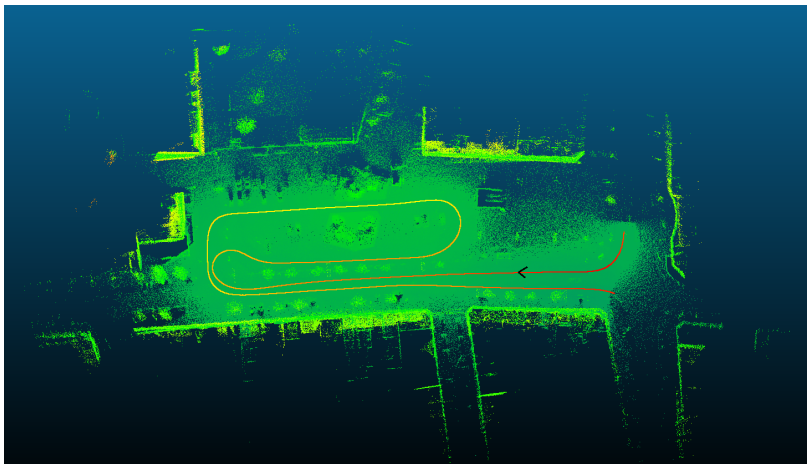
Dans la suite de cette section, nous allons présenter 2 résultats d'optimisation des paramètres de translations de la trajectoire : nous avons un nuage simulé et un nuage réel, qui ont les propriétés suivantes :



(a) Jeu de données simulées #6



(b) Trajectoire du nuage #6



(c) Jeu de données réelles #7, avec sa trajectoire

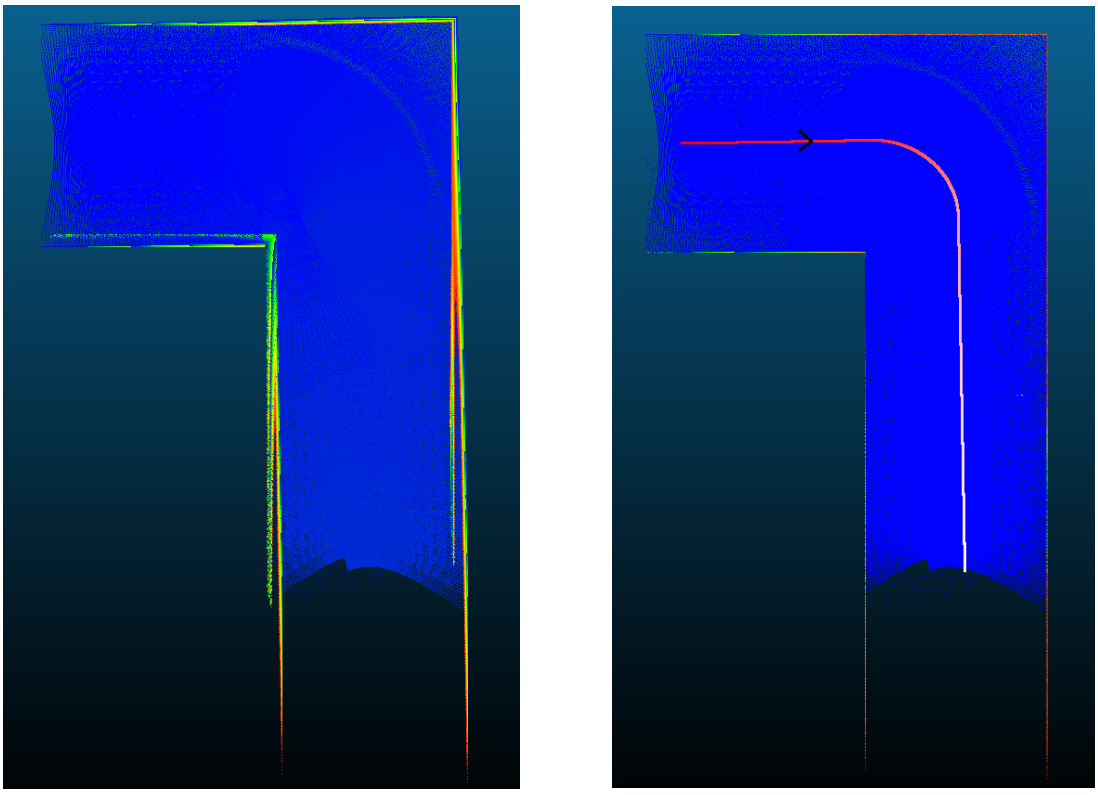
FIGURE 5.4 – Jeux de données utilisés pour l'optimisation de la trajectoire

- la figure 5.4 a) présente le jeu de données simulées utilisé pour valider notre approche d'optimisation. La scène est composé d'un sol et de 4 plans verticaux représentant des façades, et le véhicule effectue un virage lors de son déplacement. Enfin, il n'y a pas de variation d'altitude. Le nuage est composé d'environ 4 millions de points. Un nombre de 5 points de contrôles a été retenu pour ce nuage simulé, afin que l'on ait des points de contrôles dont les temps d'acquisitions soient espacés d'environ 1 seconde. La figure 5.4 b) présente la trajectoire du nuage simulé #6.

- la figure 5.4 c) présente le nuage réel utilisé pour l'optimisation de la trajectoire. Il s'agit d'une acquisition effectuée à Lille, et la portion de l'acquisition utilisée représente une place de marché sur laquelle le véhicule d'acquisition effectue une boucle. Cette fois-ci, nous prenons 50 points de contrôles, car le nuage possède environ 50 millions de points, et que cela nous permet d'avoir un intervalle entre les temps d'acquisitions des points de contrôles de l'ordre de la seconde.

5.3.1 Résultats sur un jeu de données simulées

Pour le jeu de données simulées #6, nous avons la vérité terrain : il est donc possible de comparer notre résultat d'optimisation à cette vérité. Pour les tests, nous sommes partis de la vérité terrain à laquelle nous avons rajouté une erreur linéaire par morceaux à la trajectoire. Pour vérifier le résultat d'optimisation, c'est-à-dire si la trajectoire a correctement été optimisée pour que le nuage après optimisation soit très proche de la vérité terrain, nous avons mesuré la distance nuage à nuage entre la vérité terrain et le nuage avant optimisation d'une part, puis entre la vérité terrain et le nuage après optimisation. Avant optimisation, nous avons une distance de 13.36 cm environ, alors qu'après optimisation, cette distance vaut environ 6 mm : il y a une très nette amélioration de la qualité du nuage. La figure 5.5 présente visuellement le résultat de l'affinement par optimisation de la trajectoire, et on peut voir que l'on a bien une nette amélioration par rapport au nuage de départ avec une trajectoire erronée.
Le temps de calcul pour cette optimisation est d'environ 2 minutes car le nombre de temps de contrôles est assez faible : nous avons 15 paramètres seulement à optimiser.



(a) (b)

FIGURE 5.5 – Résultats de l'optimisation de la trajectoire pour le nuage #6 : a) jeu de données avant optimisation ; b) jeu de données après optimisation

5.3.2 Résultats sur un jeu de données réelles

Le jeu de données réelles #7 provient d'une acquisition effectuée à Lille, dans le cadre du projet Terramobilita. La zone présentée sur la figure 5.4 c) est une place dans laquelle le véhicule d'acquisition a effectué un aller-retour, comme le montre la trajectoire présentée sur la même figure. Ce jeu de données est intéressant parce que la trajectoire n'est pas optimale, et les recouvrements dus à l'aller-retour lors de la cartographie ne se superposent pas comme ils devraient, comme le montrent

les figures 5.7 et 5.8, qui sont des vues de dessus de l'acquisition. Pour ce jeu de données, nous n'avons pas de vérité terrain, et la validation est visuelle : nous savons qu'avec la trajectoire avant optimisation, certaines zones du nuage ont des problèmes, et nous voulons que l'optimisation des paramètres de trajectoire règle en grande partie ces problèmes de recouvrement. C'est le cas, comme le montrent les figures 5.7 b) et 5.8 b) : en effet, on voit une nette amélioration au niveau des zones entourées, avec un meilleur recouvrement des données issues de différents instants d'acquisitions. Le temps de calcul pour cette optimisation est d'environ 15 minutes.

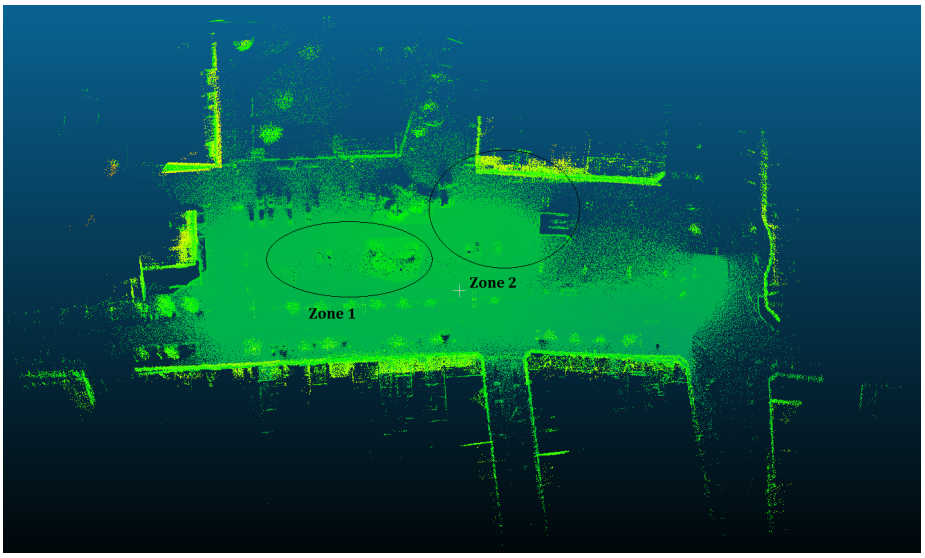
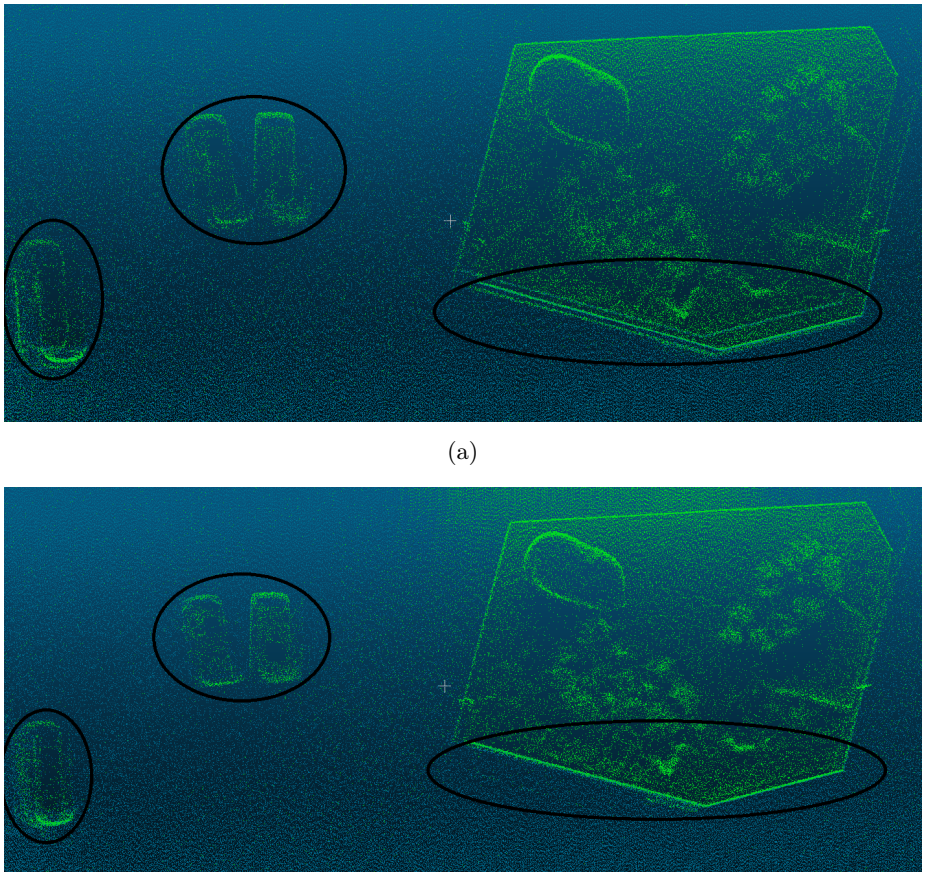


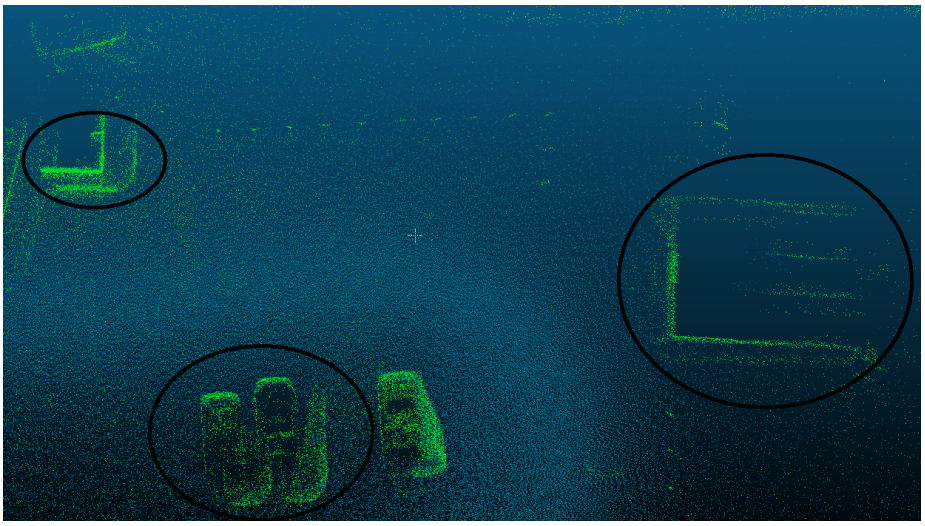
FIGURE 5.6 – Nuage #7, présentant les deux zones que l'on compare avec l'optimisation



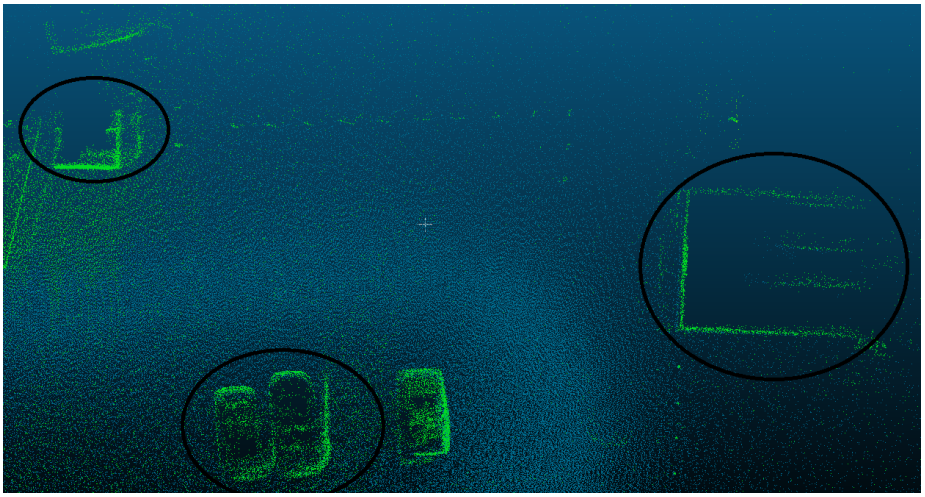
(a)

(b)

FIGURE 5.7 – Résultats de l'optimisation de la trajectoire pour le nuage #7 sur une première zone du nuage : a) jeu de données avant optimisation ; b) jeu de données après optimisation



(a)



(b)

FIGURE 5.8 – Résultats de l'optimisation de la trajectoire pour le nuage #7 sur une autre zone du nuage : a) jeu de données avant optimisation ; b) jeu de données après optimisation

5.4 Conclusion

Dans ce chapitre, nous nous sommes intéressés à l'affinement de nuages de points par optimisation de la trajectoire du véhicule lors de l'acquisition. Nous n'avons traité que de l'optimisation des paramètres de translations de la trajectoire pour avoir une première approche d'optimisation de la trajectoire. Au vu des résultats obtenus, l'affinement par optimisation de la trajectoire donne des résultats satisfaisants, qui nous permettent de valider l'approche. Les résultats sont satisfaisants pour l'approche que l'on a retenue : par rapport aux approches qui ont été présentées dans les chapitres 3 et 4, nous ne nous sommes intéressés qu'à l'optimisation des paramètres de translations. Pour le jeu de données réelles #7, les figures 5.7 et 5.8 présentent des améliorations, mais une erreur de recalage persiste, et cette erreur semble due à un décalage angulaire : la centrale inertielle donne des valeurs de roulis et de tangage précises, mais le lacet est plus difficilement mesurable par celle-ci. Aussi, comme indiqué dans la section 5.3.2, les images des résultats sont présentés en vues de dessus, et le décalage angulaire est alors essentiellement représenté par un biais au niveau du lacet. C'est pour cela qu'une piste d'amélioration de notre approche d'optimisation de la trajectoire est de prendre en compte les angles : les premiers résultats obtenus indiquent assez clairement qu'une optimisation des angles de rotations du véhicule permettrait d'affiner le nuage réel #7 de façon plus significative.

Chapitre 6

Conclusions et perspectives

Sommaire

6.1 Résultats par rapport aux objectifs fixés

Dans le chapitre 1, nous avons défini l'objectif principal de la thèse qui était de proposer différentes méthodes d'affinement de relevés laser mobiles. Pour cela, nous avons présenté trois contributions principales dans les chapitres 3, 4 et 5, avec des résultats d'optimisations sur des jeux de données simulées et réelles.

Le chapitre 3 a présenté une méthode d'affinement des relevés laser par optimisation des paramètres de calibrage extrinsèque du capteur LIDAR. Les résultats sont très satisfaisants : en effet, nous avons testé et montré la robustesse de notre méthode en initialisant l'optimisation avec des paramètres de calibrage très éloignés de la vérité terrain dans le cas des jeux de données simulées, et des paramètres arbitraires pour les jeux de données réelles. Aussi, nous avons présenté visuellement les résultats d'optimisation, où l'on voit une nette amélioration de la qualité des relevés : les éléments plans sont correctement planaires, les données des nuages sont plus homogènes. Pour cette méthode d'affinement, nous avons validé les résultats d'optimisations en fonction de la valeur finale de l'énergie : c'est pour cela que pour l'affinement proposé dans le chapitre 3, nous pouvons dire que nous satisfaisons à notre objectif d'affinement des relevés laser.

Le chapitre 4 a introduit une méthode d'affinement par optimisation des paramètres de calibrage intrinsèque du capteur LIDAR : nous avons proposé une méthode qui va dans la continuité du chapitre 3, puisque la fonctionnelle que l'on cherche à minimiser pour optimiser les nouveaux paramètres est la même que celle introduite dans le chapitre 3. Nous utilisons la même approche, et les différents résultats d'optimisation que l'on présente montrent une nette amélioration de la qualité des données. Nous avons aussi proposé une amélioration de cette méthode d'affinement en effectuant conjointement l'optimisation des paramètres extrinsèques et intrinsèques : là aussi, nous montrons une amélioration des résultats d'optimisation, ce qui permet de valider l'approche introduite et de répondre à notre objectif principal.

Enfin, le chapitre 5 présente une méthode d'affinement par optimisation des paramètres de translations de la trajectoire liés à l'acquisition du relevé concerné. En effet, comme nous l'avons présenté en section 5.2.1, la trajectoire du véhicule lors de l'acquisition d'un nuage de points peut être subdivisée en plusieurs « morceaux », qui sont repérés par des instants de contrôle : ces « instants » ont une position donnée et un temps GPS associé, et ce sont les translations liées à chaque instant de contrôle que l'on a cherché à optimiser. Là aussi, nous avons utilisé la même approche que dans les chapitres 3 et 4, ce qui a permis de créer une continuité dans les approches d'affinement que l'on a proposé. Les résultats que l'on présente montrent une amélioration de la qualité des relevés laser,

et permettent de valider l'optimisation que l'on propose : nous avons là aussi répondu à l'objectif principal de la thèse.

Concernant les autres objectifs, nous voulions des méthodes qui :

- **soient robustes** ; c'est le cas, notamment du fait que nos différentes méthodes n'ont pas besoin d'initialisations précises. Nous prenons des paramètres initiaux arbitraires, éloignés du résultat ou du « voisinage » attendus, et les résultats sont satisfaisants.
- **soient rapides** ; les temps de calcul que l'on présente prennent en compte les temps de lecture et d'écriture, ainsi que le temps pris par l'ensemble des traitements directement ou indirectement liés à notre optimisation. Ces temps de calculs sont acceptables, et assez courts par rapport aux résultats obtenus et aux volume des données qui sont utilisées dans les optimisations.
- **proposent des critères précis de validation des résultats** ; nous avons proposé tout d'abord comme critère une comparaison de la valeur finale de la fonctionnelle après optimisation des paramètres qui soit inférieure à un certain seuil, ce qui a été vérifié pour l'ensemble des résultats présentés. Aussi, nous avons des précisions sur les valeurs finales des paramètres concernés, qui permettent de valider ou non ces valeurs optimisées des paramètres : il convient de noter que ces précisions ne permettent pas de valider l'affinement (c'est la valeur finale de la fonctionnelle qui le permet), mais donnent une indication sur la confiance que l'on peut accorder aux paramètres optimisés pour une utilisation ultérieure. Enfin, différentes métriques ont aussi été proposées pour valider les résultats d'optimisation sur les jeux de données simulés.

Dans l'ensemble, les objectifs qui ont été fixés au début de la thèse ont été respectés : toutefois, plusieurs voies d'améliorations sont envisageables.

6.2 Voies d'améliorations possibles

Pour les différentes méthodes d'affinements que l'on a proposé, les résultats d'optimisation des chapitres 3 et 4 ont utilisés les mêmes jeux de données, qui sont des nuages de points plutôt « petits ». Lors d'une cartographie mobile, le volume de données est bien plus important que ce que l'on a présenté, et les jeux de données que l'on a utilisé sont des portions d'acquisitions plus importantes. Ces jeux de données ont été utilisés parce que cela permettait d'avoir des temps de calculs faibles, et de paramétrer au mieux les différentes optimisations. Des volumes de données plus importants (quelques centaines de millions de points, ce qui équivaudrait à une acquisition de quelques kilomètres) permettraient d'améliorer l'observabilité sur les paramètres que l'on optimise : en effet, nous avons vu dans le chapitre 3, section 3.5 qu'une acquisition où la trajectoire du véhicule est fortement contrainte (virages et variation d'altitude) permettait d'apporter de l'observabilité aux différents paramètres de calibrage. Aussi, un jeu de données représentant une acquisition à « grande échelle » permettrait lors des tests d'avoir une bonne observabilité, et de correctement optimiser les paramètres de calibrage, en plus d'affiner plus précisèment le nuage de points.
Une première amélioration possible irait dans le sens de ce que l'on vient de présenter, et viserait à accélérer les calculs : les optimisations proposées sont purement exécutées sur CPU, parallélisées sur l'ensemble des processeurs disponibles pour accélérer les algorithmes, mais avec le « faible » nombre de processeurs, les temps de calculs ne sont pas optimaux, notamment avec les remplissages des matrices concernées par nos optimisations, et une programmation sur GPU permettrait de considérablement accélérer nos algorithmes.

Le chapitre 5 a présenté une méthode d'affinement par optimisation de la trajectoire : des tests nous ont permis de déterminer qu'il n'était pas crucial d'optimiser les paramètres angulaires de la trajectoire, car ceux-ci sont assez précis, leurs observabilités étant bien meilleures que pour les paramètres de translations, et c'est pour cela que nous n'avons traité que de l'optimisation des paramètres de translations. Aussi, ces travaux n'ont pas pu être autant développés que les méthodes

d'optimisations présentées dans les chapitres 3 et 4 : nous avons présenté des résultats sur un jeu de données réelles de volume plus important (50 millions de points environ), mais avec un nombre de temps de contrôle assez faible par rapport au nombre de points du nuage, 50 seulement.
Comme pour les optimisations des paramètres extrinsèques et intrinsèques, une première amélioration serait de programmer l'algorithme sur GPU pour accélérer les temps de calculs : cela permettrait aussi de tester notre optimisation avec un plus grand nombre de temps de contrôle, et d'étudier l'influence du nombre de temps de contrôle sur les résultats d'optimisation.
Une deuxième piste d'amélioration serait de prendre en compte les rotations de la trajectoire dans l'affinement proposé : en effet, nous avons pu voir avec le résultat d'optimisation sur le jeu de données réelles que l'affinement proposé permet de recaler les données entre elles et de correctement affiner le nuage de points, mais le résultat ne semble pas optimal. La vue de dessus des résultats d'optimisation indique clairement que l'affinement n'est pas complet, et qu'il est possible d'améliorer le résultat, notamment en ajoutant les rotations à l'optimisation de la trajectoire.

6.3 Conclusion générale

Pour chaque contribution que nous avons proposée, nous avons cherché à répondre à notre problématique principale, et à respecter des contraintes de robustesse et de rapidité. Les objectifs fixés ont été respectés, et les optimisations que l'on a proposées permettent de correctement affiner les relevés laser issus de LIDARs multi-couches. Ces affinements peuvent être utiles pour de nombreuses applications : segmentation et classification des nuages ; mesures dans les nuages. Plus généralement, les affinements que l'on propose permettent d'améliorer la qualité d'acquisitions sans informations a priori sur celles-ci, et de les préparer pour des traitements ultérieurs comme de la classification par exemple.

Annexe A

Publications

Sommaire

A.1 Conferences internationales avec comite de relecture

1. ***Target-Free Extrinsic Calibration of a Mobile Multi-Beam LIDAR System***, Houssem Nouira, Jean-Emmanuel Deschaud, François Goulette, Laserscanning, Geospatial week 2015, Sep 2015, La grande Motte, France. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume II-3/W5*
2. ***Point Cloud Refinement with a Target-Free Intrinsic Calibration of a mobile Multi-Beam LIDAR System***, Houssem Nouira, Jean-Emmanuel Deschaud, François Goulette, ISPRS Congress, July 2016, Prague, Czech Republic, *International Society for Photogrammetry and Remote Sensing*

A.2 Article de revue internationale avec comité de relecture

Point Cloud Refinement with a Self-Calibration of a Mobile Multi-Beam LIDAR System, Houssem Nouira, Jean-Emmanuel Deschaud, François Goulette, The Photogrammetric Record (Article de revue accepté pour publication le 05 janvier 2017)

A.3 Autre article

Experimental Assessment of the Quanergy M8 LIDAR Sensor, Marie-Anne Mittet, Houssem Nouira, Xavier Roynard, François Goulette, Jean-Emmanuel Deschaud, ISPRS Congress, July 2016, Prague, Czech Republic, *International Society for Photogrammetry and Remote Sensing*

Annexe B

Compléments au chapitre 3

Sommaire

B.1 Justification du choix de certains paramètres pour notre optimisation

La figure B.1 donne l'évolution de l'énergie pour une optimisation effectuée avec différentes valeurs de N sur le nuage simulé #3. On peut voir sur la figure que l'optimisation converge le plus vite lorsque N=2, et que la valeur d'énergie est le plus faible aussi lorsque N=2.

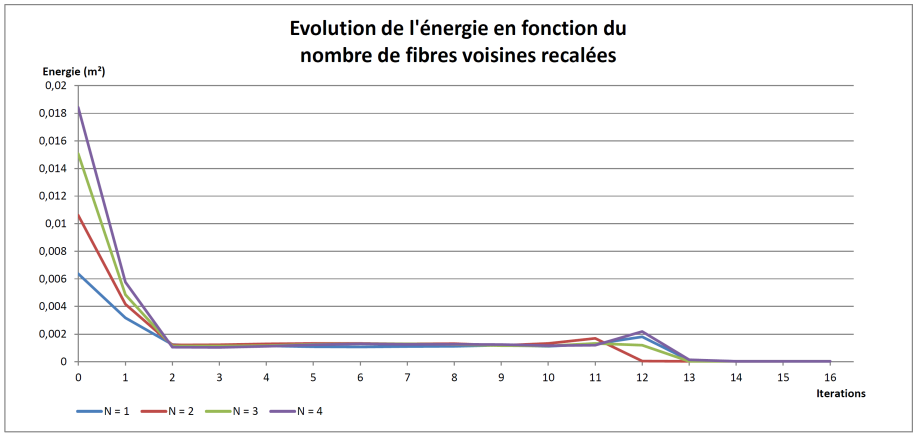


Figure B.1 – Évolution de l'énergie pour différentes valeurs de N

La figure B.2 montre que plus le nombre de voisins est grand, plus l'algorithme converge rapidement. Cependant, la valeur de l'énergie est aussi plus grande, bien que l'ordre de grandeur soit le même (les valeurs finales vont de $1.52 * 10^{-05} m^2$ pour un nombre de voisins de 30 à $2.66 * 10^{-05} m^2$ pour un nombre de voisins de 200). Un nombre de voisins de 150 permet ainsi d'avoir une convergence rapide en terme d'itérations.

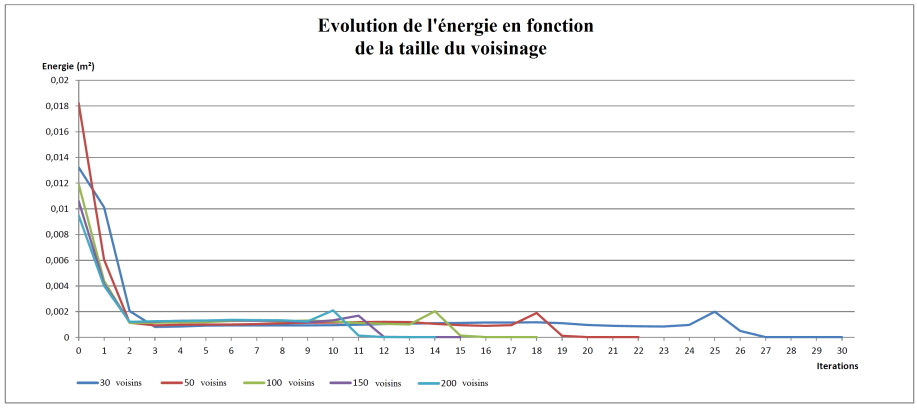


FIGURE B.2 – Évolution des énergies pour plusieurs tailles de voisinage pour le calcul des normales

B.2 Intermédiaires de calcul pour l'optimisation des paramètres de calibrage extrinsèque

Dans cette section, nous allons présenter les intermédiaires de calculs qui nous ont permis de trouver la fonction objectif (3.7).

Nous partons de notre énergie définie avec l'équation (3.6) :

$$J(R,T) = \frac{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_{k} w_{i,j,k} * d^2_{i,j,k}(R,T)}{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_{k} w_{i,j,k}} \tag{B.1}$$

avec :

$$\begin{cases} d_{i,j,k}(R,T) = n^T_{i,k} \times (p_{i,k}(R,T) - m_{j,k}(R,T)) \\ R(\alpha,\beta,\gamma) \text{ et } T(t_x,t_y,t_z) \\ p_{i,k}(R,T) = R_{nav}(p'_{i,k}) \times (R \times p'_{i,k} + T) + T_{nav}(p'_{i,k}) \\ m_{j,k}(R,T) = R_{nav}(m'_{j,k}) \times (R \times m'_{j,k} + T) + T_{nav}(m'_{j,k}) \end{cases}$$

Comme indiqué, nous linéarisons l'énergie (3.6) pour simplifier le problème, et la distance $d_{i,j,k}(R(\alpha,\beta,\gamma),T(t_x,t_y,t_z))$ devient une distance $d_{i,j,k}(R(\alpha+\delta\alpha,\beta+\delta\beta,\gamma+\delta\gamma),T(t_x+\delta t_x,t_y+\delta t_y,t_z+\delta t_z))$.

$R(\alpha,\beta,\gamma)$ est la matrice de rotation composée des angles d'Euler, avec $R = R_z(\gamma) \times R_y(\beta) \times R_x(\alpha)$, produit des 3 matrices de rotations autour de chacun des axes du repère body. Avec la linéarisation, $R_x(\alpha)$ devient avec un développement de Taylor au premier ordre :

$$R_x(\alpha+\delta\alpha) \simeq \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) - sin(\alpha)\delta\alpha & -sin(\alpha) - cos(\alpha)\delta\alpha \\ 0 & sin(\alpha) + cos(\alpha)\delta\alpha & cos(\alpha) - sin(\alpha)\delta\alpha \end{pmatrix}$$

En utilisant les formules trigonométriques, on trouve :

$$R_x(\alpha+\delta\alpha) \simeq R_x(\alpha) + \delta\alpha \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha+90) & -sin(\alpha+90) \\ 0 & sin(\alpha+90) & cos(\alpha+90) \end{pmatrix} + \delta\alpha \begin{pmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

On obtient finalement :

$$R_x(\alpha+\delta\alpha) \simeq R_x(\alpha) + \delta\alpha * R_x(\alpha+90) + \delta\alpha * A_x$$

avec : $A_x = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$

De même, on obtient pour les rotations autour des axes y et z :

$$\begin{cases} R_y(\beta + \delta\beta) \simeq R_y(\beta) + \delta\beta * R_y(\beta + 90) + \delta\beta * A_y \\ R_z(\gamma + \delta\gamma) \simeq R_z(\gamma) + \delta\gamma * R_y(\gamma + 90) + \delta\gamma * A_z \end{cases}$$

avec : $A_y = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ *et* $A_z = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{pmatrix}$

En reprenant la matrice des angles d'Euler R, et avec les développements précédents, on obtient :

$$\begin{aligned} R(\alpha + \delta\alpha, \beta + \delta\beta, \gamma + \delta\gamma) \simeq\ & R_z(\gamma) * R_y(\beta) * R_x(\alpha) \\ & + \delta\alpha * [R_z(\gamma) \times R_y(\beta) \times (R_x(\alpha + 90) + A_x)] \\ & + \delta\beta * [R_z(\gamma) \times (R_y(\beta + 90) + A_y) \times R_x(\alpha)] \\ & + \delta\gamma * [(R_z(\gamma + 90) + A_z) \times R_y(\beta) \times R_x(\alpha)] \end{aligned}$$

que l'on peut écrire sous la forme :

$$\begin{aligned} R(\alpha + \delta\alpha, \beta + \delta\beta, \gamma + \delta\gamma) \simeq\ & R(\alpha, \beta, \gamma) \\ & + R_\alpha * \delta\alpha \\ & + R_\beta * \delta\beta \\ & + R_\gamma * \delta\gamma \end{aligned}$$

Annexe C

Résultats supplémentaires pour l'optimisation des paramètres extrinsèques

Sommaire

C.1 Résultat supplémentaire sur un jeu de données simulées

Comme pour le jeu de données simulées #2, la figure C.1 donne l'évolution des énergies pour les deux optimisations comparées du nuage #3, et le résultat est meilleur avec l'utilisation des attributs : l'énergie évolue d'une valeur de 106.11 cm^2 à une valeur de 0.24 cm^2 sans l'utilisation des attributs, et évolue d'une valeur de 82.89 cm^2 à une valeur de 0.04 cm^2 avec. Là aussi, la convergence est plus rapide avec l'utilisation des attributs de dimensionnalité : à chaque mise à jour des attributs, l'énergie diminue fortement et le poids total augmente fortement.
La figure C.2 présente le nuage avec les paramètres initiaux en haut, et le même nuage avec les paramètres optimisés en dessous : on voit une nette amélioration de la structure du nuage, qui a été affiné. La table C.2 donne l'erreur initiale entre la vérité terrain et les paramètres initiaux, ainsi que la différence finale entre la vérité terrain et les paramètres extrinsèques optimisés : comme avec le nuage #2, les erreurs finales sont très petites, du même ordre de grandeur, avec et sans utilisation des attributs de dimensionnalité, sauf pour le paramètre de translation en z : l'erreur entre le paramètre optimisé et la vérité terrain n'a pas changé, bien que l'énergie finale soit très faible et que sur la figure C.2, aucune erreur ne soit visible selon l'axe z. Ce problème vient du fait qu'il n'y a pas de variation d'altitude dans le nuage, ce qui rend la direction verticale non observable : ce problème est présenté plus en détail dans la section 3.5, mais un résultat de ce manque d'observabilité est que bien que l'on a dans l'ensemble des paramètres correctement optimisés, sauf pour la translation en z, et cela n'affecte pas le résultat d'optimisation ou la valeur d'énergie finale. La table C.1 donne la valeur des précisions pour chacun des paramètres extrinsèques, et on peut voir que les précisions sont bonnes, sauf pour le paramètre de translation z, ce qui confirme l'observation faite avec le nuage #2. Enfin, pour les temps de calcul, nous avons des temps similaires à ceux présentés pour le nuage simulé #2, vu que le nombre de points des jeux de données est le même, seul le nombre d'appariements validés pour l'optimisation change entre les 2 nuages.

	σ_{t_x}(cm)	σ_{t_y}(cm)	σ_{t_z}(cm)	$\sigma_\alpha(^\circ)$	$\sigma_\beta(^\circ)$	$\sigma_\gamma(^\circ)$
Nuage de points simulé #3	5.49	4.26	$1.00 * 10^{20}$	0.12	0.15	0.20

Table C.1 – Précision des paramètres extrinsèques pour le nuage de points simulé #3

	t_x(cm)	t_y(cm)	t_z(cm)	roulis $\alpha(^\circ)$	tangage $\beta(^\circ)$	lacet $\gamma(^\circ)$
Erreurs initiales par rapport à la vérité terrain	-150.00	250.00	-200.00	5.00	-7.00	-5.50
Différence entre la vérité terrain et notre résultat d'optimisation (sans utilisation de la dimensionnalité)	-0.046	-0.024	-200.000	0.000	0.001	0.001
Différence entre la vérité terrain et notre résultat d'optimisation (avec utilisation de la dimensionnalité)	0.091	-0.104	-200.000	0.000	-0.000	0.003

TABLE C.2 – Erreurs sur les paramètres extrinsèques avant et après optimisation par rapport à la vérité terrain pour le nuage #3

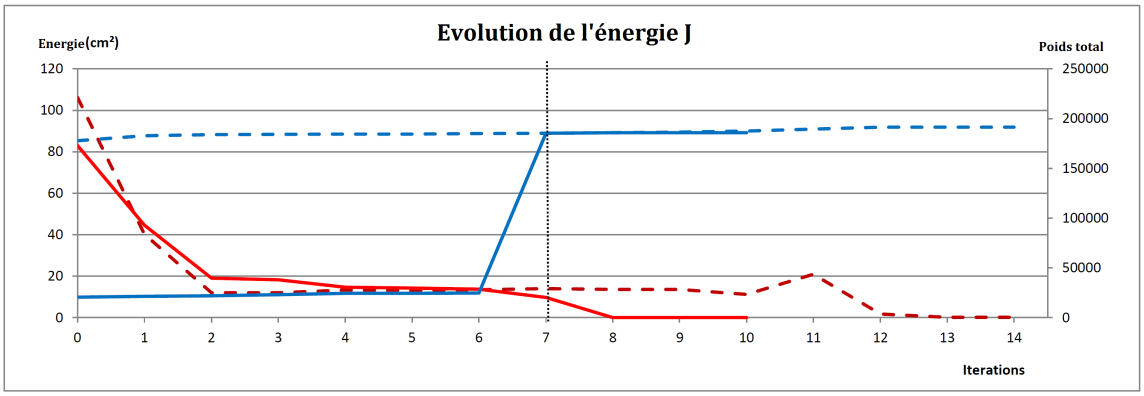


FIGURE C.1 – Évolution de l'énergie au cours de l'optimisation pour le nuage simulé #3. La ligne rouge pleine représente l'évolution de l'énergie avec l'utilisation des attributs de dimensionnalité ; la ligne rouge pointillé représente la même énergie sans l'utilisation des attributs ; la ligne bleue pleine représente l'évolution du poids total utilisé pour normaliser notre énergie, avec l'utilisation des attributs de dimensionnalité ; la ligne bleue pointillé représente la même donnée, mais sans l'utilisation des attributs de dimensionnalité ; enfin, les lignes noires verticales marquent la mise à jour des attributs de dimensionnalité, toutes les 7 itérations.

C.2 Résultat supplémentaire sur un jeu de données réelles

Pour le nuage #5, on a une configuration de type « couloir » : il y a deux façades parallèles entre elles, et le véhicule se déplace en ligne droite entre ces deux façades. Il y a aussi de nombreux éléments non plans, comme les arbres et leur feuillages ou encore les câbles électriques. L'observabilité sur ce type de jeu de données est assez mauvaise dans plusieurs directions, notamment pour les translations dans les trois directions vu qu'il y a peu de variation dans ces mêmes directions. Malgré cela, on peut voir avec la figure C.4 que l'on affine correctement le nuage de points : là aussi, un seul résultat d'optimisation est présenté car la différence entre le résultat des deux optimisations n'est pas visible à l'oeil. La figure C.3 présente l'évolution des énergies pour le nuage #5, avec et sans utilisation des attributs de dimensionnalité : l'énergie varie d'une valeur de 205.79 cm^2 à une valeur de 60.88 cm^2 sans l'utilisation des attributs, et d'une valeur de 125.50 cm^2 à une valeur de 46.22 cm^2 avec. Comme pour le nuage réel #4, l'énergie finale est plus faible avec l'utilisation des attributs de

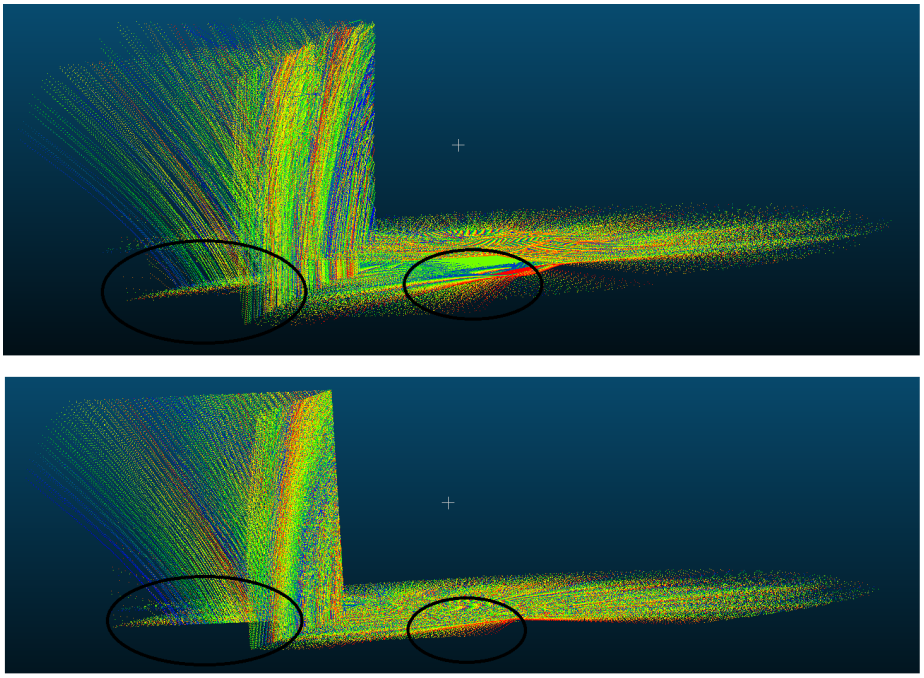


FIGURE C.2 – Nuage de points simulé #3 : en haut, nuage avec les paramètres initiaux ; en dessous, le résultat de notre optimisation. Les deux images sont vues avec le même point de vue.

dimensionnalité, et le seuil de 75 cm^2 est aussi respecté. La table C.4 donne les précisions pour les paramètres extrinsèques de ce nuage : pour les paramètres de translations, ces précisions ne sont pas bonnes, mais restent acceptables pour les paramètres de rotations : cela confirme les hypothèses faites vis à vis de la structure du nuage et de l'observabilité des paramètres qui en découlent. Parmi les trois rotations, c'est la rotation de tangage qui a la meilleure observabilité, et la table C.3 donne des résultats d'optimisation qui vont dans le sens des précisions mesurées : pour les translations, les paramètres optimisés sont trop éloignés des valeurs mesurées sur le véhicule, mais cela n'affecte pas la qualité finale du nuage ; pour les rotations, c'est la rotation de tangage qui semble être le mieux optimisée, puisque la valeur reste assez proche de celle que l'on a mesuré. Avec ce nuage de points, on peut voir les effets d'une mauvaise observabilité des données sur notre optimisation : le nuage est correctement affiné, ce qui est le résultat que l'on voulait, mais dans des cas d'applications ou le but est de retrouver des paramètres extrinsèques proches des paramètres réels, cela n'est pas possible à moins de contraindre le problème. Enfin, les temps de calculs sont similaires à ceux présentés pour le nuage #4, de un peu moins de 2 minutes sans utilisation des attributs de dimensionnalité et de environ 5 minutes avec.

	t_x(m)	t_y(m)	t_z(m)	roulis $\alpha(^\circ)$	tangage $\beta(^\circ)$	lacet $\gamma(^\circ)$
Valeurs des paramètres extrinsèques initiale	0.00	0.00	0.00	0.00	-45.00	90.00
Paramètres mesurés à la main sur notre véhicule d'acquisition	-0.21	-1.22	0.95	0.00	-60.00	90.00
Paramètres obtenus après optimisation (sans dimensionnalité)	-1.01	0.26	-3.31	-1.97	-59.47	92.27
Paramètres obtenus après optimisation (avec dimensionnalité)	-1.07	0.46	-3.39	-0.05	-59.51	90.61

TABLE C.3 – Erreurs sur les paramètres extrinsèques après optimisation pour le nuage #5

	σ_{tx}(cm)	σ_{ty}(cm)	σ_{tz}(cm)	$\sigma_\alpha(^\circ)$	$\sigma_\beta(^\circ)$	$\sigma_\gamma(^\circ)$
Nuage réel #5	180.98	131.37	177.04	4.78	0.51	4.14

TABLE C.4 – Précision des paramètres extrinsèques pour le nuage de points réel #5

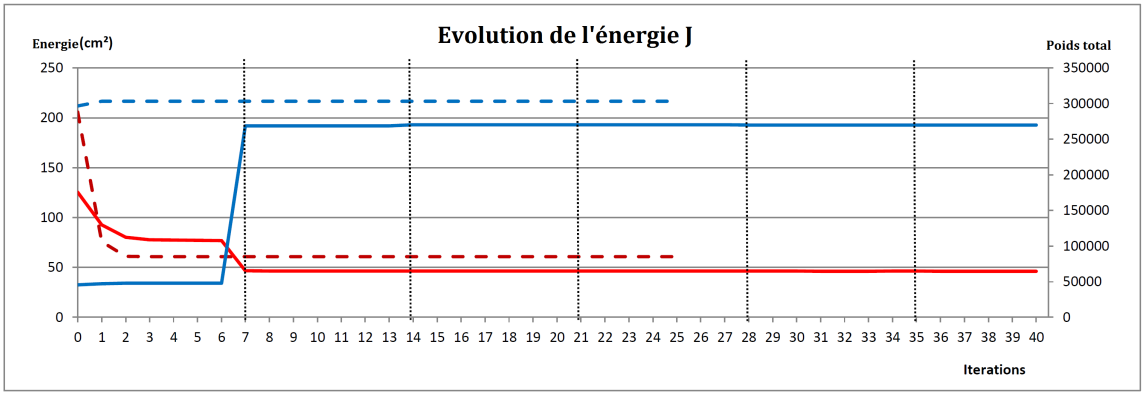


FIGURE C.3 – Évolution de l'énergie au cours de l'optimisation pour le nuage réel #5. La ligne rouge pleine représente l'évolution de l'énergie avec l'utilisation des attributs de dimensionnalité ; la ligne rouge pointillé représente la même énergie sans l'utilisation des attributs ; la ligne bleue pleine représente l'évolution du poids total utilisé pour normaliser notre énergie, avec l'utilisation des attributs de dimensionnalité ; la ligne bleue pointillé représente la même donnée, mais sans l'utilisation des attributs de dimensionnalité ; enfin, les lignes noires verticales marquent la mise à jour des attributs de dimensionnalité, toutes les 7 itérations.

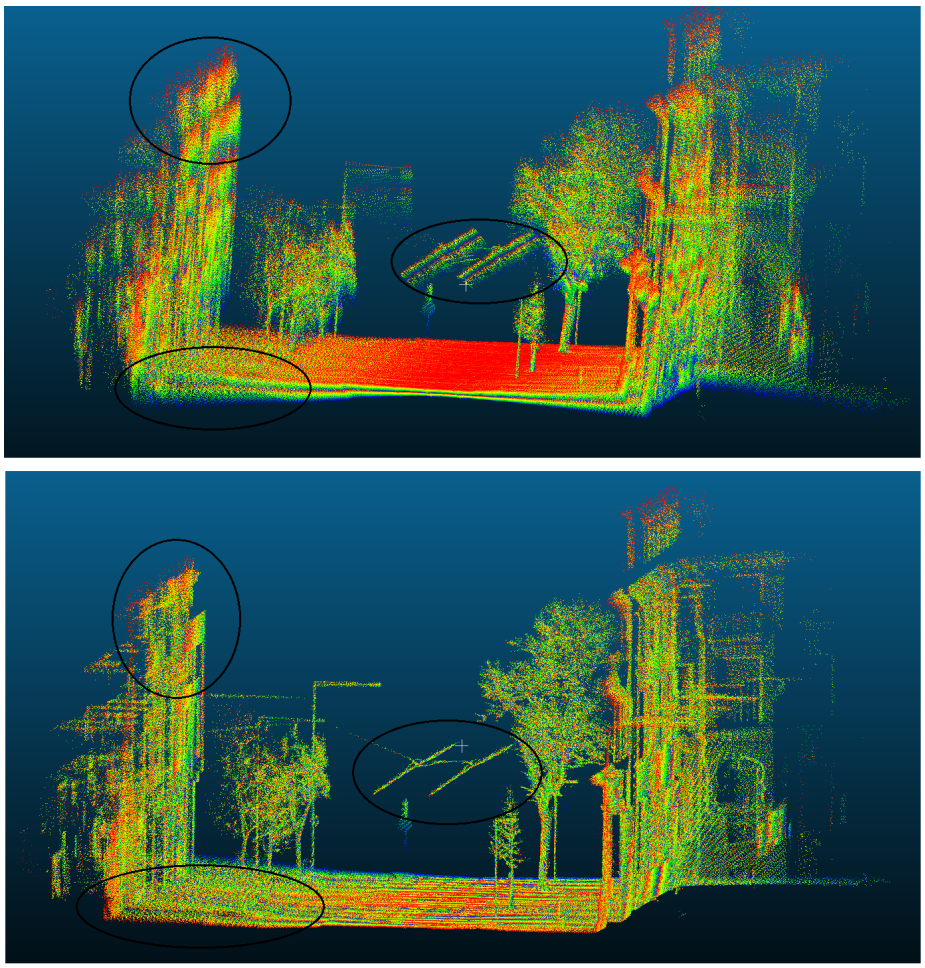


FIGURE C.4 – Nuage de points réel #5 : en haut, nuage avec les paramètres de calibrage extrinsèque initiaux ; en dessous, le résultat de notre optimisation. Les deux images sont vues depuis le même point

Annexe D

Résultats supplémentaires pour l'optimisation des paramètres intrinsèques

Sommaire

D.1 Résultat supplémentaire pour l'optimisation des paramètres intrinsèques sur un jeu de données simulées

La figure D.2 présente visuellement le résultat de l'optimisation, et on peut voir que l'on a bien affiné le nuage point avec l'optimisation des paramètres intrinsèques, comme par exemple avec les surfaces planaires qui sont correctement planes. La figure D.1 présente l'évolution de l'énergie au cours de l'optimisation pour le nuage #3 : on peut voir qu'elle a une évolution strictement décroissante. L'énergie évolue d'une valeur de 380.94 cm^2 à une valeur de 0.16 cm^2, qui est très faible. La table D.1 présente les erreurs sur les offsets intrinsèques avant et après optimisation : les erreurs sont très proches de 0 après optimisation, ce qui est le résultat que l'on attendait, et le résultat peut aussi être validé.

	$\Delta\rho(cm)$	$\Delta\theta(^\circ)$	$\Delta\phi(^\circ)$	$\Delta H_z(cm)$
Erreurs initiales	10	2.5	3	10
Erreurs après notre optimisation	$2.55 * 10^{-2}$	$7.97 * 10^{-4}$	$1.27 * 10^{-3}$	$6.62 * 10^{-2}$

Table D.1 – Erreurs entre les offsets et la vérité terrain pour le nuage simulé #3

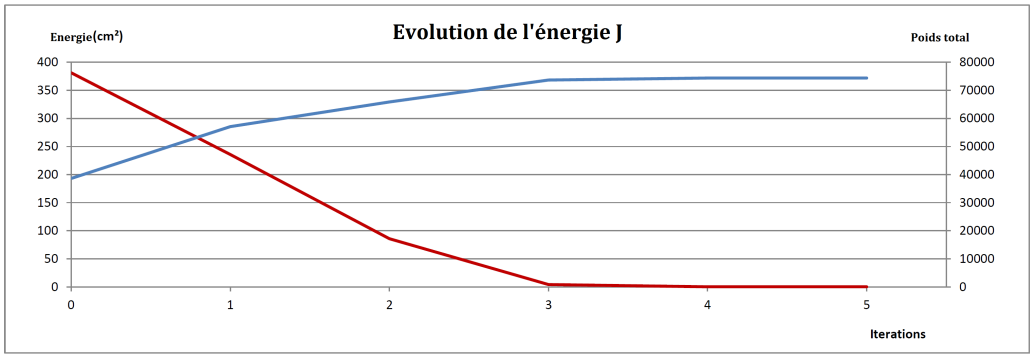


FIGURE D.1 – Évolution de l'énergie pour le nuage simulé #3 : en rouge, nous avons la courbe d'énergie en fonction des itérations ; en bleu, nous avons la courbe représentant le poids total utilisé pour normaliser l'énergie.

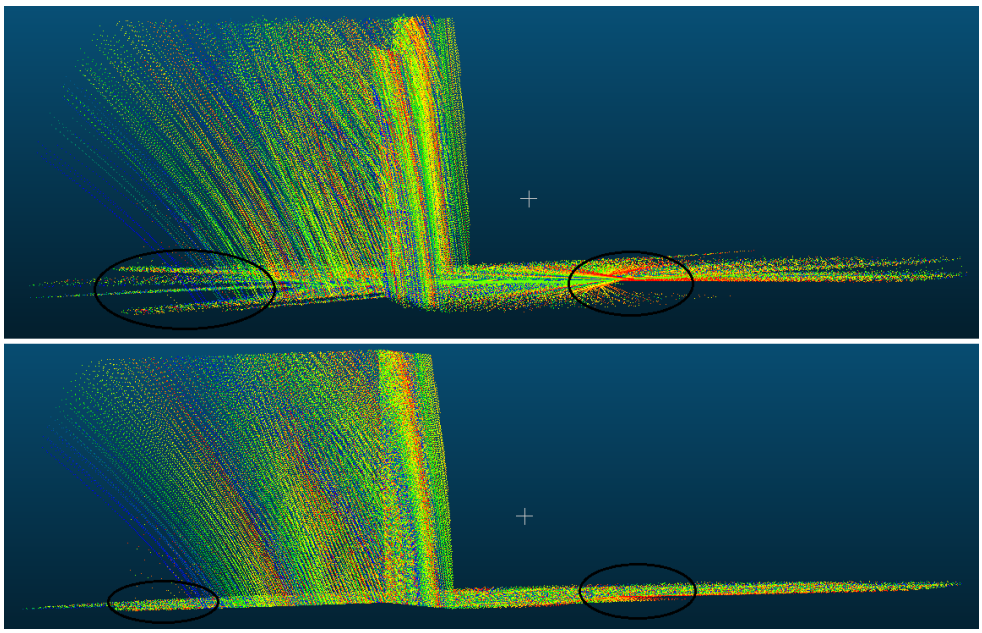


FIGURE D.2 – Nuage simulé #3 : en haut, nuage avant optimisation des paramètres intrinsèques ; en dessous, le même nuage après optimisation des paramètres intrinsèques. Les deux images sont vus depuis la même position.

D.2 Résultat supplémentaire pour l'optimisation des paramètres intrinsèques sur un jeu de données réelles

Pour le nuage réel #5, la figure D.4 présente les améliorations apportées au nuage de points après optimisation des paramètres intrinsèques : le nuage est correctement affiné après optimisation des paramètres. La figure D.3 présente l'évolution de l'énergie pendant cette optimisation, qui a la même allure que pour le nuage #4 : l'énergie évolue d'une valeur de 293.81 cm^2 à une valeur de 45.70 cm^2, ce qui permet de valider le résultat d'optimisation.

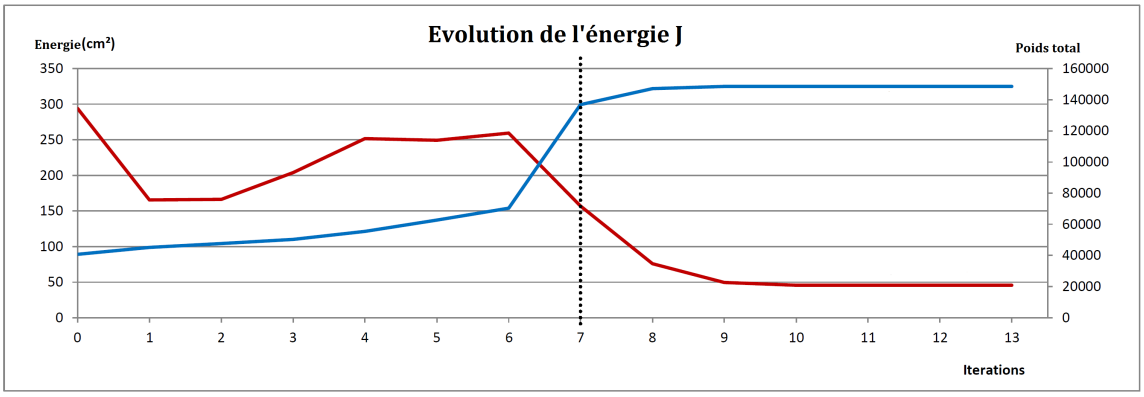


FIGURE D.3 – Évolution de l'énergie pour le nuage simulé #5 : en rouge, nous avons la courbe d'énergie en fonction des itérations ; en bleu, nous avons la courbe représentant le poids total utilisé pour normaliser l'énergie ; les lignes noires verticales mettent en évidence les itérations où les attributs de dimensionnalité sont mis à jour.

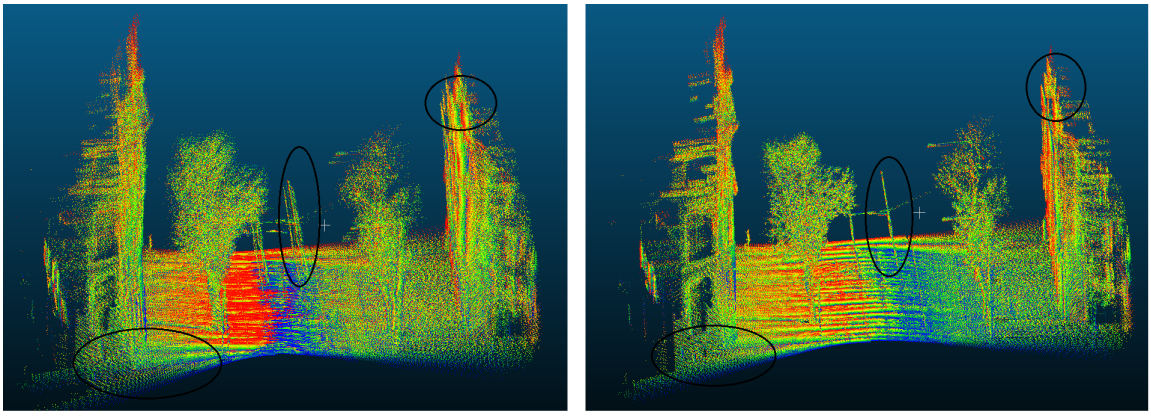


FIGURE D.4 – Nuage réel #5 : à gauche, nuage avant optimisation des paramètres intrinsèques ; à droite, le même nuage après optimisation des paramètres intrinsèques. Les deux images sont vues depuis le même point.

D.3 Résultat supplémentaire d'optimisation conjointe des paramètres intrinsèques et extrinsèques sur un jeu de données simulées

La figure D.5 présente les résultats des optimisations sur le nuage #3 : nous pouvons remarquer que le meilleur résultat d'optimisation est obtenu lorsque l'ensemble des paramètres de calibrage sont considérés. Pour le nuage #3, la figure D.6 présente l'évolution des énergies pour les deux approches d'optimisations comparées. Nous pouvons faire les mêmes observations que pour le nuage #2 : l'optimisation conjointe des paramètres de calibrage met plus d'itérations pour converger, mais l'énergie finale est plus petite. Pour l'optimisation des paramètres extrinsèques seuls, l'énergie évolue d'une valeur de 193.16 cm^2 à une valeur de 162.20 cm^2, alors que pour l'optimisation conjointe, l'énergie évolue d'une valeur de 193.16 cm^2 à une valeur de 0.49 cm^2. La table D.2 donne les erreurs entre la vérité terrain et les paramètres de calibrages : cette fois-ci, les paramètres extrinsèques, sont mieux optimisés lorsque l'ensemble des paramètres sont considérés dans l'optimisation ; seul le paramètre de translation en z n'est pas correctement optimisé, mais cela est dû au problème d'observabilité que nous avons évoqué dans le chapitre 3, et aussi au fait que l'altitude ne varie pas

dans le nuage. Cette table donne aussi les erreurs sur les paramèrtes intrinsèques définies dans la section 4.3.2 : avec l'optimisation de l'ensemble des paramètres, ces erreurs sont considérablement réduites et assez proche de 0, comme pour le nuage #2. L'énergie est aussi la plus petite dans ce cas d'optimisation.
Enfin, concernant les temps de calcul, ils sont du même ordre de grandeur que pour le nuage #2, environ 5 minutes pour l'optimisation des paramètres extrinsèques seuls, et 20 minutes pour l'optimisation conjointe des paramètres extrinsèques et intrinsèques : là aussi, ces temps de calculs sont acceptables.

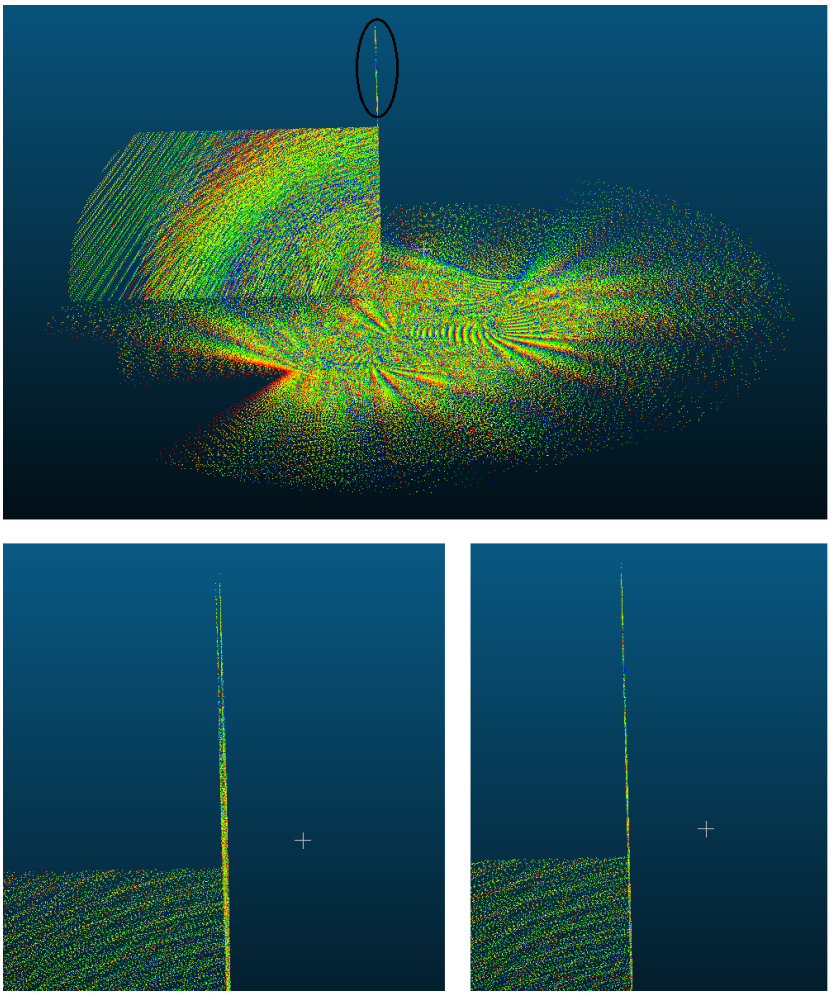


FIGURE D.5 – Nuage simulé #3 : en haut, nous avons le nuage avec des paramètres intrinsèques et extrinsèques corrects, correspondant à la vérité terrain ; en bas à gauche, nous montrons un morceau de plan après l'optimisation des paramètres extrinsèques seuls ; en bas à droite, nous montrons le même morceau de plan, mais après l'optimisation conjointe des paramètres intrinsèques et extrinsèques.

	t_x(cm)	t_y(cm)	t_z(cm)	$\alpha(^\circ)$	$\beta(^\circ)$	$\gamma(^\circ)$	$\Delta\rho(cm)$	$\Delta\theta(^\circ)$	$\Delta\phi(^\circ)$	$\Delta H_z(cm)$
Erreurs initiales par rapport à la vérité terrain	-50.00	60.00	-80.00	2.50	3.00	-2.00	2	0.3	0.2	3
Différence entre la vérité terrain et notre résultat d'optimisation (Optimisation des paramètres extrinsèques seuls)	-1.678	-0.817	-80.000	-0.537	0.024	0.453	-	-	-	-
Différence entre la vérité terrain et notre résultat d'optimisation (Optimisation conjointe des paramètres extrinsèques et intrinsèques)	-0.047	0.145	-80.000	0.174	0.121	0.008	0.048	0.050	0.035	0.277

TABLE D.2 – Erreurs entre les paramètres de calibrages et la vérité terrain pour le nuage simulé #3

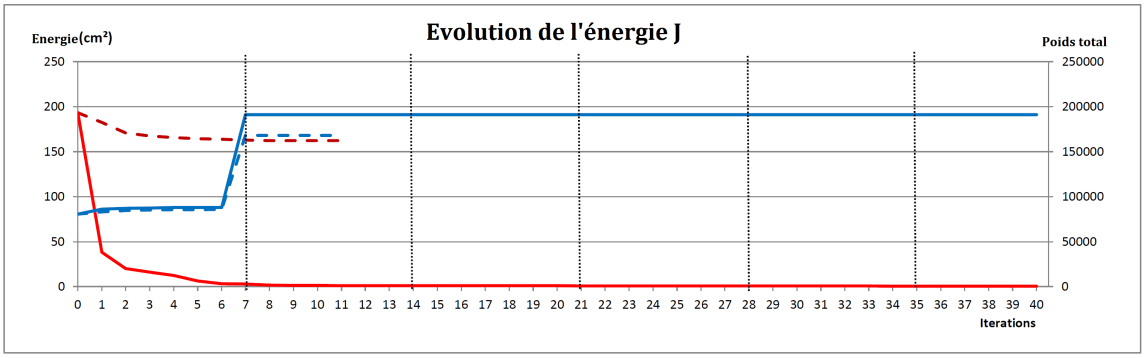


FIGURE D.6 – Évolution des énergies pour les optimisations comparées sur le nuage simulé #3. La ligne pleine rouge représente l'évolution de l'énergie avec l'optimisation conjointe des paramètres intrinsèques et extrinsèques ; la ligne pointillé rouge représente l'évolution de l'énergie pour l'optimisation des paramètres extrinsèques seuls ; la ligne bleue pleine représente le poids total avec lequel on normalise l'énergie pour l'optimisation conjointe des paramètres ; la ligne bleue en pointillé représente le poids total avec lequel on normalise l'énergie dans le cas de l'optimisation des paramètres extrinsèques seuls ; enfin, les lignes pointillés verticales noires mettent en évidence les itérations où les attributs de dimensionnalité sont mis à jour.

D.4 Résultat supplémentaire d'optimisation conjointe des paramètres intrinsèques et extrinsèques sur un jeu de données réelles

Un deuxième résultat d'optimisation est présenté avec le jeu de données réelles #5. La figure D.7 donne l'évolution des énergies d'optimisation pour les deux approches, et elle varie d'une valeur de 125.50 cm^2 à une valeur de 46.22 cm^2 pour l'optimisation des paramètres extrinsèques seuls ; dans le cas de l'optimisation de l'ensemble des paramètres de calibrage, l'énergie évolue de la même valeur de départ jusqu'à une valeur de 42.61 cm^2, qui est plus faible pour l'autre type d'optimisation, mais cette fois-ci la différence est moins notable. Ce résultat s'explique avec l'observabilité des paramètres : comme le véhicule a un déplacement rectiligne entre deux surfaces planaires, et que de nombreux éléments non plans sont présents dans le nuage, les paramètres de calibrage sont plus difficilement optimisable. Le résultat est qu'entre les deux approches d'optimisations, les paramètres extrinsèques finaux sont proches les uns des autre comme montré avec la table D.3 : pour les translations, les différences entre les deux optimisations sont inférieures à 10 cm, et pour les rotations, les différences sont inférieure au degré. Là aussi, les erreurs par rapport aux paramètres mesurés sur le véhicule sont très importantes, mais comme l'observabilité est mauvaise dans plusieurs directions, l'énergie finale est la plus faible avec l'optimisation de l'ensemble des paramètres. Concernant les offsets finaux sur les paramètres intrinsèques, ils sont compris entre -4 cm et 3 cm pour l'offset vertical H_z, entre -3 et 8 cm pour l'offset de distance sur ρ, et entre -0.14 °et 0.25 °pour les offsets sur les angles ϕ et θ. Comme pour le nuage #4, les offsets angulaires sont acceptables, mais pour les translations, ils sont légèrement élevés par rapport à ce que l'on pouvait attendre, toujours à cause de la mauvaise observabilité des translations.
Pour les temps de calculs, nous avons un temps de calcul de 6 minutes environ pour l'optimisation des paramètres extrinsèques seuls, et de 25 minutes environ pour l'optimisation de l'ensemble des paramètres.

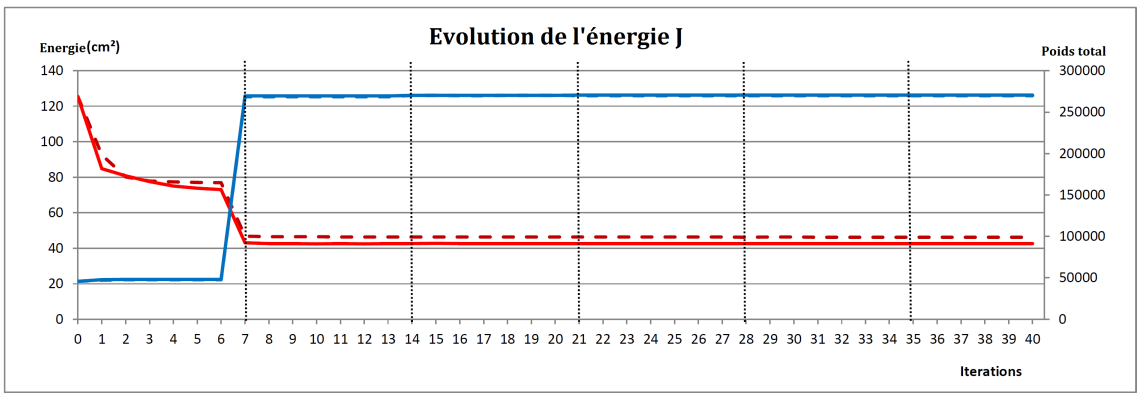


FIGURE D.7 – Évolution des énergies pour les optimisations comparées sur le nuage simulé #5. La ligne pleine rouge représente l'évolution de l'énergie avec l'optimisation conjointe des paramètres intrinsèques et extrinsèques ; la ligne pointillé rouge représente l'évolution de l'énergie pour l'optimisation des paramètres extrinsèques seuls ; la ligne bleue pleine représente le poids total avec lequel on normalise l'énergie pour l'optimisation conjointe des paramètres ; la ligne bleue en pointillé représente le poids total avec lequel on normalise l'énergie dans le cas de l'optimisation des paramètres extrinsèques seuls ; enfin, les lignes pointillés verticales noires mettent en évidence les itérations où les attributs de dimensionnalité sont mis à jour.

	t_x(cm)	t_y(cm)	t_z(cm)	$\alpha(^\circ)$	$\beta(^\circ)$	$\gamma(^\circ)$	$\Delta\rho(cm)$	$\Delta\theta(^\circ)$	$\Delta\phi(^\circ)$	$\Delta H_z(cm)$
Paramètres initiaux	0.00	0.00	0.00	0.00	-45.00	90.00	?	?	?	?
Paramètres après optimisation (Optimisation des paramètres extrinsèques seuls)	-0.21	-1.22	0.95	0.00	-60.00	90.00	-	-	-	-
Paramètres après optimisation (Optimisation conjointe des paramètres extrinsèques et intrinsèques)	-1.07	0.46	-3.39	-0.05	-59.51	90.61	1.01	0.01	0.15	3.10

TABLE D.3 – Paramètres de calibrages pour le nuage réel #5

Bibliographie

[3DR site web, 2017] 3DR site web (2017). https://3dr.com/support/sections/201195363/solo/. Dernier accés le 12-02-2017. (Cité en page 9.)

[Abshire, 2010] Abshire, J. B. (2010). NASA's Space Lidar Measurements of Earth and Planetary Surfaces. (Cité en page 23.)

[Atanacio-jiménez *et al.*, 2011] Atanacio-jiménez, G., Hurtado-ramos, J. B. et González-barbosa, R. (2011). Lidar Velodyne HDL-64E Calibration using Pattern Planes". *International Journal of Advanced Robotic Systems*, pages 70–82. (Cité en page 70.)

[Bailey, 2002] Bailey, T. (2002). *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. Thèse de doctorat, University of Sydney, Australian Centre for Field Robotics. (Cité en page 90.)

[Bailey et Durrant-Whyte, 2006a] Bailey, T. et Durrant-Whyte, H. F. (june 2006a). Simultaneous Localization and Mapping : Part I - State of the Art. *Robotics and Aut. Magazine*. (Cité en page 91.)

[Bailey et Durrant-Whyte, 2006b] Bailey, T. et Durrant-Whyte, H. F. (september 2006b). Simultaneous Localization and Mapping : Part II - State of the Art. *Robotics and Aut. Magazine*. (Cité en page 91.)

[Besl et McKay, 1992] Besl, P. J. et McKay, N. D. (1992). A method for registration of 3D shapes. *Transactions on Pattern Analysis and Machine Intelligence, IEEE,*, Vol. 14:239–256. (Cité en pages 37, 38, 39, 76 et 91.)

[Biber et Straber, 2003] Biber, P. et Straber, W. (2003). The Normal Distributions Transform : A New Approach to Laser Scan Matching. *Proceedings of the 2003 IEEVRSJ, InU. Conference on Intelligent Robots and Systems, Las Vegas. Nevada,*, pages 239–256. (Cité en page 38.)

[Censi, 2008] Censi, A. (2008). An ICP variant using a point-to-line metric. pages 19–25. (Cité en page 40.)

[Chan et Lichti, 2013] Chan, T. O. et Lichti, D. D. (2013). Feature-based self-calibration of Velodyne HDL-32E LiDAR for terrestrial mobile mapping applications. *The 8th International Symposium on Mobile Mapping Technology, Tainan, Taiwan*. (Cité en pages 70 et 72.)

[Chan et Lichti, 2015] Chan, T. O. et Lichti, D. D. (2015). Automatic in Situ Calibration of a Spinning Beam LIDAR System in Static and Kinematic Modes. *Remote Sensing*, pages 10480–10500. (Cité en page 70.)

[Chan *et al.*, 2013] Chan, T. O., Lichti, D. D. et Belton, D. (2013). Temporal Analysis and Automatic Calibration of the Velodyne HDL-32E LiDAR System. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 61–66. (Cité en page 70.)

[Chen *et al.*, 2012] Chen, C.-Y., Chien, H.-J., Huang, P.-S., Hong, W.-B. et Chen, C.-F. (2012). Intrinsic Parameters Calibration for Multi-beam LiDAR Using the Levenberg-Marquardt Algorithm. *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, pages 19–24. (Cité en page 70.)

[Chen et Medioni, 1991] Chen, Y. et Medioni, G. (1991). Object modeling by registration of multiple range images. *Proceedings of the 1991 IEEE International Conference on Robotics and Automation, Sacramento, California,*, Vol. 3:2724–2729. (Cité en pages 37, 38 et 40.)

[Choi *et al.*, 2012] Choi, W.-S., Kim, Y.-S., Oh, S.-Y. et Lee, J. (2012). Fast Iterative Closest Point Framework for 3D LIDAR data in Intelligent Vehicle. *Intelligent Vehicles Symposium Alcalá de Henares, Spain, June 3-7,*. (Cité en page 41.)

[Demantke *et al.*, 2011] Demantke, J., Mallet, C., David, N. et Vallet, B. (2011). Dimensionnality based scale selection in 3d lidar point clouds. (Cité en pages 47 et 51.)

[Désilles *et al.*, 2011] Désilles, G., Mary, A., Feugnet, G., Gutty, F., Pocholle, J.-P. et Schwartz, S. (2011). Gyrométrie interférométrique. *Photoniques, Cahier technique*. (Cité en page 25.)

[Eigen library, 2017] Eigen library (2017). http://eigen.tuxfamily.org/index.php?title=Main_Page. Dernier accés le 12-02-2017. (Cité en page 50.)

[Ellum et El-Sheimy, 2002] Ellum, C. et El-Sheimy, N. (2002). Land-based mobile mapping systems. *Photogrammetric engineering and remote sensing*, pages 13–17. (Cité en page 9.)

[Elseberg *et al.*, 2013] Elseberg, J., Borrmann, D. et Nüchter, A. (2013). Algorithmic Solutions for Computing Precise Maximum Likelihood 3D Point Clouds from Mobile Laser Scanning Platforms. Vol. 5(11):5871–5906. (Cité en page 37.)

[Faro LIDAR sensor, 2017] Faro LIDAR sensor (2017). http://www.faro.com/fr-fr/produits/releve-3d/scanner-laser-3d-faro-focus-3d/apercu. Dernier accés le 12-02-2017. (Cité en page 9.)

[FLANN library, 2017] FLANN library (2017). http://www.cs.ubc.ca/research/flann. Dernier accés le 12-02-2017. (Cité en page 50.)

[Flying eye site web, 2017] Flying eye site web (2017). https://www.flyingeye.fr/produits-services/drones/. Dernier accés le 12-02-2017. (Cité en page 9.)

[Gao et Spletzer, 2010] Gao, C. et Spletzer, J. R. (2010). On-line Calibration of Multiple LIDARs on a Mobile Vehicle Platform. *International Conference on Robotics and Automation, Anchorage, Alaska, IEEE*. (Cité en page 36.)

[Geolocation Technologies, 2017] Geolocation Technologies (2017). https://www.locationsmart.com/cms/resources/gaming-geolocation.pdf. Dernier accés le 12-02-2017. (Cité en page 15.)

[Glennie et Lichti, 2010] Glennie, C. et Lichti, D. D. (2010). Static Calibration and Analysis of the Velodyne HDL-64E S2 for High Accuracy Mobile Scanning. *Remote Sensing*, Vol. 2(6):1610–1624. (Cité en page 70.)

[Goad et Novak, 1991] Goad, C. et Novak, N. (1991). The Ohio State University Mapping System : The Stereo Vision System Component. *Proceedings of the 47th annual meeting, The Institute of Navigation(ION), June 10-12, Willamsburg, VA*, pages 121–124. (Cité en page 9.)

[Goulette *et al.*, 2007] Goulette, F., Nashashibi, F., Abuhadrous, I., Ammoun, S. et Laurgeau, C. (2007). An integrated on-board laser range sensing system for on-the-way city and road modelling. pages 78–83. (Cité en page 11.)

[Gérossier, 2012] Gérossier, F. (2012). *Localisation et cartographie simultanées en environnement extérieur à partir de données issues d'un radar panoramique hyperfréquence.* Thèse de doctorat, Université Blaise Pascal - Clermont II, LASMEA. (Cité en pages 21, 41, 42 et 90.)

[Gyroscope et gyromètres à éléments rotatifs, 2017] Gyroscope et gyromètres à éléments rotatifs (2017). http://www.techniques-ingenieur.fr/base-documentaire/mesures-analyses-th1/grandeurs-mecaniques-42407210/gyroscopes-et-gyrometres-mecaniques-avec-element-rotatif-r1940. Dernier accés le 12-02-2017. (Cité en page 17.)

[Heon-Cheol *et al.*, 2011] Heon-Cheol, L., Seung-Hee, L., Seung-Hwan, L., Tae-Seok, L., Doo-Jin, K., Kyung-Sik, P., Kong-Woo, L. et Beom-Hee, L. (Nov. 23-26, 2011). Comparison and Analysis of Scan Matching Techniques for Cooperative-SLAM. *The 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2011), Songdo ConventiA, Incheon, Korea*. (Cité en page 91.)

[Histoire de la cartographie, 2017] Histoire de la cartographie (2017). http://http://le-cartographe.net/dossiers-carto/histoire-de-la-cartographie/55-precurseurs. Dernier accés le 12-02-2017. (Cité en page 1.)

[Hokuyo LIDAR sensor, 2017] Hokuyo LIDAR sensor (2017). http://www.hokuyo-aut.jp/02sensor/07scanner/download/products/. Dernier accés le 12-02-2017. (Cité en page 24.)

[Horn, 1987] Horn, B. (1987). Closedform solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4:629–642. (Cité en page 40.)

[Huang et Barth, 2009] Huang, L. et Barth, M. (2009). A novel Multi-Planar LIDAR and Computer Vision Calibration Procedure Using 2D Patterns for Automated Navigation. *Intelligent Vehicles Symposium, Xi'an, China, IEEE*. (Cité en pages 11, 22 et 35.)

[Huang *et al.*, 2013] Huang, P. S., Hong, W. B., Chien, H. J. et Chen, C. Y. (2013). Extrinsic Calibration of a Multi-Beam LIDAR System with improved Intrinsic Laser Parameters using V-shaped Planes and Infrared Images. *IVMSP Workshop, 2013 IEEE 11th, Seoul, South Korea*. (Cité en pages 37 et 70.)

[Kannala et Brandt, 2006] Kannala, J. et Brandt, S. (2006). A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1335–1340. (Cité en page 28.)

[Lategahn *et al.*, 2011] Lategahn, H., Geiger, A. et Kitt, B. (2011). Visual SLAM for autonomous ground vehicles. *2011 IEEE International Conference on Robotics and Automation*, pages 1732–1737. (Cité en page 91.)

[Leica GNSS sensor, 2017] Leica GNSS sensor (2017). http://leica-geosystems.com/fr-fr/products/gnss-systems. Dernier accés le 12-02-2017. (Cité en page 16.)

[Levinson et Thrun, 2010] Levinson, J. et Thrun, S. (2010). Unsupervised Calibration for Multi-beam Lasers. *International Symposium on Experimental Robotics*. (Cité en pages 37, 43, 44, 53 et 70.)

[Liao *et al.*, 2013] Liao, H.-M., Liu, H.-S. et Chen, C. (2013). Develop Backpack Mobile Mapping System Based Open Source Softaware and Hardware Platform. *8th International Symposium on Mobile Mapping Technology, 1-3 May, Tainan, Taiwan*. (Cité en page 10.)

[Lin *et al.*, 2013] Lin, C.-c., Liao, Y.-d. et Luo, W.-j. (2013). Calibration Method for Extending Single-Layer LIDAR to Multi-layer LIDAR. *Proceedings of the 2013 IEEE/SICE International Symposium on System Integration, Kobe, Japan, December 15-17*, pages 677–681. (Cité en pages 23 et 36.)

[Lu et Milios, 1997] Lu, F. et Milios, E. (1997). Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4:333–349. (Cité en pages 41 et 42.)

[Maddern *et al.*, 2012] Maddern, A., Harrison, A. et Newman, P. (2012). Lost in Translation (and Rotation) : Rapid Extrinsic Calibration for 2D and 3D LIDARs. *International Conference on Robotics and Automation (ICRA)*. (Cité en pages 11, 35 et 37.)

[Magnusson *et al.*, 2009a] Magnusson, M., Andreasson, H., Nüchter, A. et Lilienthal, A. J. (2009a). Appearance-Based Loop Detection from 3D Laser Data Using the Normal Distributions Transform. *2009 IEEE International Conference on Robotics and Automation Kobe International Conference Center Kobe, May 12-17, Japan,*, pages 23–28. (Cité en page 39.)

[Magnusson *et al.*, 2007] Magnusson, M., Lilienthal, A. et Duckett, T. (2007). Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics*, 24:803–827. (Cité en page 39.)

[Magnusson *et al.*, 2009b] Magnusson, M., Nüchter, A., Lörken, C., Lilienthal, A. J. et Hertzberg, J. (2009b). Evaluation of 3D Registration Reliability and Speed – A Comparison of ICP and NDT. *2009 IEEE International Conference on Robotics and Automation Kobe International Conference Center Kobe, May 12-17, Japan,*, pages 3907–3912. (Cité en page 39.)

[Mazzei *et al.*, 2012] Mazzei, L., Medici, P. et Panciroli, M. (2012). A Lasers and Cameras Calibration Procedure for VIAC Multi-Sensorized Vehicles. *In Proceedings of Intelligent Vehicles Symposium, IEEE*, pages 548–553. (Cité en pages 11, 22 et 35.)

[Mengwen *et al.*, 2014] Mengwen, H., Huijing, Z., Jinshi, C. et Hongbin, Z. (2014). Calibration Method for Multiple 2D LIDARs System. *International Conference on Robotics and Automation (ICRA), Hong Kong, China.* (Cité en pages 37 et 43.)

[Mirzaei *et al.*, 2012] Mirzaei, F. M., Kottas, D. G. et Roumeliotis, S. I. (2012). 3D LIDAR–camera intrinsic and extrinsic calibration : Identifiability and analytical least-squares-based initialization. *The International Journal of Robotics Research*, 31(4):452–467. (Cité en page 70.)

[Monnier *et al.*, 2013] Monnier, F., Vallet, B., Paparoditis, N., Papelard, J. et David, N. (2013). Registration of Terrestrial Mobile Laser Data on 2D or 3D Geographic Database by use of a Non-Rigid ICP Approach. *ISPRS Workshop Laser Scanning 2013, Antalya, Turkey, 11 – 13 November*, II-5/W2:548–553. (Cité en pages 41, 92, 93 et 95.)

[Montemerlo et Thrun, 2006] Montemerlo, M. et Thrun, S. (2006). *Large-Scale Robotic 3-D Mapping of Urban Structures*, pages 141–150. Springer Berlin Heidelberg, Berlin, Heidelberg. (Cité en page 10.)

[Mourikis et Roumeliotis, 2006] Mourikis, A. I. et Roumeliotis, S. I. (Dec. 2006). Predicting the Performance of Cooperative Simultaneous Localization and Mapping (C-SLAM). *The International Journal of Robotics Research*, 25(12):1273–1286. (Cité en page 91.)

[Muhammad et Lacroix, 2010] Muhammad, N. et Lacroix, S. (2010). Calibration of a rotating Multi-Beam LIDAR. *IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan.* (Cité en page 70.)

[Newman *et al.*, 2006] Newman, P., Cole, D. et Ho, K. (2006). Outdoor SLAM using visual appearance and laser ranging. pages 1180–1187. (Cité en page 42.)

[Nüchter *et al.*, 2015] Nüchter, A., Borrmann, D., Koch, P., Kühn, M. et May, S. (2015). A Man-Portable, Imu-Free Mobile Mapping System. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-3/W5:17–23. (Cité en page 10.)

[Nüchter *et al.*, 2010] Nüchter, A., Elseberg, J., Schneider, P. et Paulus, D. (2010). Linearization of rotations for globally consistent n-scan matching. pages 1373–1379. (Cité en pages 41 et 42.)

[Nüchter *et al.*, 2007] Nüchter, A., Lingemann, K., Hertzberg, J., Birlinghoven, S. et Augustin, D.-S. (2007). 6D SLAM - 3D Mapping Outdoor Environments. 24:699–722. (Cité en pages 10, 23, 40, 41 et 42.)

[Nüchter *et al.*, 2004] Nüchter, A., Surmann, H., Lingemann, K., Hertzberg, J. et Thrun, S. (2004). 6D SLAM with an Application in Autonomous Mine Mapping. *Proceedings of the IEEE International Conference on Robotics and Automation, New Orleans, L.A.*, (April):1998–2003. (Cité en pages 10, 40, 41 et 91.)

[Parrot ebee site web, 2017] Parrot ebee site web (2017). https://www.sensefly.com/drones/overview.html. Dernier accés le 12-02-2017. (Cité en page 9.)

[Poullain, 2013] Poullain, E. (2013). *Exploitation de l'intensité du signal LASER d'un LiDAR topographique aéroporté pour des environnements littoraux sableux.* Thèse de doctorat, Université de Caen Basse-Normandie. (Cité en page 23.)

[Press *et al.*, 1992] Press, W. H., Teukolsky, S. A., Vetterling, W. T. et Flannery, B. P. (1992). *Numerical recipes in C (2nd ed.) : the art of scientific computing*, pages 671–675. Cambridge University Press, New York, NY, USA. (Cité en page 47.)

[Projet Terra Mobilita, 2017] Projet Terra Mobilita (2017). https://sites.google.com/site/terramobilita2012/home. Dernier accés le 12-02-2017. (Cité en page 1.)

[Quanergy product page, 2017] Quanergy product page (2017). http://quanergy.com/m8/. Dernier accés le 12-02-2017. (Cité en page 37.)

[Riegl LIDAR sensor, 2017] Riegl LIDAR sensor (2017). http://www.riegl.com/nc/products/mobile-scanning/. Dernier accés le 12-02-2017. (Cité en page 24.)

[Rusinkiewicz et Levoy, 2001] Rusinkiewicz, S. et Levoy, M. (2001). Efficients Variants of the ICP Algorithm. *Proceedings. Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152. (Cité en pages 39 et 40.)

[Sick LIDAR sensor, 2017] Sick LIDAR sensor (2017). https://www.sick.com/media/dox/7/07/207/Product_catalog_Detection_and_Ranging_Solutions_en_IM0044207.PDF. Dernier accés le 12-02-2017. (Cité en page 24.)

[Strecha *et al.*, 2008] Strecha, C., Von Hansen, W., Van Gool, L., Fua, P. et Thoennessen, U. (2008). On benchmarking camera calibration and multi-view stereo for high resolution imagery. *IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2008.* (Cité en page 28.)

[Tan et Park, 2002] Tan, C.-W. et Park, S. (2002). Design and Error Analysis of Accelerometer-Based Inertial Navigation Systems. *California PATH Research Report, California PATH Program, Institue of Transportation Studies, University of California, Berkeley.* (Cité en page 26.)

[Thrun et Montemerlo, 2005] Thrun, S. et Montemerlo, M. (2005). The GraphSLAM algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25:403–430. (Cité en page 91.)

[Trimble GNSS sensor, 2017] Trimble GNSS sensor (2017). http://www.trimble.com/Survey/GNSS-Surveying-Systems.aspx. Dernier accés le 12-02-2017. (Cité en page 16.)

[Trimble LIDAR sensor, 2017] Trimble LIDAR sensor (2017). http://www.trimble.com/3d-laser-scanning/index.aspx. Dernier accés le 12-02-2017. (Cité en page 12.)

[Tsai, 1987] Tsai, R. (1987). A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal on Robotics and Automation*, 3:323–344. (Cité en page 28.)

[Turk et Levoy, 1994] Turk, G. et Levoy, M. (1994). Zippered Polygon Meshes from Range Images. *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, New York, NY, USA*, pages 311–318. (Cité en page 40.)

[Velodyne LIDAR sensors, 2017] Velodyne LIDAR sensors (2017). http://velodynelidar.com/products.html. Dernier accés le 12-02-2017. (Cité en pages 24 et 37.)

[Vivet, 2011] 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 extérireur.* Thèse de doctorat, Université Blaise Pascal - Clermont II, LASMEA. (Cité en page 21.)

[Weng *et al.*, 1992] Weng, J., Cohen, P. et Herniou, M. (1992). Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:965–980. (Cité en page 28.)

[Wikipedia Géolocalisation, 2017] Wikipedia Géolocalisation (2017). https://fr.wikipedia.org/wiki/Géolocalisation. Dernier accés le 12-02-2017. (Cité en page 15.)

[Wikiwand site web, 2017] Wikiwand site web (2017). http://www.wikiwand.com/fr/Gyrolaser. Dernier accés le 12-02-2017. (Cité en page 18.)

[Woodman, 2007] Woodman, O. J. (2007). An introduction to inertial navigation. *Technical report, University of Cambridge.* (Cité en pages 18 et 26.)

[Zhu *et al.*, 2009] Zhu, J., Zheng, N., yuan, Z. et Du, S. (2009). Point-to-line metric based Iterative Closest Point with bounded scale. *Proceedings of the 4th IEEE Conference on Industrial Electronics and Applications*, pages 3032–3037. (Cité en page 40.)

[Zhu et Liu, 2013] Zhu, Z. et Liu, J. (2013). Unsupervised Extrinsic Parameters Calibration for Multi-Beam LIDARs. *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering, Paris, France*, pages 1110–1113. (Cité en page 37.)

Résumé

Les Systèmes Mobiles de Cartographie basés LIDAR permettent d'obtenir des cartes 3D de l'environnement, qui sont géo-référencées grâce à d'autres capteurs embarqués sur le véhicule : GPS, centrale inertielle, ou encore odomètre sont de tels capteurs qui permettent de localiser le véhicule mobile pendant la campagne d'acquisition. Toutefois, ces cartes manquent de précisions et un affinage des cartes est essentiel dans de nombreux cas d'applications où une précision fine est requise sur les cartes 3D, comme pour des applications de classifications par exemple.

Lors de la création de cartes 3D géo-référencées, les données sont tout d'abord acquises par le capteur LIDAR et référencées dans le repère cartésien du laser à l'aide d'un calibrage intrinsèque du capteur d'acquisition. Ensuite, un calibrage extrinsèque du capteur permet de caractériser la transformation entre le capteur et le véhicule, et permet de référencer les données dans le repère « body », lié au véhicule d'acquisition. Enfin, avec la trajectoire du véhicule obtenue en fusionnant les données issues des GPS, centrale inertielle et odomètre, il est possible de géo-référencer les données lasers.

Nous proposons dans cette thèse d'affiner les relevés laser issus d'acquisitions effectuées à l'aide d'un véhicule mobile de cartographie, en optimisant plusieurs paramètres différents qui entrent en compte dans le géo-référencement des données. Nous nous sommes intéressés à l'affinement des nuages de points par optimisation des paramètres de calibrage extrinsèque dans un premier temps, puis par optimisation des paramètres de calibrage intrinsèque, et enfin par optimisation des paramètres de translations liés à la trajectoire du véhicule mobile.

Mots Clés

Affinement de relevés laser ; LIDAR multi-couches ; calibrage ; véhicule mobile de cartographie ; cartographie terrestre ; recalage de données

Abstract

LIDAR based Mobile Mapping Systems allow to get 3D maps of the environment, which are globally referenced with the help of others sensors embedded on the vehicle: GPS, Inertial Measurement Unit, or odometer are such sensors which allow localizing the vehicle during the acquisition process. However, these maps lack of precision, and a refinement of the maps is necessary for many work where a good precision is needed on the 3D maps, like classification applications for example.

When creating the globally referenced 3D maps, the data are firstly acquired by the LIDAR sensor and referenced in the Cartesian reference frame of the sensor with an intrinsic calibration of the sensor. Then, an extrinsic calibration gives the transformation between the sensor and the vehicle, and gives data referenced in the « body » reference frame, linked to the vehicle. Finally, with the fusion of the data coming from the GPS, the Inertial Measurement Unit and the odometer, the laser data can be globally referenced.

In this thesis, we propose to refine the point clouds coming from acquisitions done with a mobile mapping system, by optimizing some parameters which are used in the geo-referencing process of the data. Firstly, we were interested in the refinement of point clouds by optimizing the extrinsic calibration parameters, and then we were interested in the refinement of point clouds by optimizing the intrinsic calibration parameters; finally by optimizing the translation parameters of the mobile vehicle trajectory.

Keywords

Point clouds refinement; multi-beam LIDAR; calibration; mobile mapping vehicle; terrestrial mapping; data adjustment