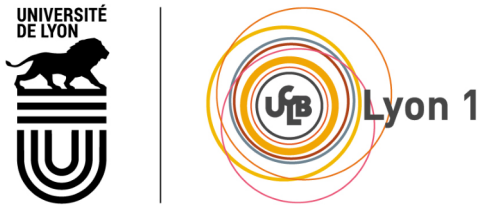
Reconstruction 3D d'environnements intérieurs à partir d'acquisitions LiDAR

Julia Sanchez



N° d'ordre NNT : 2020LYSE1049

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'Université Claude Bernard Lyon 1

École Doctorale 512
InfoMaths

Spécialité de doctorat : Informatique

Soutenue publiquement le : 24/06/2020 , par :
Julia Sanchez

Reconstruction 3D d'environnements intérieurs à partir d'acquisitions LiDAR

Devant le jury composé de :

M. Vallet Bruno, Chargé de recherche, Institut national de l'information géographique et forestière, Saint-Mande, Rapporteur ;
Mme. Morin Géraldine, Pr. des Universités, Univ. de Toulouse, Rapporteure ;
Mme. Denis Florence, Maître de conférences, Univ. Lyon 1, Directrice de thèse ;
M. Checchin Paul, Pr. des Universités, Univ. Clermont Auvergne, Directeur de thèse ;
M. Dupont Florent, Pr. des Universités, Univ. Lyon 1, Encadrant ;
M. Trassoudaine Laurent, Pr. des Universités, Univ. Clermont Auvergne, Encadrant ;
M. Goulette François, Pr. des Universités, MINES ParisTech, Examinateur ;
Mme. Marcotegui Beatriz, Pr. des Universités, MINES ParisTech, Examinatrice ;
M. Jaillet Fabrice, Maître de conférences, Univ. Lyon 1, Examinateur.

Reconstruction 3D d'environnements intérieurs à partir d'acquisitions LiDAR
3D Reconstruction of indoor environments from LiDAR acquisitions

Julia Sanchez

Université de Lyon
Laboratoire d'InfoRmatique en Image et Systèmes d'information - UMR 5205
en collaboration avec l'*Institut Pascal* (Université Clermont Auvergne) - UMR 6602

Bâtiment Nautibus
43, bd du 11 novembre 1918
69622 VILLEURBANNE CEDEX

20 Décembre 2019
Relecteurs : Géraldine Morin et Bruno Vallet
Directeurs : Florence Denis et Paul Checchin

UNIVERSITÉ CLAUDE BERNARD - LYON 1

Président de l'Université	M. Frédéric FLEURY
Président du Conseil Académique	M. Hamda BEN HADID
Vice-président du Conseil d'Administration	M. Didier REVEL
Vice-président du Conseil des Etudes et de la Vie Universitaire	M. Philippe CHEVALIER
Vice-président de la Commission de Recherche	
Directrice Générale des Services	M. Damien VERHAEGHE

COMPOSANTES SANTÉ

Faculté de Médecine Lyon-Est - Claude Bernard	Doyen : M. Gilles RODE
Faculté de Médecine et Maïeutique Lyon Sud Charles. Mérieux	Doyenne : Mme Carole BURILLON
UFR d'Odontologie	Doyenne : Mme Dominique SEUX
Institut des Sciences Pharmaceutiques et Biologiques	Directrice : Mme Christine VINCIGUERRA
Institut des Sciences et Techniques de la Réadaptation	Directeur : M. Xavier PERROT
Département de Formation et Centre de Recherche en Biologie Humaine	Directrice : Mme Anne-Marie SCHOTT

COMPOSANTES ET DEPARTEMENTS DE SCIENCES ET TECHNOLOGIE

UFR Biosciences	Directrice : Mme Kathrin GIESELER
Département Génie Electrique et des Procédés (GEP)	Directrice : Mme Rosaria FERRIGNO
Département Informatique	Directeur : M. Behzad SHARIAT
Département Mécanique	Directeur : M. Marc BUFFAT
UFR - Faculté des Sciences	Administrateur provisoire : M. Bruno ANDRIOLETTI
UFR (STAPS)	Directeur : M. Yannick VANPOULLE
Observatoire de Lyon	Directeur : Mme Isabelle DANIEL
Ecole Polytechnique Universitaire Lyon 1	Directeur : M. Emmanuel PERRIN
Ecole Supérieure de Chimie, Physique, Electronique (CPE Lyon)	Directeur : M. Gérard PIGNAULT
Institut Universitaire de Technologie de Lyon 1	Directeur : M. Christophe VITON
Institut de Science Financière et d'Assurances	Directeur : M. Nicolas LEBOISNE
ESPE	Administrateur provisoire : M. Pierre CHAREYRON

Résumé

Ce travail de thèse porte sur la reconstruction 3D d'environnements structurés à partir d'acquisitions LiDAR. L'étude a pour but d'automatiser et d'améliorer la chaîne de traitements allant de l'acquisition de nuages de points à la modélisation 3D d'intérieurs de bâtiments. Actuellement, ces traitements sont majoritairement manuels, l'acquisition LiDAR dresse de nombreux obstacles à la reconstruction automatique (anisotropie, bruit, occultations, etc.) et les méthodes actuelles manquent de précision et ne résolvent pas tous les cas de figure. Dans un premier temps, l'étude est orientée sur la modélisation de nuages de points scan par scan. Les méthodes automatiques existantes reposent sur de nombreuses hypothèses de construction qui mènent à des résultats relativement éloignés des données initiales. Le choix a été fait de proposer une nouvelle méthode de modélisation au plus proche des données, en ne reconstruisant que les zones mesurées de chaque scène et en excluant les zones occultées. Pour cela, nous nous intéressons ici au processus de modélisation locale de nuages de points et nous proposons un nouvel estimateur de normales adapté aux environnements structurés. L'utilisation de ce nouvel outil permet de réaliser une modélisation globale d'une scène scannée par un dispositif LiDAR à partir de polygones. Cette modélisation repose sur un traitement conjoint de l'image d'acquisition angulaire et du nuage de points.

Dans un second temps, nous abordons le sujet du recalage afin de replacer les scans dans un repère global. L'objectif principal est de rendre ce procédé automatique quels que soient la géométrie des scènes, leur pose initiale et d'obtenir de bonnes performances pour de faibles chevauchements. Les approches existantes fondées sur le traitement de nuages de points sont majoritairement locales et ne semblent pas adaptées à des environnements structurés dans lesquels les voisinages locaux apportent peu d'information d'identification. Une nouvelle approche adaptée aux scènes d'intérieur est proposée afin de pallier ces problèmes. L'erreur commise lors d'un recalage est difficilement mesurable, pourtant, cette information est nécessaire en vue de corriger une suite de recalages ou pour fusionner la pose issue du recalage avec d'autres données de localisation provenant de capteurs extérieurs. De nombreuses pistes de recherches ont été explorées pour estimer cette erreur et une méthode récente, utilisant un apprentissage automatique, est particulièrement développée dans ce travail. Une adaptation de cette méthode est également proposée et une évaluation sur une base de données synthétique permet de mettre en évidence les points forts de la méthode et certaines de ses limitations cruciales.

Mots clés : modèles 3D, BIM, scènes d'intérieur, nuage de points, estimation de normales, modélisation polygonale, recalage, estimation de covariance.

Abstract

This PHD thesis deals with 3D modelisation in the context of indoor reconstruction of structured environments using LiDAR data. The study aims at automating and improving the pipeline going from the point cloud acquisitions to the 3D reconstruction of building indoors. Indeed, currently, these processes remain mostly manual. The LiDAR data has some specific properties which make the reconstruction challenging (anisotropy, noise, clutters, etc.) and existing methods have a lack of accuracy and their performances depend on the scanned scenes geometry, on the sensor quality and on the acquisition process.

First, the study is oriented towards the point clouds modelisation, one scan at a time. The automatic methods of the state of the art rely on numerous construction hypothesis which yields 3D models relatively far from initial data. The choice has been done to propose a new modelisation method closest to point clouds data, reconstructing only measured areas of each scene and excluding occluded regions. In this objective, we look into the local modelisation process and propose a new normal estimator adapted to structured environments. This tool is integrated to a global modelisation of a scene scanned by a LiDAR device using polygones. This modelisation rely on a joint processing of the range image and the point cloud associated to one scan.

Second, we discuss the registration topic, in order to replace the scans in a global frame. The main objective is to make this process automatic regardless of the scenes geometry, of their initial pose and to obtain good performances for low overlaps. The approaches of the state of the art based on point cloud processing are mostly local and do not seem adapted to structured environments in which local neighborhoods do not carry much information for identification. A new approach adapted to building indoor scenes is proposed to address these issues. Finally, the error resulting from a registration is difficult to measure. Nevertheless, this information is necessary to correct globally a succession of registrations or to fusion the pose information extracted from the registration to other location data provided by external sensors. Some research leads have been explored to estimate the error of a registration and a recent method, based on machine learning, is particularly developped. An adaptation of this method is proposed and evaluated on a synthetic data base. The results emphasize the advantages of the method but also show some critical limitations.

Key words : 3D models, BIM, indoor scenes, point cloud, normal estimation, polygonal modelisation, registration, covariance estimation.

Remerciements

Cette thèse a été réalisée grâce à une bourse doctorale de la région Auvergne Rhône-Alpes.

Merci à mes encadrants lyonnais pour le soutien inconditionnel et la patience dont ils ont fait preuve durant mes diverses irruptions dans leurs bureaux.

Merci à mes encadrants clermontois pour les encouragements, les orientations et pour m'avoir permis de voir du pays et de rencontrer tout plein de gens formidables.

Merci à Mauro pour sa patience, son écoute, ses idées et pour avoir pris en main toutes ces soirées où mon cerveau refusait de se rallumer.

Merci à ma famille pour leur soutien à distance et pour supporter mes gémissements sans fin au téléphone, et merci à mes neveux pour égayer mes journées au bureau.

Merci à mes co-bureau, co-peines, co-rires, co-jaiRenverséMonThéSurMonClavier, co-larmes, co-rerires, Valentin et Matthieu pour me donner envie d'aller au labo qu'il pleuve ou qu'il neige.

Merci à tous les amis du labo : Alice, Agathe, Vincent, Bearzi, Caissard, Arthur, Webi, Rémy, Rémi, Jocelyn, Thibault, Charles, Maxime, Simon, Alexis, Antoine et Samuel pour tous ces débats sans fin, ces fous rires incroyables et pour m'avoir initiée au monde fabuleux des gens de lettre. Bon des mots croisés quoi.

Merci à François Pomerleau pour m'avoir reçue (et très bien) à Québec et merci à David d'avoir partagé son travail et de m'avoir donné de son temps.

Merci à Laurent Malaterre et Ahmad Kamal Aijazi pour l'acquisition des données expérimentales.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Historique et applications du LiDAR

La technologie LiDAR (*Light Detection And Ranging*) a été inventée au début des années 1960 dans le but de mesurer des grandes distances. La distance terre-lune a notamment été mesurée avec précision en 1962 par cette méthode dans le cadre du projet du MIT (*Masachussetts Institute of Technology*) nommé *Project Luna See*. Dans les années 90, son utilisation s'est répandue au travers de la modélisation de villes par acquisitions aériennes de nuages de points. Les premiers travaux de reconstruction de surfaces et de modélisation à partir de nuages de points ont alors vu le jour. Afin d'affiner les modélisations, les premiers LiDAR terrestres sont apparus accompagnés de nouvelles méthodes de reconstruction adaptées aux façades de bâtiments.

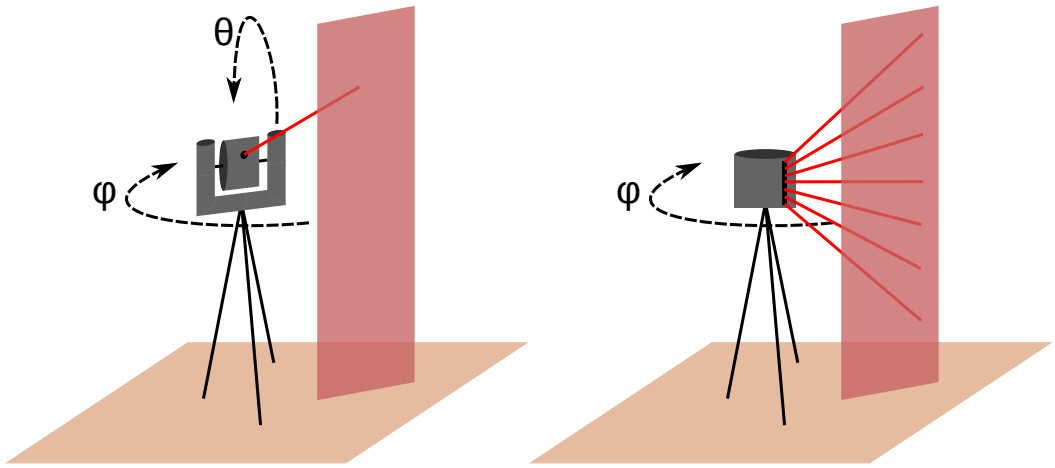
Les outils modernes d'architecture numérique sont regroupés sous le terme BIM (*Building Information Modeling*) [Hic19]. Ils ont été introduits au début des années 1980, ils étaient alors fondés sur la création de maquettes numériques 3D avant la construction d'un bâtiment. L'intérêt des industriels et des chercheurs s'est rapidement tourné vers la représentation 3D de bâtiments tels que construits. La technologie LiDAR semble alors appropriée du fait de sa précision et de sa rapidité d'acquisition. Dans ce contexte, l'objectif est d'obtenir un modèle BIM à partir des nuages de points collectés par les dispositifs LiDAR. L'intérêt d'un tel modèle est multiple :

- visualiser numériquement les bâtiments. En effet, les nuages de points sont difficilement visualisables pour des personnes non initiées du fait de leur aspect discret, des différences de densité et de leur poids en mémoire qui complique leur manipulation ;
- effectuer des calculs (volumes, aires, forces, déformations etc.) ;
- réaliser différents types de simulations (son, lumière, rénovations, etc.).

L'utilisation de cette technologie dans ce cadre est d'autant plus d'actualité depuis l'incendie de la cathédrale Notre-Dame au mois d'avril 2019. En effet, les acquisitions LiDAR, avant et après sinistre, de l'intérieur et de l'extérieur de la cathédrale, vont avoir un rôle central dans sa rénovation [Aud19]. De plus, un programme mondial d'acquisition d'édifices en péril répertoriés par l'UNESCO a été mis en place notamment sur des sites comme Palmyre en Syrie [Tex18]. En parallèle de l'expansion du LiDAR dans les applications de génie civil, cette technologie a commencé à être utilisée dans les années 1990 dans le domaine de la navigation autonome avec le développement de l'électronique embarquée. Le but n'est plus alors d'obtenir une reconstruction fidèle de l'environnement mais d'améliorer la localisation d'un véhicule. Récemment, les progrès réalisés dans la fabrication de tels équipements ont permis de réduire leur taille et leur poids, ouvrant ainsi la voie à de nouvelles applications telles que l'acquisition par drone [Kel+19].

Fonctionnement général

Un capteur LiDAR est composé d'un émetteur laser, d'un collecteur de lumière, d'un photodétecteur et d'un module de calcul. L'émetteur produit un rayon laser dont une partie va être réfléchie par la première surface non transparente rencontrée vers la source (principe de la rétrodiffusion). Le rayon réfléchi va alors être détecté par le collecteur de lumière. Le temps de vol (*Time of Flight*) du rayon lumineux jusqu'à la surface est ensuite converti en distance. Dans le cadre d'acquisitions 3D, le capteur est placé sur une plateforme rotative et un point 3D est déduit à partir de l'orientation de cette plateforme et de la distance mesurée pour cette orientation. La rotation permet d'obtenir une multitude de points à partir d'un capteur. Elle doit se faire suivant deux axes afin de couvrir tout l'espace. Les scanners communément utilisés réalisent des rotations autour de deux axes orthogonaux à intervalles angulaires réguliers. Certains scanners sont composés de plusieurs capteurs. Chaque capteur couvre alors une zone de l'espace pendant la rotation qui est communément réalisée suivant un seul axe. Le schéma ci-dessous montre la différence d'utilisation d'un scanner monofaisceau et d'un scanner multifaisceaux.



(a) Exemple de scanner monofaisceau. **(b)** Exemple de scanner multifaisceaux.

- Types de scanners LiDAR.

Dans les deux cas, la sortie d'un dispositif LiDAR est composée d'un ensemble de vecteurs comprenant deux angles nommés φ et θ (voir figure ci-dessus) et une distance. Cet ensemble peut être représenté par une image dont les lignes correspondent à l'angle θ, les colonnes correspondent à l'angle φ et la couleur du pixel correspond à la distance mesurée. Cette image est nommée « image de profondeur ». Les scanners multifaisceaux sont plus rapides mais leur couverture spatiale est souvent plus faible. Leurs caractéristiques sont adaptées à une acquisition temps réel, notamment dans le cadre de la navigation robotique. Leur faible couverture est suffisante pour résoudre des problèmes de détection et d'orientation mais les rend difficilement utilisables, sans dispositif mécanique annexe, dans un contexte de reconstruction. Dans le domaine de la navigation robotique, les dispositifs LiDAR ayant une fréquence d'acquisition suffisamment élevée peuvent également être placés sur un véhicule en mouvement continu dans l'environnement.

Les défis lancés par l'acquisition

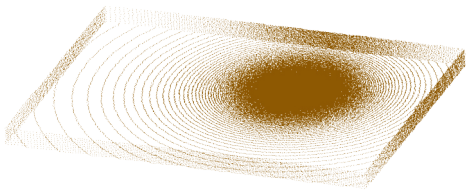
L'acquisition LiDAR, bien qu'étant rapide et précise, dresse un certain nombre d'obstacles au processus de reconstruction.

Biais et incertitudes du capteur : tout d'abord, le capteur LiDAR, comme tout appareil de mesure, donne un résultat bruité. La qualité du résultat dépend de différents facteurs dont :

- l'atténuation de la lumière pendant le parcours ;
- la précision du collecteur de lumière ;
- les biais induits par la mesure du temps de vol ;
- le matériau de la surface et son coefficient de rétrodiffusion.

De plus, certains biais ne sont pas encore pris en compte par les constructeurs. LACONTE et coll. [Lac+19] montrent par exemple que le calcul du temps de vol est corrompu lorsque la surface n'est pas parfaitement perpendiculaire au rayon. Le rayon du faisceau formant un cône, une partie de l'énergie est collectée plus tôt que ce qui est souhaité.

Echantillonnage : une des difficultés majeures engendrées par le fonctionnement d'un scanner LiDAR est l'échantillonnage 3D de la scène. En effet, l'échantillonnage des angles de rotation ne permet pas d'acquérir des points uniformément dans l'espace 3D. Le degré d'anisotropie de l'échantillonnage dépend de l'orientation des surfaces par rapport aux axes de rotation du scanner et de la distance des objets mesurés. Un exemple de cette anisotropie est donné sur le scan du sol d'une pièce dans la figure ci-après :



- Scan simulé du sol d'une pièce. Mise en évidence de l'anisotropie de l'échantillonnage.

Mouvement : bien que certains capteurs multifaisceaux soient rapides, leur fixation sur des dispositifs en mouvement peut également introduire un biais (voir jeu de données `DS3-V`, décrit en annexe). Si la fréquence d'acquisition n'est pas assez grande relativement à la vitesse du véhicule, des distorsions sont générées en raison de la latence de mesure. VIVET et coll. [Viv+12] corrigent notamment ces déformations par un modèle de projection spatio-temporel.

Masse de données : Un dispositif LiDAR peut générer un très grand volume de données pour atteindre un niveau de précision suffisant. Il est, par exemple, commun de retrouver des nuages de plusieurs millions de points. Ce volume implique des contraintes de temps et de mémoire sur les outils d'acquisition, de traitement et de visualisation.

Environnement mesuré : Les scènes d'intérieur peuvent être très différentes les unes des autres selon la structure du bâtiment ou l'arrangement des objets dans les salles. Des hypothèses structurelles ont été utilisées [CY99 ; Mac+17] mais ces dernières sont difficilement imposables du fait de la variété des architectures possibles et des normes qui varient selon les pays. De plus, une des difficultés de l'acquisition LiDAR est que le rayon est arrêté par la première surface non transparente rencontrée. De nombreuses occultations peuvent donc apparaître selon le point de vue. Certains rayons peuvent également ne jamais revenir au collecteur, notamment dans le cas d'ouvertures dans les murs (portes ou fenêtres) ou de surfaces spéculaires (miroir, vitre,etc.) pour lesquelles la rétrodiffusion est faible. Enfin, les scans étant acquis successivement dans un bâtiment, les scènes peuvent avoir évolué au cours de l'acquisition, on parle alors d'environnement dynamique (cf. `DS1-H` Annexe B.1).

Localisation du scanner : Afin de reconstruire un bâtiment complet, la localisation du dispositif LiDAR pour chaque scan doit être obtenue. Cette tâche peut être réalisée au moment de l'acquisition par des capteurs (un compteur de tours de roue, un gyroscope, un accéléromètre, un théodolite, etc.) ou en plaçant des cibles physiques dans l'espace dont le scanner peut mémoriser l'emplacement avant d'acquérir une nouvelle mesure. Néanmoins, ces méthodes présentent de nouveaux biais d'acquisition et de manipulation.

Les enjeux de la reconstruction

Dans cette thèse, nous nous intéressons particulièrement à la reconstruction de bâtiments tels que construits. Les points acquis par un dispositif LiDAR ne sont pas directement manipulables dans le cadre d'applications d'architecture ou de génie civil. En effet, ils doivent être remplacés par des surfaces et des volumes connectés les uns aux autres. De plus, les scans doivent être replacés dans un repère global. Enfin, les données peuvent nécessiter un enrichissement sémantique permettant de faciliter des traitements postérieurs. La chaîne de traitements commune est la suivante :

1. Acquisition de scans ;
2. Recalage des scans sous forme de nuages de points ;
3. Segmentation du nuage de points et enrichissement sémantique ;
4. Remplacement des points par des modèles .

La première étape consiste à acquérir les nuages de points avec un dispositif LiDAR à partir de différents points de vue afin de maximiser les zones atteintes par le capteur. Chaque scan est acquis dans le repère du capteur à la sortie de la première étape. Le recalage de scans consiste à repositionner les scans acquis dans un repère global. Cette étape est nécessaire pour aligner les scans les uns sur les autres. Les nuages de points sont alors communément fusionnés et le nuage final est utilisé dans les traitements postérieurs. La segmentation du nuage consiste à détecter des entités géométriques échantillonnées par les points. L'enrichissement sémantique

peut concerner la nature précise des entités géométriques (porte, fenêtre, table, etc.) ou les interactions entre les entités (occultation, ouverture et connexion). La dernière étape consiste à remplacer les points segmentés par des modèles permettant à la fois de réduire le coût en mémoire des données, de réaliser des calculs et des simulations dans l'environnement et de permettre une interaction de l'utilisateur avec les objets de la scène.

Actuellement, cette chaîne de traitements est majoritairement manuelle [Hic19]. De nombreux outils sont mis à la disposition des utilisateurs mais leur utilisation et leur paramétrage peuvent être complexes et une validation experte du résultat est souvent nécessaire.

Objectifs et plan de la thèse

L'objectif de cette thèse est de participer à l'automatisation d'une chaîne de traitements allant de l'acquisition LiDAR à la reconstruction 3D de la structure d'un bâtiment. Nous avons choisi de décliner l'objectif principal en sous-objectifs traités en partie dans la littérature et auxquels nous allons essayer de répondre par une nouvelle approche plus efficace tout au long de cette thèse :

Objectifs

1. Faciliter l'étape d'acquisition, en réduisant les besoins de localisation du capteur et de précision des mesures et en la guidant par des informations sémantiques.
2. Alléger le coût en mémoire des données LiDAR sans perdre d'information en remplaçant les points par des entités géométriques.
3. Aligner spatialement les différents scans sans utiliser d'information extérieure aux nuages de points.
4. Obtenir une reconstruction d'un bâtiment au plus proche des données acquises à partir de modèles de primitives connectés les uns aux autres. Cette reconstruction devra présenter des caractéristiques adaptables à une utilisation dans le cadre du BIM.

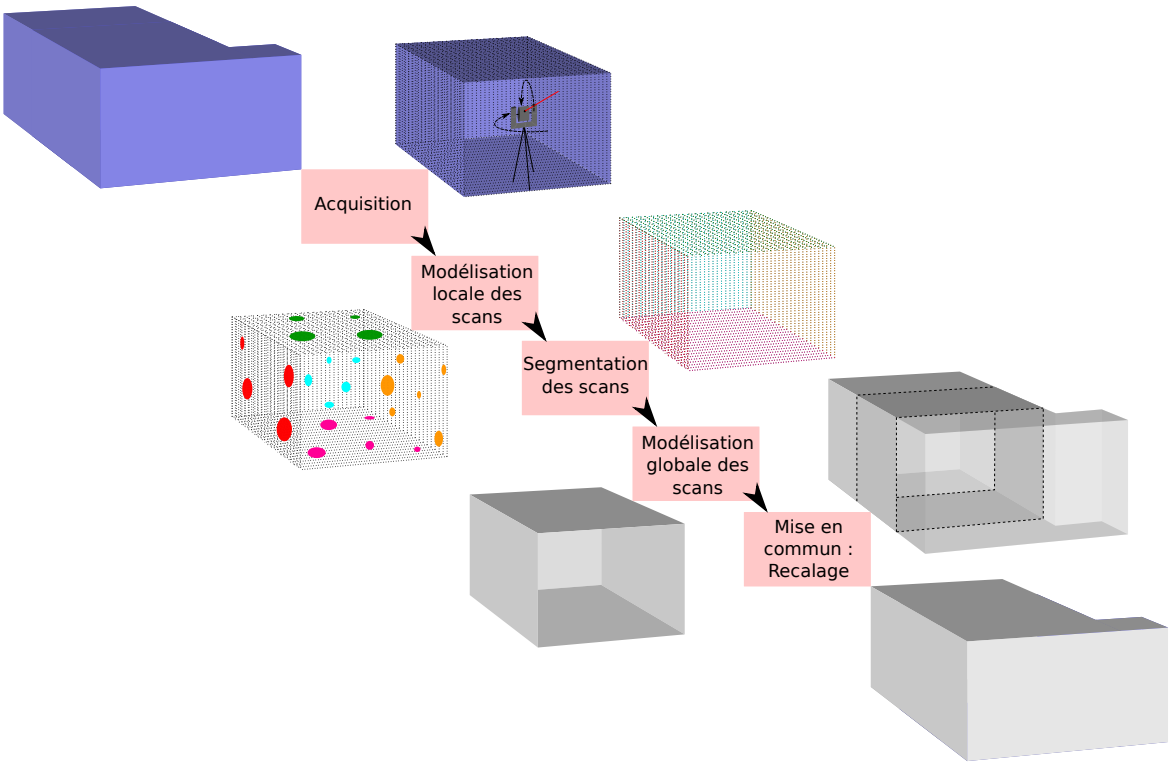
La généralisation des méthodes répondant à ces objectifs à différents capteurs et à différents environnements est le verrou scientifique majeur. En effet, ces méthodes doivent faire face aux différents obstacles induits par le processus d'acquisition tels que décrits plus haut et respecter des contraintes de temps et de mémoire, particulièrement dans le traitement de gros volumes de données.

Plus précisément, on distingue deux étapes qui pourraient être améliorées :

La modélisation : de nombreuses méthodes existent pour modéliser des nuages de points. Nous restreignons notre recherche à des approches adaptées aux scènes d'intérieur afin d'obtenir un format exploitable par des acteurs du génie civil. Les approches les plus performantes reposent sur de fortes hypothèses de construction et créent un arrangement de plans à partir de primitives planaires. La détection des primitives n'a pas une précision suffisante pour fonctionner de manière non-supervisée dans toutes les configurations. En outre, ces méthodes sont difficiles à paramétrer et peuvent produire de forts écarts à la géométrie réelle. Nous cherchons une modélisation plus proche des données en limitant les extrapolations.

Les méthodes de recalage : les algorithmes les plus utilisés sont fondés sur une mise en correspondance locale et ne sont pas robustes dans le contexte d'acquisitions LiDAR en environnement intérieur. Nous cherchons un algorithme capable de résoudre le recalage en environnement structuré dans des cas de faibles chevauchements avec de fortes variations de densité et un bruit de mesure non négligeable.

Afin de remplir les objectifs précédemment listés tout en palliant les problèmes induits par les méthodes actuelles, nous proposons la chaîne de traitements suivante :



- Chaîne de traitements utilisée dans ce travail.

Contrairement aux chaînes de traitements classiques, la reconstruction proposée passe par une modélisation de chaque scan indépendamment des autres avant de les mettre en commun. Cette technique permet de tirer parti du fonctionnement interne du scanner et plus précisément de l'uniformité de l'échantillonnage dans l'espace des angles d'acquisition pour réaliser la modélisation. Afin de modéliser un scan, nous proposons de réaliser une étape de modélisation locale. Cette modélisation doit être robuste aux différents biais induits par le capteur et prendre en compte la géométrie de la scène et les différentes discontinuités de courbure. Elle est utilisée dans une segmentation précise du nuage de points par des primitives planaires. A partir de ces primitives, un ensemble de polygones est modélisé sur la scène et des informations sémantiques sur la nature des polygones et sur leurs interactions sont déduites. Le recalage de scans peut alors être réalisé de façon à n'obtenir qu'une seule entité géométrique connectant tous les scans entre eux. Ce recalage adapté aux environnements structurés utilise la géométrie globale de la scène.

Ainsi, cette thèse s'articule autour de deux parties principales : la modélisation individuelle de scans en environnement intérieur et leur recalage. Dans la première partie, nous verrons comment, à partir d'une modélisation locale, obtenir un modèle construit par des polygones plans. Dans la seconde partie, nous étudierons comment aligner spatialement les différents scans acquis afin d'obtenir un seul objet dans un repère global et comment évaluer la validité de cet alignement.

Partie I

Modélisation

La modélisation se caractérise par le remplacement de données mesurées par un modèle. On comprend alors que ce procédé est adapté à un contexte d'acquisition LiDAR de nuages de points en intérieur. En effet, la majorité des objets scannés dans une scène d'intérieur correspondent à des modèles connus du fait de leur aspect synthétique. On retrouve notamment des travaux sur la modélisation de chaises ou de tables [Nan+12], de tuyaux [Kaw+14] ou d'escaliers [SZ12] par exemple. Nous nous intéressons particulièrement à l'enveloppe de la scène scannée et plus largement à la structure du bâtiment. Cette structure est communément composée d'un ensemble principal de plans qui sont des modèles faciles à représenter en 3D. Le verrou scientifique majeur à la modélisation de cette structure se situe dans la localisation précise de ces plans, de leurs contours et de leurs connexions. La difficulté de cette tâche provient des défauts d'acquisition d'un système LiDAR tels que décrits en introduction et principalement de l'anisotropie d'échantillonnage de la scène, du bruit de mesure et des problèmes d'occultation. En outre, la modélisation de scènes d'intérieur est dépendante du degré de détail souhaité. En ajustant un modèle géométrique, certaines caractéristiques de l'objet sous-jacent seront perdues. Il est alors important de connaître les objectifs de la modélisation afin de savoir quelle méthode employer tout en tenant compte du fait que toute modélisation entraîne une perte d'information. D'un autre côté, la modélisation d'une scène peut apporter des informations sémantiques sur la scène, et réduire l'effet du bruit de mesure.

Nous choisissons d'étudier les reconstructions polygonales et polyédriques qui sont les reconstructions les plus adaptées aux scènes d'intérieur. Ces reconstructions sont notamment facilement transférables à un format de BIM utilisé dans le milieu du génie civil. Ces méthodes reposent toutes sur une segmentation précise de primitives planaires dans la scène. Cependant, les méthodes actuelles ne peuvent pas détecter précisément les contours de telles primitives. D'un autre côté, le champ de normales correspondant à un nuage de points pourrait être utilisé dans la détection de plans pour pallier ce problème. En effet, le vecteur normal est la caractéristique du plan local en chaque point. La mise en commun de ces informations locales peut mener à la détection de primitives globales dans le nuage de points. Or, au vu de l'état de l'art, la qualité des normales autour des zones d'intersection de plans reste faible et la robustesse des algorithmes actuels au bruit et aux points aberrants n'est pas suffisante pour effectuer une détection de primitives fiable.

Le premier chapitre de cette partie concerne l'estimation de normales à partir de nuages de points ; l'étude examine le cas des surfaces à courbure continue par morceaux telles que celles qu'on retrouve en environnement structuré. Nous étudierons tout d'abord l'état de l'art afin de mettre en évidence les quatre grandes catégories de méthodes existantes : les ajustements de surface, les méthodes fondées sur un tirage aléatoire de points voisins, les corrections d'un champ de normales *a posteriori* et les méthodes fondées sur la construction d'un diagramme de Voronoi. Puis, nous proposerons une nouvelle méthode d'estimation adaptée aux environnements structurés qui permet à la fois de résoudre les problèmes d'estimation aux voisinages de discontinuité de courbure et de conserver une bonne précision sur les zones lisses et bruitées. De plus, l'anisotropie de l'échantillonnage et de la géométrie est prise en compte dans cette approche.

Le second chapitre traite de la modélisation d'un nuage de points acquis dans un bâtiment. Nous verrons que les méthodes actuelles dépendent d'une détection fiable de primitives planaires. Trois méthodes principales sont utilisées pour détecter des plans : la transformée de Hough, RANSAC et la croissance de régions 3D. Ces méthodes souffrent toutes de l'anisotropie de l'échantillonnage dans l'environnement 3D, et nécessitent un paramétrage complexe. A l'instar de différents auteurs [Cha+10 ; BM16], nous choisissons de travailler sur les scans après acquisition pour conserver le lien avec l'image de profondeur et donc la régularité de l'échantillonnage angulaire (des angles $\{\varphi, \theta\}$ définis en introduction). Cette technique permet de tirer parti des voisinages locaux définis dans l'image. Nous expliquerons comment le champ de normales peut être intégré à la segmentation de primitives planaires et nous montrerons que réaliser cette segmentation dans l'image $\{\varphi, \theta\}$ est approprié. De plus, nous montrerons que la majorité des méthodes actuelles mènent à une forte extrapolation des données et à des écarts importants avec la réalité. Nous proposerons alors une nouvelle méthode de modélisation polygonale du nuage de points qui permet de mettre en évidence les interactions entre les polygones en extrapolant au minimum les informations fournies par le nuage de points. La modélisation proposée est enrichie d'informations sémantiques telles que les occultations de la scène, les ouvertures ou les objets.

1 Modélisation locale : estimation de normales

1.1 Problématique

La normale est un attribut couramment utilisé dans le traitement de surfaces échantillonnées par des points. On estime communément une normale en un point à l'aide de son voisinage local. Les normales sont notamment utilisées dans des processus de débruitage, de recalage, de segmentation et de reconstruction de surface par exemple. Bien que de nombreux travaux aient été proposés dans ce domaine de recherche [Hop+92 ; Fle+05 ; BM12 ; Hua+13], l'estimation de normales présente encore des limites. La précision des algorithmes actuels sur des surfaces lisses n'est pas encore optimale, particulièrement en présence de bruit ou lorsque l'échantillonnage n'est pas uniforme. De plus, les discontinuités dans la courbure sont rarement prises en compte dans l'estimation. Ce dernier point est un verrou scientifique critique au traitement de scènes d'intérieur. Enfin, les temps de calcul des algorithmes les plus performants restent élevés et leur paramétrage peut s'avérer complexe.

Dans ce chapitre, nous allons décrire les principales méthodes de l'état de l'art et nous allons tenter de pallier leurs limitations en orientant notre étude vers un contexte d'acquisition LiDAR en intérieur. Nous nous intéresserons à des nuages de points qui échantillonnent des surfaces lisses par morceaux présentant des singularités. De plus, ces nuages sont susceptibles de contenir du bruit de mesure et une anisotropie dans l'échantillonnage qui sont des défauts directement liés au fonctionnement d'un scanner LiDAR et qu'il convient de prendre en compte lors du traitement local du nuage. La problématique majeure de ce chapitre est la suivante :

Problématique

Comment estimer les normales sur une surface continue par morceaux échantillonnée par des points ?

Afin de pouvoir utiliser le champ de normales dans une modélisation globale postérieure, l'estimateur de normales doit pouvoir fonctionner sous les contraintes suivantes :

Contraintes

— la surface est échantillonnée de manière anisotrope ;
— le nuage de points est bruité ;
— le temps de calcul est limité.

Nous commencerons par détailler les méthodes de l'état de l'art et leurs lacunes. Nous proposerons ensuite notre approche du problème. La méthode sera évaluée et comparée à 9 autres méthodes de l'état de l'art.

1.2 Etat de l'art

Dans l'estimation de normales, trois grandes étapes peuvent être distinguées : premièrement, le voisinage local du point étudié est sélectionné, deuxièmement, la direction de la normale est calculée et troisièmement, l'orientation de la normale est déterminée. La première étape peut être réalisée avec une recherche de plus proches voisins en utilisant des structures de données telles que des arbres k-d ou avec une tesselation de Delaunay [OY05]. La dernière étape peut, elle, être effectuée en utilisant une croissance de régions et des *spanning trees* [Hop+92]. La méthode que nous proposons permet de résoudre la seconde étape du processus. Nous nous efforcerons donc de décrire les travaux relatifs à ce sujet particulier dans ce chapitre. Le problème peut être formulé comme suit : soit $\mathbf{p}_i$ un point extrait d'un nuage de points $\mathscr{S}$, l'ensemble des N_s points voisins est appelé $S(\mathbf{p}_i) = \{\mathbf{p}_j\}_{j=1,\dots,N_s}$. Nous cherchons la normale $\mathbf{n}_i$ calculée en $\mathbf{p}_i$ en utilisant son voisinage.

Il existe quatre grandes catégories de méthodes permettant de résoudre ce problème : les méthodes fondées sur un ajustement de modèles prédéfinis, celles fondées sur un tirage aléatoire de voisins, les procédures *a posteriori* qui corrigent une estimation initiale et enfin, celles qui utilisent un diagramme de Voronoi.

1.2.1 Ajustement de modèles

Les méthodes de la première catégorie consistent à ajuster des modèles sur le voisinage sélectionné afin d'en déduire une normale par dérivation. Les modèles les plus employés sont : les plans [HK87 ; Hop+92 ; Mit+03], les sphères [GG07], les quadriques [YL99], [OY05], ou plus largement les n-jets [CP05] locaux.

L'ajustement le plus commun se réalise par moindres carrés. On cherche à estimer les paramètres de la surface modèle (contenus dans le vecteur θ) en minimisant la somme des normes L_2 des résidus r_j^θ, *c.-à-d.* les distances des voisins à la surface :

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{\mathbf{p}_j \in S(\mathbf{p}_i)} \left(r_j^\theta\right)^2 . \tag{1.1}$$

Le modèle de degré minimal qui peut être ajusté est le plan [HK87]. La solution à la minimisation par moindres carrés a alors une forme fermée qui peut être calculée par Analyse en Composantes Principales (ACP) [Hop+92 ; Mit+03]. En effet, on cherche $\mathbf{n}_i{}^*$ qui résout l'équation suivante :

$$\mathbf{n}_i{}^* = \underset{||\mathbf{n}_i||_2=1}{\operatorname{argmin}} \sum_{\mathbf{p}_j \in S(\mathbf{p}_i)} \left(\mathbf{n}_i \cdot (\mathbf{p}_j - \overline{\mathbf{p}})\right)^2 , \tag{1.2}$$

où $\overline{\mathbf{p}}$ est le centroïde du voisinage et $r_j = (\mathbf{n}_i \cdot (\mathbf{p}_j - \overline{\mathbf{p}}))$ est le résidu du voisin $\mathbf{p}_j$ par rapport au plan de normale $\mathbf{n}_i$ défini au centroïde. On peut réécrire le problème sous forme matricielle :

$$\mathbf{n}_i{}^* = \underset{||\mathbf{n}_i||_2=1}{\operatorname{argmin}} \mathbf{n}_i{}^T P^T P \mathbf{n}_i \,, \tag{1.3}$$

avec

$$P = \begin{pmatrix} (\mathbf{p}_1 - \overline{\mathbf{p}})^T \\ (\mathbf{p}_2 - \overline{\mathbf{p}})^T \\ \vdots \\ (\mathbf{p}_n - \overline{\mathbf{p}})^T \end{pmatrix}$$

On reconnaît la matrice de covariance définie par :

$$C := \frac{1}{N_s} P^T P \,. \tag{1.4}$$

Si on intègre la contrainte $||\mathbf{n}_i{}^*||_2 = 1$ à la fonction de coût, on obtient la forme lagrangienne suivante :

$$\mathbf{n}_i{}^* = \underset{\mathbf{n}_i}{\operatorname{argmin}} \left(\mathbf{n}_i{}^T N C \mathbf{n}_i + \lambda(\mathbf{n}_i{}^T \mathbf{n}_i - 1)\right). \tag{1.5}$$

En dérivant la fonction de coût matricielle et en l'annulant, on trouve :

$$C\mathbf{n}_i = -\frac{\lambda}{N_s} \mathbf{n}_i \,. \tag{1.6}$$

Par conséquent, l'équation (1.6) a trois solutions qui sont les trois vecteurs propres de C et la solution correspondant au minimum de l'équation (1.5) est le vecteur propre de la valeur propre la plus faible.

Lorsqu'on souhaite ajuster des surfaces de type polynomial comme dans la méthode n-jets [CP05], il est nécessaire de définir un repère (x, y, z) où (x, y) est le domaine de définition de la surface modèle locale. La contrainte sur ce repère est que l'axe z n'appartienne pas au plan tangent à la surface. Le domaine est communément défini par la méthode de modélisation de plan décrite précédemment, *c.-à-d.* qu'une ACP est préalablement effectuée sur le voisinage : x, y sont les vecteurs propres de la matrice de covariance correspondant aux valeurs propres les plus grandes. z est alors le vecteur propre correspondant à la valeur propre la plus faible.

Bien que la norme L_2 soit communément employée dans les ajustements de modèles de par sa dérivabilité et sa robustesse au bruit gaussien, d'autres optimisations ont été proposées afin d'exclure les points aberrants du processus. FLEISHMAN et coll. [Fle+05] utilisent une extension de la minimisation des moindres carrés médians (*Least Median of Squares* - LMS) pour initialiser leur estimation de normales. La méthode LMS consiste à chercher la surface θ qui minimise la fonction de coût suivante :

$$\underset{\mathbf{p}_j \in S(\mathbf{p}_i)}{\text{médiane}} \, | \, r_j^\theta \, | \,, \tag{1.7}$$

où r_j^θ est le résidu du point $\mathbf{p}_j$, *c.-à-d.* la distance du point à la surface θ. FLEISHMAN et coll. remplacent la médiane par un centile pour plus d'adaptabilité. Comme cette fonction de coût n'est pas dérivable, ils utilisent une méthode de type RANSAC (*RANdom SAmple Consensus*) pour résoudre l'optimisation. Pour cela, ils réalisent un tirage aléatoire de d points, d correspondant au degré de la surface modèle que l'on souhaite ajuster. Le processus est répété k fois et une surface est ajustée à chaque ensemble de points tirés. La surface finale sélectionnée est alors celle qui minimise (1.7). LIPMAN et coll. [Lip+07] utilisent une minimisation L_1-médiane dans leur méthode nommée « projection localement optimale » (*Locally Optimal Projection* - LOP) dans un contexte de reconstruction et d'échantillonnage de surface. Ils définissent la surface sous-jacente à un nuage de points comme le point fixe d'un opérateur de projection. L'opérateur de projection prend en entrée un nombre de points à projeter et est conditionné par deux critères : être au plus près localement des points initiaux et être assez éloigné des autres points échantillonnés pour éviter un effet d'agglomération. Cette méthode permet d'obtenir un emplacement des points précis sur la surface sous-jacente et est étendue par HUANG et coll. [Hua+13] pour prendre en compte les singularités comme expliqué plus bas. Cependant, les opérations de projection successives n'étant pas dérivables, elles ne permettent pas de déduire une normale du nuage de points final.

Une autre manière de traiter le problème est d'introduire une pondération dans l'optimisation. Dans le cas de l'ajustement de plan, PAULY et coll. [Pau+02] considèrent une version pondérée de la matrice de covariance. Plus précisément, pour n'importe quel vecteur m tel que chacun de ses éléments m_i est le poids associé au voisin $\mathbf{p}_i$, ils cherchent à résoudre l'équation suivante :

$$\mathbf{n}_i{}^* = \underset{||\mathbf{n}_i||_2=1}{\operatorname{argmin}} \sum_{\mathbf{p}_j \in S(\mathbf{p}_i)} m_j (\mathbf{n}_i \cdot (\mathbf{p}_j - \overline{\mathbf{p}}))^2 . \tag{1.8}$$

La nouvelle matrice de covariance pondérée est définie telle que :

$$C_M := \frac{1}{\sum_j m_j} P^T diag(m) P . \tag{1.9}$$

La solution de la minimisation (1.8) est le vecteur propre de C_M correspondant à la plus faible valeur propre (voir (1.5) et (1.6)).

Dans la même optique, la projection par moindres carrés glissants (*Moving Least Square* - MLS - *Projection*) [Lip+03 ; Ale+03] permet de reconstruire des surfaces définies par des nuages de points. Comme les autres méthodes d'ajustement de polynômes, cette méthode peut être séparée en deux étapes : un ajustement de plan pour définir le domaine et un ajustement de surface sur ce domaine. Ainsi, un plan H est ajusté sur le voisinage. On appelle q_i la projection du point p_i sur H. La particularité de cette méthode provient de l'attribution de poids aux voisins dépendamment de leur distance à q_i dans le processus d'ajustement du plan H. Ensuite, un polynôme est ajusté sur les points avec la pondération finale relative au plan H.

Bien que la majorité de ces méthodes permettent d'obtenir des estimations fiables, elles restent sensibles au bruit et ne traitent pas les zones de discontinuité dans la courbure. FLEISHMAN et coll. [Fle+05] intègrent un traitement des singularités dans l'ajustement de surface dans une méthode appelée « moindres carrés glissants robuste » (*Robust Moving Least-Squares* - RMLS). A partir d'un ajustement initial de surface par LMS tel que décrit ci-dessus, cette méthode consiste à ajouter itérativement le point du voisinage présentant le résidu le plus faible relativement à la surface et à actualiser la surface avec ce nouveau point. Le processus est arrêté lorsque le résidu étudié est supérieur à un seuil défini par l'utilisateur. Ensuite, une surface polynomiale est ajustée sur le voisinage extrait. Cette méthode requiert une forte densité de points et dépend principalement de la première étape de LMS qui n'est pas robuste à l'anisotropie du voisinage. De plus, le paramétrage de cette méthode peut s'avérer complexe lorsque les modèles sont bruités et contiennent de la courbure.

MEDEROS et coll. [Med+03] proposent d'utiliser une optimisation d'estimateurs robustes à la place d'une minimisation L_2 classique afin de rejeter les points n'appartenant pas au morceau de surface lisse du point étudié. Ils choisissent d'utiliser les M-estimateurs [Hub11]. Soient θ un vecteur de paramètres recherchés et $\{r_j\}_{j=1...N}$ les résidus associés au système pour un vecteur θ donné. La M-estimation est une méthode qui estime un vecteur θ^* en recherchant le minimum de vraisemblance des résidus d'ajustement de la façon suivante :

$$\theta^* := \underset{\theta}{\operatorname{argmin}} \sum_j \rho(r_j(\theta)) \,, \tag{1.10}$$

où la fonction $\rho(x)$ est la fonction noyau appelée M-estimateur. Cette fonction doit être positive, symétrique, elle doit passer par 0 et être décroissante sur l'intervalle $[0, +\infty[$ afin de neutraliser l'effet des points dont les erreurs sont trop fortes. Le choix de cette fonction dépend de l'utilisation et de la rapidité d'exclusion des points non-pertinents souhaitée. MEDEROS et coll. résolvent donc (1.10), les résidus étant les distances des points voisins au plan estimé. L'estimateur choisi est le suivant :

$$\rho(x) = 1 - e^{-x^2/(2\sigma^2)} \,. \tag{1.11}$$

Ils ajoutent également une seconde pondération relative à la distance entre la position initiale du point étudié $\hat{\mathbf{p}}_i$ et ses voisins. La fonction minimisée est donc la suivante :

$$\sum_{\mathbf{p}_j \in S(\mathbf{p}_i)} w_d(||\mathbf{p}_j - \hat{\mathbf{p}}_i||_2) \rho(\mathbf{n}_i \cdot (\mathbf{p}_j - \mathbf{p}_i)) \,, \tag{1.12}$$

où les poids de distance sont définis par :

$$w_d(x) := e^{-x^2/2{\sigma_d}^2} \,. \tag{1.13}$$

Ils minimisent cette fonction par une méthode de Newton alternativement selon l'orientation de $\mathbf{n}_i$ et selon la position de $\mathbf{p}_i$. Cette méthode est coûteuse en temps de calcul et sa convergence est fortement dépendante du niveau de bruit et de l'anisotropie dans le voisinage traité.

L'optimisation par M-estimation (1.10) peut aussi être réalisée avec les moindres carrés repondérés itérativement (*Iteratively Reweighted Least Square* - IRLS) [Wil79]. En effet, cette minimisation revient à résoudre l'équation :

$$\sum_j w_j(\theta)\frac{dr_j}{d\theta}r_j(\theta) = 0\,, \quad (1.14a) \qquad \text{avec} \qquad w_j(\theta) := \frac{1}{r_j(\theta)}\frac{\partial\rho}{\partial r_j}(r_j(\theta))\,. \quad (1.14b)$$

Si on considère des poids w_j fixes, cette équation revient donc à résoudre un problème quadratique du type :

$$\underset{\theta}{\operatorname{argmin}} \sum_j w_j r_j^2(\theta)\,. \tag{1.15}$$

Par conséquent, à chaque itération k de IRLS, deux processus sont appliqués successivement : premièrement, les poids sont attribués en utilisant (1.16) avec θ^{k-1}, puis, le problème d'optimisation (1.17) est résolu pour actualiser θ^k :

$$w_j{}^k := \frac{1}{r_j(\theta^{k-1})}\frac{\partial\rho}{\partial r_j}(r_j(\theta^{k-1}))\,, \tag{1.16}$$

$$\theta^k := \underset{\theta}{\operatorname{argmin}} \sum_j w_j{}^k \cdot (r_j(\theta))^2\,. \tag{1.17}$$

Des algorithmes relatifs aux minimisations quadratiques peuvent alors être appliqués pour résoudre l'équation (1.17) (*Gauss-Newton* ou *Levenberg-Marquardt* par exemple). La difficulté du problème d'optimisation dépend donc majoritairement de la régularité de r_j en fonction de θ.

1.2.2 Tirage aléatoire dans le voisinage

La deuxième catégorie de méthodes se fonde sur la sélection de triplés de points dans le voisinage étudié et sur le calcul d'un ensemble de normales correspondant aux plans définis par les triplés. La normale au point $\mathbf{p}_i$ peut ensuite être estimée de différentes façons, en calculant, par exemple, la moyenne [Gou71] ou la moyenne pondérée [Jin+05] de l'ensemble. Cependant, ces méthodes lissent les arêtes vives. BOULCH et MARLET [BM12] construisent un histogramme de l'ensemble des normales et utilisent une stratégie de vote inspirée de la transformée de Hough pour extraire la classe appropriée de l'histogramme. Selon les auteurs, la normale peut être calculée à partir de cette classe par moyennage des normales dans la classe. Ces méthodes mettent en évidence les singularités dans la courbure mais peuvent mener à des erreurs non négligeables sur des surfaces lisses. De plus, leur robustesse au bruit est limitée, comme nous le verrons dans la suite de ce chapitre, et le traitement de l'anisotropie requiert de nouveaux paramètres et une forte augmentation du temps de calcul. Une extension de cette méthode est proposée par les auteurs [BM16] qui remplacent la procédure de vote par un réseau de neurones pour prendre la décision de l'estimation finale à partir de l'histogramme. Malheureusement, cette méthode n'est pas aussi précise que la première lorsque le bruit est limité comme cela sera montré ci-après. En outre, la qualité du résultat dépend essentiellement de la base de

données d'entraînement et de la vérité terrain associée qui est difficile à obtenir pour des données réelles.

Récemment, GUERRERO et coll. [Gue+18] ont proposé une nouvelle méthode fondée également sur des réseaux neuronaux. L'entrée du réseau neuronal est le résultat de combinaisons symétriques entre voisins et de rotations du voisinage permettant au résultat d'être indépendant de l'ordre spatial. Nonobstant, cet algorithme a de faibles performances sur des données simples non bruitées comme nous le verrons dans la suite du chapitre.

1.2.3 Post-traitements

Optimisation locale

D'autres auteurs suggèrent l'utilisation d'un post-traitement d'un nuage de normales initial pour mettre en évidence les discontinuités de courbure.

Le champ de normales peut être filtré localement. YAGOU et coll. introduisent trois filtres classiques du champ de normales sur des maillages ([Yag+02 ; Yag+03]). Le filtre moyenneur effectue une moyenne locale pondérée par l'aire des triangles, il est robuste au bruit gaussien mais lisse les discontinuités de courbure. Le filtre médian sélectionne la normale correspondant à l'angle médian entre la normale du point étudié et celles de ses voisins. Ce filtre rend bien compte des discontinuités mais est sensible au bruit. Le filtre nommé « division-alpha » (*alpha-trimming*) est un compromis entre les deux filtres précédents : une fenêtre est fixée sur l'angle défini précédemment (on ne prend ni les normales trop proches, ni celles trop éloignées) et une moyenne des normales ayant un angle inclus dans cette fenêtre est réalisée. D'autres extensions existent telles que le vecteur médian flou (*fuzzy vector median*) [YE04] qui pondère la moyenne des normales voisines par leur distance à la médiane définie plus haut, ou encore la méthode proposée par SUN et coll. [Sun+07] qui utilise un seuil de différence entre normales permettant d'opérer une moyenne pondérée sur les normales les plus proches localement.

ZHENG et coll. [Zhe+11] adaptent le filtre bilatéral préalablement développé par TOMASI et MANDUCHI [TM98] pour débruiter des images afin de l'appliquer sur le champ de normales. Le nouveau vecteur normal à l'itération t est calculé comme suit :

$$\mathbf{n}_i^t = \frac{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} \mathbf{n}_j^{t-1} w_s(||\hat{\mathbf{n}}_j - \hat{\mathbf{n}}_i||_2) w_d(||\mathbf{p}_j - \mathbf{p}_i||_2)}{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} w_s(||\hat{\mathbf{n}}_j - \hat{\mathbf{n}}_i||_2) w_d(||\mathbf{p}_j - \mathbf{p}_i||_2)}, \tag{1.18}$$

où $\{\hat{\mathbf{n}}\_k\}_{k=\{i,j\}}$ désigne les vecteurs initiaux. w_d est défini dans la formule (1.13). De la même façon, w_s est défini par :

$$w_s(x) := e^{(-x^2/2{\sigma_s}^2)}. \tag{1.19}$$

Il est intéressant de noter que ZHENG et coll. travaillent avec des maillages et l'aire du triangle relatif au point voisin est ajoutée à la pondération pour minimiser les problèmes liés à l'anisotropie de l'échantillonnage. Cela est uniquement possible lorsque les informations de connectivité sont connues. Le moyennage est répété itérativement pour converger vers la solution, le vecteur estimé est normalisé à chaque itération.

ZHENG et coll. [Zhe+18] utilisent le même filtre en échangeant les termes initiaux et ceux optimisés à chaque itération. Le nouveau vecteur normal à l'itération t est alors :

$$\mathbf{n}_i^t = \frac{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} \hat{\mathbf{n}}_j w_s(||\mathbf{n}_j^{t-1} - \mathbf{n}_i^{t-1}||_2) w_d(||\mathbf{p}_j - \mathbf{p}_i||_2)}{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} w_s(||\mathbf{n}_j^{t-1} - \mathbf{n}_i^{t-1}||_2) w_d(||\mathbf{p}_j - \mathbf{p}_i||_2)}, \tag{1.20}$$

JONES et coll. [Jon+03] proposent une autre adaptation non-itérative du filtre bilatéral, toujours sur des maillages, pour projeter les points bruités sur la surface sous-jacente. La nouvelle position du point p_i est alors :

$$\mathbf{p}_i^* = \mathbf{p}_i + \frac{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} (\mathbf{n}_j \cdot (\mathbf{p}_j - \mathbf{p}_i)) w_s(\mathbf{n}_j \cdot (\mathbf{p}_j - \mathbf{p}_i)) w_d(||\mathbf{p}_j - \mathbf{p}_i||_2)\ \mathbf{n}_j}{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} w_s(\mathbf{n}_j \cdot (\mathbf{p}_j - \mathbf{p}_i)) w_d(||\mathbf{p}_j - \mathbf{p}_i||_2)}. \tag{1.21}$$

La ressemblance des normales voisines n'est plus directement exploitée, les auteurs préférent utiliser la notion de résidu du point étudié avec les plans tangents voisins dans la pondération. Comme précédemment [Zhe+11], ils ajoutent une pondération par l'aire des triangles voisins dans le moyennage.

Une variante itérative à cette procédure est proposée par FLEISHMAN et coll. [Fle+03], dans laquelle le résidu utilisé n'est pas celui de $\mathbf{p}_i$ sur les surfaces locales voisines mais celui des points voisins sur le plan prédéfini en $\mathbf{p}_i$. Le filtre devient donc :

$$\mathbf{p}_i^t = \mathbf{p}_i^{t-1} + \frac{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} (\mathbf{n}_i \cdot (\mathbf{p}_j^{t-1} - \mathbf{p}_i^{t-1})) w_s(\mathbf{n}_j \cdot (\mathbf{p}_j^{t-1} - \mathbf{p}_i^{t-1})) w_d(||\mathbf{p}_j^{t-1} - \mathbf{p}_i^{t-1}||_2)}{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} w_s(\mathbf{n}_i \cdot (\mathbf{p}_j^{t-1} - \mathbf{p}_i^{t-1})) w_d(||\mathbf{p}_j^{t-1} - \mathbf{p}_i^{t-1}||_2)}\ \mathbf{n}_i \tag{1.22}$$

Dans le cas d'un nuage de points, JONES et coll. [Jon+04] démontrent que la normale peut être déduite grâce aux résultats trouvés par BARR [Bar84] sur l'impact d'une déformation des points sur la normale. En effet, ce dernier démontre que si le point $\mathbf{p}_i$ subit une déformation de type $\mathbf{p}_i^* = F(\mathbf{p}_i)$ et qu'on définit J_F comme la matrice jacobienne de F au point $\mathbf{p}_i$, la nouvelle normale peut être exprimée par :

$$\mathbf{n}_i^* = \frac{J^{-T}\mathbf{n}_i}{||J^{-T}\mathbf{n}_i||_2}. \tag{1.23}$$

Dans ce cas, $F(\mathbf{p}_i)$ est assimilée à l'expression (1.21) et la normale peut être déduite de ce schéma par calcul de la jacobienne.

Öztireli et coll. [Özt+09] et Zheng et coll. [Zhe+18] utilisent ce procédé pour initialiser le champ de normales utile à leurs méthodes de reconstruction et débruitage respectivement. Ce type de méthodes requiert un champ initial fiable de normales et un échantillonnage dense. Leurs performances sont limitées sur des surfaces continues courbes.

Afin de pallier les problèmes précédents, Zhang et coll. [Zha+18] proposent une autre méthode. Ils isolent les points dont les voisinages contiennent potentiellement une discontinuité de courbure et cherchent des plans qui minimisent une fonction de coût faisant intervenir des couples de points dans le voisinage local. Une pondération (w_a) est associée à chaque couple de voisins dépendamment de la ressemblance entre leurs normales. Cette fonction de coût permet d'isoler les points du voisinage dont la normale a été affectée par une discontinuité de courbure car ils auront peu de voisins avec une normale similaire et donc peu de poids dans la minimisation. Les auteurs utilisent également les M-estimateurs pour chaque voisin dans la fonction de coût afin d'exclure les points les plus éloignés du plan étudié, ceux-ci pouvant correspondre à des points bruités ou à des points appartenant à une autre face.

$$\mathbf{n}_i{}^* = \underset{\mathbf{n}_i}{\operatorname{argmax}} \sum_{\mathbf{p}_j,\mathbf{p}_k \in S(\mathbf{p}_i)} \rho(\mathbf{p}_j,\mathbf{n}_i)\rho(\mathbf{p}_k,\mathbf{n}_i)w_a(\mathbf{n}_j,\mathbf{n}_k)\,, \tag{1.24}$$

où ρ est le M-estimateur tel que :

$$\forall\, \mathbf{p}_l \;\in\; S(\mathbf{p}_i),\; \rho(\mathbf{p}_l,\mathbf{n}_i) = \exp\left(\frac{-(\mathbf{n}_i\cdot(\mathbf{p}_l-\mathbf{p}_i))^2}{\sigma_r^2}\right), \tag{1.25}$$

et w_a est le poids associé au couple de voisins de la façon suivante :

$$w_a(\mathbf{n}_j,\mathbf{n}_k) = \exp\left(\frac{\cos^\alpha\left(\widehat{\mathbf{n}_j,\mathbf{n}_k}\right)}{\sigma_a^\alpha}\right). \tag{1.26}$$

α, σ_r et σ_a sont des paramètres de l'algorithme permettant d'adapter la sélection des points du voisinage. Cette optimisation est résolue d'une manière similaire à Fleishman et coll. [Fle+05] en testant des plans candidats ajustés à trois points aléatoirement choisis dans le voisinage. La normale du plan maximisant la fonction (1.24) est sélectionnée. Des poids de densité sont ajoutés à la fonction pour rendre l'algorithme robuste à l'anisotropie. Cette méthode est efficace sur des données peu bruitées mais est lente en comparaison des autres méthodes de l'état de l'art.

Optimisation globale

D'autres méthodes de correction de normales s'appuient sur une optimisation globale sur le nuage de points en prenant en compte les discontinuités de courbure. En partant d'un nuage de normales initial $\hat{N}$ constitué des éléments $\{\hat{\mathbf{n}}_1 \cdots \hat{\mathbf{n}}_{N_s}\}$, on peut chercher à estimer un vecteur de normales N^*, constitué des éléments $\{\mathbf{n}_1^* \cdots \mathbf{n}_{N_s}^*\}$, qui minimise une fonction de coût reflétant les différences entre chaque normale et ses voisines. En effet, lorsque la surface est lisse par morceaux, on s'attend à trouver une différence proche de 0 pour une majorité de couples de normales et

quelques valeurs élevées en cas de discontinuités. Afin de refléter la parcimonie du problème, ZHENG et coll. [Zhe+11] proposent une minimisation en utilisant la norme L_2 avec une pondération bilatérale. On appelle la matrice de poids W entre tous les points du nuage. Pour un ensemble de normales $N = \{\mathbf{n}_1, \ldots, \mathbf{n}_{N_x}\}$, $D(N)$ est un vecteur défini par :

$$D(N) := [\delta\mathbf{n}_0^T \; \delta\mathbf{n}_1^T \; \ldots \; \delta\mathbf{n}_{N_s}^T] \,, \tag{1.27}$$

avec :

$$\delta\mathbf{n}_i := \sum_{\mathbf{p}_j \in S(\mathbf{p}_i)} W_{ij}(\mathbf{n}_i - \mathbf{n}_j) \,, \tag{1.28}$$

les éléments de poids W_{ij} sont identiques à ceux du filtre bilatéral classique :

$$W_{ij} := \frac{w_s(||\hat{\mathbf{n}}_j - \hat{\mathbf{n}}_i||_2) w_d(||\mathbf{p}_j - \mathbf{p}_i||_2)}{\sum\limits_{\mathbf{p}_j \in S(\mathbf{p}_i)} w_s(||\hat{\mathbf{n}}_j - \hat{\mathbf{n}}_i||_2) w_d(||\mathbf{p}_j - \mathbf{p}_i||_2)} \,, \tag{1.29}$$

w_d et w_s étant définis dans les formules (1.13) et (1.19). La minimisation réalisée par ZHENG et coll. [Zhe+11] est donc la suivante :

$$N^* = \underset{N}{\operatorname{argmin}} \left(\, ||D(N)||_2^2 + \lambda ||N - \hat{N}||_2^2 \, \right) , \tag{1.30}$$

Le second terme de l'équation (1.30) est le terme d'attache aux données et permet de s'assurer que les normales estimées ne sont pas trop différentes des normales initiales $\hat{N}$. λ est le coefficient de Lagrange permettant d'équilibrer les deux termes.

Pour accentuer encore l'hétérogénéité du problème, AVRON et coll. [Avr+10] choisissent d'utiliser la norme L_1 dans leur optimisation. Une nouvelle pondération est également introduite, la matrice de poids est appelée W'. $M(a, b)$ est l'indice du $b^{\text{ème}}$ voisin du point d'indice a. Soit $D'(N)$ un vecteur dont chaque élément est défini par :

$$D'(N)_{iN_s+k} = {W'}_{iM(i,k)} ||\mathbf{n}_i - \mathbf{n}_{M(i,k)}||_2 \,. \tag{1.31}$$

Les poids ${W'}_{ij}$ étant définis comme suit :

$${W'}_{ij} := e^{-\left(\widehat{\mathbf{n}_j, \hat{\mathbf{n}}_i}/\sigma_\theta\right)^4} \,, \tag{1.32}$$

avec σ_θ le rayon d'influence de la fonction, paramétré par l'utilisateur. Les auteurs cherchent alors à résoudre :

$$N^* = \underset{N}{\operatorname{argmin}} \, ||D'(N)||_1 \quad t.q. \quad ||N - \hat{N}||_\infty < \gamma_n \,, \tag{1.33}$$

γ_n étant un paramètre fixé par l'utilisateur. A noter que la contrainte $||n_i|| = 1$ n'est pas prise en compte dans cette minimisation ; les auteurs normalisent le résultat après optimisation.

Avec le même objectif, SUN et coll. [Sun+15] vont plus loin et utilisent une minimisation L_0 pour estimer les normales. Soit $D''(N)$ un vecteur tel que chacun de ses éléments est défini par :

$$D''(N)_{iN_s+k} = ||n_i - n_{M(i,k)}||_2 \, . \tag{1.34}$$

Ils cherchent alors à résoudre l'équation suivante :

$$N^* = \underset{N}{\operatorname{argmin}} \left(\, ||D''(N)||_0 + \lambda ||N - \hat{N}||_2^2 \, \right) \tag{1.35}$$

Cette fonction de coût est difficile à minimiser par des méthodes classiques. Les auteurs s'appuient donc sur les travaux de XU et coll. [Xu+11] et HE et SCHAEFER [HS13] pour résoudre ce problème de manière itérative. Comme ZHENG et coll. [Zhe+11], ils négligent la contrainte sur la norme de chaque normale et normalisent les vecteurs à la fin du traitement.

Optimisation semi-locale

Une troisième approche est proposée à travers les méthodes de reconstruction de noyau de faible rang (*Kernel Low-rank Recovery*) dans un contexte de débruitage de maillages [Che+19] ou de suppression de texture [Wei+18]. Elles consistent à regrouper des patchs de surfaces (ou voisinages de faces) similaires via une comparaison de leur champ de normales afin d'enrichir les informations de courbure des voisinages sans les étendre spatialement. De ces groupes de patchs, les auteurs extraient de nouvelles normales qui servent de références dans les poids w_s d'un filtre bilatéral (voir équation (1.19)). Pour créer les groupes, les normales sont tout d'abord placées dans une matrice. Les auteurs cherchent alors à remettre en évidence la redondance d'information de courbure en modifiant les éléments de cette matrice de façon à abaisser son rang tout en garantissant une modification minimale du champ de normales. Cette procédure améliore le débruitage d'un champ de normales issu d'un maillage en cas de fort bruit, et évite une trop forte distorsion due à l'accroissement du voisinage mais n'est pas encore adaptée au traitement de nuages de points.

Couplage

Dans le contexte du débruitage de maillages, certains auteurs proposent de coupler itérativement le filtrage du champ de normales avec un déplacement des points, c'est ce que SUN et coll. [Sun+07] appellent la procédure « deux-étapes » (*two-steps*). Deux schémas peuvent alors s'appliquer :

- deux-étapes/une-phase (*two-steps/one-stage*) : à chaque itération, le champ de normales est actualisé et les points sont déplacés afin d'être ajustés au mieux sur le nouveau champ de normales selon leurs connexions dans le maillage [Tau01] ;
- deux-étapes/deux-phases (*two-steps/two-stages*) : les normales sont d'abord optimisées à travers différentes itérations, puis les points sont également optimisés par un traitement similaire.

La procédure deux-étapes/une-phase est utilisée notamment dans les travaux de YAGOU et coll. [Yag+02], CHEN et CHENG [CC08] et HUANG et coll. [Hua+13]. HUANG et coll. utilisent cette méthode dans le but de ré-échantillonner des surfaces définies par des nuages de points. Ils couplent un filtre bilatéral [Zhe+11] et un déplacement de points dans la direction opposée aux discontinuités de courbure en utilisant une LOP pondérée. Les zones de discontinuités sont ré-échantillonnées ultérieurement en utilisant un algorithme d'intégration. Cette méthode est évaluée dans la suite de ce chapitre.

1.2.4 Diagramme de Voronoi et ACP

La dernière catégorie de méthodes est fondée sur la construction d'un diagramme de Voronoi à partir de cellules 3D. Le vecteur normal estimé en un point du nuage peut alors être estimé à partir de la géométrie de sa cellule de Voronoi. Cette géométrie peut être extraite à partir de pôles de Voronoi [AB99 ; DS06] ou en réalisant une ACP à partir de la mesure de covariance des cellules de Voronoi (*Voronoi Covariance Measure*) [All+07]. Un exemple du procédé de la dernière méthode est montré figure 1.1. MÉRIGOT et coll. [Mér+11] introduisent la mesure de covariance de Voronoi convoluée (*Convolved Voronoi Covariance Measure*), afin de rendre l'algorithme précédent robuste au bruit. Nous appellerons cette dernière version VCM par la suite. La covariance en un point est alors la somme des covariances des cellules de Voronoi voisines. La dernière approche détecte intrinsèquement les discontinuités de courbure mais l'aspect convolutif de l'algorithme mène encore à un résultat lissé et sa robustesse au bruit est limitée comme nous le verrons dans la section d'évaluation. Par ailleurs, CHE et OLSEN [CO18] introduisent un détecteur d'arêtes à l'aide d'une triangulation locale dans le but d'améliorer une première estimation de normales.

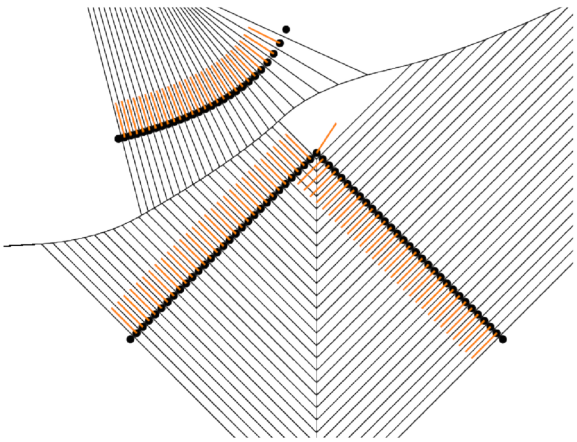


Figure 1.1.: Estimation de normales (oranges) par construction de cellules de Voronoi en 2D par MÉRIGOT et coll. Image extraite de leur article [Mér+11].

Les méthodes listées dans cette section allient difficilement une précision d'estimation fiable sur les zones lisses avec une prise en compte des discontinuités de courbure.

Leurs performances restent très dépendantes de l'uniformité de l'échantillonnage, de sa densité et du niveau de bruit. De plus, les temps de calculs de certains algorithmes sont rédhibitoires pour le traitement de nuages contenant plusieurs millions de points. Cet état de l'art nous permet donc de fixer les objectifs d'une nouvelle approche que nous développons dans la section suivante :

Objectifs

1. Prendre en compte les singularités dans l'estimation d'une normale sans perdre en précision sur les surfaces lisses.
2. Ne pas utiliser d'information de courbure associée aux points voisins, afin de mieux localiser l'estimation.
3. Intégrer des outils de robustesse au bruit et à l'anisotropie.
4. Conserver des temps de calcul compétitifs avec l'état de l'art.

1.3 Méthode

Notre algorithme réalise un ajustement de plan avec un M-estimateur dans le voisinage d'un point étudié. Il produit une direction de normale et une séparation du voisinage. Dans la section 1.3.1, nous expliquons comment la M-estimation entraîne l'intégration d'une ACP pondérée à un processus itératif. Ensuite, une description détaillée de l'algorithme est présentée dans la section 1.3.2.

1.3.1 ACP pondérée itérative

Notre méthode a pour but de séparer le voisinage en deux parties. La première contient les points échantillonnant le morceau de surface lisse auquel appartient $\mathbf{p}_i$, sous un niveau de bruit donné, et qui participent au calcul de la normale. La seconde contient les autres points qui sont exclus de l'estimation. Il semble donc adéquat d'utiliser un M-estimateur [Med+03] dans le processus d'ajustement. Supposons que nous disposions d'une estimation de normale $\mathbf{n}_i$ au point $\mathbf{p}_i$, nous définissons le résidu r_j relatif au voisin $\mathbf{p}_j$ par $r_j(\mathbf{n}_i) := \mathbf{n}_i \cdot (\mathbf{p}_j - \mathbf{p}_i)$. La sélection du M-estimateur est critique, car on ne souhaite pas ajouter de filtre supplémentaire sur le voisinage étudié. Nous avons choisi l'estimateur suivant :

$$\rho(r_j(\mathbf{n}_i)) := \frac{\mu r_j(\mathbf{n}_i)^2}{2(\mu + r_j(\mathbf{n}_i)^2)}, \tag{1.36}$$

qui est une extension de l'estimateur de *Geman-McClure* [He+14], paramétrée par un scalaire μ pour définir son rayon d'action. Contrairement au travail de Mederos et coll. [Med+03] dans lequel les auteurs ont recours à une méthode de Newton

coûteuse en temps de calcul pour résoudre l'optimisation, une minimisation par IRLS est réalisée (voir § 1.2.1). Les poids sont calculés comme suit :

$$w_j(\mathbf{n}) := \left(\frac{\mu}{\mu + r_j(\mathbf{n}_i)^2} \right)^2 , \tag{1.37}$$

puis, la fonction ci-dessous est minimisée en considérant des poids constants :

$$\sum_j w_j \cdot r_j(\mathbf{n})^2 . \tag{1.38}$$

On retrouve alors la fonction minimisée dans (1.8), la référence de l'analyse en composantes principales n'étant plus le centroïde mais le point étudié $\mathbf{p}_i$. Par conséquent, l'étape d'optimisation (1.38) a une solution fermée qui est le vecteur propre correspondant à la valeur propre la plus faible de la matrice de covariance telle que définie dans la formule (1.9), le centroïde $\overline{\mathbf{p}}$ étant remplacé par le point étudié $\mathbf{p}_i$. Cette modification permet de discriminer de manière plus agressive les points appartenant à une autre face. En partant d'une estimation initiale n_i^{init} en $\mathbf{p}_i$, on alterne donc deux étapes : les poids sont actualisés relativement à l'estimation courante de la normale et une nouvelle estimation est déduite d'une ACP pondérée. La fonction de poids (1.37) est représentée figure 1.2 pour différentes valeurs du rayon μ.

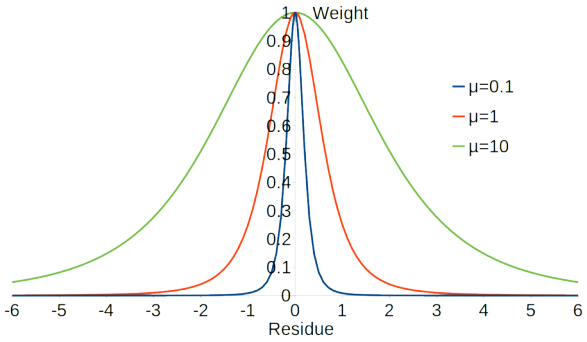


Figure 1.2.: Fonction de poids pour trois valeurs du paramètre μ.

On peut observer que si $r_j^2 \ll \mu$, alors w_j est proche de 1 et $\mathbf{p}_j$ a un fort impact dans l'estimation. Au contraire, si $r_j^2 \gg \mu$, le poids w_j est faible et $\mathbf{p}_j$ est exclu de l'estimation. Une conséquence de l'utilisation de IRLS pour résoudre l'optimisation est que la fonction de coût peut ne plus être convexe si le noyau choisi est trop petit. En effet, si μ est fixé, pour toutes les itérations, à une valeur supérieure aux carrés de toutes les erreurs, l'estimation finale restera éloignée de la direction optimale de la normale. Si μ est fixé à une valeur inférieure aux carrés de toutes les erreurs, des minima locaux peuvent apparaître et l'algorithme peut ne pas converger vers la solution optimale. Par conséquent, le paramètre μ dans l'équation (1.37) est divisé à chaque itération k par un facteur de division qu'on nomme χ pour calibrer la fonction d'erreur sur les valeurs des résidus qui diminuent également :

$$\mu_k = \frac{\mu_{k-1}}{\chi} . \tag{1.39}$$

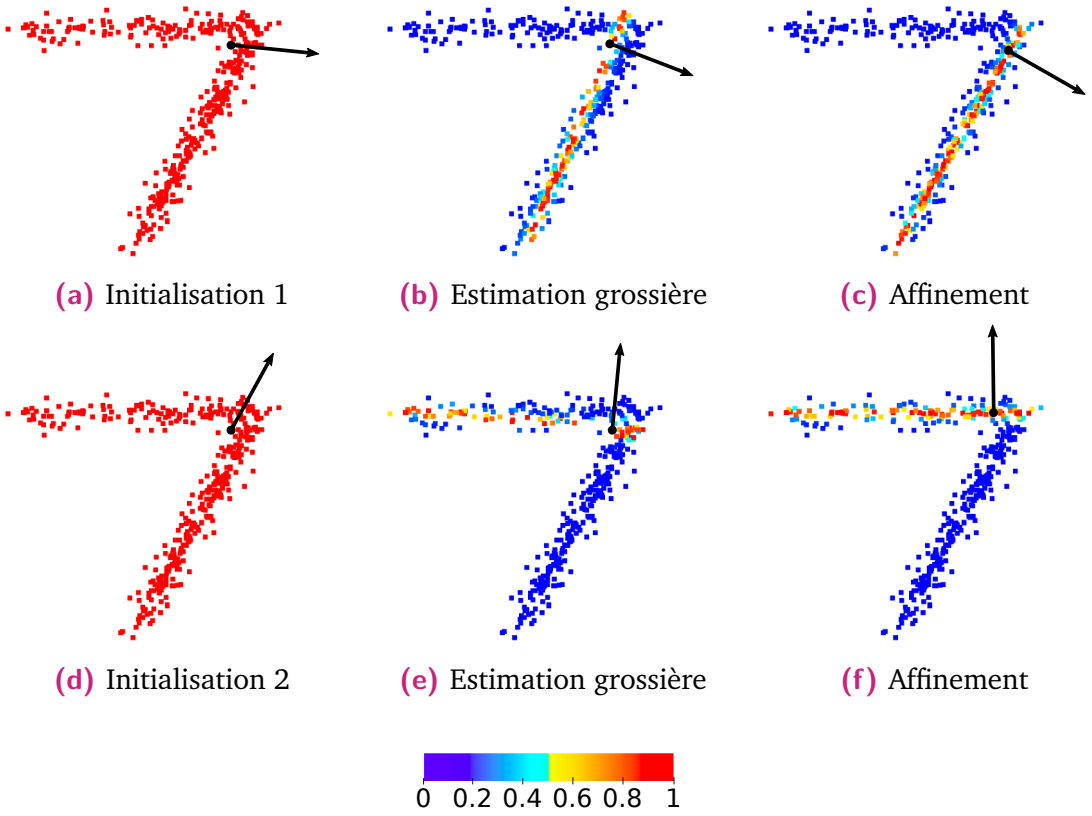
1.3.2 Description de l'algorithme

Dans cette section, nous expliquons comment paramétrer la procédure décrite précédemment. Cette procédure permet d'obtenir une normale fiable lorsque le nuage de points n'est pas bruité et présente un échantillonnage uniforme. Cependant, elle n'est pas suffisante pour traiter des nuages de points bruités et des voisinages anisotropes (de par la géométrie de l'objet ou son échantillonnage). Afin de réduire l'influence du bruit, nous expliquons comment l'optimisation est divisée en deux étapes successives : une estimation grossière de la normale est obtenue puis une étape d'affinement est réalisée. Ensuite, nous expliquons comment traiter le problème de confusion entre faces, dû à l'anisotropie présente au voisinage des arêtes.

Etapes élémentaires

Dans ces étapes, le vecteur normal $\mathbf{n}_i$ est estimé itérativement en utilisant une ACP pondérée et en actualisant les poids à chaque itération selon la formule (1.37). L'estimation initiale est calculée avec une ACP classique (voir figure 1.3a), on la nomme $\mathbf{n}_{i1}^{init}$ (voir la section 1.3.2 pour l'explication derrière l'indice « 1 »). La première étape actualise uniquement la direction du vecteur normal. Dans la deuxième étape, la position de la référence de l'ACP est également modifiée pour affiner l'estimation de la normale en présence de bruit.

- **Estimation grossière :** μ est initialisé comme le carré du résidu maximum calculé dans le voisinage : $\mu_{init} = (\max(r_j))^2$. A chaque itération, μ est diminué pour ajuster les poids sur les valeurs des résidus. Si la diminution est trop rapide, l'algorithme peut tomber dans un minimum local. Si la diminution est trop lente, l'algorithme peut être plus long que nécessaire. Nous avons utilisé un facteur de division χ de 1.01 à chaque itération k dans tous nos tests. L'effet de ce facteur de division χ est détaillé sur la figure 1.10. Les itérations sont réalisées jusqu'à ce que μ atteigne un seuil μ_{lim} qui dépend de la courbure et du bruit dans le modèle. Le choix de cette valeur est discutée dans la section 1.3.2. Le nombre d'itérations est donc : $\ln(\mu_{init} \,/\, \mu_{lim})/\ln(\chi)$. La normale résultante est illustrée sur un exemple planaire dans la figure 1.3b.
- **Affinement :** cette étape permet d'affiner la première estimation de normale en présence de bruit. Jusqu'à présent, le point de référence de l'ACP est p_i et il est considéré fixe. L'algorithme ne converge pas vers la solution optimale à cause du bruit sur p_i. Nous affinons donc le résultat en actualisant la référence de l'ACP dans un schéma itératif. Alternativement à chaque itération, un vecteur normal $\mathbf{n}$ est calculé (tel que décrit précédemment) et le point de référence est déplacé le long de l'axe de la normale. Le déplacement est défini comme la moyenne pondérée des résidus : $\sum_i w_i r_i(\mathbf{n}) / \sum_i w_i$. La normale estimée résultante est appelée $\mathbf{n}_{i1}$, son estimation est illustrée sur l'exemple figure 1.3c. La position finale du point de référence de l'ACP est appelée $\mathbf{p}_{i1}$. L'effet de cette étape d'affinement est mis en évidence sur la figure 1.6.



(a) Initialisation 1 **(b)** Estimation grossière **(c)** Affinement

(d) Initialisation 2 **(e)** Estimation grossière **(f)** Affinement

Figure 1.3.: Evolution de l'estimation de la normale et des poids sur le voisinage d'une discontinuité de courbure avec un échantillonnage non-uniforme : la face horizontale contient moins de points que l'autre face. Les couleurs représentent les poids associés aux voisins de 0 (bleu) à 1 (rouge). **(a)**, **(b)** et **(c)** correspondent aux étapes de calculs menant à la normale estimée $\mathbf{n}_{i1}$, **(d)**, **(e)** et **(f)** correspondent à celles menant à $\mathbf{n}_{i2}$. **(a)** et **(d)** correspondent aux deux étapes d'initialisation. **(b)** et **(e)** correspondent aux résultats des estimations grossières. **(c)** et **(f)** correspondent aux résultats des étapes d'affinement.

Traitement de l'anisotropie : deuxième initialisation

Si une arête se trouve dans le voisinage de $\mathbf{p}_i$, la normale résultante de l'étape précédente peut être attirée par le côté de l'arête contenant le plus de points et/ou par le côté présentant le moins de courbure. Pour ces raisons, une seconde direction initiale est calculée et une seconde optimisation est réalisée (voir dernière ligne de la figure 1.3) pour obtenir une nouvelle estimation de normale. Afin de minimiser les chances de retrouver la première alternative de normale, la seconde normale initiale doit être à $90°$ de $\mathbf{n}_{i1}$ et de la direction de l'arête ν. La direction de l'arête est déduite du produit vectoriel entre $\mathbf{n}_{i1}^{init}$ et $\mathbf{n}_{i1}$, en faisant l'hypothèse que le vecteur normal a été optimisé en suivant un plan perpendiculaire à l'arête en raison de sa symétrie. La seconde normale initiale, définie par :

$$\mathbf{n}_{i2}^{init} := \mathbf{n}_{i1} \wedge \nu \,,$$

est montrée sur l'exemple planaire, figure 1.3d. Puis, les deux étapes élémentaires sont réalisées menant respectivement aux résultats des figures 1.3e et 1.3f. Pour cette seconde initialisation, μ doit être initialisé par une valeur faible pour orienter le résultat vers un minimum local et une estimation différente de $\mathbf{n}_{i1}$. Dans tous nos tests, nous avons choisi d'initialiser μ à la valeur correspondant au tiers des résidus du voisinage ordonnés. On appelle cette seconde estimation $\mathbf{n}_{i2}$ et la nouvelle position de $\mathbf{p}_i$, $\mathbf{p}_{i2}$.

Après la seconde optimisation, un choix doit être fait entre les deux vecteurs estimés, $\mathbf{n}_{i1}$ et $\mathbf{n}_{i2}$. Premièrement, les normales estimées sont orientées vers l'extérieur de la courbure, *c.-à-d.* que chaque vecteur $\{\mathbf{n}_{ik}\}_{k=1,2}$ doit satisfaire la condition suivante :

$$\sum_j (\mathbf{n}_{ik} \cdot (\mathbf{p}_j - \mathbf{p}_i)) < 0, \;\; k \in \{1,2\}\,.$$

Deuxièmement, pour les deux vecteurs $\mathbf{n}_{i1}$ et $\mathbf{n}_{i2}$, les valeurs d'erreur $((\mathbf{p}_{ik} - \mathbf{p}_i).\mathbf{n}_{i1})$ et $((\mathbf{p}_{ik} - \mathbf{p}_i).\mathbf{n}_{i2})$ sont calculées et l'estimation menant à la valeur d'erreur minimum est sélectionnée. L'effet de cette double initialisation est évaluée figure 1.7.

Propriétés de convergence et paramétrage

Notre algorithme s'arrête quand μ atteint la valeur prédéfinie μ_{lim}. Rappelons que μ représente la bande passante de l'estimateur, *c.-à-d.* que les voisins ayant un résidu r_j tel que ${r_j}^2 \leq \mu_{lim}$ ont un fort impact sur l'estimation de la normale tandis que les voisins ayant un résidu tel que ${r_j}^2 > \mu_{lim}$ ont un faible impact. Si μ_{lim} est trop grand, la normale optimale peut ne pas être retrouvée à l'arrêt de l'algorithme. Si μ_{lim} est trop petit, l'algorithme peut diverger. L'erreur est limitée par l'erreur de l'estimation par une ACP lorsque la valeur de μ_{lim} est augmentée.

Une valeur optimale de μ_{lim} peut être recherchée puisque cette valeur dépend uniquement du niveau de bruit et de la courbure maximum tolérée dans le modèle. Pour calculer cette valeur optimale, nous recherchons le résidu maximum acceptable r_{max} dans le voisinage. Puis, μ_{lim} est déduit avec $\mu_{lim} = {r_{max}}^2$, pour que le poids soit de 0.5 pour les voisins ayant un résidu égal à r_{max}. Si le modèle étudié peut être représenté par un ensemble de surfaces planes, r_{max} dépend uniquement du niveau de bruit gaussien dans le modèle, évalué grâce à son écart type σ. Pour les autres modèles, l'effet induit par la courbure doit être ajouté. En effet, comme exposé dans la figure 1.4 sur un exemple 2D de zone à courbure lisse sans bruit, tous les voisins doivent participer à l'estimation de la normale pour préserver la symétrie après convergence. Par conséquent, un rayon de courbure minimal acceptable R_{min} doit être défini par l'utilisateur. Dans un cas non bruité, r_{max} est alors défini comme le résidu du voisin le plus éloigné de p_i sur une surface théorique de rayon de courbure R_{min}. Dans le cas général, on appelle cette valeur de résidu X_{max} (voir figure 1.4). X_{max} est lié à R_{min} et à la distance du voisin le plus éloigné d_{max} par la relation : $X_{max} = d_{max}^2/2R_{min}$.

Par conséquent, en pratique nous utilisons la valeur : $r_{max} = X_{max} + 0.5\sigma'$, avec $\sigma' = \frac{\sigma}{\sqrt{3}}$ l'écart type du bruit gaussien dans une direction de l'espace. Dans la section 1.4,

l'effet de μ_{lim} sur la précision des résultats est évalué afin de corroborer les choix pratiques. Plus R_{min} est définie par une valeur élevée, plus le nombre de points exclus de l'estimation à travers la pondération sera important.

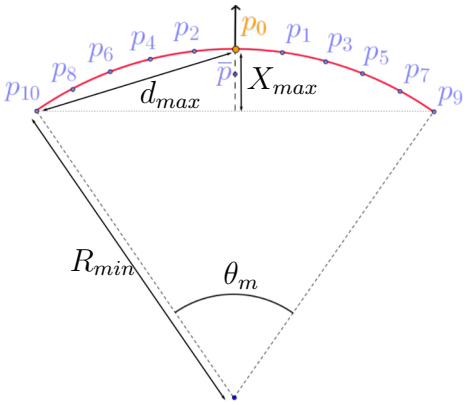


Figure 1.4.: Schéma d'une surface courbe projetée en 2D. X_{max} est la valeur du résidu maximum acceptable d'un voisin pour participer à l'estimation de la normale dans ce voisinage. La normale est calculée en $\mathbf{p}_i$ en utilisant les voisins $\{\mathbf{p}_1 \dots \mathbf{p}_{10}\}$. Si certains voisins ne sont plus pris en compte dans l'estimation, l'orientation de la normale risque d'être erronée.

Présélection des points

Afin de rendre l'algorithme plus efficace, le processus itératif d'optimisation peut être appliqué uniquement pour les points détectés au voisinage des arêtes. En effet, après la première initialisation par une ACP classique, on calcule l'écart type des résidus relatifs au plan défini au centroïde. Si l'écart type est inférieur à un seuil τ, on suppose que le voisinage ne contient pas de discontinuité de courbure et les itérations peuvent être évitées. τ est défini comme l'écart type théorique de tous les points d'une surface de rayon de courbure R_{min}. Cet écart type théorique peut être calculé par intégration des résidus au carré sur la surface :

$$\tau := \sqrt{\frac{R_{min}^2}{2}\left(1 + \frac{\sin(\theta_m)}{\theta_m}\right) - \overline{p}^2}\,, \tag{1.40}$$

où θ_m est l'angle d'ouverture (voir figure 1.4) défini de la façon suivante :

$$\theta_m := 2\arccos\left(\frac{R_{min} - X_{max}}{R_{min}}\right)\,, \tag{1.41}$$

et $\overline{p}$ est le centroïde calculé théoriquement par :

$$\overline{p} := 2R_{min}\frac{\sin(\frac{\theta_m}{2})}{\theta_m}\,. \tag{1.42}$$

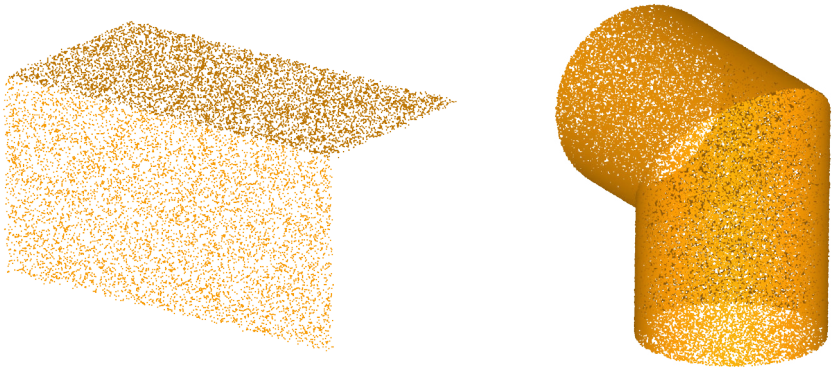
L'effet de la présélection sur les résultats est évalué section 1.4.

1.4 Evaluation interne de l'algorithme

1.4.1 Méthodologie

Afin d'évaluer les étapes et paramètres de l'algorithme proposé, nous utilisons deux modèles synthétiques représentés figures 1.5a et 1.5b. Le premier modèle est une intersection de deux plans formant une arête, échantillonnée aléatoirement par $15\,000$ points. Chacun de ces plans a une longueur de $1\,\mathrm{m}$ et une largeur de $0.5\,\mathrm{m}$. L'arête mesure $1\,\mathrm{m}$. L'angle d'intersection des plans est $90°$. Un bruit gaussien est ajouté aux points avec un écart type compris entre 0 et $2\,\mathrm{cm}$. Le voisinage est défini par 300 points. Ce modèle est appelé modèle biplanaire par la suite.

Le second modèle synthétique est une intersection de cylindres formant un coude échantillonné aléatoirement par $100\,000$ points. Chaque cylindre a une longueur de $0.2\,\mathrm{m}$ et un rayon de $0.1\,\mathrm{m}$. Les cylindres s'intersectent à $90°$ et un bruit gaussien est ajouté aux points avec un écart type compris entre 0 et $1\,\mathrm{cm}$. Le voisinage est défini par 200 points. Ce modèle est nommé modèle bicylindrique par la suite.



(a) Modèle biplanaire **(b)** Modèle bicylindrique

Figure 1.5.: Modèles synthétiques utilisés pour l'évaluation interne de notre algorithme.

Nous allons étudier successivement les effets de l'étape d'affinement, de la double initialisation, du préfiltrage des points ainsi que la convergence de notre algorithme. Pour ce faire, nous utiliserons l'erreur angulaire moyenne des normales avec la vérité terrain et les temps de calcul.

1.4.2 Etape d'affinement

Les normales résultantes de l'algorithme proposé sont évaluées avec ou sans l'étape d'affinement sur le modèle biplanaire et les résultats sont mis en évidence figure 1.6. Cette figure montre que l'étape d'affinement rend l'algorithme plus robuste au bruit gaussien. L'erreur angulaire moyenne est réduite jusqu'à six fois lorsque cette étape est ajoutée au traitement.

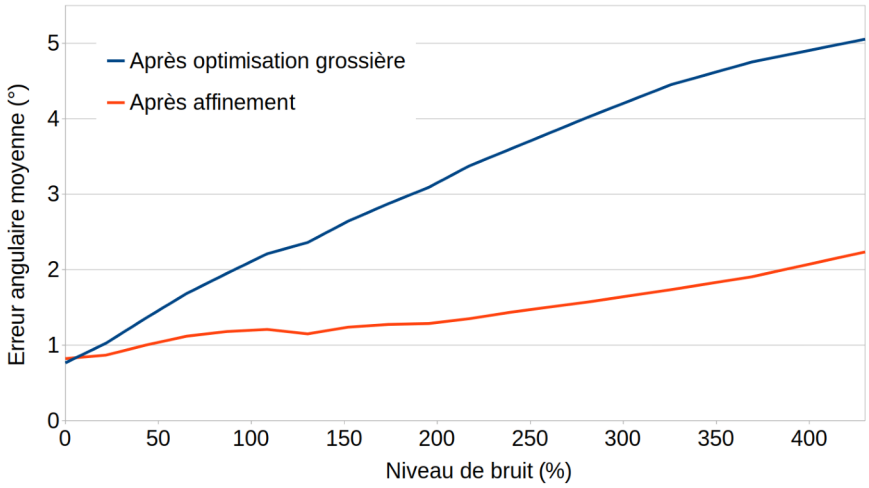


Figure 1.6.: Evaluation de l'étape d'affinement de l'algorithme proposé. L'erreur angulaire moyenne est évaluée sur le modèle biplanaire avec un bruit gaussien croissant (le niveau de bruit est défini comme le pourcentage de l'écart type relativement à la distance moyenne entre plus proches voisins). L'algorithme proposé est évalué avec et sans l'étape d'affinement avec 300 points voisins.

1.4.3 Double initialisation

Sur la figure 1.7, les normales résultantes de notre algorithme sont comparées avec ou sans la deuxième optimisation sur le même modèle, sans bruit, et en faisant varier la densité des points indépendamment sur les deux plans. On observe que cette étape entraîne une diminution de l'erreur angulaire moyenne. Par exemple, lorsqu'une face contient deux fois plus de points que l'autre, la seconde optimisation permet une réduction de l'erreur angulaire moyenne d'un facteur 90.

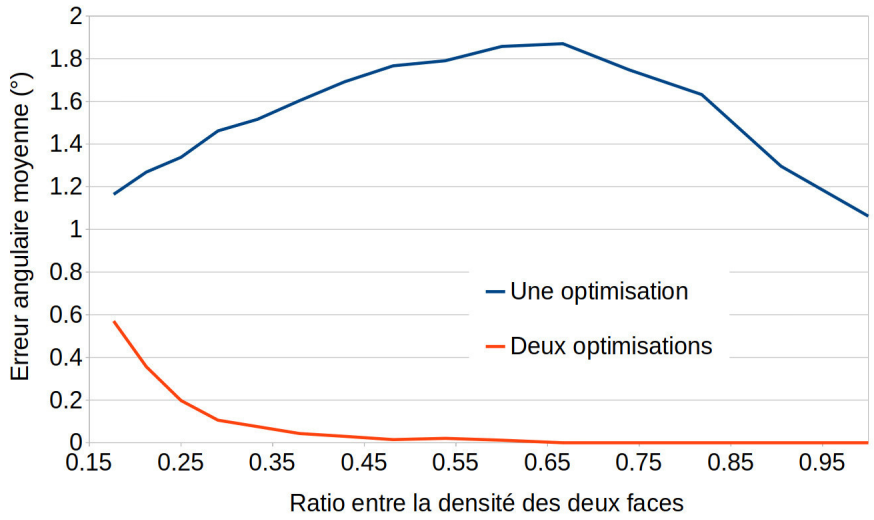


Figure 1.7.: Evaluation de la double initialisation de notre algorithme. L'erreur angulaire moyenne est évaluée sur le modèle biplanaire avec un échantillonnage non-uniforme : une face contient plus de points que l'autre. Le ratio entre la densité des deux faces est augmenté. Le voisinage est défini par 300 points.

1.4.4 Paramétrage de la convergence

Troisièmement, les paramètres de convergence sont étudiés. La figure 1.8 illustre la variation de l'erreur angulaire moyenne selon la valeur de r_{max} (voir § 1.3.2) sur le modèle biplanaire, pour quatre niveaux de bruit différents. On remarque que l'erreur angulaire moyenne diminue quand μ_{lim} est diminué jusqu'à un point optimal. Si μ_{lim} est trop petit, l'algorithme diverge (*c.-à-d.* l'erreur augmente à nouveau de façon marquée). De plus, ce graphique confirme expérimentalement qu'une valeur adaptée pour r_{max} est $(0.5\ \sigma')$ quand le modèle ne contient pas de courbure, avec σ' l'écart type du bruit dans une direction de l'espace.

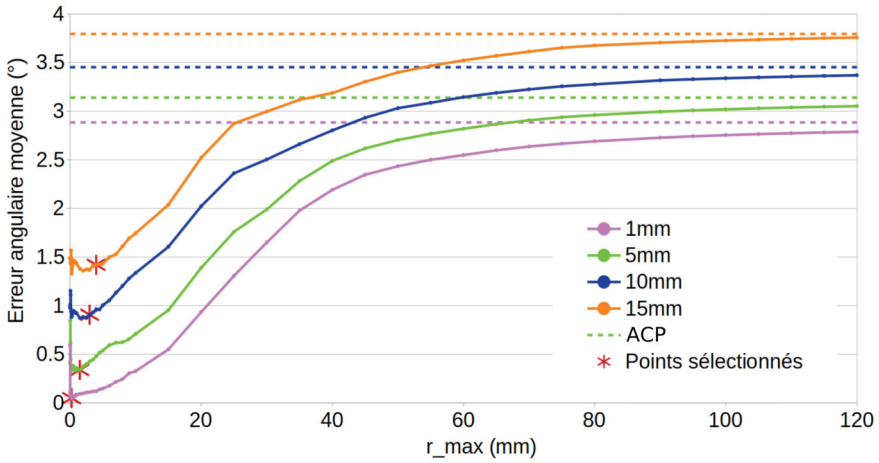


Figure 1.8.: Erreur angulaire moyenne évaluée sur le modèle biplanaire en fonction de la valeur de e_{max}. L'évaluation est réalisée pour 4 niveaux de bruit gaussien (niveau de bruit défini par l'écart type en mm). Les erreurs résultantes des estimations par ACP sont représentées par des pointillés.

Pour des modèles présentant de la courbure, r_{max} peut être défini par $r_{max} := X + 0.5\sigma'$, X étant l'effet induit par la courbure et déterminer X. Nous utilisons à présent le modèle bicylindrique pour tester la convergence en présence de courbure. La figure 1.9 montre l'évolution de l'erreur angulaire moyenne en fonction de X pour quatre niveaux de bruit gaussien. Elle montre que les meilleurs résultats sont obtenus pour une valeur proche de $X = 0.8\,\mathrm{mm}$ qui correspond à la longueur X_{max} définie dans la section 1.3.2. En effet, cela confirme que X_{max} est un bon choix pour définir μ_{lim} dans un cas non-bruité et en présence de zones courbes. En outre, sur ces deux graphiques, on peut vérifier que les estimations par ACP correspondent aux asymptotes des courbes relatives à l'algorithme proposé quand μ_{lim} tend vers l'infini.

L'effet du facteur de division χ sur la précision du résultat est illustré figure 1.10a. Son impact sur le temps de calcul est représenté figure 1.10b. Ces figures montrent que le facteur de division n'a qu'un impact limité sur la précision du résultat : l'erreur angulaire moyenne est augmentée de $0.4°$ quand le facteur de division passe de 1 à 1.14. Néanmoins, un compromis doit être trouvé afin de limiter le temps de calcul tout en optimisant la précision. Nous avons choisi expérimentalement 1.01 dans tous les tests inclus dans cette étude.

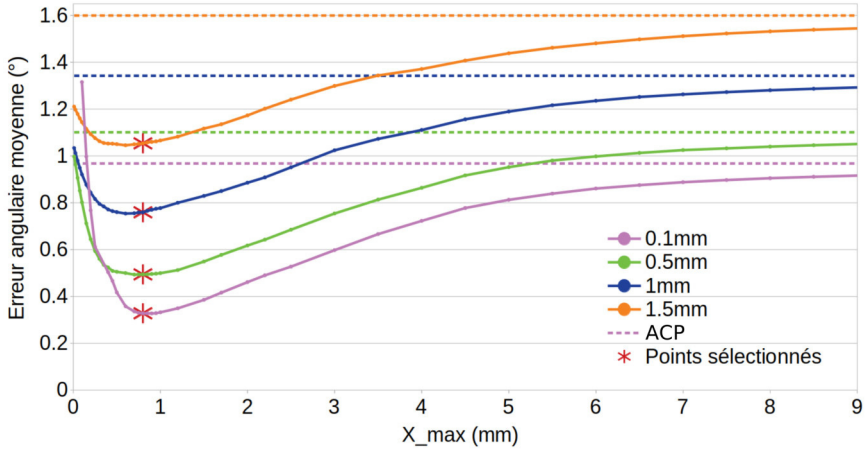
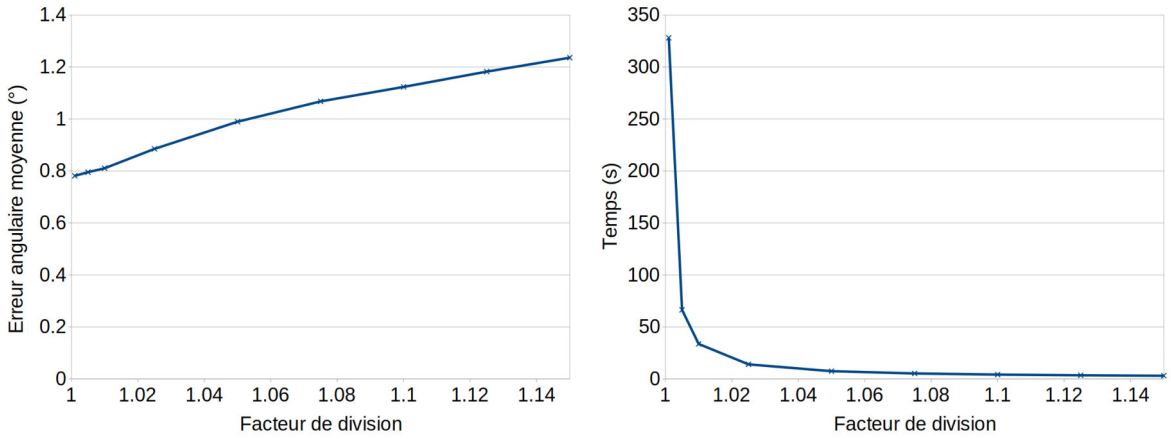


Figure 1.9.: Erreur angulaire moyenne évaluée sur le modèle bicylindrique en fonction de la valeur de X. L'évaluation est réalisée pour 4 niveaux de bruit gaussien (le niveau de bruit est défini comme l'écart type en mm). Les erreurs résultantes des estimations par ACP sont représentées en pointillés.



(a) Erreur angulaire moyenne **(b)** Temps de calcul

Figure 1.10.: Evaluation de l'erreur angulaire et du temps de calcul en fonction du facteur de division, sur le modèle biplanaire. Une moyenne est calculée pour 16 niveaux de bruit gaussien avec des écarts types compris entre 0 % et 400 % de la moyenne des distances entre les plus proches voisins.

1.4.5 Présélection des points

Pour finir, nous présentons le tableau 1.1 pour prouver l'efficacité d'une présélection des points d'entrée pour limiter le nombre d'optimisations à résoudre comme expliqué dans la section 1.3.2. Ce tableau montre qu'il y a peu de différence de précision lorsque tous les points passent par l'optimisation itérative ou lorsque ces points sont présélectionnés. Cependant, le temps de calcul peut être significativement réduit (jusqu'à quatre fois). En utilisant la présélection, le temps de calcul devient directement lié à la géométrie du modèle : l'algorithme devient plus lent lorsque le nombre d'arêtes vives augmente dans un modèle.

Tableau 1.1.: Evaluation de la sélection des points d'entrée à l'optimisation sur les deux modèles synthétiques. Le temps de calcul moyen et l'erreur angulaire moyenne sont évalués pour 16 niveaux de bruit gaussien. L'écart type de ce bruit est compris entre 0 % et 400 % de la distance moyenne entre plus proches voisins pour le modèle biplanaire et de 0 % à 200 % pour le modèle bicylindrique.

		Plans	Cylindres
ACP	Erreur angulaire moyenne (°)	3.35	1.27
	Temps (s)	1	2
Sans présélection	Erreur angulaire moyenne (°)	0.72	0.68
	Temps (s)	53	59
Avec présélection	Erreur angulaire moyenne	0.83	0.71
	Temps (s)	34	14

1.5 Evaluation comparative

1.5.1 Procédure d'évaluation comparative

Pour évaluer notre algorithme, nous effectuons une comparaison avec huit autres méthodes de l'état de l'art : ACP [Hop+92], 2-jets [CP05], VCM [Mér+11], le filtre bilatéral [Hua+13], la méthode de Boulch et Marlet [BM12] (nommée Hough par la suite), son extension par réseau de neurones [BM16] (nommée Hough-CNN par la suite), PCV [Zha+18] et PCP-net [Gue+18]. Ces méthodes représentent les catégories décrites dans la section 1.2 : les algorithmes d'ajustement de surfaces, les méthodes fondées sur une décomposition en cellules de Voronoi, les post-traitements et les algorithmes fondés sur un tirage aléatoire de voisins.

Les méthodes ont été évaluées en utilisant le même nombre de points voisins sauf PCP-net qui est entraîné avec un rayon de voisinage fixe par défaut et VCM qui prend également un rayon en paramètre. ACP et 2-jets ne nécessitent que ce paramètre en entrée. VCM prend deux rayons en entrée. Le rayon d'offset est défini comme cinq fois la distance moyenne entre les points et leurs voisins les plus proches. Le rayon de convolution r est calculé tel qu'il corresponde au nombre de voisins requis dans les autres méthodes. Une approximation est faite : l'aire du voisinage nommée A_{neigh} est liée à l'aire totale de l'objet A_{tot} telle que :

$$A_{neigh} = \pi r^2 = \frac{N_S}{N_{tot}} A_{tot} \,,$$

N_S et N_{tot} étant respectivement le nombre de points du voisinage et le nombre de points total dans le nuage de points. r peut être déduit de cette équation. Le filtre bilatéral est appliqué après une initialisation par ACP. Son seuil angulaire est défini à $25°$ et 5 itérations sont réalisées. Hough est utilisée avec un nombre de paires maximum de $20\,000$ et l'histogramme contient 10 classes. La méthode de regroupement est utilisée pour pallier les problèmes de discrétisation. L'extension

CNN de cette méthode est utilisée avec le réseau pré-entraîné fourni par les auteurs. Nous utilisons les implémentations de la librairie CGAL (Computational Geometry Algorithms Library) pour tester ACP, 2-jets, VCM et le filtre bilatéral. Nous utilisons le code mis en ligne par les auteurs pour tester Hough, son extension par apprentissage automatique et PCP-net ainsi que le code fourni par les auteurs pour tester PCV avec les paramètres par défaut. Notre algorithme est évalué en faisant varier deux paramètres selon l'ensemble de données testé : l'écart type estimé du bruit et le rayon de courbure minimum toléré (voir section 1.3.2). Si rien n'est précisé, la présélection des points est réalisée (voir section 1.3.2). Il est à noter que le filtre bilatéral déplace intrinsèquement le point $\mathbf{p}_0$. Cependant, le débruitage induit n'est pas évalué dans ce travail et tous les points sont représentés à leurs positions initiales. Les outils utilisés pour évaluer les algorithmes sont : l'erreur moyenne angulaire, l'erreur angulaire RMS (Root Mean Square), qui sont des outils statistiques communs et l'erreur angulaire RMS seuillée (RMS$_\tau$) telle qu'introduite par BOULCH et MARLET [BM12] et utilisée dans [Zha+18] avec le même seuil, *c.-à-d.* $10°$. Cette dernière valeur résulte d'un calcul d'erreur RMS dans lequel les angles plus grands que le seuil sont remplacés par $90°$ dans le moyennage. Elle permet de pénaliser l'effet de lissage des algorithmes.

Dans la section suivante, premièrement, nous évaluons notre algorithme sur des modèles synthétiques classiques échantillonnés aléatoirement afin de mettre en évidence les caractéristiques majeures de chaque classe d'algorithmes. Puis, une évaluation est réalisée sur des nuages de points extraits de maillages CAO d'objets contenant des zones à forte courbure, des détails et des arêtes. Un scanner LiDAR est également simulé dans des scènes d'intérieur de bâtiments à partir de maillages CAO. Ce procédé permet d'obtenir un nuage de points proche d'acquisitions LiDAR réelles avec de hauts niveaux d'anisotropie tout en conservant une vérité terrain fiable. Enfin, des données réelles fournies par des scanners LiDAR sont utilisées pour valider visuellement la méthode proposée. Les valeurs des paramètres de notre algorithme fixées pour chaque jeu de données étudié sont regroupées dans le tableau 1.2. Le bruit est estimé directement à partir du type de données étudiées. Dans le cadre d'acquisitions LiDAR, les caractéristiques du capteur peuvent être utilisées. Le rayon de courbure est estimé selon les modèles étudiés. Il permet de faire un compromis entre la précision sur les zones de discontinuité et les zones courbes lisses.

Tableau 1.2.: Paramétrage de notre estimateur de normales sur tous les jeux de données.

	Ecart type estimé du bruit (cm)	Rayon de courbure minimum toléré (cm)
Modèle biplanaire	$[0, 2.0]$	∞
Modèle bicylindrique	$[0, 1.0]$	10.0
Données CAO	$[0, 0.2]$	8.0
LiDAR simulé	0.1	∞
Données LiDAR réelles	1.0	∞

1.5.2 Modèles synthétiques

Premièrement, toutes les catégories de méthodes sont testées sur le modèle biplanaire pour mettre visuellement en évidence les avantages et inconvénients de chaque groupe de méthodes décrit section 1.2. Un bruit gaussien avec un écart type de $0.5\,\mathrm{mm}$ est ajouté aux points. Une partie de ce modèle est extraite et illustrée sur la figure 1.11. Les normales résultantes sont présentées sur la figure 1.12.

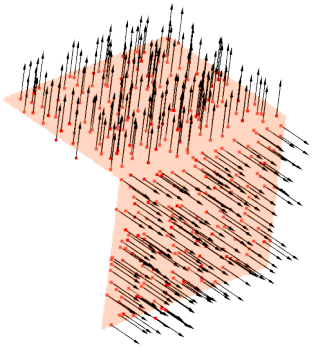


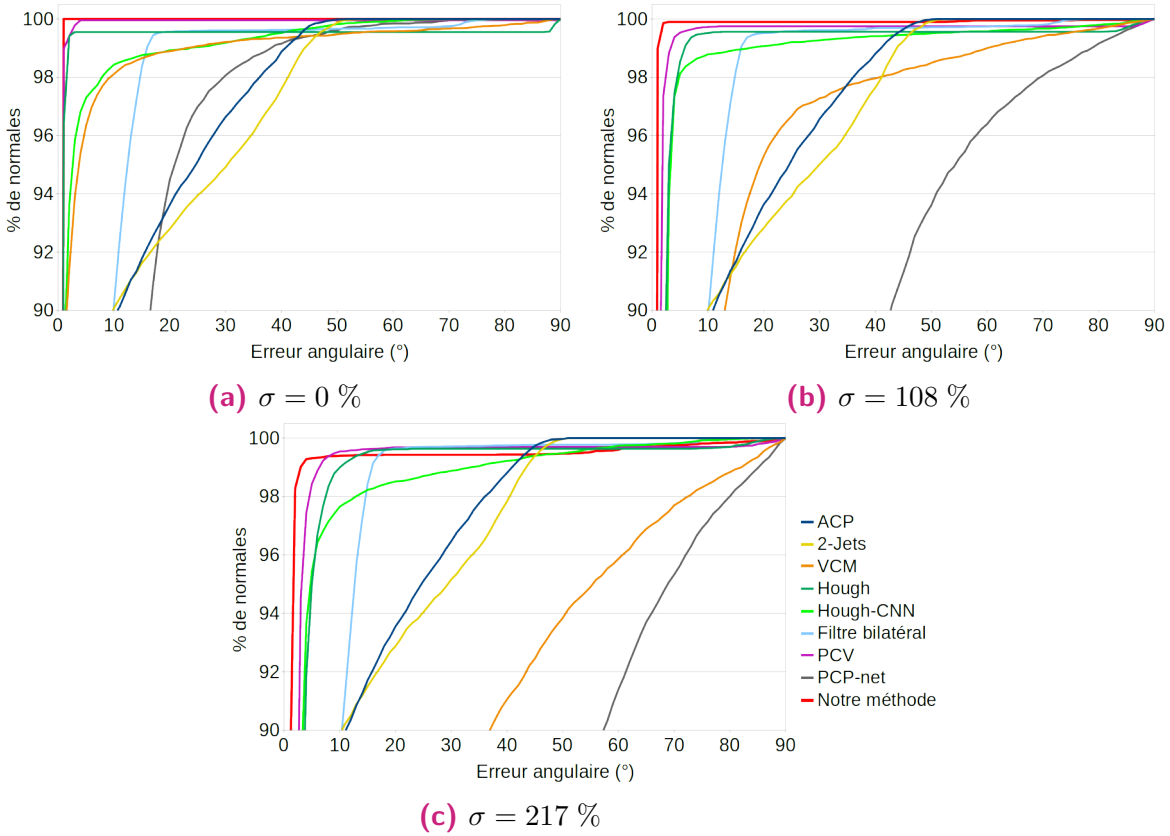
Figure 1.11.: Extrait du modèle biplanaire et illustration des normales vérité terrain. Ce modèle est utilisé pour évaluer les algorithmes d'estimation de normales.

Figure 1.12.: Normales estimées sur le modèle biplanaire. ACP, 2-jets, VCM, le filtre bilatéral, Hough, Hough-CNN, PCV et PCP-net sont comparés avec notre méthode.

Les techniques d'ajustement de surfaces (ACP et 2-jets) lissent l'arête. VCM présente des résultats légèrement meilleurs mais les normales restent éloignées de leurs directions optimales. Les post-traitements tels que le filtre bilatéral testé ici corrigent les normales dont le voisinage contient l'arête mais tendent à diminuer la précision des autres normales. Les méthodes de tirage aléatoire (Hough ici) mettent correctement en évidence les discontinuités de courbure mais certains points ont une normale associée au mauvais plan. Avec le réseau de neurones (Hough-CNN), certaines normales restent lissées près de l'arête. Enfin, notre méthode peut traiter ce type de données de manière fiable.

Ensuite, des histogrammes cumulés et un histogramme simplifié des erreurs angulaires sont représentés sur les figures 1.13 et 1.14, pour le même modèle en ajoutant un bruit gaussien avec un écart type allant jusqu'à $2\,\mathrm{cm}$. Comme tous les algorithmes testés ont un bon comportement sur les zones planes et $84\ \%$ des points ont un voisinage exclusivement sur un plan, la partie intéressante du graphique de la figure 1.13 se situe dans l'intervalle de $90\ \%$ à $100\ \%$. ACP, 2-jets et VCM produisent des erreurs angulaires inférieures à $50°$ mais elles ne traitent pas les zones de discontinuité de courbure : de nombreuses normales ont une faible erreur angulaire ($< 5°$). Hough traite correctement l'arête du modèle mais a un taux important de normales inversées, *c.-à-d.* appartenant à l'autre plan présent dans leur voisinage. Ce phénomène se traduit par un nombre conséquent de points ayant une haute erreur angulaire ($> 80°$). L'extension CNN de cette méthode ne sépare pas les normales aussi clairement que la méthode classique lorsqu'il n'y a pas de bruit, mais elle semble plus robuste à un bruit gaussien limité ($< 200\ \%$ de la distance moyenne entre les points et leurs plus proches voisins). Le filtre bilatéral corrige les normales proches de l'arête et semble robuste au bruit gaussien, cependant, il aboutit à de nombreuses normales ayant une faible erreur angulaire ($< 10°$) même sans bruit. Les résultats de PCV semblent fiables sur ce modèle mais il est moins robuste au bruit que notre algorithme. Ce dernier parvient à un nombre maximum de normales présentant une erreur angulaire inférieure à $5°$ dans tous les cas. Ses performances sont réduites lorsque du bruit est ajouté mais il reste plus précis que les autres algorithmes.

Afin de comparer les algorithmes sur des modèles planaires et courbes, on introduit le modèle bicylindrique (voir section 1.4). Les erreurs angulaires sont représentées pour ce modèle sur la figure 1.15. Cela confirme les observations précédentes et met en évidence que VCM, le filtre bilatéral, Hough et Hough-CNN entraînent un fort taux d'erreur sur les zones lisses. PCV sépare correctement les parties lisses de la surface mais sa précision reste plus faible que celle de notre méthode. Cette dernière présente une bonne précision sur les zones lisses et au voisinage de l'arête. De plus, elle peut traiter un angle d'arête fin entre les deux parties lisses comme on peut l'observer sur la partie latérale de l'objet.



(a) $\sigma = 0\ \%$ **(b)** $\sigma = 108\ \%$

(c) $\sigma = 217\ \%$

Figure 1.13.: Histogrammes cumulés des erreurs angulaires évaluées sur le modèle biplanaire. Trois niveaux de bruit gaussien différents sont testés sur le modèle. Les meilleurs résultats correspondent aux courbes dans la partie haute/gauche. ACP, 2-jets, VCM, le filtre bilatéral, Hough, Hough-CNN, PCV et PCP-net sont comparés avec notre méthode. Le voisinage est défini par 300 points. σ est l'écart type du bruit gaussien évalué en % de la distance moyenne entre les plus proches voisins.

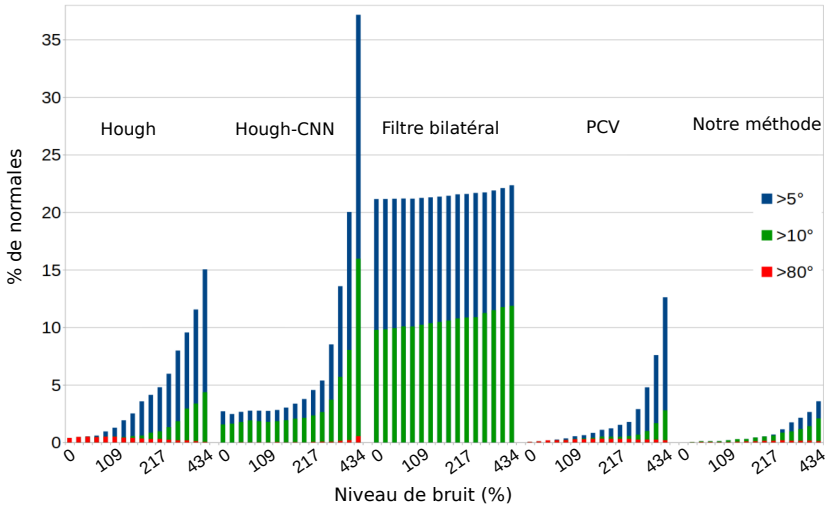
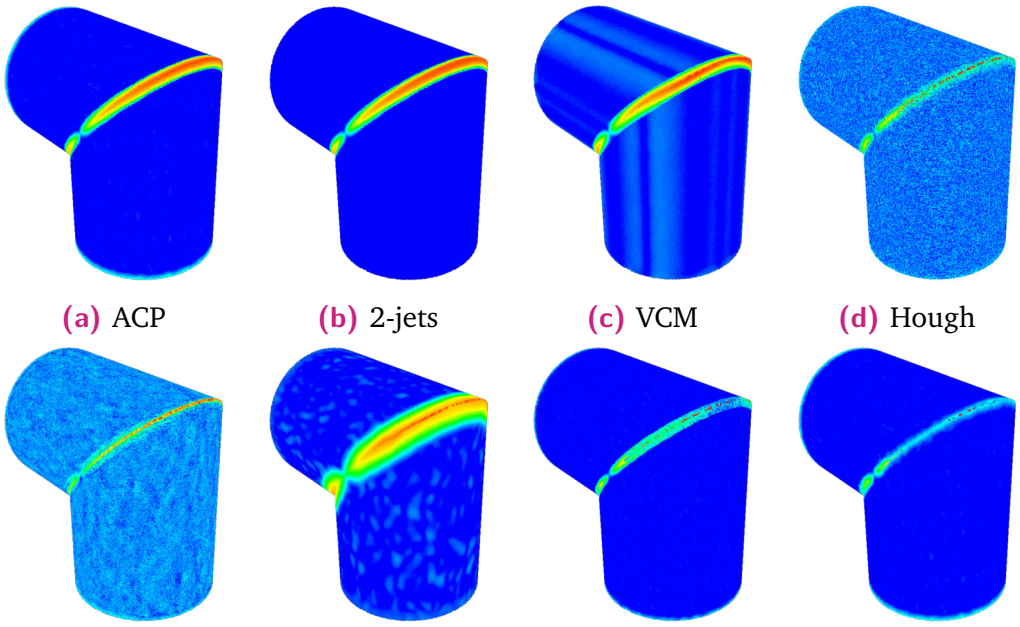


Figure 1.14.: Pourcentage de normales résultantes ayant une erreur angulaire supérieure à trois seuils ($5°$, $10°$, $80°$) sur le modèle biplanaire pour 16 niveaux de bruit gaussien (de 0 % à 400 % de la distance moyenne entre les plus proches voisins). Le filtre bilatéral, Hough, Hough-CNN et PCV sont comparés avec notre méthode.



(a) ACP **(b)** 2-jets **(c)** VCM **(d)** Hough

(e) Hough-CNN **(f)** Filtre bilatéral **(g)** PCV **(h)** Notre méthode

Figure 1.15.: Erreur angulaire évaluée sur le modèle bicylindrique. ACP, 2-jets, VCM, le filtre bilatéral, Hough, Hough-CNN, PCV et notre méthode sont comparés.

Sur les figures 1.16 et 1.17, la robustesse au bruit gaussien est testée sur les deux modèles. Dans les deux cas, l'erreur angulaire moyenne de notre algorithme est plus faible que celle des autres algorithmes pour tous les niveaux de bruit. De plus, la précision de notre algorithme diminue plus lentement que celles de Hough et PCV lorsque le bruit augmente. Hough-CNN a un mauvais comportement pour de hauts

niveaux de bruit et n'atteint pas les performances de Hough dans les cas non-bruités. La précision de PCV diminue plus rapidement avec le niveau de bruit lorsqu'on traite des surfaces courbées que lorsqu'on traite des surfaces planaires.

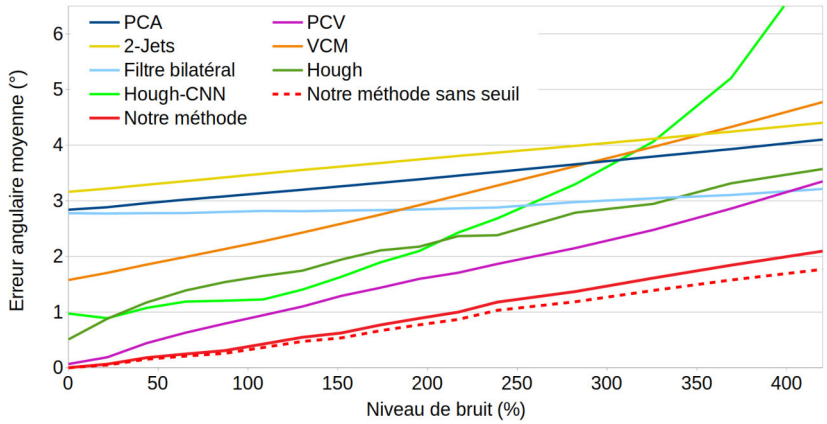


Figure 1.16.: Erreur angulaire moyenne évaluée en fonction du niveau de bruit sur le modèle biplanaire. Le bruit est défini par un pourcentage de l'écart type relativement à la distance moyenne entre plus proches voisins. ACP, 2-jets, VCM, le filtre bilatéral, Hough, Hough-CNN, PCV et notre méthode sont comparés. Notre algorithme est évalué en présélectionnant ou pas les points pour l'optimisation.

A noter que PCP-net n'est pas inclus dans cette partie de l'évaluation car ses performances sur des données synthétiques ne sont pas suffisantes. Le tableau 1.3 met en évidence ce point en réunissant les valeurs d'erreur angulaire moyenne de ACP, de PCP-net et de notre méthode pour les deux modèles synthétiques. La figure 1.18 montre le résultat de cet algorithme sur le modèle bicylindrique. L'échelle est la même que celle utilisée figure 1.15.

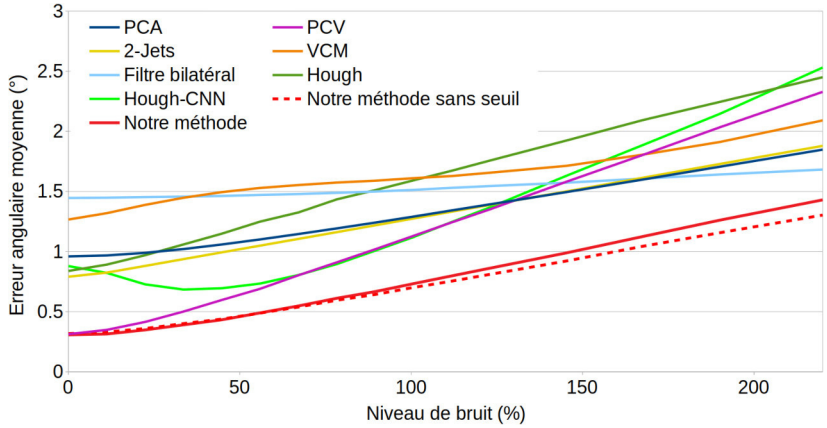


Figure 1.17.: Erreur angulaire moyenne évaluée en fonction du niveau de bruit sur le modèle bicylindrique. Le bruit est défini par un pourcentage de l'écart type relativement à la distance moyenne entre plus proches voisins. ACP, 2-jets, VCM, le filtre bilatéral, Hough, Hough-CNN, PCV et notre méthode sont comparés. Notre algorithme est évalué en présélectionnant ou pas les points pour l'optimisation.

Tableau 1.3.: Erreur angulaire moyenne (en degrés) des normales résultantes de PCP-net sur les modèles synthétiques.

	ACP	Notre méthode	PCP-net
Modèle biplanaire	3.35	0.81	23.86
Modèle bicylindrique	1.27	0.71	3.02

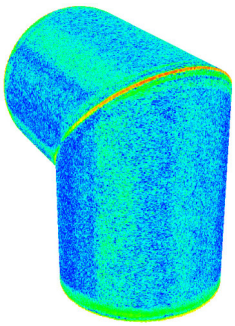


Figure 1.18.: Erreur angulaire des normales résultantes de PCP-net sur le modèle bicylindrique. L'échelle de couleurs est la même que celle utilisée figure 1.15.

1.5.3 Modèles CAO

Ensuite, les méthodes sont évaluées sur un ensemble nommé « CAO » composé de 14 modèles CAO mélangeant plans, zones courbes et incluant des discontinuités dans les courbures. Une vue d'ensemble de ces données est représentée figure 1.19. Chaque maillage est mis à l'échelle afin que la diagonale de la boîte englobante soit unitaire, et sur-échantillonné en divisant les arêtes par deux jusqu'à ce que leurs longueurs soient inférieures à 1 % de cette diagonale. Les centroïdes des faces sont extraits. A la fin du traitement, les objets contiennent entre 54 000 et 493 000 points. Un bruit gaussien est ajouté aux modèles avec un écart type compris entre 0 % et 0.2 % de la longueur de la diagonale avec 9 valeurs espacées régulièrement. Pour chaque modèle CAO, le voisinage est défini par 200 voisins et le paramètre de rayon de courbure minimale de notre algorithme est défini empiriquement à 8 % de la longueur de la diagonale pour tous les objets.

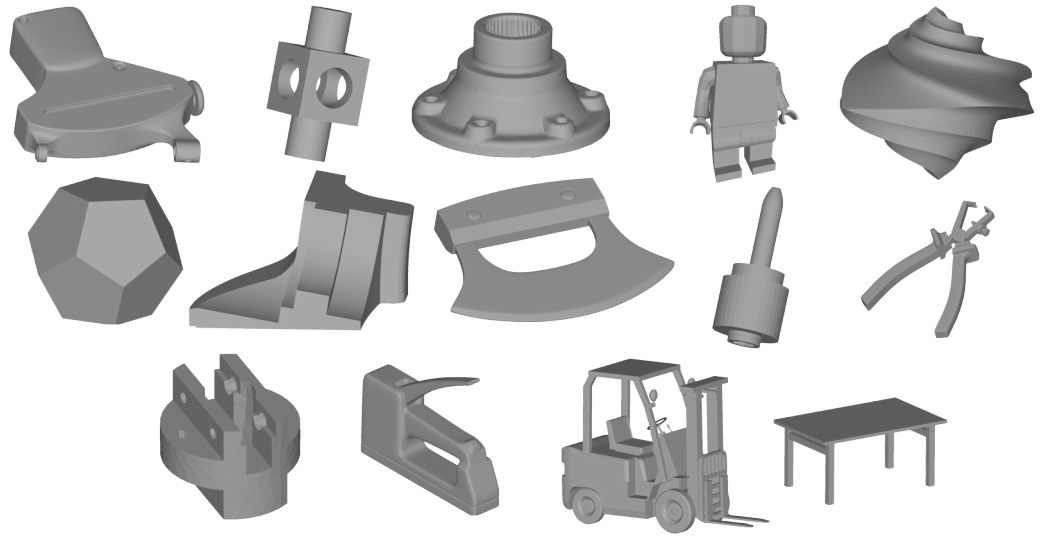
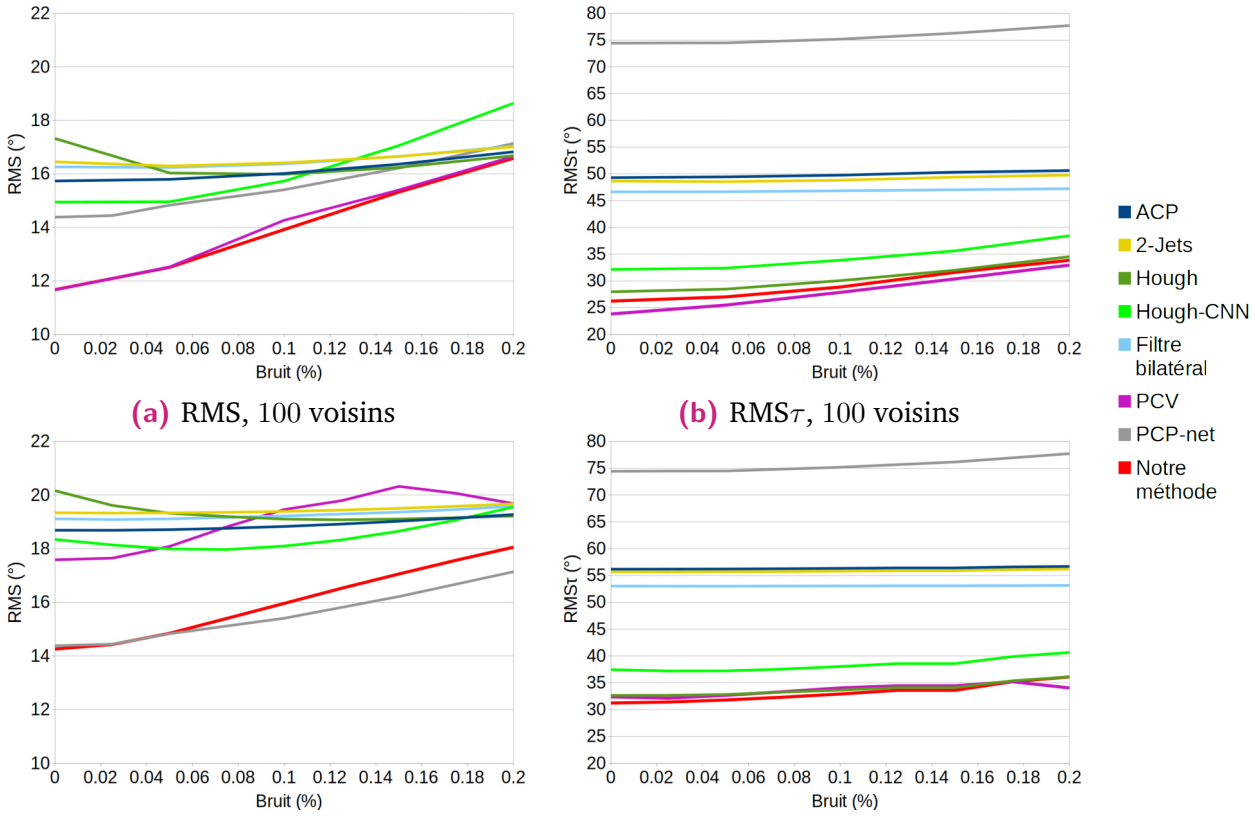


Figure 1.19.: Maillages du jeu de données CAO utilisé pour l'évaluation. Les maillages sont normalisés pour que la diagonale de leurs boîtes englobantes soit 1. Les nuages de points utilisés correspondent aux centroïdes de toutes les faces. Les objets contiennent entre 54 000 et 493 000 points.

Les résultats sont présentés figure 1.20. Les comportements de PCV et de notre algorithme sont les meilleurs lorsque le voisinage est défini par 100 points (RMS et RMSτ). Cependant, lorsque le voisinage est augmenté à 200 voisins, l'erreur RMS de PCV a entre 3° et 4° de plus que notre algorithme. PCP-net qui est calculé pour un rayon de voisinage fixe, présente une meilleure erreur RMS lorsque le nombre de voisins des autres algorithmes est fixé à 200. Cependant, ses performances en termes d'erreur RMS sont moins bonnes lorsque le voisinage est défini par 100 voisins. Son erreur RMSτ est toujours très élevée en comparaison des autres algorithmes, cela met en évidence son manque de précision sur les zones lisses.



(a) RMS, 100 voisins

(b) RMSτ, 100 voisins

(c) RMS, 200 voisins

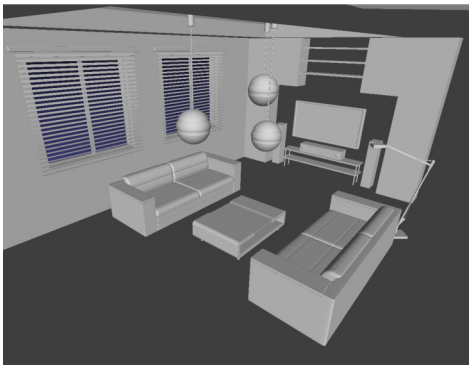
(d) RMSτ, 200 voisins

Figure 1.20.: Erreur angulaire en fonction du niveau de bruit sur un ensemble de 14 objets CAO contenant entre 54 000 et 493 000 points. Le voisinage est défini par 100 points sur la première ligne ((a) et (b)) et 200 points sur la deuxième ligne ((c) et (d)).

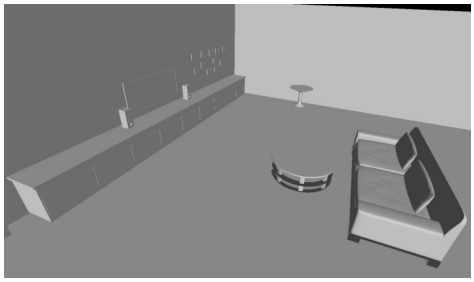
1.5.4 Simulations LiDAR

Pour évaluer notre algorithme dans des conditions proches de la réalité, nous avons simulé un scanner LiDAR dans quatre scènes d'intérieur CAO (voir figure 1.21) extraites du site `free3d.com`. Les nuages de points ont alors de hauts niveaux d'anisotropie qui découlent directement du fonctionnement interne d'un scanner LiDAR comme cela peut être observé sur la figure 1.22a. Nous avons ajouté un bruit gaussien de 0.1 % de la diagonale de la boîte englobante, car c'est un bruit qu'on retrouve communément dans des données réelles. Comme les modèles sont majoritairement planaires, le paramètre de rayon de courbure minimal de notre algorithme est défini comme infini dans tous les cas. Un des nuages de points résultants est illustré sur la figure 1.22b.

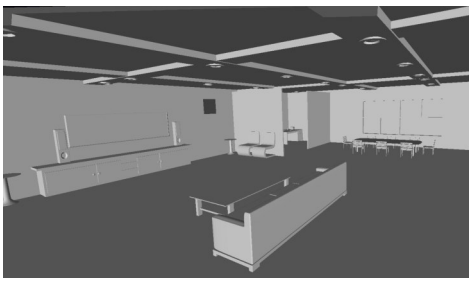
(a) Conference room



(b) Living room 1

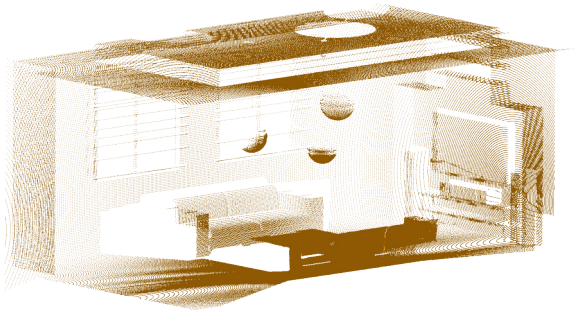


(c) Living room 2

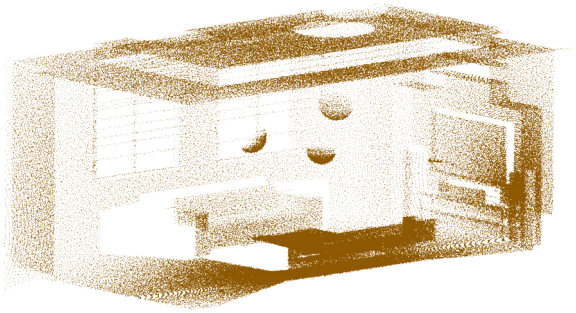


(d) House

Figure 1.21.: Maillages CAO de scènes d'intérieur utilisés pour évaluer les algorithmes. Un scanner LiDAR est simulé à l'intérieur des maillages.



(a)



(b)

Figure 1.22.: Nuage de points obtenu après simulation d'un scanner LiDAR dans une scène d'intérieur représentée par un maillage appelé *Living room 1*. (a) sans bruit et (b) avec bruit gaussien.

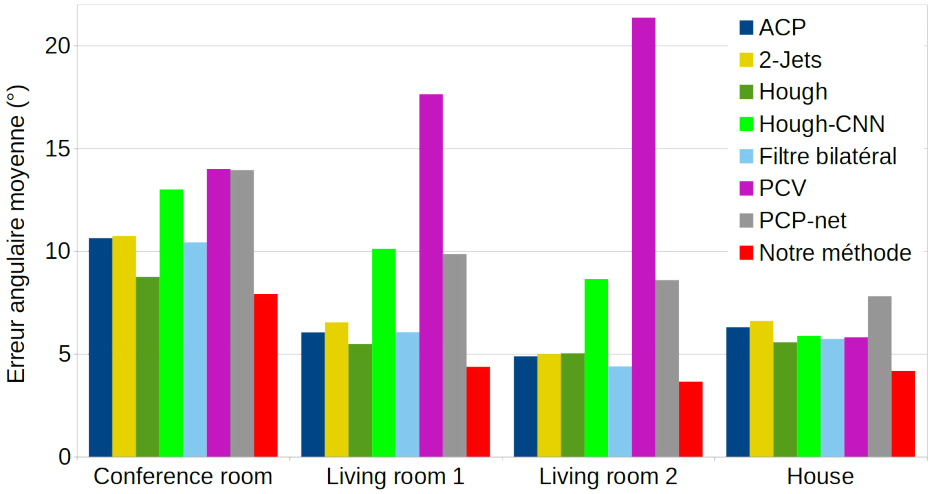
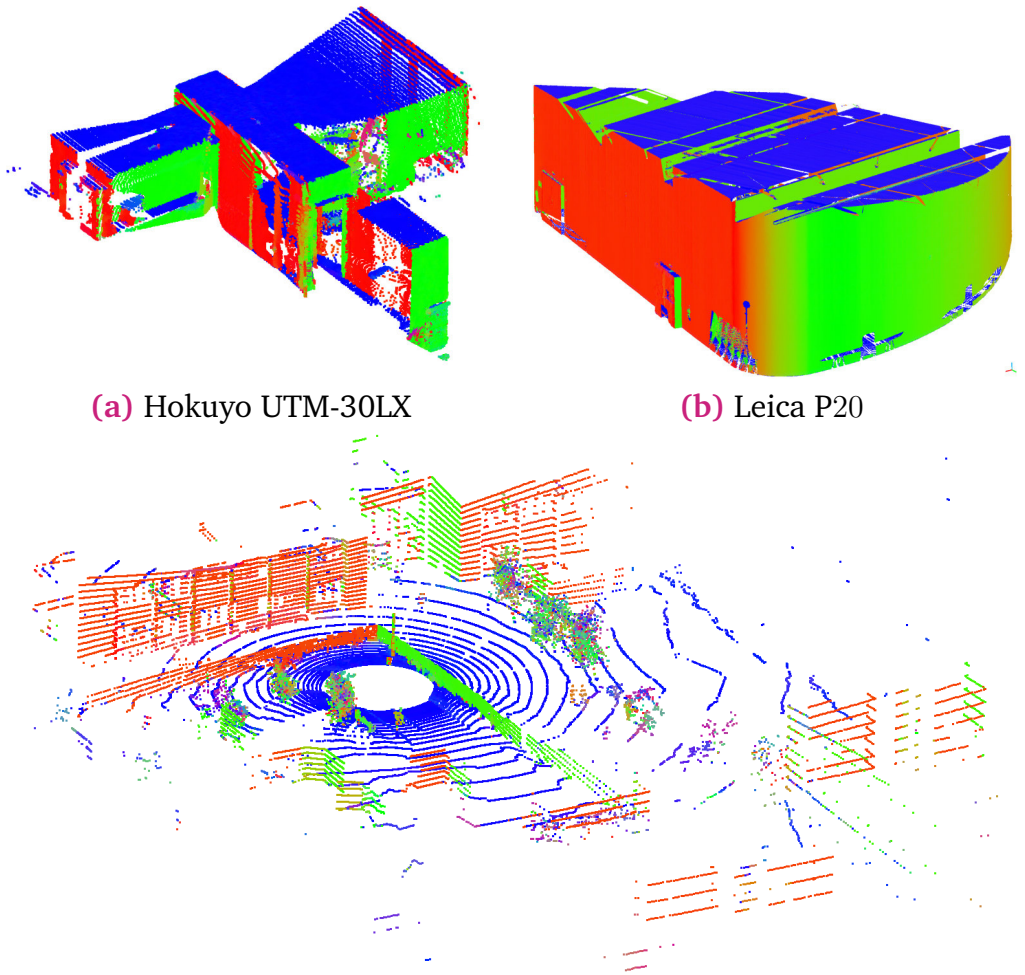


Figure 1.23.: Erreur angulaire moyenne calculée sur les normales estimées dans un nuage de points issu d'une simulation d'un scanner LiDAR dans des scènes d'intérieur représentées sous forme de maillages. Un bruit gaussien est ajouté avec un écart type égal à 0.1 % de la diagonale de la boîte englobante.

1.5.5 Acquisitions LiDAR

Enfin, nous avons testé notre méthode sur des données réelles. Les nuages de points testés sont plus denses, ils présentent différents niveaux de bruit et des artefacts causés par les capteurs utilisés. Le premier nuage de points est extrait du jeu de données « apartment » disponible sur le site de l'ASL (Autonomous System Lab) [Pom+12]. Le second est extrait du jeu de données « Patio », acquis par notre équipe et disponible en ligne [San+17a]. Il a été produit par un scanner Leica P20, qui permet d'obtenir une meilleure précision et une plus haute densité de points. Le troisième est extrait du jeu de données « PAVIN », également acquis par notre équipe et disponible en ligne [San+17a]. Il a été scanné par un capteur Velodyne HDL-32E qui a une faible précision et produit un échantillonnage très anisotrope. Tous les nuages de points ont été acquis dans des environnements structurés. La description des jeux de données (capteurs et environnements) est détaillée dans l'annexe B. 100 points sont utilisés pour définir un voisinage local pour les trois nuages de points, le paramètre de rayon de courbure minimal est défini comme infini et le bruit estimé est paramétré à $1\,\mathrm{cm}$. Les résultats de notre algorithme sont présentés figure 1.24 et montrent qu'il est robuste à des échantillonnages fortement non-uniformes (particulièrement remarquables avec les capteurs Hokuyo et Velodyne) et qu'il est capable de traiter les zones courbes avec une bonne précision. De plus, il peut détecter des plans étroits tels que les bordures de porte comme dans la scène scannée par le dispositif Leica de la figure 1.24.



(a) Hokuyo UTM-30LX **(b)** Leica P20

(c) Velodyne HDL-32E

Figure 1.24.: Normales résultantes de notre estimateur sur trois nuages de points acquis dans des environnements structurés par trois capteurs différents. Les couleurs RGB représentent les trois coordonnées des normales.

1.5.6 Temps de calcul

La moyenne des temps de calcul sur le modèle bicylindrique pour différents niveaux de bruit est présentée dans le tableau 1.4. Les voisinages sont définis par 50, 100, 200, 300 et 400 voisins (processeur : Intel i5–7440HQ, 2.80 GHz, RAM 16Go). Les algorithmes comparés sont exécutés dans un seul thread. Le niveau de bruit gaussien est augmenté jusqu'à 200 % avec 16 valeurs. Il est à noter que les algorithmes qui ne traitent pas les discontinuités de courbure sont toujours plus rapides sur ce type de données. Notre algorithme est plus rapide que la technique Hough. Le filtre bilatéral est plus rapide que notre algorithme lorsque le voisinage est petit mais devient plus lent dès que le nombre de voisins est supérieur à 100. Hough est la seule méthode dont le temps de calcul diminue avec le nombre de voisins. En effet, celui-ci dépend principalement du nombre de paires testées dans le voisinage afin d'atteindre un seuil de certitude.

Tableau 1.4.: Temps de calcul moyens (en secondes) sur le modèle bicylindrique en fonction du nombre de voisins pour 16 niveaux de bruit gaussien compris entre 0 % et 200 % de la distance moyenne entre les plus proches voisins. ACP, 2-jets, VCM, le filtre bilatéral, Hough, Hough-CNN et notre méthode sont comparés.

Méthode	Nombre de voisins				
	50	100	200	300	400
ACP	0.5	1	2	3	4
2-Jets	24	4	8	11	14
VCM	10	10	11	12	12
Hough	111	76	57	56	54
Hough-CNN	36	38	42	40	43
Filtre bilatéral	7	13	27	42	52
Notre méthode	11	14	16	23	29

Sur la figure 1.25, le temps de calcul des algorithmes est représenté en fonction du nombre total de points qui échantillonnent le modèle bicylindrique. Cette figure met en évidence le fait que le temps de calcul de notre algorithme est plus faible et croît plus lentement que celui de Hough ou du filtre bilatéral relativement au nombre de points total. En effet, dans notre cas, le temps de calcul ne dépend que du nombre de points au voisinage d'une discontinuité de courbure.

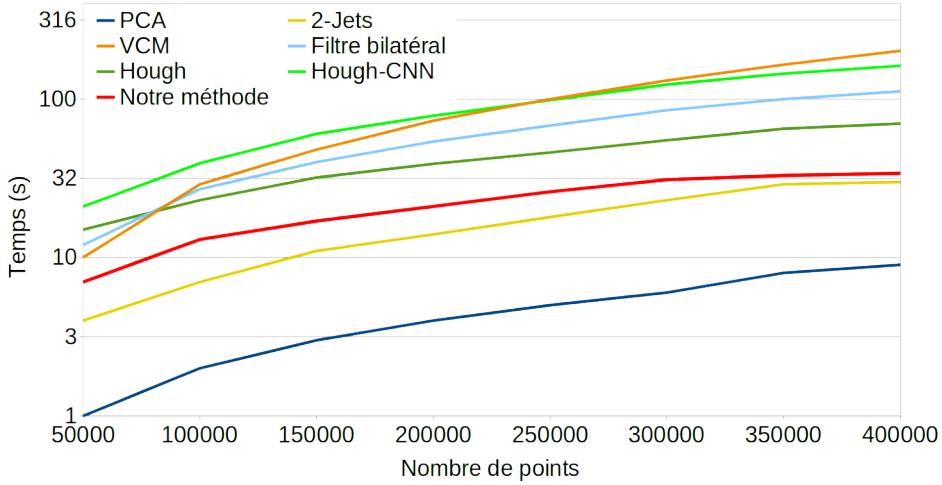


Figure 1.25.: Temps de calcul sur le modèle bicylindrique en fonction du nombre de points. Un bruit gaussien correspondant à 55 % de la distance moyenne des plus proches voisins. ACP, 2-jets, VCM, le filtre bilatéral, Hough, Hough-CNN et notre méthode sont comparés.

Les temps de calcul sont également évalués sur le jeu de données CAO (voir tableau 1.5) avec le niveau de bruit maximum (0.2 %). Dans ce cas, notre algorithme est plus lent que Hough-CNN et le filtre bilatéral à cause de la grande quantité d'arêtes vives présentes.

Tableau 1.5.: Temps de calcul moyen (en secondes) sur 14 objets CAO avec 0.2 % de bruit.

ACP	2-Jets	Hough	Hough-CNN	Filtre bilatéral	Notre méthode
2	9	410	46	32	131

Enfin, les temps de calcul sont évalués sur les acquisitions réelles. Ils sont présentés dans le tableau 1.6. Notre algorithme présente un temps de calcul légèrement plus haut que Hough-CNN pour le premier et le troisième nuage de points. Cependant, il est six fois plus rapide sur le second. Hough est la méthode la plus lente et le filtre bilatéral n'a pas pu être testé sur le second nuage de points pour cause d'une trop grande consommation de mémoire RAM.

Tableau 1.6.: Temps de calcul moyen (en secondes) pour estimer des normales sur trois nuages de points acquis avec différents scanners LiDAR dans des environnements structurés. ACP, 2-jets, VCM, le filtre bilatéral, Hough, Hough-CNN et notre méthode sont comparés avec un voisinage de 100 points. *n.a.* indique qu'aucun résultat n'a été obtenu (excès en mémoire).

Méthode	Capteur LiDAR		
	Hokuyo UTM-30LX	Leica P20	Velodyne HDL-32E
ACP	3	28	1
2-Jets	15	120	2
VCM	43	308	12
Filtre bilatéral	46	*n.a.*	7
Hough	1163	1058	107
Hough-CNN	137	1116	32
Notre méthode	143	180	48

Comme PCV a été implémenté pour Matlab et PCP-net a été implémenté en Python pour être exécuté sur GPU, leurs temps de calcul moyens n'ont pas pu être intégrés à cette étude. Cependant, ils sont lents comparés à notre algorithme. Afin de confirmer ce point, le tableau 1.7 rassemble les temps de calcul maximaux de ces algorithmes en comparaison du notre et de la méthode fondée sur la transformée de Hough qui est la plus lente des méthodes évaluées ci-dessus.

Il est à noter que, comme la majorité des algorithmes testés ici, la méthode proposée peut être entièrement parallélisée car chaque estimation de normale est indépendante de ses voisines.

Tableau 1.7.: Temps de calcul maximum dans différents jeux de données. Hough et notre algorithme sont implémentés en C++ et sont exécutés sur un thread sur CPU. PCV est implémenté sous MATLAB sur un thread et PCP-net est implémenté en Python et exécuté sur le GPU sur plusieurs threads.

	CAO	Simulations LiDAR
Hough	40 min	25 min
Notre méthode	12 min	35 s
PCV	6 h	80 min
PCP-net	65 min	5 min

1.6 Conclusion

Le calcul de normales sur les nuages de points est issu d'une étude locale de voisinage. Actuellement, il est difficile de trouver une méthode qui réunisse tous les atouts, permettant d'être appliquée à des données LiDAR acquises en environnement intérieur. En effet, les scènes d'intérieur de bâtiment peuvent être représentées par des surfaces continues par morceaux, les estimateurs doivent donc maintenir une haute précision sur les zones lisses de la scène tout en résolvant le problème d'estimation au voisinage des discontinuités. De plus, les données présentent une forte anisotropie et un bruit induit par le capteur qui compliquent les traitements locaux. Dans ce chapitre, nous avons proposé une nouvelle approche adaptée à ce type de données dans le but d'obtenir une modélisation fiable. L'estimation proposée utilise une ACP pondérée intégrée à un processus itératif. Ce dernier est divisé en deux phases pour assurer la robustesse au bruit gaussien et une double initialisation automatique est réalisée pour résoudre les problèmes liés à l'anisotropie au voisinage des arêtes vives. Notre algorithme a été évalué sur des données synthétiques, sur des objets issus d'une construction CAO, sur des nuages de points obtenus par simulations LiDAR et sur des données acquises par trois capteurs LiDAR différents. En réponse aux objectifs fixés section 1.2, nous pouvons conclure que l'algorithme proposé répond en partie au premier objectif puisqu'il met bien évidence les discontinuités de courbure. Cependant, étant destinée à l'étude d'environnements intérieurs, notre méthode est, de fait, moins performante pour des nuages de points ne présentant pas les mêmes caractéristiques. Par exemple, certaines méthodes de l'état de l'art permettent de traiter avec plus d'efficacité les détails lisses en hautes fréquences ce qui peut rendre la réponse à l'objectif 1 de conservation de la précision sur des zones lisses plus mitigée. L'objectif 2 est atteint puisque l'estimation ne prend en compte que la position des points voisins. L'algorithme permet de traiter des données bruitées et de faibles densités de points répondant ainsi à l'objectif 3. Enfin, sa rapidité le rend approprié pour estimer les normales de très grands nuages de points tel que requis par l'objectif 4. Dans le prochain chapitre, nous allons utiliser ce nouvel outil pour réaliser une modélisation globale des scènes d'intérieur.

Modélisation globale : reconstruction d'un scan

2

2.1 Problématique

La modélisation de nuages de points peut être réalisée directement par une triangulation et une représentation des surfaces sous forme de maillages. Cependant, pour un environnement d'intérieur de bâtiment fortement structuré, ce type de modèles mène à de nombreuses erreurs géométriques, notamment dans les zones d'interaction entre objets (entre le sol et les murs, ou entre une chaise et le sol par exemple) et ne permet pas de corriger l'information de bruit du capteur [Lai+15]. De plus, bien que le maillage permette une visualisation plus simple pour des personnes non initiées en comparaison au nuage de points, ce modèle n'est pas manipulable par des professionnels du BIM car les objets ne sont pas indépendants les uns des autres. Le problème est donc généralement décomposé en deux phases : un enrichissement sémantique bas niveau du nuage de points est réalisé, autrement dit, le nuage de points est segmenté, puis les modèles sont ajustés sur les segments.

La structure d'un bâtiment est généralement constituée d'un ensemble de surfaces planaires connectées les unes aux autres. Les modèles les plus utilisés en pratique sont des ensembles de polygones ou des polyèdres dans le cas d'environnements fermés ou dits « étanches ». Les informations d'interaction entre polygones sont également extraites dans une majorité de méthodes de reconstruction. On peut donc remarquer que la sémantisation des scènes mesurées est intrinsèquement liée à leur modélisation : plus on a d'informations sémantiques et plus précise sera la modélisation. Néanmoins, une sémantisation de plus haut niveau (détection de bureaux, armoires, etc.) peut aussi enrichir le modèle *a posteriori* pour faciliter le travail des utilisateurs de modèles BIM mais nous ne traiterons pas cet aspect.

Pour segmenter le nuage de points, une majorité de méthodes cherchent à détecter des primitives planaires dans le nuage [Mac+17 ; BM16 ; Och+16]. Pour ce faire, trois algorithmes principaux sont couramment employés : la transformée de Hough, RANSAC et la croissance de régions. Ces méthodes nécessitent souvent un paramétrage complexe afin d'être adaptées à un type de données. Leur comportement est peu robuste à l'anisotropie de l'échantillonnage et à la complexité de la scène. En outre, leur fonctionnement est itératif, *c.-à-d.* que les primitives sont détectées les unes après les autres ce qui peut entraîner une mauvaise détection de leurs contours. Pour pallier ce problème, les auteurs utilisant ce type de méthodes passent par un arrangement de lignes ou de plans, en prolongeant les primitives détectées. Puis, les intersections sont retrouvées grâce à des heuristiques ou par optimisation. Le problème d'un tel procédé est que l'information fournie par le nuage de points peut être fortement extrapolée ce qui peut mener à des écarts élevés avec les données réelles. La problématique majeure de ce chapitre est la suivante :

Problématique

Comment modéliser une scène d'intérieur échantillonnée par des points par acquisition LiDAR en vue d'une utilisation dans le cadre du BIM ?

Les contraintes auxquelles doit faire face l'algorithme recherché sont les suivantes :

Contraintes

— l'acquisition n'est pas nécessairement orientée ;
— les murs peuvent être courbés ;
— Le nuage de points est bruité ;
— La scène est échantillonnée de manière anisotrope ;
— De nombreuses occultations peuvent apparaître ;
— Le temps de calcul doit être limité.

Nous proposons une méthode alternative fondée sur le traitement conjoint d'un nuage de points correspondant à un scan et de son image d'acquisition angulaire relative. Cette méthode repose sur la précision de l'estimation de normales telle que proposée dans le chapitre précédent afin de réaliser une croissance de région 2D permettant d'extraire les primitives planes et leurs contours directement dans l'image d'acquisition angulaire. Une étude d'intersection est ensuite réalisée afin de classifier les interactions entre plans (occultations ou connexions) et les contours sont modélisés par un ensemble de segments. Les ouvertures sont également détectées et l'ensemble des polygones à trous connectés est ensuite représenté sous forme de maillage.

Dans ce chapitre, nous détaillerons tout d'abord les méthodes existantes dans le domaine de la modélisation de scènes d'intérieur. Ensuite, nous proposerons une approche permettant de modéliser un scan. Enfin, nous évaluerons brièvement la méthode sur des données réelles avant de conclure sur les avantages et inconvénients d'une telle modélisation.

2.2 Etat de l'art

La modélisation de bâtiments à partir de nuages de points se décompose généralement en deux tâches distinctes : la segmentation, qui correspond à un étiquetage des points selon l'objet échantillonné, et l'ajustement des modèles géométriques sur les points.

Ces deux tâches sont réalisées par une multitude de méthodes différentes qui varient suivant :

- l'objet reconstruit : l'objectif de la modélisation peut être de reconstruire un bâtiment complet, un étage de bâtiment, une pièce ou un scan ;
- la précision souhaitée ;
- les hypothèses initiales : verticalité des murs et horizontalité du sol et du plafond, planéité des surfaces, parallélisme et/ou perpendicularité entre éléments, etc. ;
- les données initiales : nuage de points, couleur, normales, photographies, scans individuels ou recalés, position du scanner, etc.

Par conséquent, les sections suivantes dressent un inventaire non exhaustif des tâches réalisables dans une chaîne de traitements de reconstruction d'intérieurs de bâtiments à partir d'hypothèses et de données variées.

2.2.1 Décomposition en étages

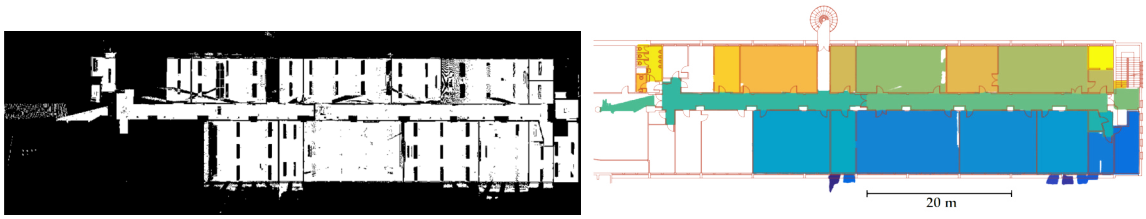
La modélisation d'un bâtiment peut être réalisée à partir d'un nuage de points intégrant tous les scans issus des différentes poses du scanner, après recalage (voir partie II). Dans ce cadre, certaines méthodes proposent de décomposer le nuage global en sous-nuages qui peuvent être segmentés et/ou modélisés indépendamment. Cette décomposition permet donc de sélectionner les modèles adéquats et d'enrichir sémantiquement la reconstruction finale.

Le bâtiment peut notamment être décomposé en étages. Pour ce faire, les méthodes font l'hypothèse que l'axe vertical réel correspond à l'axe z de l'acquisition. Ainsi, un histogramme de points est calculé sur cet axe et les pics de l'histogramme sont associés aux sols et plafonds des différents étages [Hub11 ; Oes+14 ; KDV14 ; Mac+17]. Oesau et coll. [Oes+14] effectuent cette opération sur le nuage complet en filtrant préalablement les points dont la normale est verticale. Une étape de *mean-shift* sur les classes de l'histogramme permet de retrouver précisément l'altitude des sols et plafonds de l'immeuble. Macher et coll. [Mac+17] créent un histogramme pour chaque scan recalé. Les auteurs identifient donc l'altitude du sol et du plafond dans chaque scan et regroupent ces scans en fonction de leurs altitudes. Une étude statistique est menée afin d'exclure les objets présentant des caractéristiques similaires au sol et au plafond (les tables ou les bureaux par exemple), ces groupes correspondent alors aux différents étages. Khoshelham et Díaz-Vilariño [KDV14] s'affranchissent de l'hypothèse de verticalité en ajoutant un traitement préliminaire permettant de réaligner le nuage de points. Pour cela, ils repèrent les directions principales du nuage grâce au champ de normales projeté sur la sphère gaussienne (voir chapitre 3) et alignent ces directions sur le repère $\{x, y, z\}$.

2.2.2 Décomposition en pièces

Les étages peuvent à leur tour être décomposés en pièces [Mac+17]. Pour ce faire, les auteurs extraient une tranche de points horizontale contenant le plafond qu'ils projettent sur le plan $\{x, y\}$. A partir de cette projection, ils créent une image binaire d'occupation (voir figure 2.1a) : chaque pixel peut être vide ou occupé. A cette

altitude, les pièces sont séparées par des murs à l'intérieur desquels aucun point n'a été acquis. Pour s'assurer que les pièces sont bien séparées sur l'image, les pixels contenant des points avec un fort écart type sont considérés comme vides : des murs verticaux sont potentiellement compris dans ces pixels. Une croissance de régions sur l'image d'occupation permet alors d'identifier les pièces (voir figure 2.1b) et d'attribuer une étiquette, correspondant à une pièce, à chaque point.



(a) **(b)**

Figure 2.1.: Segmentation du nuage de points d'un étage de bâtiment par pièces par MACHER et coll. [Mac+17]. Images extraites de leur article. **(a)** Image binaire créée à partir des projections des points situés autour du plafond. **(b)** Résultat de la segmentation par croissance de régions. Le plan de l'étage est superposé (en rouge).

2.2.3 Filtrage d'objets occultants

Plusieurs auteurs choisissent d'éliminer les points échantillonnant des objets occultants qui n'appartiennent pas à l'enveloppe avant segmentation du nuage. Les points ayant une normale horizontale peuvent être filtrés pour isoler les murs [Oes+14] et les points ayant une normale verticale peuvent être filtrés pour isoler le plafond et le sol [SZ12]. MACHER et coll. [Mac+17] réalisent cette opération en deux phases. Dans un premier temps, ils placent une grille sur le nuage de points et créent des voxels d'occupation de l'espace. Ils effectuent ensuite une croissance de région 3D sur ces voxels. Les points de la région la plus large sont alors considérés comme comprenant l'enveloppe et les points des autres régions sont éliminés. Dans un second temps, après détection de la pièce par croissance de région 2D (voir paragraphe précédent), ils filtrent les points correspondant aux pixels de contours des régions. XIONG et coll. [Xio+13] choisissent de réaliser une classification de patchs planaires détectés par croissance de régions 3D (voir section 2.2.4). La classification est réalisée par apprentissage automatique en incluant des informations d'interaction entre les patchs. Les classes sont les murs, le sol, le plafond et les objets.

2.2.4 Détection des éléments de l'enveloppe

Afin d'être modélisés, les éléments de l'enveloppe d'une pièce doivent être détectés, *c.-à-d.* que les points des différents murs, du sol et du plafond doivent être étiquetés. Pour ce faire, la plupart des méthodes cherchent à détecter des plans dans les scènes.

Détection de plans en 3D

Dans une majorité de méthodes de reconstruction de bâtiments, la recherche de plans est réalisée dans le nuage de points, cette recherche est un problème récurrent en reconstruction 3D. Les méthodes classiques de détection de plans sont : la transformée de Hough 3D [DH71], le paradigme RANSAC [Sch+07] et la croissance de régions 3D [BJ88]. Ces méthodes peuvent être adaptées au contexte de la reconstruction de bâtiments tel que nous le verrons dans la suite de cette section. D'autres méthodes peuvent être employées en utilisant des hypothèses fortes sur l'environnement.

Un plan est communément caractérisé par sa normale et sa distance à l'origine $\{\mathbf{n}, d\}$. Un point $\mathbf{p}$ appartenant à ce plan respecte alors l'équation suivante :

$$\mathbf{n} \cdot \mathbf{p} = d\,. \tag{2.1}$$

Le résidu d'un point $\mathbf{p}$ par rapport à ce plan est :

$$r = d - (\mathbf{n} \cdot \mathbf{p})\,. \tag{2.2}$$

Transformée de Hough 3D La normale est paramétrée par deux angles, φ la longitude et θ la colatitude. L'équation (2.1) peut donc être réécrite comme suit :

$$\cos(\varphi)\sin(\theta)\mathbf{p}_x + \sin(\varphi)\sin(\theta)\mathbf{p}_y + \cos(\theta)\mathbf{p}_z = d\,. \tag{2.3}$$

La transformée de Hough 3D classique [DH71] consiste, pour chaque point du nuage étudié, à créer une surface de paramètres dans le repère $\{\varphi, \theta, d\}$ représentant les paramètres des plans potentiels passant par ce point. Les intersections de surfaces correspondent alors aux paramètres des plans détectés dans le nuage. Pour retrouver ces intersections tout en prenant en compte le bruit de mesure, la méthode classique construit un histogramme 3D appelé accumulateur et compte le nombre de surfaces par classe de l'histogramme. La figure 2.2 donne un exemple de surfaces de paramètres. Cette méthode peut être lente et certaines extensions permettent de sélectionner un sous-ensemble de points du nuage par des méthodes probabilistes afin d'en réduire le temps de calcul [Kir+91 ; YJ94]. De plus, elle rencontre des problèmes de discrétisation qui la rendent dépendante de la largeur de classe choisie et de l'orientation de l'histogramme. D'autres formes d'accumulateurs ont été proposées pour pallier ces problèmes [CC09 ; Bor+11]. Toutes ces méthodes sont résumées et évaluées par BORRMANN et coll. [Bor+11].

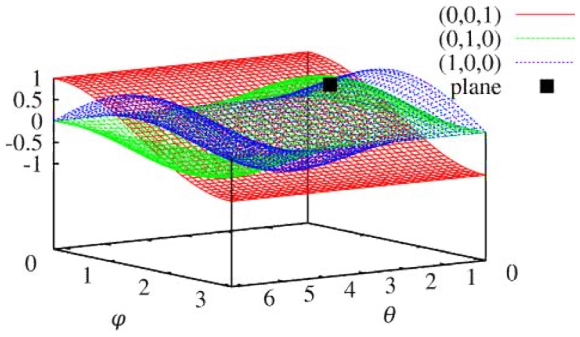


Figure 2.2.: Surfaces de paramètres de plans construites à partir de trois points. Les paramètres du plan commun à ces trois points sont repérés par le carré noir. Image extraite de l'article [Bor+11].

RANSAC Le paradigme RANSAC [Fis+81] consiste à extraire des triplés de points aléatoirement dans le nuage et à calculer un plan pour chaque triplé. Ensuite, Les points appartenant à chaque plan sont comptés, ils sont définis comme ceux dont la distance au plan est inférieure à un seuil. Le plan contenant le plus de points est sélectionné. Les points étiquetés peuvent être retirés de l'ensemble afin de réitérer sur le sous-ensemble restant. Cette méthode est efficace et rapide. Cependant, sa précision dépend du paramétrage du seuil, du critère d'arrêt des itérations et elle peut être compromise par des connexions entre plans et par l'anisotropie de l'échantillonnage. De nombreuses méthodes de reconstruction de bâtiments utilisent cet outil dans leurs chaînes de traitements [Och+16 ; Hon+15 ; Cze+18 ; SZ12 ; TB15 ; BR05]. BRETAR et ROUX proposent d'intégrer l'information des normales locales dans l'algorithme RANSAC dans leur méthode *Normal Driven RANSAC* en ajoutant un critère de ressemblance entre les normales des trois points d'un triplé. THOMSON et BOEHM contraignent directement les modèles proposés par RANSAC pour retrouver des plans horizontaux ou verticaux.

TORR et ZISSERMAN introduisent deux extensions de RANSAC nommées MSAC et MLESAC dans lesquelles le critère de sélection du plan n'est plus le nombre de points pouvant lui être attribués mais dépend des résidus de ces points par rapport au modèle étudié [TZ98 ; TZ00]. MSAC sélectionne le plan qui minimise :

$$\sum_i \rho(r_i^2) \text{ avec } \rho(x) = \begin{cases} x & \text{si } x < T \\ T & \text{si } x \geq T \end{cases} . \tag{2.4}$$

MLESAC sélectionne le plan qui maximise le produit des probabilités $P(r_i)$:

$$\prod_i P(r_i) \text{ avec } P(x) = \frac{\gamma}{\sqrt{(2 * \pi)\sigma}} e^{\frac{-x^2}{2\sigma^2}} + (1 - \gamma)\frac{1}{\nu}, \tag{2.5}$$

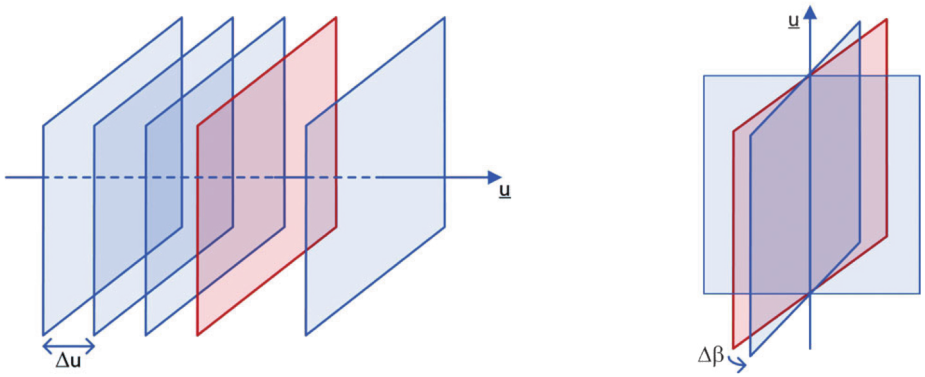
où le premier terme correspond à une probabilité gaussienne d'écart type σ et le deuxième terme correspond à une probabilité uniforme paramétrée par ν. γ est déterminé automatiquement pour chaque plan. Cette procédure est préférée par MACHER et coll. [Mac+17] notamment. Ces formulations permettent de sélectionner les plans les mieux ajustés et pas ceux contenant le plus de points en premier, tout en prenant en compte les points bruités ou appartenant à d'autres objets qui peuvent apparaître au voisinage de chaque hypothèse de plan. Cette méthode a prouvé son efficacité et est majoritairement employée dans les chaînes de traitements de reconstruction. Cependant, sa précision reste dépendante du type de données et de l'anisotropie d'échantillonnage de la scène.

Combinaison Hough/RANSAC TARSHA-KURDI et coll. [TK+07] proposent une comparaison des algorithmes Hough3D et RANSAC pour la détection de plans et affirment que RANSAC a une meilleure gestion du bruit et une plus grande efficacité. Une combinaison des méthodes RANSAC et Hough est proposée par XU et coll. [Xu+90] dans leur méthode nommée transformée de Hough aléatoire (*Randomized Hough Transform* - RHT). Ces derniers utilisent des triplés de points extraits aléatoirement

dans le nuage et déterminent un plan pour chaque triplé comme dans la méthode RANSAC. Cependant, ils intègrent ces paramètres dans un accumulateur identique à celui de l'approche Hough classique et sélectionnent la classe de l'accumulateur contenant le plus de points.

Croissance de régions 3D La croissance de régions sur une image est un procédé qui consiste à partir d'un pixel appelé « graine » ou « germe » et à faire grandir une région de proche en proche en utilisant un critère relatif à la graine sélectionnée. Une fois cette région détectée, on l'extrait de l'image et on recommence le processus à partir d'une nouvelle graine. Une croissance de région peut être réalisée sur les images de profondeur [BJ88 ; Fit+97]. Dans le cadre de la reconnaissance de formes, cette méthode est couramment employée pour détecter des plans [PV06 ; Pop+08 ; Bou+14]. Une limitation de cette méthode est la difficulté de son paramétrage (sélection des graines et contraintes de croissance) afin de limiter le chevauchement d'une région sur une autre tout en prenant en compte l'effet du bruit de mesure et d'estimation des normales. La croissance de régions 2D a été étendue aux nuages de points 3D dans le contexte de la reconstruction de bâtiments en définissant un voisinage local à l'aide d'un rayon ou d'un nombre de points [Cha+10 ; Xio+13]. DESCHAUD et GOULETTE [DG10] préfèrent utiliser une voxelisation de l'espace pour agrandir les régions afin d'optimiser le processus. Cependant, le voisinage dans un nuage de points est difficile à définir, principalement à cause de la non-uniformité de l'échantillonnage. Un voisinage peut ainsi être représenté par une ligne d'acquisition du scanner et perdre l'information de planéité. Le critère généralement employé pour agrandir la région est la distance des points voisins au plan local défini sur la graine [PV06 ; Pop+08] mais la ressemblance des normales peut également être prise en compte [DG10 ; PY09 ; Bou+14]. Des seuils sur ces valeurs doivent alors être paramétrés selon le contexte d'acquisition.

Hypothèses du monde de Manhattan Certains auteurs fondent leur approche sur les hypothèses du monde de Manhattan [CY99]. Sous ces hypothèses, un bâtiment est constitué de plans orientés suivant trois directions principales orthogonales. BUDRONI et BOEHM [BB10] ajoutent l'hypothèse que le sol et le plafond sont des plans parallèles au plan $\{x, y\}$, ils procèdent alors par balayage. Ils font tout d'abord un balayage du plan $\{x, y\}$ par translation suivant l'axe z pour retrouver le sol et le plafond puis un balayage suivant la direction des murs principaux et un dernier balayage suivant une direction orthogonale pour retrouver les murs secondaires (voir figure 2.3a). La direction des murs principaux est déterminée en réalisant un balayage de plans verticaux par rotation autour d'axes perpendiculaires au sol (voir figure 2.3b). Pour ce faire, les auteurs construisent chaque axe vertical sur un point sélectionné aléatoirement dans le nuage duquel les points du sol et du plafond ont été retirés. Un histogramme est créé dans lequel, pour chaque orientation de plan suivant tous les axes, les points appartenant au plan sont comptés. La classe de l'histogramme contenant le plus de points correspond à l'orientation des plans principaux de la scène.



(a) (b)

Figure 2.3.: Balayage de plans utilisé par BUDRONI et BOEHM pour retrouver la direction des murs principaux d'une pièce. Image extraite de leur article [BB10]. **(a)** Balayage de plans par translation. Cette opération est réalisée suivant l'axe z pour retrouver le sol et le plafond puis suivant la direction principale et sa direction orthogonale pour détecter les murs. **(b)** Balayage par rotation : un histogramme d'orientation est créé à partir de ce balayage afin de détecter la direction principale.

VANEGAS et coll. [Van+12] supposent que le bâtiment est orienté sur les axes $\{x, y, z\}$ du repère. Ils cherchent alors à étiqueter le voisinage de chaque point relativement à des modèles locaux définis selon ce repère : des plans, des arêtes, des coins et des coins concaves (les angles des coins et des arêtes sont fixés à $90°$). Les points sont alors regroupés selon leurs étiquettes et leur proximité, puis, chaque groupe est triangulé. Un lancer de rayon permet ensuite de séparer les espaces intérieur et extérieur afin de déduire le volume adapté aux maillages tout en complétant les zones non mesurées. Bien que l'efficacité de ces méthodes ait été prouvée, les hypothèses de construction sur lesquelles elles sont fondées réduisent leur applicabilité sur des données réelles.

Détection de plans en 2D

Si on cherche les plans correspondant aux murs, on peut supposer que les murs sont perpendiculaires avec le plan du sol. Le problème 3D peut alors se transformer en recherche de droites 2D après projection des points sur ce plan. OESAU et coll. [Oes+14] ajustent des droites locales en chaque point projeté. Afin de pouvoir modéliser les zones d'intersection, ils considèrent deux hypothèses : un modèle d'une droite et un modèle de deux droites sécantes. Les auteurs ajustent le meilleur modèle par RANSAC sur le voisinage le plus large possible de chaque point. OESAU et coll. corrigent ces droites locales par un filtre bilatéral et les regroupent par une croissance de régions. Les groupes trouvés sont à nouveau corrigés par une étape de classification en construisant un accumulateur. Cette méthode a l'avantage de pouvoir détecter des plans étroits et de reconstruire des murs courbes sous forme d'un ensemble de plans verticaux fins. MACHER et coll. [Mac+17] réalisent également une détection de droites 2D sur les projections en utilisant MLESAC 2D pour guider une détection postérieure de murs par MLESAC 3D.

2.2.5 Modélisation

Modélisation polygonale

Les éléments de l'enveloppe peuvent être modélisés de manière indépendante sous forme de polygones plans. Pour ce faire, les caractéristiques de chaque plan sous-jacent sont calculées et le contour du polygone est détecté et modélisé.

Certains auteurs choisissent de modéliser directement les points des différentes primitives planaires par le modèle produit par l'algorithme RANSAC [TB15 ; SZ12]. Afin de définir le contour des points des primitives, THOMSON et BOEHM [TB15] utilisent l'enveloppe convexe (*convex hull* voir figure 2.4a), tandis que SANCHEZ et ZAKHOR [SZ12] préfèrent l'α-*shape*. Ces deux modèles forment des polygones de contour construits à partir d'un sous-ensemble de points du plan (voir figure 2.4). Dans le cas de l'α-*shape*, ce sous-ensemble est nommé « complexe α ». L'α-*shape* d'un nuage de points est une enveloppe non-convexe plus proche des données que l'enveloppe convexe (voir figure 2.4b). Ce modèle assure que n'importe quel cercle de rayon $\sqrt{\alpha}$ (α étant le paramètre de l'algorithme) passant par deux points voisins dans le complexe α est vide. Les problèmes majeurs de cette méthode pour une application LiDAR sont sa sensibilité à l'anisotropie de l'échantillonnage et sa difficulté de paramétrage (cf. [Bou+14]). CHAUVE et coll. [Cha+10] préfèrent employer cette méthode afin d'obtenir le contour approximatif de la primitive dans le but d'alléger une future détection plus précise sur un arrangement de plans (voir paragraphe suivant).

(a) Enveloppe convexe **(b)** Complexe α

Figure 2.4.: Détection de contour à partir d'un nuage de points 2D par deux méthodes différentes. Images extraites de la page internet `en.wikipedia.org/wiki/Alpha_shape`.

Avec le même objectif, BOULCH et coll. [Bou+14] préfèrent travailler directement sur l'image d'acquisition angulaire (définie plus en détail dans la suite de ce chapitre) et détectent le contour sous forme de pixels. Ils simplifient ensuite ce contour par fusion des arêtes successives selon un critère de distance.

MACHER et coll. [Mac+17] utilisent la projection des points de chaque mur en 2D sur le plan $\{x, y\}$. Pour chaque segment représentant un mur, ils détectent les deux extrémités qui correspondent aux arêtes verticales des polygones. Les autres arêtes

sont modélisées par l'intersection avec le sol et le plafond qui sont, par hypothèse, des plans horizontaux. Toutes ces méthodes ne permettent pas de mettre en évidence les interactions entre plans et mènent à des modélisations polygonales indépendantes pour chaque plan.

Modélisation polyhédrique

Une majorité de méthodes cherchent à reconstruire directement un polyèdre en repérant ses faces sur un arrangement de droites ou de plans. La méthode classique décrite par BUDRONI et BOEHM [BB10] consiste à projeter en 2D les plans des murs sous forme de segments puis à extrapoler les segments jusqu'à intersecter la boîte englobante (voir figure 2.5). Ensuite, les points du nuage sur le plan $\{x, y\}$ sont également projetés et les cellules 2D sont étiquetées selon leur occupation par des points ou non. Les murs correspondent aux segments séparant une cellule vide d'une cellule pleine. Les auteurs modélisent ensuite les murs en extrudant ces segments suivant l'axe z. Cette méthode est sensible aux occultations et aux points aberrants et peut mener à de mauvaises reconstructions.

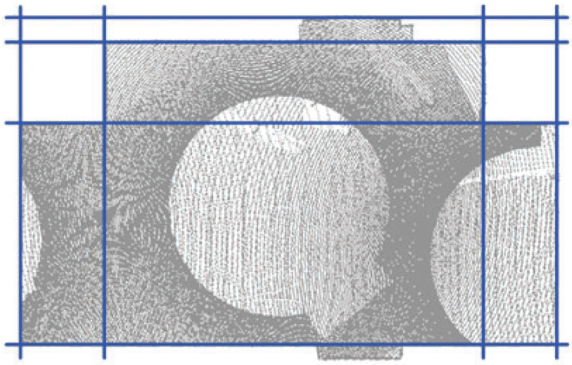
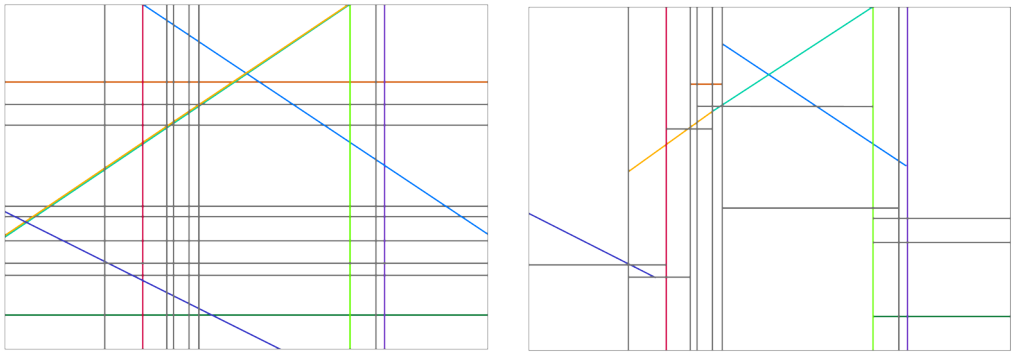


Figure 2.5.: Arrangement de droites (bleu) créé à partir de plans projetés en 2D. Image extraite de l'article [BB10].

OCHMANN et coll. [Och+16] étudient un étage complet. Dans ce cadre, ils construisent plusieurs polyèdres. A partir de l'étiquetage par scan initial donné par l'acquisition, les points sont réétiquetés par salle. Les auteurs détectent ensuite les murs entre deux salles à partir de plans proches parallèles issus d'une première détection par RANSAC. Ils construisent un arrangement 2D en remplaçant les segments de ces plans par un segment intermédiaire contenant les points des deux surfaces du mur. Les cellules sont ensuite étiquetées par salle en optimisant une fonction de coût faisant intervenir deux termes : le premier terme assure que l'étiquette de chaque cellule correspond à l'étiquette des points à l'intérieur ; le second terme assure que les étiquettes de deux cellules séparées par le segment d'un mur correspondent aux étiquettes des points appartenant à ce mur. Comme précédemment, les murs sont modélisés en extrudant les segments qui séparent deux cellules d'étiquettes différentes.

D'autres auteurs travaillent sur des arrangements de plans en 3D. Cet arrangement peut être construit par extrusion des arrangements 2D de chaque étage sur toute la hauteur du bâtiment [Oes+14] ou par extrapolation de primitives planaires en

3D [Cha+10 ; Bou+14]. Le but est toujours d'étiqueter les cellules 3D ainsi créées comme vides ou pleines afin de définir la surface du bâtiment. Les auteurs cités précédemment proposent de réaliser un étiquetage par optimisation d'une fonction de coût composée de deux termes principaux : un terme d'attache aux données mettant en jeu le nuage de points et un terme de régularisation qui limite la complexité de la surface. OESAU et coll. cherchent à construire plusieurs polyèdres et définissent le terme d'attache aux données en traçant des rayons depuis chaque centre de cellule et en comptant le nombre d'intersections de ces rayons avec des faces du polyèdre (une cellule est étiquetée comme intérieure si une majorité de rayons a un nombre d'intersections impair et extérieure si le nombre d'intersections est pair). Dans les méthodes de CHAUVE et coll., et BOULCH et coll., le terme d'attache aux données encourage les étiquettes de cellules adjacentes à être identiques si la face qui les sépare est sur la trajectoire de rayons lancés par le LiDAR (*c.-à-d.* sur la trajectoire point/capteur). BOULCH et coll. ajoutent un terme qui encourage les étiquettes de cellules adjacentes à être identiques si la face les séparant ne contient pas de point. Les termes de régularisation peuvent tendre à minimiser l'aire de la surface [Cha+10], la longueur des arêtes et le nombre de coins du polyèdre [Bou+14]. De plus, CHAUVE et coll. et BOULCH et coll. améliorent l'implémentation classique de l'arrangement de plans en insérant les primitives successivement. Les primitives insérées ne sont pas extrapolées jusqu'aux limites de la boîte englobante mais les auteurs se servent de leurs contours détectés au préalable, grâce à l'α-*shape* et sur l'image d'acquisition angulaire respectivement. Elles sont alors extrapolées à partir de leurs contours jusqu'à la prochaine primitive rencontrée. Un exemple 2D d'un tel arrangement est donné figure 2.6. La modélisation finale dépend alors de l'ordre d'insertion des plans.



(a) **(b)**

Figure 2.6.: Comparaison des arrangements de plans en insérant tous les plans en une fois **(a)** ou avec insertions successives **(b)**. Images extraites de l'article [Cha+10].

Ces méthodes permettent d'obtenir une reconstruction étanche, crédible et facilement manipulable dans une application BIM. Cependant, elles sont sensibles à l'anisotropie de l'échantillonnage et les auteurs recourent à des mécanismes de pondération pour pallier ce problème [Oes+14 ; Och+16 ; Bou+14]. De plus, la reconstruction est approximative, les détails sont souvent éliminés et de nombreux artefacts peuvent intervenir dans la reconstruction (oubli d'un mur, copies, fausses intersections).

Les méthodes listées précédemment utilisent un grand nombre d'hypothèses de construction et d'extrapolations à partir des données initiales ce qui peut mener à de forts écarts avec le nuage de points initial. De plus, les méthodes plus proches des données ne cherchent pas à connecter les objets les uns aux autres. Nous souhaitons proposer une méthode de modélisation de scènes d'intérieur plus proche des données et avec un enrichissement sémantique des zones d'occultation et des ouvertures détectées. Cette modélisation peut alléger les données de nuages de points sans perdre les informations principales contenues dans ces dernières. En outre, une application de cette nouvelle reconstruction pourrait être d'orienter un scanner automatique pendant la prise de points en identifiant les nouvelles positions pouvant maximiser la visibilité des zones occultées. Au vu des lacunes des méthodes listées ci-dessus, nous avons choisi de développer une nouvelle approche répondant aux objectifs suivants :

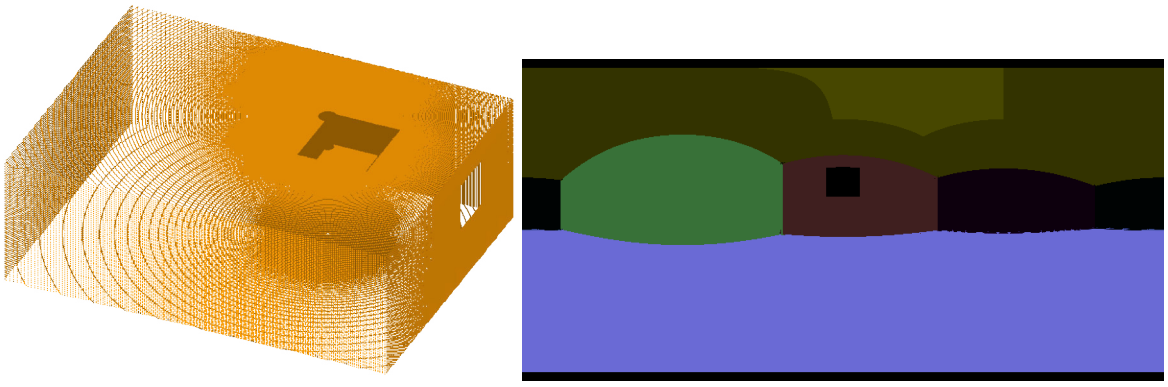
Objectifs

1. Implémenter une reconstruction de scènes d'intérieur en utilisant un minimum d'*a priori* géométriques.
2. Pallier les problèmes liés à l'anisotropie de l'échantillonnage 3D dans la segmentation de primitives.
3. Limiter l'extrapolation des données.
4. Retrouver les informations d'interactions entre entités géométriques (*c.-à-d.* occultations, connexions).
5. Modéliser les ouvertures dans la reconstruction.

2.3 Méthode de modélisation de scènes d'intérieur

Notre objectif est d'obtenir un ensemble de polygones permettant de représenter la scène sans extrapoler d'information à partir du nuage de points. La méthode proposée repose principalement sur un traitement conjoint de l'image d'acquisition angulaire et du nuage de points 3D relatifs à l'acquisition LiDAR. En effet, comme présenté en introduction de cette thèse, le dispositif LiDAR acquiert des mesures de distance pour une liste de rayons correspondants à des angles d'orientation φ et θ. L'objectif est donc de tirer parti de la régularité de l'échantillonnage dans le repère 2D (φ, θ) pour améliorer la segmentation du nuage de points dans le repère 3D. Le domaine 2D régulier permet une représentation de la distance mesurée ou de la distance du plan local sous forme d'images ; φ étant associé aux lignes et θ étant associé aux colonnes. Nous appelons celles-ci, images $\{\varphi, \theta\}$ par la suite. Chaque point 3D de l'ensemble $\mathcal{S} = \{\mathbf{p}_i\}_{i=\{1,\dots,N_S\}}$ est associé à un pixel de l'ensemble $Q = \{q_i\}_{i=\{1,\dots,N_S\}}$. La figure 2.7 présente un exemple d'image $\{\varphi, \theta\}$, dans laquelle on a choisi de représenter le plan local en chaque pixel : les éléments de couleur $\{R, G, B\}$ expriment les valeurs $\{d_{\mathbf{p}_i}\mathbf{n}^x_{\mathbf{p}_i}, d_q\mathbf{n}^y_{\mathbf{p}_i}, d_{\mathbf{p}_i}\mathbf{n}^z_{\mathbf{p}_i}\}$ où $\mathbf{n}_{\mathbf{p}_i}$ et $d_{\mathbf{p}_i}$ sont respectivement la normale et la distance

à l'origine du plan local ajusté en $\mathbf{p}_i$. Cette représentation permet d'identifier les polygones plans comme des groupes de pixels de couleurs similaires.



(a) (b)

Figure 2.7.: Acquisition d'une pièce par dispositif LiDAR. **(a)** Nuage de points. **(b)** Image $\{\varphi, \theta\}$ avec représentation en chaque pixel du plan local. Les polygones peuvent être identifiés comme des ensembles de pixels voisins de couleurs similaires. L'utilisation de cette image permet de pallier les problèmes dus à l'anisotropie en 3D mise en évidence sur l'image de gauche.

Avant de modéliser les objets, nous réalisons une segmentation de l'espace à partir de l'image $\{\varphi, \theta\}$. En effet, nous allons détecter les groupes de pixels et les points 3D associés appartenant à des murs, puis ceux appartenant aux lignes de contour de ces murs. La modélisation des polygones sur ces paires pixel/point 3D est ensuite réalisée en deux temps. Premièrement, des segments de droites sont ajustés sur les lignes de contour et deuxièmement, les coins des polygones sont déduits d'une analyse des intersections de ces segments. Tous les paramètres de la méthode définis ci-après sont résumés à la fin de cette section, dans le tableau 2.1 avec leurs valeurs utilisées dans tous nos tests.

2.3.1 Segmentation de l'image $\{\varphi, \theta\}$ par primitives planaires

Vue générale de l'algorithme

Définition 3.1

On appelle l'ensemble des primitives planaires $\Pi = \{\pi_k\}_{k=\{1,\dots,N_\Pi\}}$. Une primitive π_k modélisant un sous-ensemble de points du nuage $\mathcal{S}$ est définie par :

- $P^{\pi_k} = \{\mathbf{p}_i^{\pi_k}\}_{i=\{1,\dots,N_{\pi_k}\}}$: l'ensemble des points 3D modélisés ;
- $Q^{\pi_k} = \{q_i^{\pi_k}\}_{i=\{1,\dots,N_{\pi_k}\}}$: l'ensemble des pixels modélisés ;
- $\mathbf{n}^{\pi_k}$: sa normale ;
- d^{π_k} : sa distance à l'origine.

Comme expliqué dans la section 2.2, la croissance de régions 3D est couramment réalisée à partir d'un voisinage de points et dépend donc de l'anisotropie de l'échantillonnage. C'est pourquoi, à l'instar de BOULCH et coll. [Bou+14], nous choisissons de réaliser une croissance de régions 2D sur l'image $\{\phi, \theta\}$ avec un critère de croissance 3D. Les régions contenues dans l'ensemble $R = \{r_k\}_{k=\{1,\dots,N_R\}}$ sont alors des ensembles de pixels. Un problème relatif à l'utilisation de cette méthode est que, selon l'erreur résiduelle limite choisie, une région peut chevaucher partiellement une autre région à la frontière avec cette dernière. Un autre problème est qu'une région peut se propager sur une surface d'orientation différente si des points se retrouvent alignés avec les points de la région étudiée (voir la région jaune figure 2.8). BOULCH et coll. [Bou+14] proposent d'ajouter un nouveau seuil sur la ressemblance des normales locales des points avec la graine pour pallier ce problème mais cela peut compliquer le paramétrage de l'algorithme. Nous utilisons une variante à la croissance de régions qui autorise un chevauchement de certaines régions entre elles afin de résoudre ces ambiguités. Les régions sont corrigées par la suite. L'algorithme est détaillé dans les paragraphes suivants.

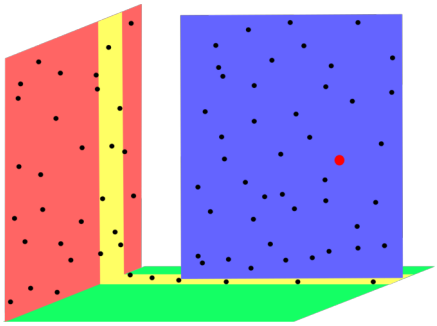


Figure 2.8.: Exemple de configuration de primitives menant à une mauvaise propagation lors d'une croissance de région à partir de la graine rouge (bande jaune).

Sélection des graines

Définition 3.2

L'ensemble des graines sélectionnées est appelé $\mathcal{G} = \{g^{r_k}\}_{k=1,\dots,N_R}$. Chaque graine g^{r_k} est définie par :

- $\mathbf{p}^{g^{r_k}}$: son point 3D ;
- $q^{g^{r_k}}$: son pixel ;
- $\pi^{g^{r_k}}$: son plan local.

A chaque début d'une croissance d'une région, une graine g^{r_k} est déterminée. Pour ce faire, on teste tous les pixels du scan dans leur ordre d'apparition. Si le pixel q_i respecte les conditions suivantes :

1. q_i n'appartient à aucune région ;
2. la différence angulaire maximale entre la normale en p_i et une normale voisine est inférieure à un seuil τ_α^g (voir tableau 2.1) ;

3. le résidu maximal d'un voisin de p_i relativement au plan local de p_i est inférieur à un seuil nommé τ_d^g (voir tableau 2.1) ;

alors $\mathbf{p}_i$ est sélectionné pour créer une graine : $\mathbf{p}^{g^{r_k}} = \mathbf{p}_i$ et $q^{g^{r_k}} = q_i$. Le plan local $\pi^{g^{r_k}}$ est ensuite déterminé par moyennage des points 3D et des normales dans le voisinage de $q^{g^{r_k}}$ (défini par les 8 pixels voisins de $q^{g^{r_k}}$).

Croissance de régions

La croissance d'une région est réalisée grâce à un voisinage de 8 pixels. Après la détection d'une région, les pixels de cette région sont retirés de la liste des graines potentielles mais sont à nouveau mis en jeu dans l'accroissement des régions suivantes. Un pixel peut alors être sélectionné dans plusieurs régions, autrement dit, les régions se superposent.

Attribution d'une région par point

A cette étape, on souhaite ré-attribuer une seule région à chacun des pixels partagés. Ainsi, pour un pixel q_i partagé par les régions r_a et r_b ($r_a, r_b \in R$), on compare la normale locale $\mathbf{n}_i$ avec les normales des plans locaux de g^{r_a} et de g^{r_b} et on sélectionne la région correspondant à l'écart angulaire le plus faible. Cette méthode permet de résoudre les ambiguités aux voisinages d'intersections entre plans de manière précise et de pallier les problèmes des propagation sur d'autres primitives tout en s'affranchissant d'un nouveau paramètre de seuil angulaire.

Détermination des primitives planaires

Après extraction des régions planaires R, l'ensemble des primitives Π peut être déterminé. Pour chaque région r_k, une primitive π_k est créée à partir de la définition 3.1 :

- Q^{π_k} contient les pixels de r_k et P^{π_k} est l'ensemble des points 3D correspondants ;
- $\mathbf{n}^{\pi_k}$ est déterminée par ACP sur l'ensemble P^{π_k} ;
- d^{π_k} est déterminée par produit scalaire du centroïde de P^{π_k} avec $\mathbf{n}^{\pi_k}$.

Fusion

Lorsque des points 3D d'une région sont éloignés du point de la graine, l'erreur angulaire d'estimation de la normale du plan local π^{g_k} implique une forte erreur résiduelle des points au plan local et plusieurs régions non complètes peuvent alors être détectées sur un même plan. La figure 2.9 présente un exemple schématique d'une telle situation. Une région (en hachuré) est détectée sur une surface plane rouge à partir de la graine du point A associée au plan local bleu. La région obtenue ne représente pas le plan rouge total à cause de l'imprécision de la normale locale en A ; le point C est pris en compte dans la région mais pas le point B. Après sélection de la région associée à la graine du point A, le point B peut à son tour être sélectionné comme une graine d'une nouvelle région et l'imprécision de sa normale locale peut entraîner l'insertion du point C et l'exclusion du point A. Les caractéristiques des différents plans segmentés sont donc comparées et, si elles sont proches et que les régions ont des pixels en commun, les plans sont fusionnés : leurs points et pixels sont mis en commun et une nouvelle paire de caractéristiques (normale et distance) est calculée.

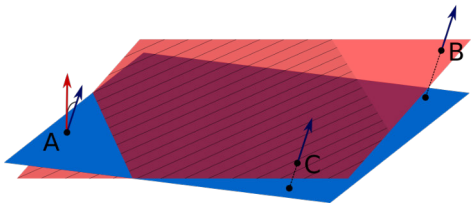
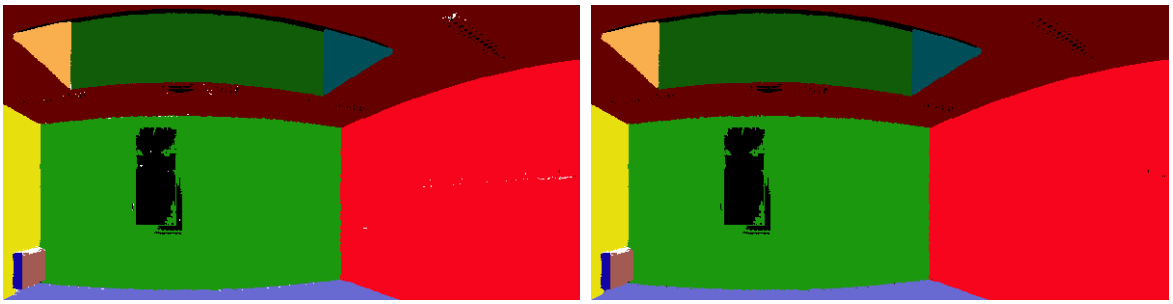


Figure 2.9.: Représentation d'une région (en hachuré) sur une surface plane rouge à partir de la graine du point A associée au plan local bleu. La région hachurée ne couvre pas la totalité du plan rouge à cause de l'imprécision de la normale locale en A.

A la fin de ce procédé, on obtient une image où toutes les régions planaires sont segmentées. Les points n'appartenant à aucune région sont supposés être des objets non planaires. L'image est ensuite filtrée pour éliminer les fausses détections d'objets non planaires dues à l'erreur d'estimation des normales et au bruit sur le nuage de points. Si les objets contiennent moins de S_{min} pixels (voir tableau 2.1), ils sont réattribués au plan le plus proche dans l'image. Le résultat de cette étape est présenté figure 2.10. L'image finale segmentée et filtrée est appelée I. Un exemple de résultat de ces traitements est donné figure 2.11 pour le nuage présenté figure 2.7a. La couleur de chaque pixel correspond à l'indice k de la primitive π_k à laquelle il appartient.



(a) **(b)**

Figure 2.10.: Exemple de filtrage d'une image $\{\varphi, \theta\}$ segmentée pour éliminer les fausses détections d'objets non planaires : **(a)** avant filtrage et **(b)** après filtrage. Les objets non planaires sont représentés en blanc sur l'image.

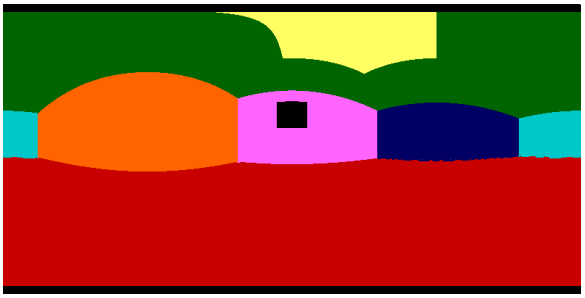


Figure 2.11.: Segmentation de l'image $\{\varphi, \theta\}$ représentée figure 2.7b par plans pour un nuage de points synthétique.

2.3.2 Détection des lignes de contour

Vue générale de l'algorithme

Afin de modéliser des polygones, nous cherchons à détecter les contours des primitives planaires extraites dans la section précédente et à les étiqueter. Une ligne de contour peut séparer deux primitives, indiquer une ouverture ou la présence d'un objet non planaire. Des exemples de lignes de contour sont présentés sur la figure 2.12.

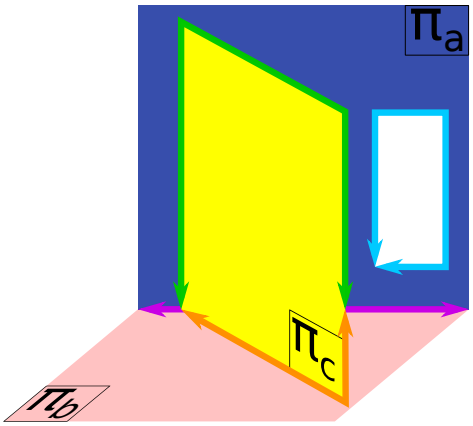


Figure 2.12.: Exemple de lignes de contour pour trois primitives planaires en interaction. La ligne orange représente l'interaction des primitives π_b et π_c, la ligne verte celle de π_a et π_c, la ligne violette celle de π_a et π_b et la ligne bleue met en évidence l'ouverture dans la primitive π_a.

Définition 3.3

L'ensemble des lignes de contour est appelé $\mathcal{L} = \{\ell_l\}_{l=\{1,\dots,N_\mathcal{L}\}}$. Chaque ligne de contour ℓ_l est définie par :

- $P^{\ell_l} = \left\{\mathbf{p}_i^{\ell_l}\right\}_{i=\{1,\dots,N_{\ell_l}\}}$: un ensemble de points 3D ;
- $Q^{\ell_l} = \left\{q_i^{\ell_l}\right\}_{i=\{1\dots N_{\ell_l}\}}$: l'ensemble de pixels correspondants ;
- Π^{ℓ_l} : un ensemble de primitives en interaction. $\Pi^{\ell_l} \subset \Pi$;
- e^{ℓ_l} : une étiquette qui peut être : « interaction avec une autre primitive planaire », « ouverture », ou « interaction avec un objet ».

Les lignes de contour sont extraites successivement de chaque primitive. Pour chaque primitive, une image binaire est créée et filtrée. Les lignes de contour sont alors détectées sur cette image et étiquetées à l'aide de l'image segmentée I. Une ligne de contour entre deux primitives n'est détectée qu'une fois. Ce mécanisme est décrit en détails ci-dessous. On considérera l'étude du contour d'une primitive particulière nommée π. On note π^C le complémentaire de π.

Extraction d'un plan

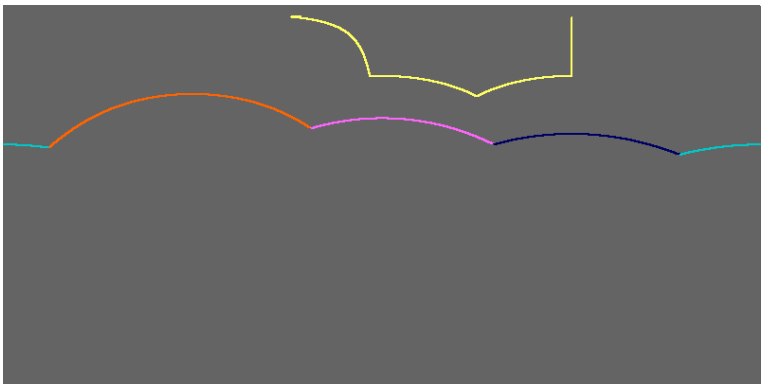
L'ensemble des pixels Q^{π} est isolé dans une image binaire. Cette image est filtrée afin de corriger d'éventuels défauts dus à l'acquisition et/ou à l'erreur d'estimation des normales. Ainsi, un filtre morphologique de dilatation est appliqué avec un masque carré de 6 pixels de largeur pour remplir les pixels vides. Un exemple de résultat de cette opération est montré figure 2.13a.

Extraction des pixels de contour

Les pixels de contour de π sont alors extraits de l'image ainsi que leurs plus proches voisins dans π^{C}. Ce nouvel ensemble de pixels est nommé Q^{π}_{bound}. A noter que les pixels ajoutés par la dilatation morphologique n'ont pas de correspondance de point dans P^{π}, donc si un de ces pixels est détecté comme appartenant au contour, ce sont ses pixels les plus proches et appartenant à Q^{π} qui seront ajoutés à sa place.

Pour créer chaque ligne de contour, il faut regrouper les pixels de Q^{π}_{bound} correspondant à un même élément (une primitive planaire voisine, un objet non planaire ou une ouverture). Un exemple d'un tel regroupement est donné figure 2.13b, l'étiquetage représenté sous forme de couleur correspond à l'objet en interaction avec la primitive étudiée.

(a) Primitive planaire extraite de l'image $\{\varphi, \theta\}$



(b) Lignes de contour étiquetées

Figure 2.13.: Détection du contour d'une primitive planaire.

Création des lignes de contour

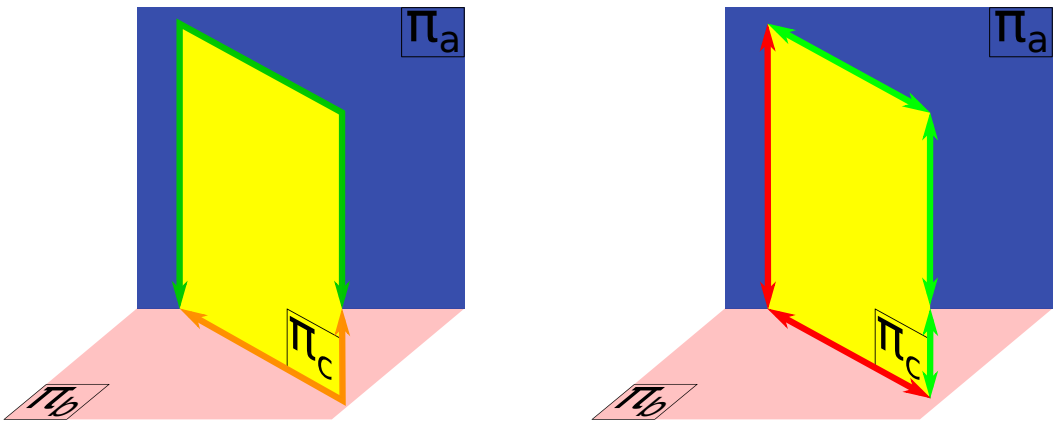
Pour une primitive π, un groupe de pixels de contour extrait de Q^{π}_{bound} permet de créer une ligne ℓ relativement à la définition 3.3 de la façon suivante :

- Q^{ℓ} est le groupe de pixels et P^{ℓ} est l'ensemble des points 3D correspondants.
- Π^{ℓ} contient le plan π et le plan en interaction le cas échéant.
- L'étiquette e^{ℓ} est appliquée comme suit :
  - Si les pixels sont voisins d'un autre plan, l'étiquette est « interaction avec autre plan » ;
  - Si les pixels n'ont pas de voisin (les pixels voisins sont vides), l'étiquette est « ouverture » ;
  - Sinon, l'étiquette est « interaction avec objet non planaire ».

2.3.3 Modélisation des lignes de contour

Vue générale de l'algorithme

On émet l'hypothèse que les lignes de contour peuvent être modélisées par un ensemble de segments de droites. Nous allons donc chercher à ajuster ces modèles et à les étiqueter. Un exemple de modélisation de lignes de contour est donné figure 2.14. Les lignes de contour du plan π_c (à gauche), sont modélisées par des segments (à droite). Ces segments sont étiquetés selon leur nature (ici intersection ou occultation).



(a) (b)

Figure 2.14.: Représentation de lignes de contour et des segments de droites modélisés. **(a)** Lignes de contour d'un plan π_c en interaction avec les plans π_a et π_b. La ligne verte correspond aux plans π_a et π_c et la ligne orange aux plans π_b et π_c. **(b)** Modélisation des lignes par des segments de droites. Les segments verts sont des occultations. Les segments rouges sont des intersections.

Définition 3.4

L'ensemble des segments de droites modélisant les lignes de contour est nommé $\Delta = \{\delta_d\}_{d=\{1...N_\Delta\}}$. Un segment de droite δ_d est défini par :

— $\mathbf{t}^{\delta_d}$: son vecteur directeur ;

— $\{\mathbf{p}^{\delta_d}_{init}, q^{\delta_d}_{init}\}$ et $\{\mathbf{p}^{\delta_d}_{fin}, q^{\delta_d}_{fin}\}$: les couples point 3D/pixel de ses extrémités ;

— $\ell^{\delta_d} \mid \ell^{\delta_d} \in \mathcal{L}$: sa ligne de contour référente.

— $P^{\delta_d} = \left\{\mathbf{p}^{\delta_d}_i\right\}_{i=\{1,...,N_{\delta_d}\}}$: l'ensemble des points 3D modélisés $(P^{\delta_d} \subset P^{\ell^{\delta_d}})$;

— $Q^{\delta_d} = \left\{q^{\delta_d}_i\right\}_{i=\{1,...,N_{\delta_d}\}}$: l'ensemble des pixels modélisés correspondants $(Q^{\delta_d} \subset Q^{\ell^{\delta_d}})$;

— e^{δ_d} : une étiquette qui peut être : « occultation », « intersection », « interaction avec objet » ou « ouverture ».

Les lignes de contour peuvent être de trois types différents : une interaction avec une autre primitive, une ligne d'ouverture, ou une interaction avec un objet (voir figure 2.12). Si ℓ est une interaction entre primitives, alors $\#\Pi^\ell = 2$, sinon, $\#\Pi^\ell = 1$. De plus, une ligne d'interaction entre primitives peut être modélisée par des segments de droites de type « occultation » et/ ou « intersection ». Selon leur nature, les lignes de contour sont modélisées par des méthodes différentes. Un exemple de modélisation de lignes de contour par des segments est donné figure 2.14b. Bien que les lignes puissent être étiquetées directement selon la nature du voisinage de leurs pixels comme décrit dans le paragraphe 2.3.2, il est nécessaire d'effectuer une étape d'étiquetage supplémentaire afin de séparer les lignes ne contenant que des segments de droites d'occultation et les lignes contenant un segment d'intersection. Afin de sélectionner l'une ou l'autre des étiquettes, nous réalisons un test de connectivité. Une fois l'étiquetage réalisé, les lignes sont modélisées de la manière suivante :

- Si ℓ contient un segment de droite d'intersection, on modélise cette droite par l'intersection des plans associés à ℓ ;
- Si ℓ est une ligne d'ouverture, d'occultation ou une ligne d'interaction avec un objet, on ajuste des droites par une méthode fondée sur RANSAC en 2D ;
- Si ℓ est une ligne d'occultation, l'estimation du segment du plan occulté est affinée grâce à l'estimation du segment du plan occultant et inversement.

Après la modélisation d'un segment δ dans la ligne de contour ℓ, les points P^δ et les pixels Q^δ sont retirés de P^ℓ et Q^ℓ respectivement et de nouveaux segments sont recherchés dans les points restants. Les différentes étapes décrites dans ce paragraphe sont détaillées dans les sous-sections suivantes.

Test de connectivité

Un test de connectivité doit être réalisé sur les lignes étiquetées comme interactions entre deux plans afin de déterminer si elles contiennent une intersection. Pour une ligne ℓ de ce type, nous déterminons les pixels qui pourraient être attribués à une intersection entre les plans de l'ensemble Π^{ℓ}, on nomme ce nouvel ensemble de pixels potentiels $Q^{\ell *}$. Le calcul de ces pixels est détaillé dans l'annexe A. La figure 2.15 donne un exemple de résultat obtenu pour le contour de la primitive planaire étudiée figure 2.13. Si le nombre de pixels inclus dans $\left\{Q^{\ell} \cap Q^{\ell *}\right\}$ est supérieur à la valeur seuil précédemment définie S_{min}, on déduit que la ligne contient une intersection, sinon, on déduit qu'il s'agit d'un ensemble de segments d'occultation. Dans l'exemple figure 2.15, toutes les lignes de contour ont des pixels communs avec leurs droites d'intersection théorique et sont donc étiquetées en tant qu'intersections, exceptée la ligne jaune qui résulte d'une occultation.

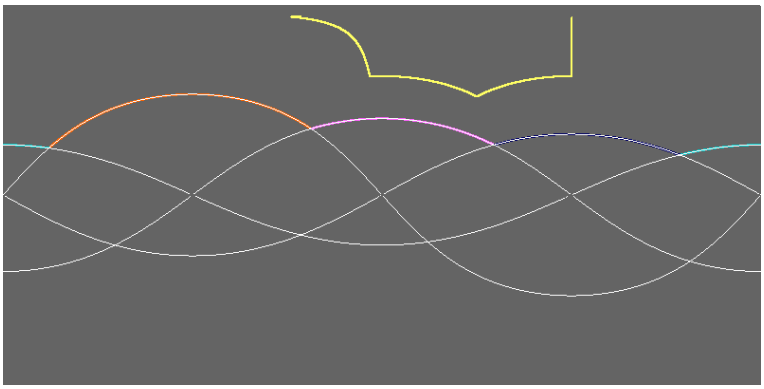


Figure 2.15.: Représentation des lignes théoriques pour test de connectivité. Les lignes blanches correspondent aux intersections théoriques calculées avec les caractéristiques des plans. Les lignes de couleurs proviennent du contour déterminé sur l'image $\{\varphi, \theta\}$. Les lignes de contour bleue, cyan, orange et rose sont confondues avec leurs lignes théoriques, elles sont donc étiquetées « intersections entre plans ». La ligne de contour jaune correspond à une occultation.

Modélisation de segments d'intersection

Après détection d'un segment de droite d'intersection δ par test de connectivité, ses caractéristiques (voir définition 3.4) peuvent être calculées de la manière suivante :

- $\mathbf{t}^{\delta}$ est déduit par produit vectoriel des normales des plans de $\Pi^{\ell^{\delta}}$;
- Q^{δ} est défini par $Q^{\delta} = Q^{\ell^{\delta}} \cap Q^{\ell^{\delta} *}$. P^{δ} est constitué des points 3D correspondants ;
- Pour déterminer $\mathbf{p}^{\delta}_{init}$ et $\mathbf{p}^{\delta}_{fin}$, on calcule les valeurs de projection $\{\mathbf{p}^{\delta}_i \cdot \mathbf{t}^{\delta}\}_{i=\{1,\ldots,N_{\delta}\}}$ des points de P^{δ} sur le vecteur directeur $\mathbf{t}^{\delta}$. Les projections minimales et maximales correspondent respectivement aux projections de $\mathbf{p}^{\delta}_{init}$ et $\mathbf{p}^{\delta}_{fin}$. On résout donc les systèmes (2.6).

$$\left\{\begin{array}{l} \mathbf{p} \cdot \mathbf{n}^{\pi_1^{\ell^{\delta}}} = d^{\pi_1^{\ell^{\delta}}} \\ \mathbf{p} \cdot \mathbf{n}^{\pi_2^{\ell^{\delta}}} = d^{\pi_2^{\ell^{\delta}}} \\ \mathbf{p} \cdot \mathbf{t}^{\delta} = \min\limits_{i \in [1, N_{\delta}]} (\mathbf{p}^{\delta}_i \cdot \mathbf{t}^{\delta}) \end{array}\right. \quad \left\{\begin{array}{l} \mathbf{p} \cdot \mathbf{n}^{\pi_1^{\ell^{\delta}}} = d^{\pi_1^{\ell^{\delta}}} \\ \mathbf{p} \cdot \mathbf{n}^{\pi_2^{\ell^{\delta}}} = d^{\pi_2^{\ell^{\delta}}} \\ \mathbf{p} \cdot \mathbf{t}^{\delta} = \max\limits_{i \in [1, N_{\delta}]} (\mathbf{p}^{\delta}_i \cdot \mathbf{t}^{\delta}) \end{array}\right. \tag{2.6}$$

Modélisation de segments d'ouverture

Afin de modéliser un segment d'ouverture δ, on cherche à ajuster une droite sur les points d'une ligne ℓ étiquetée comme « ouverture ». Pour cela, on utilise une procédure RANSAC 2D. En effet, les points de l'ouverture appartiennent à un plan unique ($\Pi^\ell = \left\{\pi^\ell\right\}$), la recherche de lignes s'opère donc dans le repère de ce plan. RANSAC est réalisé avec un nombre maximum de tests M et un seuil d'appartenance à la droite nommé τ_d. Une distance maximum entre deux points consécutifs d'un même segment est paramétrée simultanément en pixels à la valeur de τ^q_{diff} et en distance spatiale à la valeur de τ^p_{diff}. Le seuil S_{min} précédemment défini (voir tableau 2.1) est également utilisé pour définir un nombre minimum de pixels dans un segment.

Cette première modélisation est affinée par une étape d'ajustement de droite par moindres carrés, toujours en 2D, réalisé sur les points extraits par RANSAC sans les points des extrémités du segment afin de réduire l'influence des segments voisins (voir schéma de la figure 2.16).

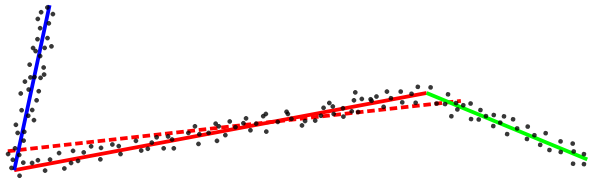


Figure 2.16.: Ajustement d'un segment de droite (rouge) sur une ligne de contour représentée par des points 2D. Le segment rouge en pointillés est détecté par recherche RANSAC et corrigé par moindres carrés en retirant les points des extrémités appartenant potentiellement aux segments bleus et verts.

Chaque ensemble P^δ est finalement défini par les points à une distance inférieure à τ_δ de la droite ajustée par moindres carrés. Les extrémités ($\mathbf{p}^\delta_{init}$ et $\mathbf{p}^\delta_{fin}$) du segment sont calculées par projection des points de P^δ sur la droite obtenue et par sélection des valeurs de projection minimale et maximale respectivement.

Les segments sont ainsi détectés successivement en retirant les points modélisés à chaque itération jusqu'à ce que le nombre de points ne soit plus suffisant (inférieur au seuil S_{min}). Après cette étape, les segments sont finalement reprojetés en 3D.

Modélisation de segments d'occultation

Contrairement à une ligne d'ouverture, une ligne d'occultation ℓ correspond à des points 3D sur deux plans distincts ($\Pi^\ell = \left\{\pi^\ell_1, \pi^\ell_2\right\}$). On cherche alors à modéliser à partir de la même ligne de contour des couples de segments, dont les premiers sont dans π^ℓ_1 et les seconds sont dans π^ℓ_2. Pour ce faire, un premier segment δ_1 est ajusté sur les points de P^ℓ appartenant à π^ℓ_1 en utilisant, comme précédemment, une procédure RANSAC 2D, doublée d'un ajustement par moindres carrés. Ensuite, un second segment δ_2 est ajusté sur les points de P^ℓ appartenant à π^ℓ_2 et dont les pixels sont voisins des pixels modélisés par δ_1. Puis, dans chaque couple de segments

$\{\delta_1, \delta_2\}$, δ_1 est projeté sur le plan π_2^ℓ et δ_2 est projeté sur le plan π_1^ℓ dans la direction du rayon laser du capteur. Pour un point $\mathbf{p}$ que l'on souhaite projeter sur le plan π, la projection $\mathcal{P}_\pi\{\cdot\}$ peut s'exprimer de la manière suivante :

$$\mathcal{P}_\pi\{\mathbf{p}\} = \frac{d^\pi}{\mathbf{p} \cdot \mathbf{n}^\pi}\mathbf{p}\,. \tag{2.7}$$

Un exemple de ce type de projection est donné figure 2.17. Les segments ajustés sont représentés par une ligne pleine et les segments projetés sont représentés par des pointillés. Une fois cette projection effectuée, on obtient deux nouveaux couples de segments : $\{\delta_1, \mathcal{P}_{\pi_1^\ell}\{\delta_2\}\}$ et $\{\mathcal{P}_{\pi_2^\ell}\{\delta_1\}, \delta_2\}$ contenus respectivement dans les plans π_1^ℓ et π_2^ℓ. Finalement, une moyenne est calculée entre les segments de chaque couple afin d'obtenir deux segments finaux, l'un dans le plan π_1^ℓ et l'autre dans le plan π_2^ℓ.

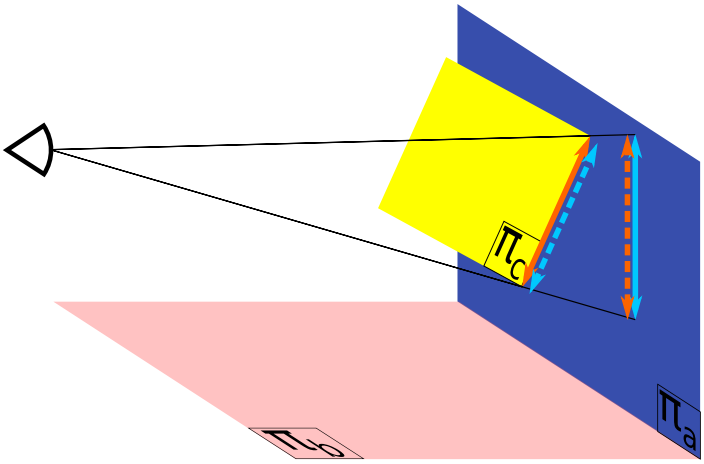


Figure 2.17.: Détection de segments de droites d'occultation à partir de lignes de contour d'interaction entre plans. Les lignes pleines représentent les segments détectés par RANSAC 2D sur les plans π_c et π_a, les lignes pointillées représentent leur projection sur l'autre plan.

Modélisation d'interactions avec des objets

Pour les interactions avec des objets non planaires, la différence avec la modélisation précédente réside dans le fait que seule la projection des points de l'objet détecté vers le plan étudié est réalisée. Le contour de l'objet n'est donc pas modélisé.

2.3.4 Modélisation des coins

Vue générale de l'algorithme

La dernière étape consiste à connecter les segments de droites pour obtenir des polygones plans. Pour cela, nous cherchons les sommets des polygones.

Définition 3.5

On appelle $\mathcal{C} = \{c_n\}_{n=\{1,\dots,N_\mathcal{C}\}}$ l'ensemble des coins de la scène. Un coin c_n de l'ensemble $\mathcal{C}$ est défini par :

- $\mathbf{p}^{c_n}$: un point 3D ;
- q^{c_n} : un pixel ;
- $\Delta^{c_n} \mid \Delta^{c_n} \subset \Delta$: un ensemble de segments de droites ;
- e^{c_n} : une étiquette qui peut être : intersection de trois plans, intersection de deux lignes, continuité de deux lignes.

Les coins des polygones plans peuvent être l'intersection de trois plans, l'intersection de deux lignes, l'intersection de trois lignes et la continuité de deux lignes. Ces différents types sont illustrés figure 2.18. Selon son type, un coin est calculé de manière différente. L'algorithme consiste tout d'abord à calculer un ensemble de coins potentiels $\mathcal{C}^*$ relativement à l'ensemble des segments de droites Δ. Ensuite, ces coins potentiels sont comparés avec les extrémités mesurées des segments afin d'être sélectionnés ou non dans l'ensemble $\mathcal{C}$. Ce processus est décrit en détails ci-après.

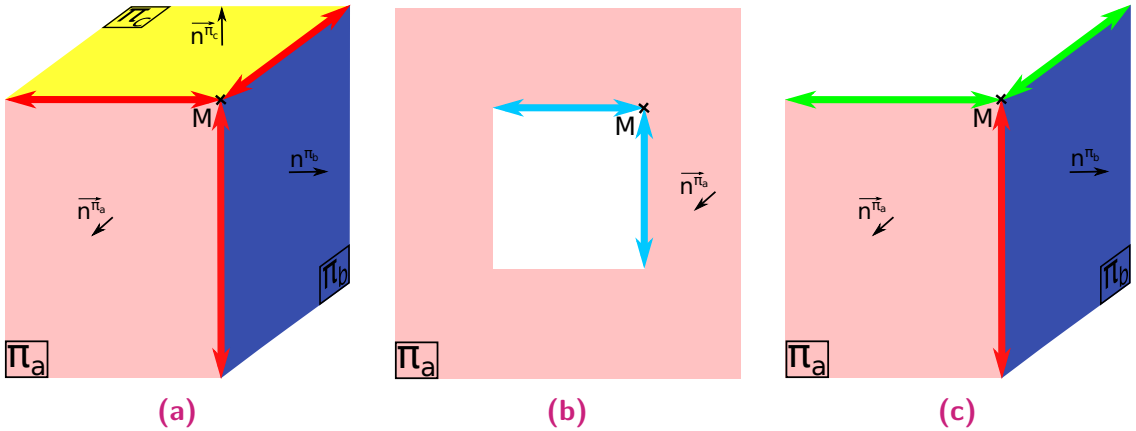


Figure 2.18.: Types de coins possibles. **(a)** Coin formé à partir de l'intersection de trois plans ; **(b)** coin formé à partir de l'intersection de deux droites ; **(c)** coin formé à partir de l'intersection de trois droites.

Intersection de trois plans

On sélectionne les couples de segments de droites étiquetés comme intersections $\{\delta_\alpha, \delta_\beta\}_{\delta_\alpha,\delta_\beta \in \Delta}$ qui ont un plan en commun. Ce plan commun est nommé $\pi^{\delta_{\alpha,\beta}}$. L'union des ensembles Π^{δ_α} et Π^{δ_β} contient alors trois plans, nommés $\left\{\pi^{\delta_\alpha}, \pi^{\delta_\beta}, \pi^{\delta_{\alpha,\beta}}\right\}$. Un nouveau coin potentiel c^* est créé dont le point $\mathbf{p}^{c^*}$ est l'intersection de ces trois plans. Les segments δ_α et δ_β forment alors l'ensemble Δ^{c^*}. Enfin, on cherche si un troisième segment représente l'intersection entre les plans π^{δ_α} et π^{δ_β}. Si ce segment existe, il est ajouté à Δ^{c^*}.

Intersection de deux lignes

L'intersection de deux lignes correspond à des couples de segments $\{\delta_\alpha, \delta_\beta\}_{\delta_\alpha,\delta_\beta \in \Delta}$ qui ont un plan commun et dont l'un des segments au moins n'est pas une intersection. Ce plan est alors nommé $\pi^{\delta_{\alpha,\beta}}$. Un nouveau coin potentiel c^* est créé dont le point $\mathbf{p}^{c^*}$ est l'intersection des droites directrices de ces segments dans le plan le $\pi^{\delta_{\alpha,\beta}}$. L'ensemble de segments de droites associé à c_n^* est alors $\Delta^{c_n^*} = \{\delta_\alpha, \delta_\beta\}$.

Continuité de deux lignes

Lorsqu'une primitive occulte différentes autres primitives, une même arête peut être divisée en plusieurs segments consécutifs. Les segments verts, figure 2.14b, sont un exemple d'un tel cas. Ces différents segments sont reconnectés. Pour cela, pour chaque couple de segments $\{\delta_a, \delta_b\}$, on vérifie les critères suivants :

1. δ_a et δ_b ont un plan en commun : $\Pi^{\delta_a} \cap \Pi^{\delta_b} \neq \emptyset$;
2. δ_a et δ_b sont parallèles : $\arccos\left(|\mathbf{t}^{\delta_a} \cdot \mathbf{t}^{\delta_b}|\right) < \tau_{//}$;
3. δ_a et δ_b ont des extrémités proches : $||\mathbf{p}_{fin}^{\delta_a} - \mathbf{p}_{init}^{\delta_b}||_2 < \tau_c^p$ ou $||\mathbf{p}_{fin}^{\delta_b} - \mathbf{p}_{init}^{\delta_a}||_2 < \tau_c^p$.

Si ces critères sont respectés un coin potentiel est créé et ajouté à $\mathcal{C}^*$ pour lier les deux segments.

Sélection des coins et fusion

Pour un segment de droite δ, on extrait les coins potentiels dont l'ensemble de segments correspondant contient δ (*c.-à-d.* que pour un coin c^*, $\delta \in \Delta^{c^*}$). On calcule alors les distances entre le point $\mathbf{p}^{c^*}$ et les extrémités de δ ($\mathbf{p}_{init}^{\delta}$ et $\mathbf{p}_{fin}^{\delta}$). Après avoir calculé toutes les distances 3D entre extrémités et coins potentiels, pour chaque extrémité on sélectionne le coin c^* pour lequel la distance est la plus faible, *c.-à-d.* celui dont le point $\mathbf{p}^{c^*}$ est le plus proche. Si cette distance est inférieure à un seuil nommé τ_c^p et si la distance entre le pixel q^{c^*} et le pixel de l'extrémité est inférieure à un seuil τ_c^q, on ajoute le coin à $\mathcal{C}$ et on modifie les extrémités du segment δ en conséquence.

Enfin, comme on peut le voir sur la figure 2.18c, il est possible que le coin (appelé M sur l'image) soit l'intersection de trois droites sans pour autant être une intersection de trois plans. Dans ce cas, on obtiendra deux coins distincts, calculés à partir de deux intersections de droites. La dernière étape de l'algorithme consiste donc à détecter ces doublons et à les fusionner. La détection est réalisée en cherchant les coins calculés à partir d'un segment commun qui sont séparés d'une distance inférieure au seuil τ_c^p. Le point moyen est calculé entre les points des deux coins.

2.3.5 Modélisation finale

Un exemple de résultat obtenu après la construction de segments et de coins est donné figure 2.19 pour l'exemple synthétique étudié dans cette section (voir figure 2.7). A partir de ce résultat, la dernière étape consiste à fermer les polygones et créer les trous. Pour ce faire, en partant d'un coin choisi aléatoirement dans un plan, les segments de ce plan sont parcourus par connexion jusqu'à retrouver le coin initial.

On obtient alors un cycle. Si un segment n'est connecté que par une extrémité, on cherche le coin du polygone non connecté le plus proche de l'autre extrémité pour fermer le cycle. De plus, si une fermeture de cycle implique de traverser une arête, le coin le plus proche de l'intersection est retiré et le parcours est poursuivi. Plusieurs cycles peuvent appartenir à une même primitive planaire, notamment lorsqu'une ouverture ou un objet occultant ont été détectés sur un mur.

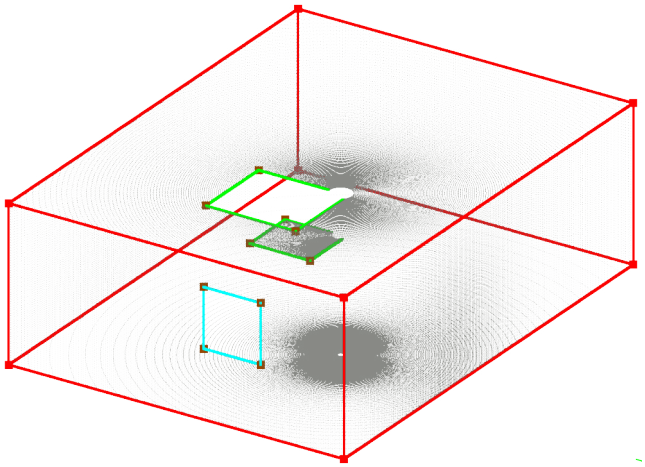
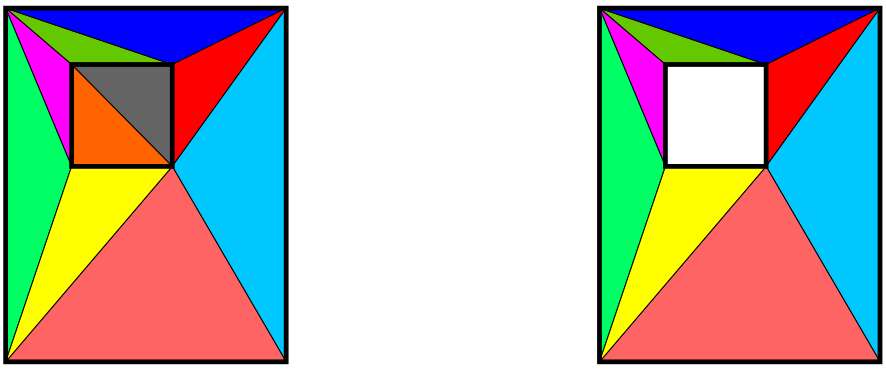


Figure 2.19.: Segments de contour des polygones et coins détectés par notre méthode, superposés sur le nuage de points issu d'une simulation LiDAR dans une scène synthétique. Les étiquettes des arêtes sont : intersection (rouge), occultation (vert) et ouverture (cyan). Les étiquettes des coins sont : intersection de trois plans (rouge), intersection de deux segments (marron).

Un maillage peut ensuite être créé par triangulation sur les coins en imposant les arêtes des polygones. Pour une même primitive plane, le cycle extérieur est détecté et les faces situées dans les cycles intérieurs sont retirées (voir figure 2.20). La triangulation de Delaunay contrainte implémentée par la librairie CGAL a été utilisée pour réaliser cette tâche. Le résultat maillé de l'exemple utilisé dans cette section (voir figure 2.7) est représenté figure 2.21.



(a) Triangulation contrainte avec les arêtes des polygones externe et interne.

(b) Elimination des faces dans le polygone interne.

Figure 2.20.: Triangulation contrainte pour faire réapparaître les trous. Les polygones initiaux sont représentés en gras.

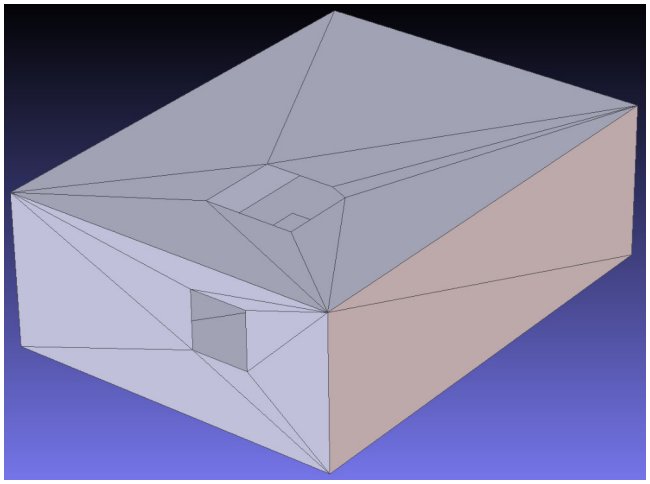


Figure 2.21.: Maillage obtenu à partir des segments et des coins représentés sur la figure 2.19.

2.3.6 Paramétrage

Pour tous les exemples traités, nous utilisons le même ensemble de paramètres tel que détaillé dans le tableau 2.1. Cet ensemble a été choisi relativement aux caractéristiques physiques des environnements intérieurs et des dispositifs LiDAR généralement employés.

Tableau 2.1.: Paramètres de la méthode de modélisation de scan.

Contexte	Nom	Signification	Valeur
Général	S_{min}	Nombre de points/pixels minimum	4
	$\tau_{//}$	Angle maximum pour définir le parallélisme	10°
Sélection de graine	τ_d^g	Distance maximale d'un voisin au plan local de la graine	2 cm
	τ_α^g	Différence angulaire maximale entre normales voisines	8°
Croissance de région	τ_d	Distance au plan local de la graine	8 cm
RANSAC	M	Nombre de tests maximum	50 000
	τ_δ	Distance spatiale maximale à une droite	2 cm
	τ_{diff}^p	Distance spatiale maximale entre deux points consécutifs d'une droite	10 cm
	τ_{diff}^q	Distance maximale en pixels entre deux pixels consécutifs d'une droite	5
Sélection de coins	τ_c^p	Distance spatiale au coin potentiel	15 cm
	τ_c^q	Distance en pixels au coin potentiel	10

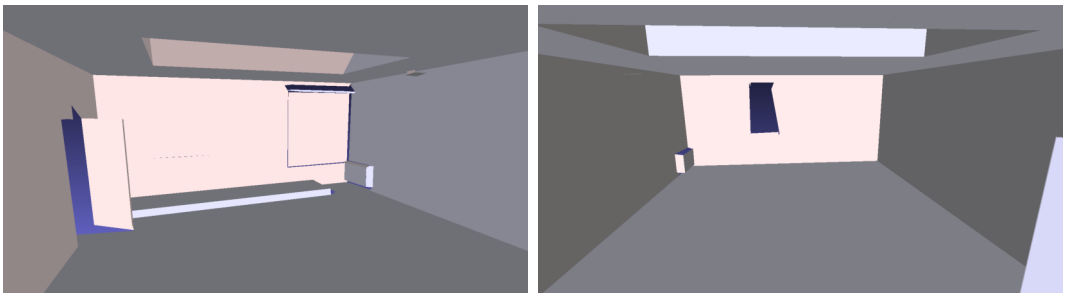
2.4 Evaluation

Nous évaluons notre méthode sur des données réelles d'acquisition LiDAR en intérieur. Ces données ont été acquises par un scanner Leica P20 dont la description complète est donnée en annexe (voir annexe B).

Premièrement, une salle est scannée de deux points de vue distincts et modélisée par notre méthode. Les résultats de modélisation sont montrés figure 2.23. Les reconstructions points/ segments des figures 2.23c et 2.23d mettent en évidence les relations entre les plans et les objets dans la scène. On remarque que, selon le point de vue, certaines régions planaires peuvent être identifiées comme plan ou objets non planaires si le nombre de points sur la région n'est pas suffisant ou si l'estimation des normales n'est pas assez précise. Les segments d'intersection courts sont tous détectés mais certains segments d'occultation n'ont pas pu être mis en évidence par l'algorithme RANSAC. Sur les images de reconstruction figures 2.23e, 2.23f, 2.23g, 2.23h et 2.22, on remarque que l'anisotropie d'échantillonnage des nuages de points est bien traitée par notre méthode. Les polygones fins tels que la bordure de la porte ou les grilles des plafonniers sont détectés s'ils sont échantillonnés par au moins 3 points sur leur largeur. Les petits objets occultants sont également détectés : l'éclairage de secours au-dessus de la porte, les radiateurs, le vidéo-projecteur, le repose-craie du tableau, etc. Cependant, on remarque que les points reflétés sur les vitres peuvent corrompre la reconstruction (voir figure 2.22b). De plus, les plafonniers ne sont pas entièrement détectés car leur altitude est dans la zone du bruit sur le plafond.

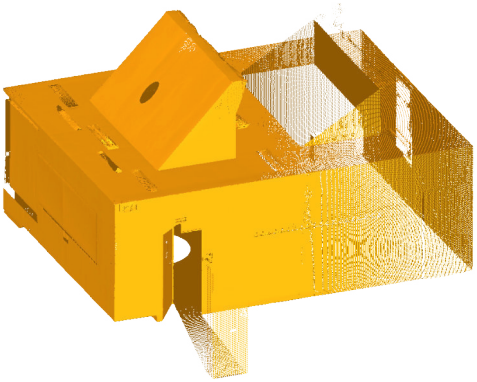
Deuxièmement, un autre exemple de scan est testé dans une scène comprenant un escalier (voir figure 2.24a). La scène est correctement reconstruite. Les problèmes résiduels concernent les formes courbées telles que la cible discoïde placée au premier plan sur le sol pour aider au recalage dont le contour est modélisé par un polygone ce qui provoque une perte d'information.

Sur les données testées, notre méthode donne des résultats visuellement corrects avec un minimum d'extrapolation des nuages de points. Elle parvient à résoudre les problèmes induits par l'échantillonnage anisotrope, les structures fines et le bruit de mesure.

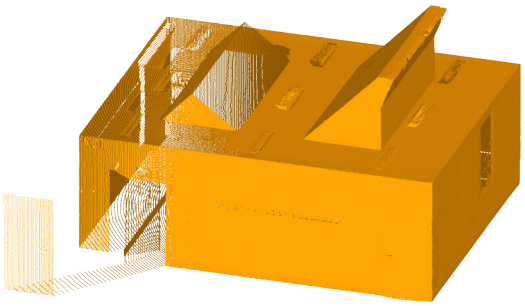


(a) Vue intérieure. **(b)** Vue intérieure opposée.

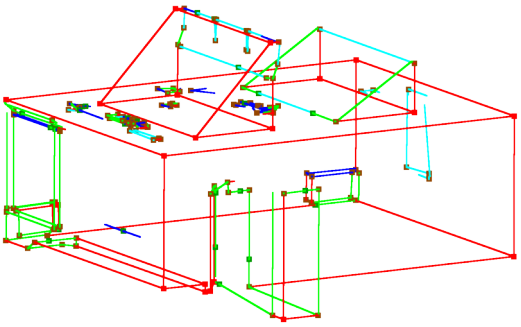
Figure 2.22.: Modélisation 3D à partir d'un nuage de points issu d'un scan LiDAR d'une scène d'intérieur de bâtiment (voir figure 2.23a).



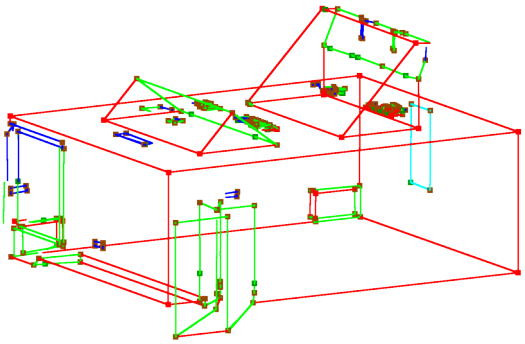
(a) Scan 1



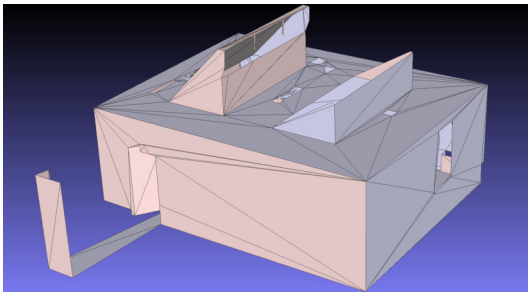
(b) Scan 2



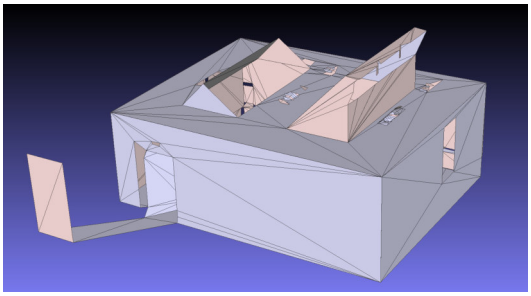
(c) Segments et coins du scan 1



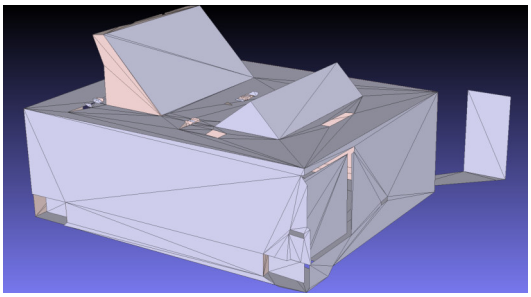
(d) Segments et coins du scan 2



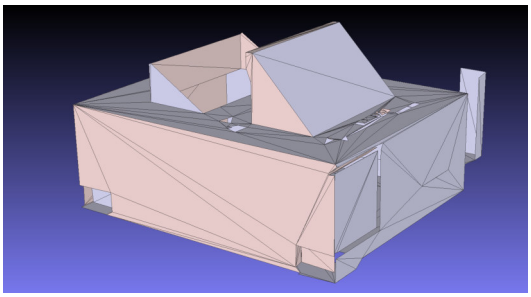
(e) Première vue extérieure du scan 1 reconstruit



(f) Première vue extérieure du scan 2 reconstruit

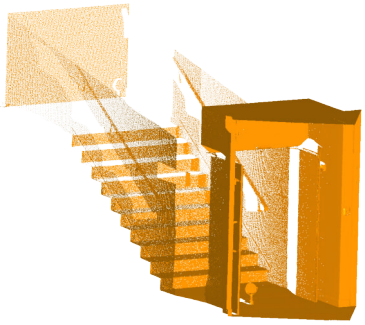


(g) Deuxième vue extérieure du scan 1 reconstruit

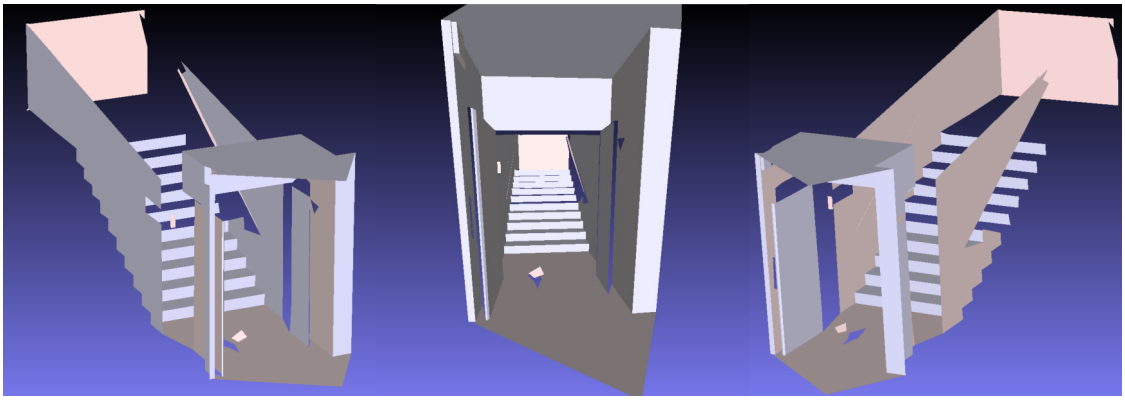


(h) Deuxième vue extérieure du scan 2 reconstruit

Figure 2.23.: Application de notre méthode à deux nuages de points acquis par un dispositif LiDAR dans une salle de classe de l'Université de Clermont-Ferrand par un scanner LiDAR Leica P20. (a) et (b) : nuages de points. (c) et (d) : segments et coins. Les étiquettes des arêtes sont : intersection (rouge), occultation (vert), interaction avec objet (bleu) et ouverture (cyan). Les étiquettes des coins sont : intersection de 3 plans (rouge), intersection de 2 segments (marron). (e), (f), (g) et (h) : modèles 3D maillés est reconstruits.



(a) Nuage de points acquis par le capteur.



(b) Vue de gauche. **(c)** Vue du capteur. **(d)** Vue de droite.

Figure 2.24.: Modélisation 3D à partir d'un nuage de points issu d'un scan LiDAR d'une scène d'intérieur de bâtiment contenant un escalier.

2.5 Conclusion

Nous avons vu que la modélisation de bâtiment était couramment divisée en deux étapes, la segmentation du nuage de points et l'ajustement de modèles sur ces points. L'étude de l'état de l'art dans le domaine de la reconstruction 3D nous a permis de mettre en évidence certaines failles des algorithmes existants. Dans le procédé de segmentation, ces algorithmes ne permettent que de détecter des patches de plans qui devront ensuite être traités pour en déduire leurs connexions et leurs contours. En outre, les méthodes principalement utilisées demandent un paramétrage complexe. Dans le procédé d'ajustement de modèles, les principaux défauts des méthodes de l'état de l'art sont la forte extrapolation produite sur les données et l'écart retrouvé avec les données réelles. De plus, la robustesse à l'anisotropie de l'échantillonnage de certaines méthodes est une limite majeure à leur application. Nous avons alors souhaité proposer une nouvelle méthode permettant de résoudre les problèmes de modélisation par scan. Elle repose sur le traitement simultané du nuage de points et de l'image d'acquisition angulaire d'un scan. La segmentation est alors obtenue par croissance de régions sur l'image, puis, l'ajustement de modèles polygonaux est réalisé par recherche de segments de contour conjointement dans l'image et dans le nuage de points. Cette méthode n'utilise pas d'*a priori* de verticalité des murs,

d'horizontalité du sol ni d'orientation du nuages de points en général. Les plans verticaux ne sont pas extrapolés du sol au plafond par défaut, et nous n'utilisons pas de caractéristique architecturale. En ce sens, notre algorithme respecte l'objectif 1 fixé section 2.2 d'utiliser un minimum d'*a priori* géométriques. Cependant, une hypothèse nécessaire à son bon fonctionnement est que les composants de la structure soient planaires. En l'état, il n'est donc pas adapté pour reconstruire des murs courbes. Ensuite, l'anisotropie de l'échantillonnage est bien résolue par le traitement conjoint du nuage de points et de l'image d'acquisition angulaire, donc l'objectif 2 est atteint. La création d'un modèle polygonal par détection des contours des primitives en utilisant notre estimateur de normales optimisé, nous a permis de nous défaire de l'arrangement de plans utilisé dans la majorité des méthodes. Cette approche respecte donc l'objectif 3, car elle permet une plus faible extrapolation des données. Enfin, les informations d'occultations, de connexions et d'ouvertures ont bien été extraites du nuage de points, ce qui répond aux objectifs 4 et 5.

Une fois les scans individuels modélisés, la question du recalage des différents scans peut être posée. La partie suivante traite du recalage des scènes sous forme de nuages de points et une extension est proposée pour recaler les scans modélisés.

Partie II

Recalage

Le processus d'acquisition LiDAR permet d'obtenir des scans de nuages de points dans le repère du capteur. Le recalage est le procédé permettant d'aligner deux entités géométriques dans un repère commun. Ce traitement fournit les poses (position et orientation) entre les différentes vues grâce à leur chevauchement. Dans la majorité des chaînes de traitements, le recalage est réalisé sur les nuages de points après acquisition. Une fois alignés, les nuages de points peuvent être fusionnés afin d'appliquer d'autres traitements. L'étape de recalage est une tâche récurrente en acquisition LiDAR. On distingue deux domaines majeurs d'application dont les objectifs et les contextes d'acquisition diffèrent :

- la navigation autonome où le recalage est intégré à un processus de SLAM (*Simultaneous Localization and Mapping*) [Cad+16] pour localiser le capteur LiDAR ;
- la reconstruction 3D qui vise à obtenir une mesure précise de l'environnement.

Le premier domaine requiert un faible temps d'acquisition et de calcul. Dans ce contexte, les nuages de points traités sont acquis en haute fréquence (les mouvements entre scans sont donc souvent faibles) et ils sont souvent peu précis. Des capteurs externes peuvent être utilisés pour élargir les données de localisation (compteur de tours de roues, accéléromètre, théodolite). Le recalage est alors généralement intégré à un filtre de Kalman dans lequel toutes les informations de localisation sont fusionnées grâce à une estimation de leurs erreurs.

Dans le second domaine, les nuages de points traités sont très denses, ils présentent une haute précision et un recouvrement important de l'espace 3D. Les mouvements entre scans peuvent être larges mais les acquisitions sont souvent réalisées à l'aide de cibles physiques placées dans l'environnement permettant un recalage plus fiable que celui réalisé par les algorithmes existants. Le placement de ces cibles est un processus long et complexe qui requiert une connaissance *a priori* de l'environnement.

La variété de nuages de points traités et la variabilité des contraintes et des objectifs de ces deux domaines compliquent la généralisation d'un algorithme de recalage. Dans cette partie, nous abordons le problème du recalage dans un contexte d'acquisition LiDAR en environnement structuré. Dans un premier chapitre, nous nous intéresserons à l'estimation de la transformation rigide permettant l'alignement de deux scans successifs. Nous mettrons tout d'abord en évidence les manques de l'état de l'art dans ce type de contexte. En effet, les algorithmes actuels les plus utilisés reposent sur une mise en correspondance de voisinages locaux et requièrent un faible mouvement entre scans, un chevauchement important et des caractéristiques locales significatives [BM92 ; Zha94 ; Zho+16]. Tous ces critères n'étant pas toujours réunis dans un contexte intérieur, nous proposerons une nouvelle méthode globale qui résout les problèmes précédents tout en conservant un temps de calcul faible et une consommation de mémoire permettant le traitement de larges données. Nous travaillerons tout d'abord sur les nuages de points, puis nous expliquerons comment généraliser cette pratique aux modélisations générées dans la partie précédente.

Dans un second chapitre, nous étudierons l'estimation de l'erreur relative à la transformation. Cette erreur peut notamment servir à ajuster les recalages successifs globalement en cas de fermeture de boucle (*c.-à-d.* si une scène est scannée plusieurs

fois pendant l'acquisition) mais elle peut également être intégrée à un filtre de Kalman dans le cadre d'un suivi de position par un processus d'estimation. Nous présenterons brièvement le contexte d'estimation de positions et les besoins d'un modèle de distribution des poses qui s'adapte au contexte d'acquisition et aux différentes scènes. Nous verrons que cette distribution est difficilement estimable du fait des nombreux facteurs agissant sur sa forme et son amplitude tels que le bruit, l'échantillonnage, l'orientation ou la géométrie de la scène. Nous choisissons de nous restreindre à des distributions d'erreur gaussiennes afin d'employer des méthodes connues et rapides d'estimation telles que le filtre de Kalman. Le modèle de distribution repose alors sur la détermination d'une matrice de covariance fiable. Dans ce contexte, nous expliquerons les bénéfices de l'utilisation d'un modèle par apprentissage à partir de matrices de covariances connues. Nous détaillerons l'algorithme CELLO proposé par Vega-Brown [VB13] et adapté en 3D par Landry et coll. [Lan+19] et nous étudierons son comportement sur un jeu de données simulées. De plus, nous proposerons une adaptation de cet algorithme au contexte d'environnement intérieur.

Estimation de transformation rigide

3

3.1 Problématique

Un des principaux défis du recalage dans des scènes d'intérieur est d'obtenir une estimation précise sur des données à faible degré de détails de manière automatique quelle que soit la configuration de départ des scans. Les algorithmes existants présentent un manque de précision et ne résolvent le recalage que dans un nombre restreint de configurations. La plupart d'entre eux sont lents et ne peuvent pas traiter des données bruitées.

L'algorithme le plus populaire dans les chaînes de traitement automatiques est ICP (*Iterative Closest Point*) [BM92 ; Zha94 ; CM92]. Le plus grand défaut de cette méthode est que la fonction de coût minimisée n'est pas convexe, ce qui peut mener à une convergence dans des minima locaux. Afin de résoudre ce problème, l'utilisateur doit fournir des nuages de points avec un faible mouvement, soit en adaptant le processus d'acquisition, soit en appliquant un alignement grossier *a priori*. Ce premier alignement peut être déterminé manuellement, par des mesures odométriques, ou par d'autres algorithmes de recalage moins précis. De plus, l'approche locale des algorithmes de type ICP implique la nécessité de disposer de caractéristiques locales significatives. Or, dans le cas d'environnements intérieurs, ces caractéristiques contiennent peu d'information d'identification à cause du manque de détails des scènes étudiées. Enfin, les approches locales souffrent également du problème d'anisotropie de l'échantillonnage et de fortes différences de densité entre deux nuages de points à recaler. Les approches globales développées pour résoudre le recalage requièrent un grand nombre d'hypothèses de construction et ne fonctionnent pas dans toutes les configurations. De plus, le temps de calcul limite les méthodes les plus précises pour le traitement de larges données. La problématique à laquelle nous allons essayer de répondre dans ce chapitre est la suivante :

Problématique

Comment estimer la transformation rigide permettant d'aligner deux nuages de points acquis par un dispositif LiDAR ?

Les contraintes que doit respecter la méthode recherchée sont résumées ci-dessous :

Contraintes

— la scène contient peu de détails ;
— la scène est échantillonnée de manière anisotrope ;
— le nuage de points est bruité ;
— le chevauchement entre les nuages de points peut être faible ;
— le temps de calcul doit être limité.

Dans un premier temps, nous détaillerons les algorithmes existants permettant d'aligner des nuages de points. Nous établirons les manques de l'état de l'art pour résoudre le recalage de manière précise dans un contexte d'intérieur de bâtiment. Nous présenterons alors une nouvelle approche adaptée aux environnements structurés tels que des scènes d'intérieur de bâtiment. Une évaluation de notre méthode sera menée en comparaison à 6 méthodes de l'état de l'art. Enfin, nous proposerons un algorithme permettant d'adapter le recalage proposé au modèle de structure de polygones développé dans la partie précédente. Nous montrerons en quoi l'approche globale permet une extension à ce type de modèles.

3.2 Etat de l'art

Le problème peut être posé de la manière suivante : à partir d'un scan appelé « source » contenant les points de l'ensemble $S = \{\mathbf{s}_i\}_{i=1...N_S}$ et d'un autre appelé « cible » et contenant les points de l'ensemble $C = \{\mathbf{c}_j\}_{j=1...N_C}$, on recherche la transformation rigide appelée T permettant d'aligner au mieux la source sur la cible.

3.2.1 Image gaussienne

La première méthode proposée pour recaler des nuages de points par rotation utilise la projection sur la sphère gaussienne [Bro84]. Cela consiste à représenter les normales comme des points sur une sphère centrée sur l'origine et de rayon unitaire. Les plans mesurés vont donc réapparaître sous forme de groupe de points sur la sphère. L'idée principale est d'échantillonner à la fois l'image gaussienne et l'espace de rotations représenté comme une sphère également, et d'évaluer les similarités entre les images gaussiennes transformées par l'ensemble des rotations. La difficulté réside dans l'échantillonnage des sphères qui doit être le plus uniforme possible et induit un effet de binarisation. HORN [Hor84] introduit l'image gaussienne étendue (*Extended Gaussian Image* - EGI) dans laquelle l'information d'aire des plans est ajoutée pour affiner la mise en correspondance. L'image gaussienne étendue complexe (*Complex Extended Gaussian Image* - CEGI) [KI91] est une autre extension qui prend également

en compte la distance des plans, ce qui permet aux auteurs de déterminer la translation alignant les deux nuages de points. Ces approches requièrent un chevauchement total des nuages de points. La dernière extension de cette méthode a été proposée par MAKADIA et coll. [Mak+06]. Elle consiste à appliquer une corrélation sphérique entre les images gaussiennes binarisées pour déterminer la meilleure rotation. Cependant, toutes ces méthodes ne peuvent mener qu'à un alignement grossier à cause de la discrétisation de la sphère.

3.2.2 ICP et ses extensions

ICP classique

La méthode ICP surpasse les méthodes vues précédemment [BM92 ; Zha94 ; CM92]. L'algorithme d'ICP établit itérativement des correspondances locales entre les points de la source et ceux de la cible, filtre ces correspondances et minimise par moindres carrés les distances entre les points de chaque paire. Si $\mathbf{m}_i$ est le point de la cible le plus proche de $\mathbf{s}_i$ (point de la source), on cherche à résoudre la minimisation suivante, au sens des moindres carrés :

$$T^* = \underset{T}{\operatorname{argmin}} \sum_{i}^{N_S} w_i ||T\mathbf{s}_i - \mathbf{m}_i||^2 , \tag{3.1}$$

avec

$$w_i = \begin{cases} 1 \text{ si } ||T\mathbf{s}_i - \mathbf{m}_i||_2 \leq D \\ 0 \text{ si } ||T\mathbf{s}_i - \mathbf{m}_i||_2 > D \end{cases} , \tag{3.2}$$

où D est le seuil permettant de filtrer les correspondances établies au préalable afin de pouvoir traiter des cas de chevauchement partiel. Cette valeur peut être adaptée manuellement ou automatiquement par analyse statistique des correspondances [Zha94].

Pour établir les correspondances $\{\mathbf{s}_i, \mathbf{m}_i\}$, l'algorithme passe par une recherche de plus proche voisin. Il s'agit d'une tâche coûteuse limitant le nombre de points pouvant être traités en un temps raisonnable. De plus, l'aspect itératif complique le paramétrage et l'implémentation en temps réel. Cependant, comme expliqué en introduction, le plus gros défaut de cette méthode vient de sa sensibilité aux minima locaux.

Beaucoup de travaux ont essayé d'améliorer la robustesse de cet algorithme [Pom+13 ; Pom+15]. Dans les paragraphes suivants, nous citons les principales extensions de cet algorithme.

Fonctions de coût alternatives

Certaines méthodes se concentrent sur la fonction de coût. Les correspondances trouvées entre les points ne peuvent pas être exactes car l'échantillonnage est différent et le capteur introduit un bruit dans les mesures. L'ICP *point à plan* a été proposé pour résoudre ce problème [CM92], [GA05]. Une approche simplifiée de cet algorithme consiste à remplacer la distance minimisée dans l'équation (3.1) par la distance entre

un point de la source et le plan local de son plus proche voisin dans la cible [Seg+09] :

$$T = \underset{T}{\operatorname{argmin}} \sum_{i}^{N_S} w_i ||\mathbf{n}_{\mathbf{m}_i} \cdot (T\mathbf{s}_i - \mathbf{m}_i)||^2 , \tag{3.3}$$

où $\mathbf{n}_{\mathbf{m}_i}$ est la normale définie au point $\mathbf{m}_i$. Il est intéressant de noter que dans l'approche initiale proposée [CM92], les points $\{\mathbf{m}_i\}_{i=1...N_S}$ ne sont plus des points extraits de C par recherche de plus proches voisins mais des points virtuels : pour chaque point $\mathbf{s}_i$, $\mathbf{m}_i$ est le point d'intersection entre la surface définie par les points de C et la normale en $\mathbf{s}_i$. Les auteurs trouvent ces points par une nouvelle minimisation itérative. Une autre extension d'ICP a été proposée par SEGAL et coll. [Seg+09]. Ils généralisent les expressions (3.1) et (3.3) en introduisant des matrices de covariance H^S et H^C définissant les degrés de liberté que l'on souhaite attribuer aux points de S et de C pendant le recalage. La nouvelle minimisation est la suivante :

$$T = \underset{T}{\operatorname{argmin}} \sum_{i} d_i^T (H_i^C + T H_i^S T^T)^{-1} d_i , \tag{3.4}$$

avec

$$d_i = ||T\mathbf{s}_i - \mathbf{m}_i||_2 . \tag{3.5}$$

SEGAL et coll. en déduisent une expression pour un recalage *plan à plan* en définissant les matrices de covariance selon l'orientation des plans locaux à chaque point $\mathbf{s}_i$ et $\mathbf{m}_i$. Cette méthode est nommée ICP généralisé.

Méthodes d'optimisation alternatives

D'autres méthodes ont pour but d'améliorer le processus de minimisation. ZHANG [Zha94] et BESL et MCKAY [BM92] utilisent une méthode de quaternions [Hor+88] qui fournit une forme fermée de la solution à l'équation (3.1). D'autres auteurs préfèrent linéariser la transformation afin d'utiliser des méthodes itératives d'optimisation de type quadratique plus rapides telles que *Gauss-Newton* [CM92 ; GA05] ou *Levenberg-Marquardt* [Cha+92 ; Fit03]. Ces dernières permettent notamment d'inclure un estimateur robuste à l'optimisation afin de rejeter les fausses correspondances de manière plus lisse. De plus, certains paramètres de la transformation peuvent être fixés de manière simple dans la minimisation selon les conditions d'acquisition (pas de rotation sur l'axe x ou y, ou pas de translation sur l'axe z notamment). YANG et coll. [Yan+13] vont plus loin en proposant la méthode Go-ICP qui permet de pallier le problème de non-convexité. Go-ICP alterne entre une phase d'ICP et une recherche *Branch-and-Bound* dans l'espace des transformations afin de sortir des minima locaux. Il a été prouvé que cet algorithme converge vers la solution optimale. Cependant, son coût computationnel est très élevé en raison des multiples recherches de plus proches voisins nécessaires à la fois dans l'algorithme d'ICP et dans les recherches de transformations. Il est difficile d'utiliser un tel algorithme sur des données de type LiDAR contenant des dizaines de milliers de points. Un autre axe d'amélioration du recalage est l'échantillonnage initial. GELFAND et coll. [Gel+03] proposent une méthode d'échantillonnage fondée sur la covariance des points d'entrée afin d'améliorer les performances d'ICP. Les algorithmes d'ICP sont largement utilisés en association avec des données odométriques dans des processus de SLAM. NÜCHTER et coll. [Nüc+07]

proposent notamment une chaîne de traitements incluant ICP pour réaliser un SLAM 6D avec des données odométriques. CADENA et coll. [Cad+16] proposent un résumé des méthodes actuelles dans ce domaine.

Mises en correspondance alternatives

D'autres auteurs s'intéressent à la mise en correspondance des points $\{\mathbf{s}_i, \mathbf{m}_i\}$. Premièrement, la correspondance peut être réalisée sur des points clés plutôt que sur tous les points du nuage. Dans le cadre du SLAM, ZHANG et SINGH [ZS14] utilisent des points clés correspondant à des points sur des plans ou sur des arêtes de la scène. Ces points clés sont extraits en utilisant le critère suivant :

$$\omega_i = \frac{1}{N_n||\mathbf{p}_i||}||(\sum_{j=1}^{N_n} \mathbf{p}_i - \mathbf{p}_j^{'})||_2 \,. \tag{3.6}$$

où $\mathbf{p}_i$ est un point testé, et $\{\mathbf{p}_j^{'}\}_{j\in[1,N_n]}$ sont les points du voisinage définis selon l'ordre d'acquisition. Les points présentant un ω_i faible sont considérés comme échantillonnant un plan tandis que les points présentant un ω_i fort sont extraits comme caractéristiques d'arêtes. Pour chaque point clé de la source, les auteurs extraient un ensemble K de points clés du même type dans son voisinage dans la cible et déduisent les caractéristiques du plan ou de l'arête selon l'analyse en composantes principales des points de K. Ils cherchent alors à minimiser la distance du point clé au modèle calculé (plan ou arête). Cependant, cet algorithme requiert un faible mouvement entre scans pour assurer un fonctionnement correct. La notion de point clé peut également être associée aux coins de la surface mesurée. Différentes méthodes permettent d'extraire ce type de points clés : Harris [HS88], ISS [Zho09] ou NARF [Ste+11]. Certaines méthodes utilisent d'autres données telles que l'intensité associée aux points pour extraire un sous-ensemble de points clés comme SIFT [Low04] ou SUSAN [SB97]. Deuxièmement, une analyse de voisinage de ces points clés peut permettre de mettre ces points en correspondance en fonction, non plus de leur distance spatiale, mais de leurs caractéristiques locales [Hol+15]. La distance entre deux points clés peut alors être évaluée par la distance entre descripteurs locaux ou une combinaison de descripteurs locaux tels que FPFH (*Fast Point Feature Histograms*) [Rus+09], Spin-Images [JH99] ou encore SHOT (*Signature of Histograms of Orientation*) [Tom+10]. Une autre façon d'obtenir des descripteurs peut être d'inclure l'information d'intensité liée au capteur [Han+13 ; Wan+16]. Ce type de méthodes est rapide mais sa précision est faible, elle sont généralement utilisées pour déduire un alignement initial permettant d'optimiser les performances d'ICP. Comme les fausses correspondances entraînent une forte erreur de convergence, certains auteurs se sont efforcés de trouver des heuristiques fortes afin de filtrer un maximum de paires [Hol+15]. ZHOU et coll. [Zho+16] vont plus loin avec leur méthode FGR (*Fast Global Registration*) qui intègre un M-estimateur à la minimisation afin de rejeter les mauvaises correspondances de manière plus lisse et plus stable qu'avec des heuristiques. Cependant, toutes les méthodes fondées sur des descripteurs locaux requièrent des détails géométriques et peuvent mener à de fausses convergences dans le cas de scènes structurées telles que rencontrées en environnement intérieur.

3.2.3 *RANdom SAmple Consensus*

Une autre façon d'obtenir un alignement initial est d'utiliser une méthode de type RANSAC (*RANdom SAmple Consensus*) [Fis+81]. La méthode classique consiste à extraire des triplés de points aléatoirement dans chaque nuage (S et C), de déduire la transformation permettant d'aligner le triplé de la source sur le triplé de la cible puis de sélectionner la transformation permettant d'obtenir un chevauchement maximal. DROR et coll. [Dro+08] proposent la méthode 4PCS (*4-Points Congruent Sets*) qui rend le procédé plus robuste en utilisant des ensembles de 4 points et en les filtrant par certaines heuristiques avant la mise en correspondance. Ce dernier algorithme a été amélioré par MELLADO et coll. [Mel+14] qui l'ont rendu plus rapide et utilisable sur de larges données. L'outil d'évaluation du chevauchement utilisé dans ces méthodes est la valeur LCP (*Largest Common Pointset*) qui prend en paramètre un rayon r_{LCP}. Elle correspond au pourcentage de points de la source ayant au moins un voisin de la cible dans la boule de rayon r_{LCP}. Des garanties de convergence ont été démontrées, cependant, le temps de calcul est élevé si on cherche à obtenir une transformation optimale. Cet algorithme est plus adapté à un objectif d'approximation pour initialiser une étape d'ICP notamment.

3.2.4 Alignement de fonctions de densité

Une approche différente consiste à construire une fonction de probabilité sur le nuage cible et à maximiser la probabilité de localisation des points de la source après transformation à partir de la distribution calculée. La méthode la plus connue est NDT (*Normal Distribution Transform*) [Mag+07 ; Mag09], dans laquelle les auteurs divisent l'espace avec une grille régulière et ajustent une loi de probabilité mélangeant une loi uniforme et une loi normale dans chaque cellule afin de réduire l'impact des points aberrants. La fonction de probabilité est alors continue par morceaux. Bien que cette méthode soit plus rapide et demande moins d'espace mémoire que ICP, elle souffre aussi du problème de minima locaux et ses performances dépendent de la taille des cellules qui est une valeur difficile à paramétrer par l'utilisateur. De plus, le fonctionnement interne du LiDAR induit des densités de points très différentes selon les points de vue ce qui peut entraîner de mauvaises convergences.

3.2.5 Utilisation de plans

D'autres auteurs se sont intéressés aux propriétés des scènes structurées. PATHAK et coll. [PB10 ; Pat+09] ont focalisé leur étude sur la mise en correspondance de plans. Des plans sont extraits du nuage de points et leurs paramètres (normale et distance à l'origine) sont calculés avec une estimation d'erreur sous forme de covariance gaussienne. A l'instar de l'ICP généralisé, cette covariance peut permettre d'orienter le recalage dans certaines directions de l'espace (voir équation (3.4)).

$$T = \underset{T}{\operatorname{argmin}} \sum_{i}^{N_p} d_i^T (H_i^Q + T H_i^P T^T)^{-1} d_i \quad \text{avec} \quad d_i = \begin{pmatrix} \mathbf{n}_i^S \\ d_i^S \end{pmatrix} - T \begin{pmatrix} \mathbf{n}_i^C \\ d_i^C \end{pmatrix}^T . \tag{3.7}$$

La transformation rigide peut alors être estimée par une maximisation pondérée de la vraisemblance pour chaque paire de plans, les poids étant associés à l'erreur d'estimation des paramètres du plan. On recherche la rotation en étudiant les normales des plans de chaque paire tandis que la translation est déduite des valeurs de distances à l'origine. L'étape la plus compliquée est l'appariement des plans. Cette étape demande beaucoup d'heuristiques dont certaines utilisent des données odométriques et d'autres requièrent un échantillonnage identique sur les deux plans mis en correspondance. De plus, l'extraction de plans est une tâche qui demande un paramétrage fiable ce qui peut s'avérer complexe dans des scènes d'intérieur.

Les méthodes décrites dans cet état de l'art se heurtent à de nombreux obstacles caractéristiques d'acquisitions LiDAR en intérieur : les ouvertures peuvent produire de larges régions mesurées en dehors de la zone de chevauchement, l'anisotropie de l'échantillonnage et le manque de spécificités locales peuvent corrompre la précision des algorithmes locaux et les nuages peuvent être très denses ce qui limite les temps de calcul acceptables des algorithmes. Après avoir observé les limites des algorithmes de l'état de l'art, nous souhaitons introduire une nouvelle méthode de recalage automatique qui répond aux objectifs suivants :

Objectifs

1. Ne pas utiliser de donnée odométrique ni de cible physique.
2. Pallier le problème des descripteurs locaux peu significatifs.
3. Résoudre le recalage en cas de faible chevauchement.
4. Résoudre le recalage quelle que soit la pose initiale des scans.
5. Obtenir une estimation précise du recalage sans besoin d'affinement.
6. Conserver des temps de calcul faibles.

3.3 Approche proposée : SSFR

Nous nous sommes intéressés aux environnements structurés, c'est pourquoi la méthode proposée est appelée SSFR (*Structured Scene Features based Registration*). La scène doit contenir des plans et les murs principaux de la cible doivent avoir une équivalence dans la source. L'algorithme développé estime la rotation et la translation à travers deux étapes distinctes. Il est présenté dans le schéma simplifié de la chaîne de traitements figure 3.1. Tout d'abord, les normales principales des nuages de points sont extraites sur la sphère gaussienne. Ensuite, un ensemble de rotations possibles est calculé et, pour chacune d'elles, une translation optimale est associée. Enfin, les transformations résultantes sont évaluées afin de sélectionner la meilleure en utilisant une estimation du chevauchement entre la source et la cible après recalage. L'approche complète est également décrite au travers de l'algorithme 2.

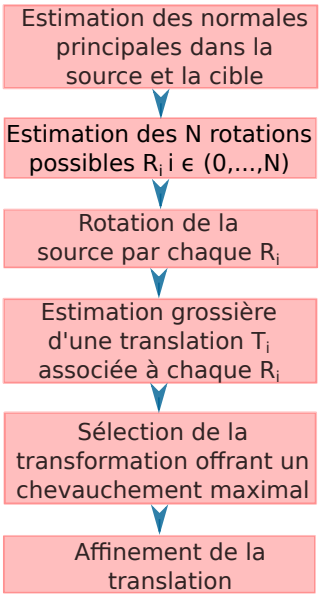


Figure 3.1.: Chaîne de traitements de l'algorithme SSFR.

3.3.1 Recherche de la rotation

Pré-traitement

La première étape consiste à calculer les vecteurs normaux unitaires en chaque point de la source et de la cible. Nous utilisons la méthode ACP telle que décrite par HOPPE et coll. [Hop+92]. Puis, les nuages de points sont sous-échantillonnés uniformément : une grille 3D est appliquée sur le nuage de points et un point par cellule est conservé.

Projection sur la sphère gaussienne

Ensuite, les normales des deux nuages sont représentées sur la sphère gaussienne. Un exemple d'image gaussienne est donné figure 3.2 pour un nuage de points acquis dans une salle polyvalente par un capteur Leica P20 (voir annexe B pour plus d'informations sur ce capteur). Dans cette représentation, l'orientation des principaux plans est mise en évidence par des zones de forte densité de points, mais l'information de distance est perdue. Si on étudie un nuage de points S qui subit une rotation R et une translation T et si l'application qui permet de transformer un nuage de points en son image gaussienne s'appelle $G\{\cdot\}$, une propriété majeure de l'image gaussienne [KI91] peut être formulée ainsi :

$$R \cdot G\{S\} = G\{R \cdot S\} \qquad T \cdot G\{S\} = G\{S\} \tag{3.8}$$

La méthode est fondée sur l'hypothèse que l'environnement est composé d'un nombre raisonnable de plans que l'on peut détecter sur l'image gaussienne et mettre en correspondance entre scans. Pour ce faire, on va chercher à isoler les groupes de normales correspondant aux murs principaux et à extraire les modes de ces groupes.

Sélection des normales principales

Un algorithme de *mean-shift* pourrait être utilisé directement sur les points de l'image gaussienne. Cependant, cet algorithme est lent pour le nombre de points traités. Les points sont donc pré-filtrés selon un critère de densité locale afin d'accélérer le processus. En effet, pour chaque point $\mathbf{p}_i$, on extrait ses N_i voisins plus proches $\{\mathbf{p}_j\}_{j=1:N_i(r)}$ contenus dans un rayon r_d. Ce rayon dépend de la précision du capteur sur les plans et de l'estimation de normales. L'utilisateur doit paramétrer l'erreur angulaire maximum tolérable sur les normales des plans, nommée Δ_θ. Le rayon r_d est alors :

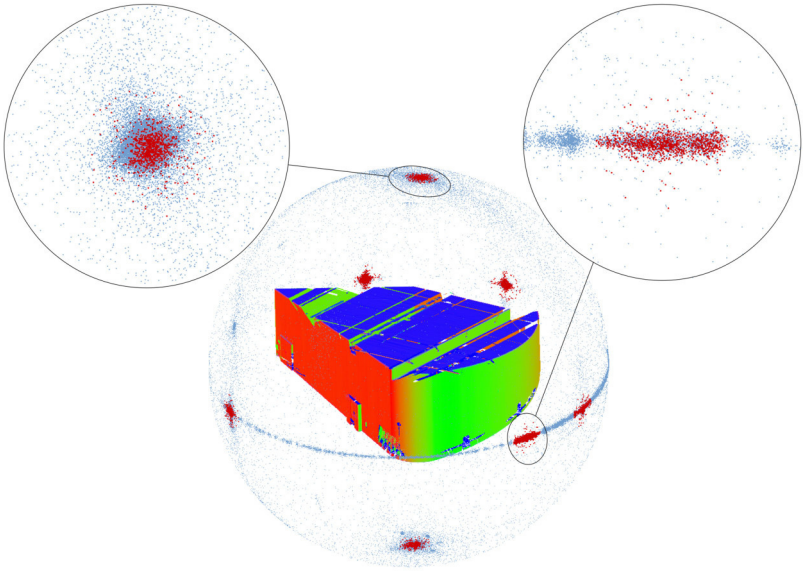
$$r_d = \sqrt{2(1 - \cos(\Delta_\theta))}\,, \tag{3.9}$$

et la densité est estimée au point p_i par la formule suivante :

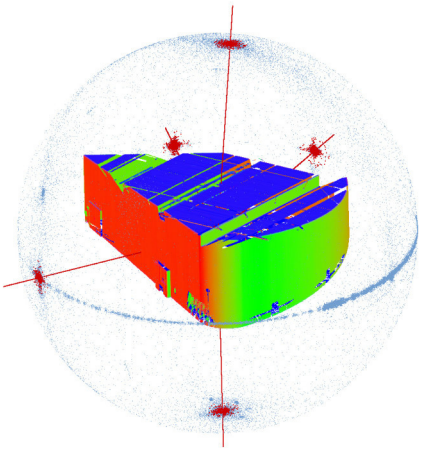
$$d_i = \sum_j e^{(\mathbf{p}_j - \mathbf{p}_i)^2/(r_d/2)^2}\,. \tag{3.10}$$

Afin d'obtenir un sous-ensemble de points pour l'algorithme *mean-shift*, on sélectionne un maximum de N_m points de plus forte densité (dont les voisinages sont distincts), ainsi que leurs points voisins. On suppose donc qu'il existe au moins trois plans principaux non coplanaires dans la zone de chevauchement des nuages et que ceux-ci sont représentés par trois des zones extraites précédemment. N_m est paramétré à 8 dans tous les tests. Afin de rendre l'algorithme plus efficace, ces zones peuvent ensuite être sous- échantillonnées aléatoirement afin d'obtenir un maximum de N_{max} points par zone à classifier par *mean-shift*. Ce sous-échantillonnage est évalué sur le graphique de la figure 3.11 qui confirme que la perte de précision est faible mais que le gain de temps est important. Un exemple de pré-filtrage par densité et de déduction de mode est donné figure 3.2. Afin de pouvoir exclure les murs courbes et les faux groupes de l'analyse, un filtre est ajouté sur la forme des groupes détectés. En effet, les murs plans sont représentés par un point sur l'image gaussienne (voir figure 3.2a zoom de gauche), tandis que les murs courbes sont représentés par un arc sur l'image gaussienne (voir figure 3.2a zoom de droite). Une ACP est donc réalisée sur un voisinage plus large (un angle de $15°$ est utilisé dans tous les tests de cette étude) afin d'obtenir les valeurs propres qui représentent les variances sur les axes principaux, $\{\lambda_0, \lambda_1, \lambda_2\}$ rangées dans l'ordre croissant. Le critère d'exclusion est le suivant :

$$\frac{\lambda_2}{\lambda_1} < 3\,. \tag{3.11}$$



(a) Avant filtrage des groupes.



(b) Après filtrage des groupes et détection des modes.

Figure 3.2.: Nuage de points 3D (colorisé par ses normales), son image gaussienne (bleue), les groupes détectés avec leurs modes respectifs (rouge).

Enfin, l'algorithme de *mean-shift* peut être appliqué avec un noyau d'Epanechnikov [Epa69] et un rayon égal à $2r$. Les modes obtenus sont les normales principales du nuage étudié $\{\mathbf{n}_i\}_{i=1...N_g}$ (N_g étant le nombre de groupes détectés, $N_g \leq N_m$). La différence entre un point détecté par filtrage par densité et après *mean-shift* est représentée figure 3.3. On note que l'algorithme de regroupement est capable d'obtenir un mode de manière précise, même en présence de bruit dû à une irrégularité sur le plan.

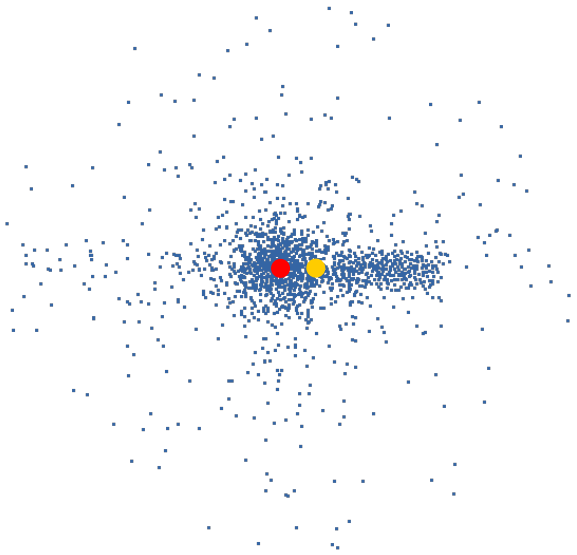


Figure 3.3.: Détection d'un mode dans un groupe par simple filtrage de densité (jaune) et par *mean-shift* (rouge).

Appariement et mise en correspondance

Une fois les ensembles de normales principales obtenus, toutes les paires $(\vec{n_i}, \vec{n_j})_S$, $(\vec{n_k}, \vec{n_l})_C$ avec $i \neq j$ et $k \neq l$ sont formées avec leurs permutations respectives dans la source (indice S) et dans la cible (indice C). Les normales d'une paire ne peuvent pas être opposées. Le nombre maximum de paires dans un nuage de points est $N_{paires} = N_g!/(N_g - 2)!$. Le but est de lier une paire de la source avec une paire de la cible. Toutes les correspondances possibles entre paires sont formées puis filtrées : les paires dont les angles $\widehat{(\vec{n_i}, \vec{n_j})}_S$ et $\widehat{(\vec{n_k}, \vec{n_l})}_C$ sont différents sont rejetées. Pour trouver la meilleure correspondance, toutes les combinaisons sont testées et après l'estimation de la translation, les résultats des recalages totaux sont comparés comme expliqué dans la section 3.3.3. Le nombre de tests est inférieur à : $M = N^S_{paires} \times N^C_{paires}$.

Déduction de la rotation

Pour estimer la rotation R entre les paires mises en correspondance, nous utilisons l'algorithme SVD (*Singular Value Decomposition*) [SHR17] sur les paires de vecteurs formées.

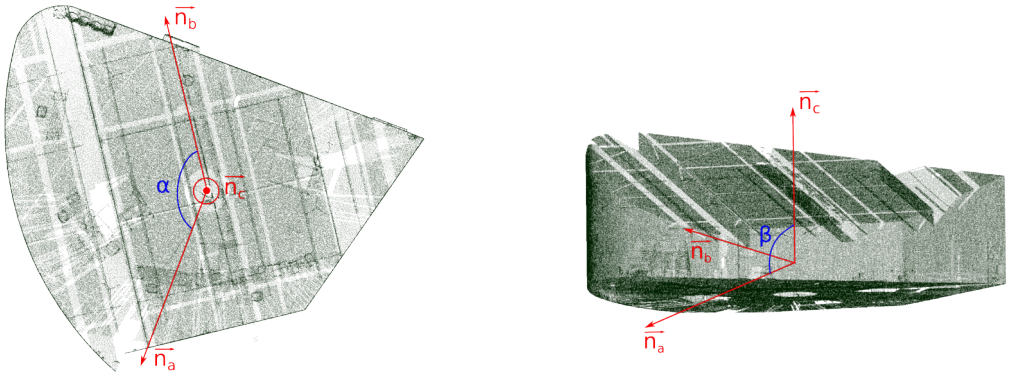
3.3.2 Recherche de la translation

Pour chaque rotation possible trouvée, une translation est déduite d'une analyse d'histogrammes de points 1D telle que décrite dans la suite de cette section.

Sélection d'axes de translation

Tout d'abord, les axes de translation sont définis. Trois normales principales $\mathbf{n}_a$, $\mathbf{n}_b$, $\mathbf{n}_c$, sont extraites des modes détectés précédemment dans la cible (voir section 3.3.1). Les deux premières normales $\mathbf{n}_a$ et $\mathbf{n}_b$ correspondent à celles utilisées pour retrouver la rotation dans l'étape précédente. La troisième normale est retrouvée après avoir appliqué la rotation au nuage source : elle correspond au mode de la cible n'appar-

tenant pas au plan formé par $\{\mathbf{n}_a, \mathbf{n}_b\}$ et présentant l'alignement le plus important avec un mode de la source. Ces normales définissent les axes de translation a, b, c. Un exemple de sélection d'axes est donné sur la figure 3.4.



(a) Vue de dessus **(b)** Vue de côté

Figure 3.4.: Sélection des axes de translation dans le scan d'une pièce. Ils sont extraits d'une analyse des normales de la scène. Scans issu de `DS2-L`.

Estimation de la translation

La translation est ensuite calculée à travers deux étapes : une estimation grossière et un affinement. Dans chacune des étapes, les points de la cible et de la source après rotation sont projetés sur les différents axes de translation déterminés précédemment et des histogrammes sont construits. La mise en correspondance de ces histogrammes indique la translation à effectuer sur les différents axes pour aligner les nuages de points. L'estimation de la translation est détaillée dans l'algorithme 1 dont une explication est donnée ci-après.

Algorithme 1 : *Translation* : estimation de la translation entre deux nuages.

Entrée: RS, C, a, b, c, $LargeurClasse$ *//RS est la source après rotation*

1: **pour tout** $w \in \{a, b, c\}$ **faire**
2: Filtrer RS et C et projeter les points sur u
3: Construire les histogrammes $hist_w^{RS}$, $hist_w^C$
4: Filtrer les histogrammes
5: Calculer la fonction de corrélation f entre $hist_w^{RS}$ et $hist_w^C$
6: $i \leftarrow argmax(f)$
7: $\Delta w \leftarrow (i + 0.5) \times LargeurClasse$
8: **fin pour**
9: $\Delta x \leftarrow \Delta a$
10: $\Delta y \leftarrow \frac{\Delta b - \Delta x \times \cos(\alpha)}{\sin(\alpha)}$
11: $\Delta z \leftarrow \frac{\frac{\Delta c - \Delta x \times \cos(\beta)}{\sin(\beta)} - \Delta y \times \cos(\gamma)}{\sin(\gamma)}$
12: **retourne** Matrice de translation T contenant les éléments Δx, Δy et Δz

Dans un premier temps, les nuages de points de la source et de la cible sont filtrés, pour chaque axe, afin de ne garder que les points qui ont une normale parallèle à l'axe étudié. Dans un second temps, ces points sont projetés sur l'axe et un histogramme est construit à partir de ces projections. Un exemple d'application de cette procédure est donné figure 3.5.

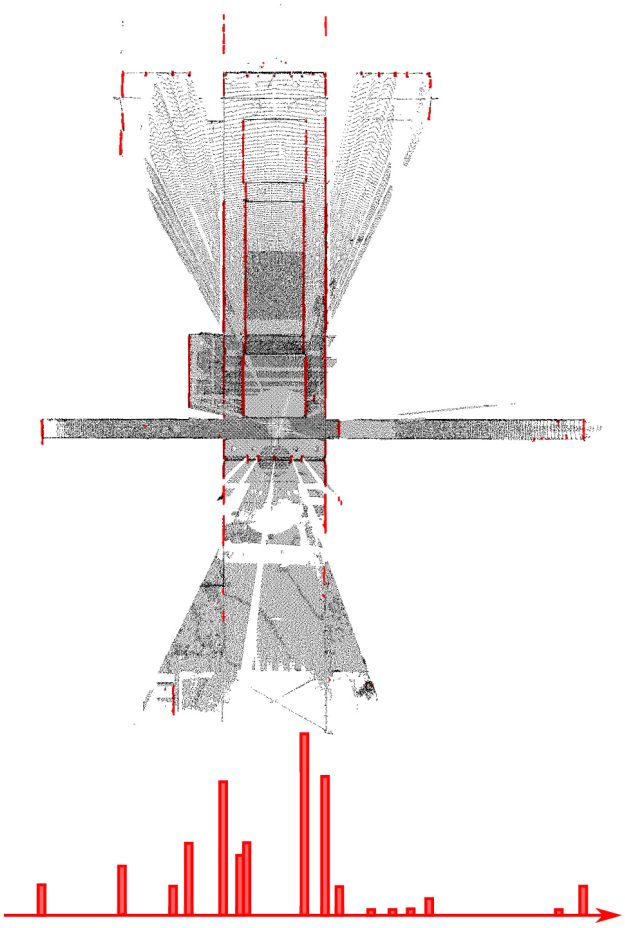


Figure 3.5.: Exemple de construction d'histogramme sur un axe de translation à partir d'un scan LiDAR acquis par un dispositif Leica P20. L'axe étudié est représenté par une flèche sur laquelle sont représentées les classes de l'histogramme. Ces dernières sont construites à partir des points dont les normales sont parallèles à l'axe (en rouge sur le nuage).

La translation sur un axe w est déduite du maximum de corrélation entre les histogrammes de la source et de la cible sur w. En effet, ce maximum correspond à la translation Δw à appliquer sur cet axe pour aligner les murs au mieux. Si les axes a, b et c ne sont pas perpendiculaires, le mouvement estimé pour aligner les murs sur un axe est corrélé avec les autres. Afin de décorréler ces mouvements, on cherche à les exprimer selon un repère orthogonal. On appelle les axes de ce nouveau repère d, e et f. Ils sont représentés par les vecteurs unitaires $\mathbf{u}_d$, $\mathbf{u}_e$ et $\mathbf{u}_f$ définis équation (3.12).

$$\begin{aligned} \mathbf{u}_d &= \mathbf{n}_a \\ \mathbf{u}_f &= \mathbf{n}_a \times \mathbf{n}_b \\ \mathbf{u}_e &= \mathbf{n}_f \times \mathbf{u}_a \end{aligned} \tag{3.12}$$

On cherche donc une relation entre les mouvements Δa, Δb, Δc et les mouvements Δd, Δe, Δf. La figure 3.6 présente une scène schématique de deux murs vus de dessus et illustre la contribution du mouvement sur l'axe a au mouvement sur l'axe b. Les formules (3.13) et (3.14) permettent de calculer Δd et Δe avec α, l'angle entre les axes a et b. La formule (3.15) permet de déduire Δf avec β, l'angle entre les axes a et c et γ, l'angle entre les axes b et c. Les angles α, β et γ sont non-signés et contenus dans l'intervalle $[0, \pi]$.

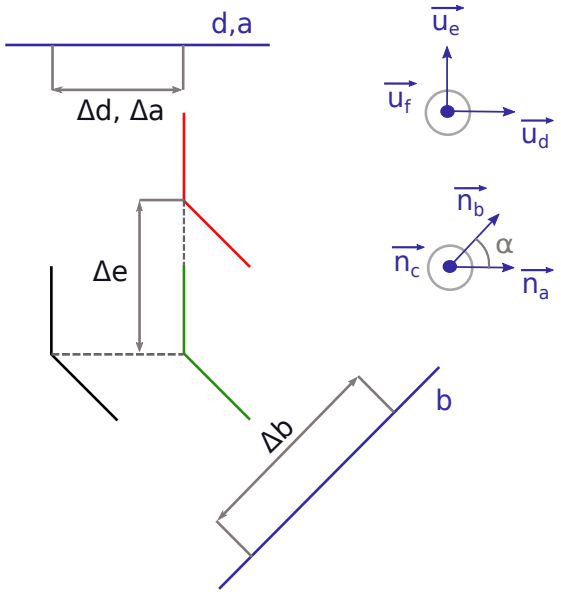


Figure 3.6.: Scène schématique représentant deux murs vus de dessus depuis une position initiale (en noir) jusqu'à une position finale (en rouge). En vert, les murs sont représentés après leur translation le long de l'axe a.

$$\Delta d = \Delta a\,, \tag{3.13}$$

$$\Delta e = \frac{\Delta b - \Delta d \times \cos(\alpha)}{\sin(\alpha)}\,, \tag{3.14}$$

$$\Delta f = \frac{\frac{\Delta c - \Delta d \times \cos(\beta)}{\sin(\beta)} - \Delta e \times \cos(\gamma)}{\sin(\gamma)}\,, \tag{3.15}$$

Le vecteur de translation final $\mathbf{t}$ est donc égal à :

$$\mathbf{t} = \Delta d\,\mathbf{u}_d + \Delta e\,\mathbf{u}_e + \Delta f\,\mathbf{u}_f\,, \tag{3.16}$$

et la matrice de transformation résultante est :

$$T := \begin{pmatrix} \ddots & & \vdots \\ & R & \mathbf{t}\,. \\ & & \ddots \quad \vdots \end{pmatrix} \tag{3.17}$$

Dans l'étape d'estimation grossière, les histogrammes sont construits avec des classes larges afin de mener rapidement à une translation approximative. A noter que, dans ce cas, les histogrammes sont seuillés avant la corrélation pour limiter les mauvaises corrélations dues à des échantillonnages trop différents entre la source et la cible. Sans cela, les murs contenant le plus de points auraient tendance à être mis en correspondance aux dépens de l'appariement des autres plans. Pour ce faire, la moyenne m et l'écart type s_d des hauteurs des classes sont calculés. Les classes présentant un nombre de points dont l'écart à m est supérieur à $3s_d$ sont seuillées à la valeur $m + 3s_d$. Dans la deuxième étape, les classes des histogrammes sont réduites et la corrélation est recalculée dans l'intervalle d'une classe de l'étape précédente afin d'affiner son estimation. La finesse des classes dans cette étape implique que les points des plans sont divisés sur plusieurs classes. Un filtre moyenneur est alors employé pour mettre en évidence la position des murs avec précision. Un exemple de moyennage d'histogramme est illustré figure 3.7.

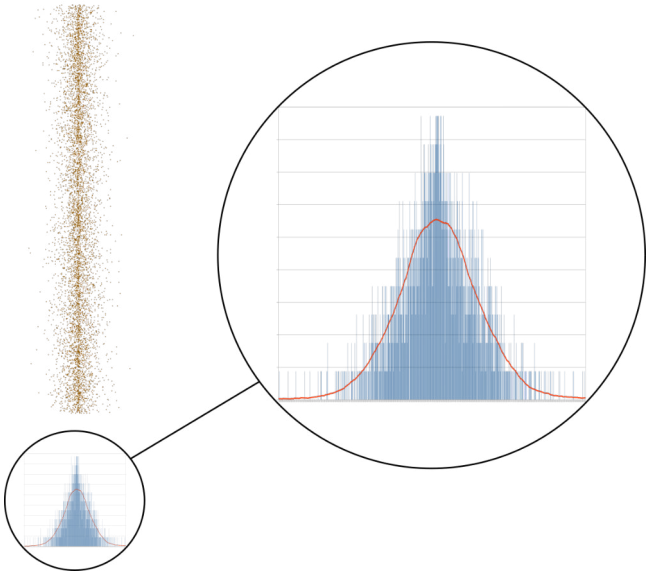


Figure 3.7.: Exemple d'histogramme construit à partir d'un mur échantillonné par des points. Les points du murs sont représentés en haut à gauche de l'image, l'histogramme est représenté en bleu et la courbe orange représente l'histogramme moyenné.

3.3.3 Sélection de la transformation

Toutes les rotations possibles pour aligner des paires de normales sont testées (voir section 3.3.1). Les translations sont estimées grossièrement (voir section 3.3.2) et les transformations totales sont comparées afin de sélectionner la meilleure. Nous choisissons d'évaluer le chevauchement résultant de la transformation de la source sur la cible en utilisant un estimateur qu'on nomme ω. Il est calculé à partir de la distance $d(p_i, m_i)$ entre chaque point de la source p_i et son voisin le plus proche dans la cible m_i et avec la résolution du nuage cible δ_c de la façon suivante :

$$\omega = \sum_{p_i \in S} k_i e^{-(d(p_i,m_i)/\delta_c)} \quad \text{avec} \quad k_i = \begin{cases} 1 & si \quad d(p_i, qm_i) < \delta_c \\ 0 & sinon. \end{cases} . \tag{3.18}$$

δ_c est un paramètre à fixer par l'utilisateur, il représente le rayon du voisinage local d'un point de la source transformée dans lequel on cherche des voisins de la cible. La transformation correspondant à la valeur de ω maximale est sélectionnée. Cette étape correspond aux lignes 16, 17 et 21 de l'algorithme 2. Après sélection de la meilleure transformation, la translation est affinée comme décrit dans la section 3.3.2 et la transformation finale estimée est déduite.

Algorithme 2 : *SSFR* : algorithme complet : estimation de transformation rigide.

Entrée: S, C, r, Δ_θ, $LargeurClasse$

//r est le rayon de voisinage utilisé pour estimer les normales

//Δ_θ est l'erreur angulaire estimée sur le calcul d'une normale sur un plan

//S est le nuage source et C est le nuage cible

1: Estimer les normales avec le rayon r
2: Sous-échantillonner S et C
3: Calculer les images gaussiennes G_S et G_C à partir de S et C
4: Calculer la densité en chaque point de G_S et G_C avec Δ_θ
5: Filtrer G_S et G_C pour garder un maximum de N_m des régions les plus denses $\{A_{S,i}\}_{i=1,\dots,N_m}$ et $\{A_{C,j}\}_{j=1,\dots,N_m}$
6: $M_S \leftarrow$ *mean-shift*(A_S) et $M_C \leftarrow$ *mean-shift*(A_C)
7: Créer les ensembles de paires (P_S et P_C) d'éléments de M_S et de M_C
8: Initialiser les listes V et Ω : $V \leftarrow [\,]$ et $\Omega \leftarrow [\,]$
9: **pour tout** $P_{S,i} \in P_S$ **faire**
10: **pour tout** $P_{C,j} \in P_C$ **faire**
11: **si** $P_{S,i}.angle - P_{C,j}.angle < \epsilon$ **alors**
12: Estimer la rotation R pour aligner $P_{S,i}$ sur $P_{C,j}$
13: Sélectionner les axes $M_{C,a}$, $M_{C,b}$, $M_{C,c}$
14: $T \leftarrow Translation\left((R \cdot S), C, M_{C,a}, M_{C,b}, M_{C,c}, LargeurClasse\right)$ *// Algo. 1*
15: Déduire la transformation totale $T_{tot} \leftarrow T \cdot R$
16: Calculer l'estimateur de chevauchement ω
17: $\Omega = [\Omega, \omega]$ et $V = [V, T_{tot}]$
18: **fin si**
19: **fin pour**
20: **fin pour**
21: $i \leftarrow \mathrm{argmax}\,(\Omega)$
22: $T \leftarrow Translation(V_i \cdot source, cible, M_{C,a}, M_{C,b}, M_{C,c}, LargeurClasse/100)$
23: **retourne** Matrice de transformation $T \cdot V_i$

3.4 Evaluation

Pour évaluer notre méthode, nous utilisons quatre jeux de données d'acquisitions LiDAR correspondant à différents contextes d'application. Ces jeux de données sont décrits précisément dans l'annexe B.

Nous comparons notre algorithme à six algorithmes principaux de l'état de l'art : ICP *point à point*, ICP *point à plan* , NDT, FGR, Super4PCS et Go-ICP. Les implémentations de la librairie PCL (Point Cloud Library) ont été utilisées pour tester les algorithmes ICP et NDT. FGR, Go-ICP et Super4PCS ont été testés en utilisant les codes en ligne fournis par les auteurs. Nous évaluons les algorithmes de recalage selon différents critères :

- l'erreur moyenne entre les points de la source après transformation par vérité terrain et après recalage ;
- l'erreur de translation et l'erreur de rotation : l'erreur de rotation est représentée par la combinaison de trois angles de rotation autour des axes de référence du capteur, on calcule la moyenne de ces trois angles ;
- le taux de réussite est aussi étudié : il est défini comme le pourcentage de recalages ayant une erreur inférieure à un seuil qui dépend de la largeur du nuage ;
- le temps de calcul des algorithmes (processeur : 4 cœurs, Intel i5–7440HQ, $2.80\,\mathrm{GHz}$, RAM 16 Go)

Avant l'application de chaque algorithme, les nuages de points sont échantillonnés uniformément, les normales sont calculées avec le même paramètre de rayon d'action et les points aberrants sont retirés. Les temps de calcul ne concernent que les étapes de recalage.

3.4.1 Résultats

3.4.2 Modèle synthétique

Tout d'abord, un modèle synthétique est testé. Il s'agit d'une surface polygonale formant un coin ($0.6 \times 0.5 \times 0.7\,\mathrm{mm}$), échantillonnée de façon à imiter le fonctionnement d'un LiDAR placé en son centre avec un pas angulaire de $3.6°$. Un bruit gaussien d'écart type $5\,\mathrm{mm}$ est ajouté au modèle. Ce dernier contient finalement $10\,000$ points et est illustré figure 3.8. Il est déplacé manuellement pour créer un deuxième nuage de points à recaler. Les deux nuages ont donc un chevauchement total. Les transformations effectuées sont des rotations suivant l'axe z d'angle croissant. Les normales sont calculées avec un rayon de voisinage de $5\,\mathrm{cm}$ en chaque point. Aucun sous-échantillonnage n'est appliqué. Le paramétrage des différents algorithmes est résumé dans le tableau 3.1.

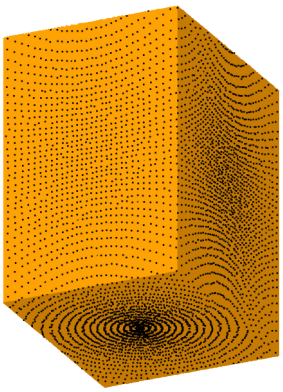
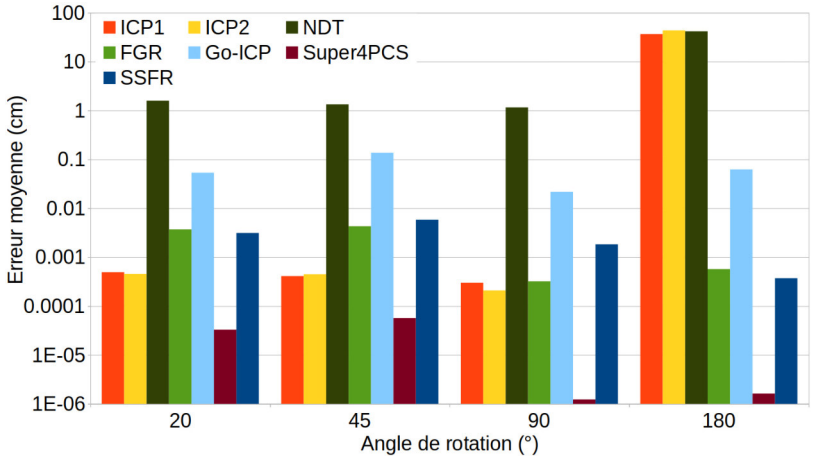


Figure 3.8.: Modèle synthétique utilisé pour évaluer les algorithmes de recalage. Coin formé par trois plans ($0.6 \times 0.5 \times 0.7\,\mathrm{mm}$) et échantillonné par $10\,000$ points.

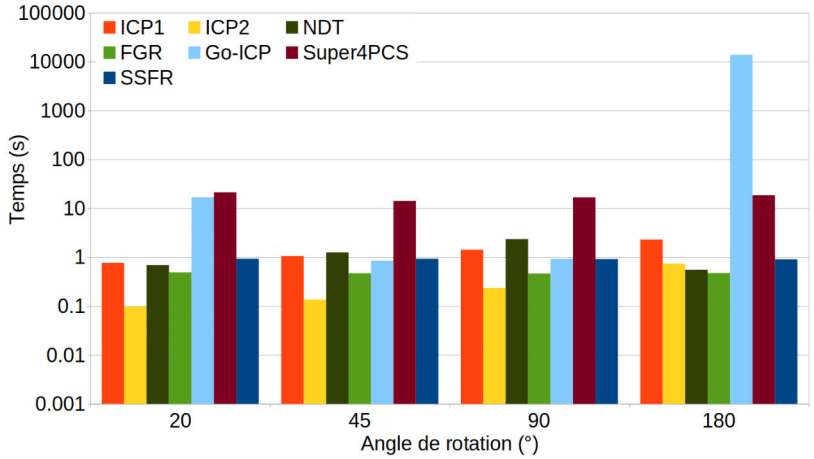
Tableau 3.1.: Paramètrage des algorithmes de recalage évalués sur le modèle synthétique.

Algorithme	**Paramètre**	**Valeur**
ICP1	Distance de rejet	$100\,\mathrm{m}$
ICP2	Distance de rejet	$100\,\mathrm{m}$
NDT	Largeur de grille	$10\,\mathrm{cm}$
FGR	Rayon FPFH	$20\,\mathrm{cm}$
	Facteur de division	1.1
	Rayon de noyau minimum	$1\,\mathrm{cm}$
	Nombre de correspondances	$10\,000$
Super4PCS	Rayon LCP	$1\,\mathrm{cm}$
	Nombre de points conservés	1500
	Chevauchement estimé	0.9
Go-ICP	Angle autorisé	$[-\pi, \pi]^3$
	Translation autorisée	$[-0.5, +0.5]^3$
	Pourcentage de rejet	$0\,\%$
	Nombre de nœuds par dimension de l'espace de transformation	100
SSFR	Erreur angulaire estimée (Δ_θ)	$1°$
	Largeur de classe avant sélection	$1\,\mathrm{cm}$
	Nombre de points maximum pour *mean-shift*	2000

L'erreur RMS et le temps de calcul sont étudiés pour toutes les méthodes et sont reportés dans la figure 3.9. Dans ce cas, on remarque que notre algorithme n'atteint pas la précision des méthodes fondées sur ICP et Super4PCS pour les trois premières rotations. Cela s'explique par le fait que le chevauchement est total et que les points de la cible et de la source sont parfaitement superposés après recalage. Ce sont des conditions qu'on ne retrouve pas dans des données réelles. L'erreur résultante du recalage par SSFR reste faible (de l'ordre du centième de millimètre) et le temps de calcul est limité. On peut remarquer que le temps de calcul de notre algorithme et sa convergence ne dépendent pas du mouvement initial entre les deux nuages contrairement aux méthodes itératives telles qu'ICP ou NDT qui ne convergent pas pour une rotation de 180°. Super4PCS obtient les meilleures précisions sur cet exemple. Cependant, son temps de calcul est plus de 10 fois supérieur aux autres méthodes (excepté Go-ICP). Go-ICP obtient de bons résultats de précision quelle que



(a) Erreur moyenne



(b) Temps de calcul

Figure 3.9.: Evaluation des algorithmes sur un coin synthétique composé de trois plans échantillonnés par 10 000 points en simulant un scanner LiDAR. **(a)** Erreur moyenne entre points de la source après recalage par les transformations réelles et les transformations estimées. **(b)** Temps de calcul. Comparaison de notre algorithme avec ICP1 (*point à point*), ICP2 (*point à plan*), NDT, FGR, Go-ICP et Super4PCS. Les échelles sont logarithmiques.

soit la rotation appliquée (de l'ordre du millimètre) mais son temps de calcul explose lorsque la première étape d'ICP ne satisfait pas les critères de convergence requis et que la méthode *Branch-and-Bound* est appliquée. Les temps de calculs de SSFR et Go-ICP pour $10\,000$ points suggèrent que leur utilisation n'est pas recommandable pour de larges données contenant différents objets complexes, telles que des scans d'intérieur de bâtiment.

3.4.3 Environnement intérieur dynamique

Nous évaluons à présent notre algorithme sur les données `DS1-H` (voir annexe B.1), pour tester sa robustesse dans le contexte de reconstruction de scènes en environnement intérieur. Le dynamisme évoqué provient du fait que des éléments sont déplacés dans l'environnement entre chaque acquisition de scan, ce qui complique le travail des algorithmes fondés sur des appariements locaux.

Evaluation interne

Nous souhaitons évaluer l'impact des paramètres de notre algorithme sur les résultats. Premièrement, nous faisons varier l'erreur angulaire estimée sur les normales Δ_θ ; les résultats sont illustrés sur le graphique de la figure 3.10. Pour cette expérience, le nombre de points maximum par groupe de normales N_{max} n'est pas limité afin d'observer uniquement l'impact de Δ_θ. Ce graphique montre que Δ_θ a un faible impact sur la précision de l'algorithme à partir d'un seuil (dépendant de la précision du capteur et de la méthode de calcul de normales) mais a un fort impact sur les temps de calcul qui sont multipliés par 10 entre $\Delta_\theta = 0.05°$ et $\Delta_\theta = 10°$. Δ_θ peut donc être défini expérimentalement, ou avec un *a priori* sur la qualité des normales, pour un jeu de données issu d'un même capteur afin d'obtenir le meilleur compromis précision/temps. En outre, si Δ_θ est trop faible, le filtre par densité ne peut pas être exécuté et l'algorithme ne va pas obtenir de résultat.

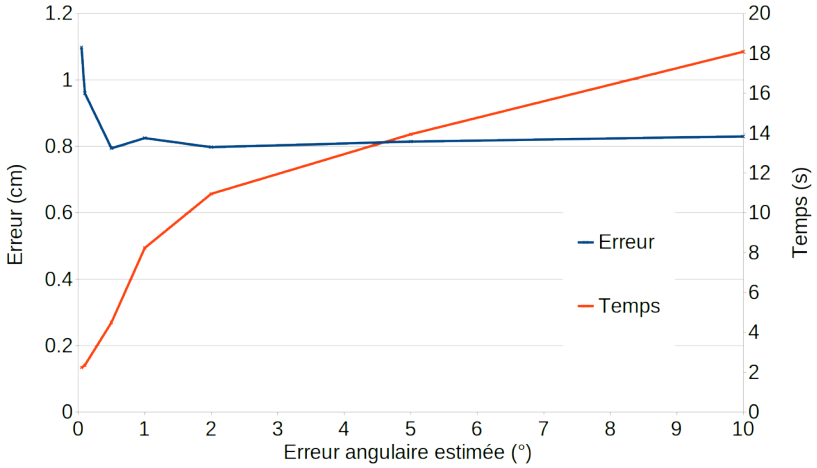


Figure 3.10.: Evaluation des résultats de notre algorithme SSFR (précision et temps de calcul) en fonction du paramètre d'erreur angulaire estimée Δ_θ.

Deuxièmement, nous évaluons le paramètre N_{max} qui limite le nombre de points par groupe de normales détecté afin d'accélérer l'algorithme. Δ_θ est fixé à $0.5°$. Les résultats de cette évaluation sont présentés sur le graphique de la figure 3.11. Sur cette figure, on remarque que cette limitation permet un gain de temps significatif et n'altère que très peu la précision jusqu'à un certain seuil (environ 50 points).

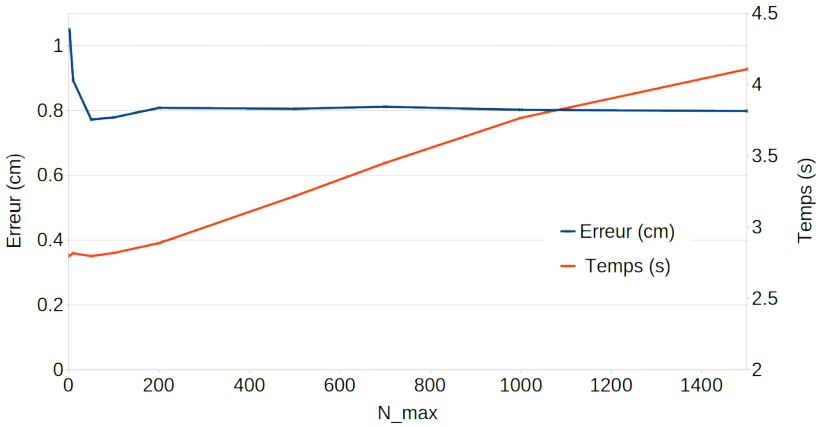
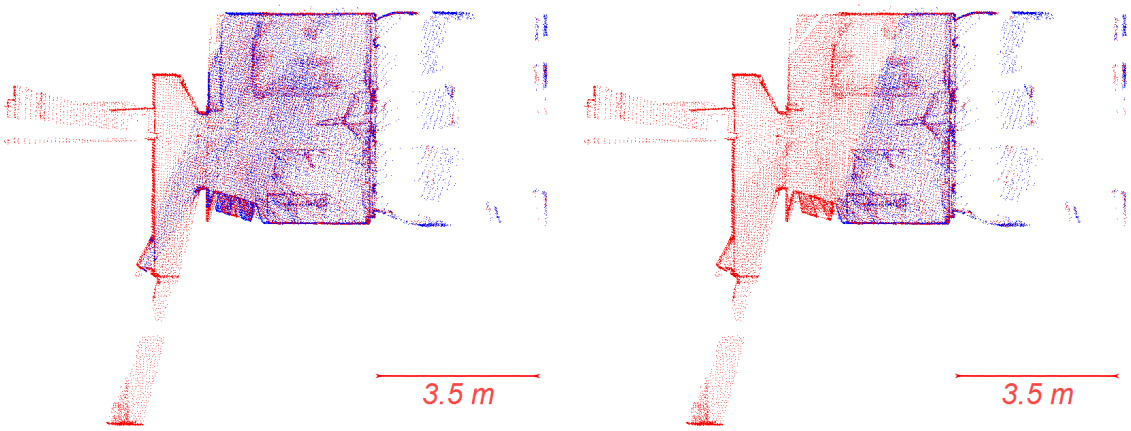
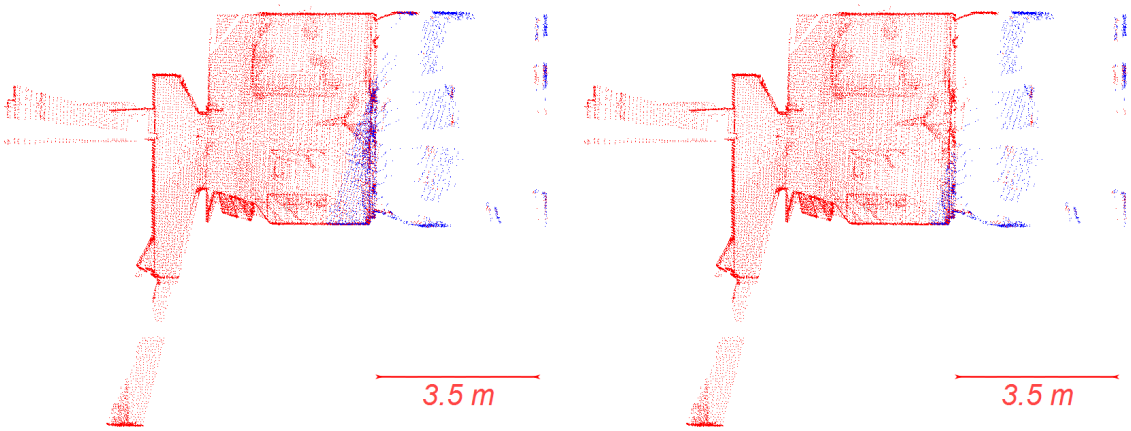


Figure 3.11.: Evaluation des résultats de notre algorithme SSFR (précision et temps de calcul) en fonction du nombre de points maximum paramétré par groupe.

Afin d'évaluer notre algorithme pour un chevauchement entre scans faible, nous extrayons une paire de scans du jeu de données `DS1-H`, et nous rognons le nuage source pour réduire la zone de chevauchement. Le chevauchement est évalué grâce à la valeur de LCP [Mel+14] (voir section 3.2.3) avec un seuil de $10\,\mathrm{cm}$. Quatre configurations différentes sont présentées sur la figure 3.12 après recalage par la transformation vérité terrain. Notre algorithme peut résoudre le recalage pour une valeur de LCP allant jusqu'à $6\,\%$ (voir figure 3.13). Pour des valeurs de chevauchement plus faibles, SSFR n'obtient pas le bon recalage. Cela peut s'expliquer par le fait qu'un mur disparaît, le système sous-contraint ne peut pas être résolu par l'étape de translation de notre algorithme. Les méthodes ICP peuvent résoudre le recalage pour un chevauchement minimum de $12\,\%$. L'algorithme FGR obtient le même chevauchement minimum que notre méthode.



(a) $LCP = 68\ \%$ **(b)** $LCP = 34\ \%$



(c) $LCP = 12\ \%$ **(d)** $LCP = 5\ \%$

Figure 3.12.: Nuages de points extraits du jeu de données `DS1-H`, utilisés pour tester la robustesse de notre algorithme au faible chevauchement. Le premier scan est rogné afin de réduire la zone commune entre les deux nuages. Les scans sont représentés après transformation du nuage source par la vérité terrain. Le chevauchement est estimé par la valeur de LCP.

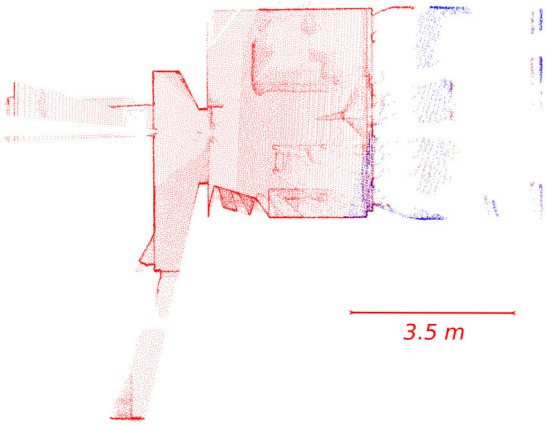


Figure 3.13.: Recalage par notre algorithme du scan n° 7 sur le scan n° 6 préalablement rogné, les scans sont extraits du jeu de données `DS1-H`. La valeur du LCP est faible (6%) et notre algorithme parvient à un résultat fiable.

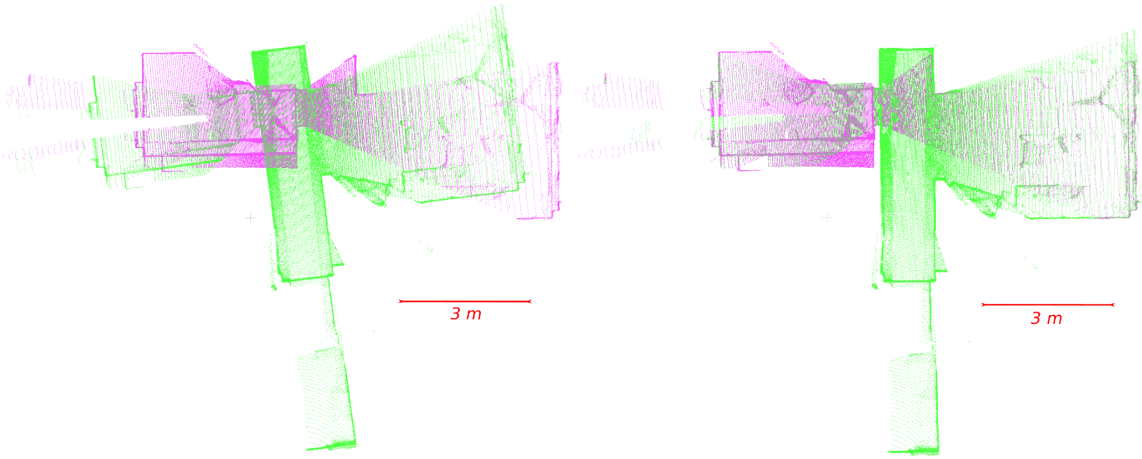
Evaluation comparative

SSFR est à présent comparé à ICP *point à point*, ICP *point à plan*, NDT, FGR et Super4PCS sur le même jeu de données : `DS1-H` (voir annexe B.1). Go-ICP n'est pas inclus à l'analyse en raison de son temps de calcul (supérieur à 2 heures par recalage).

Pour toutes les méthodes, le rayon de voisinage pour le calcul de normales est paramétré à $15\,\mathrm{cm}$. Puis, les nuages de points sont échantillonnés uniformément en utilisant une grille 3D pour obtenir une résolution moyenne de $4.0\,\mathrm{cm}$, ce qui correspond à $23\,000$ points par scan environ. Le paramétrage des différents algorithmes est résumé dans le tableau 3.2 et un exemple de recalage par notre algorithme sur ce jeu de données est représenté figure 3.14. Les erreurs moyennes entre les points de la source recalée avec la transformation exacte et avec les transformations estimées par les différents algorithmes sont représentées figure 3.15 et sont moyennées dans le tableau 3.3. Ce tableau contient également le taux de réussite des algorithmes et le temps de calcul moyen.

Tableau 3.2.: Paramétrage des algorithmes de recalage évalués sur le jeu de données `DS1-H`.

Algorithme	Paramètre	Valeur
ICP1	Distance de rejet initiale	$30\,\mathrm{cm}$
	Distance de rejet finale	$15\,\mathrm{cm}$
ICP2	Distance de rejet initiale	$30\,\mathrm{cm}$
	Distance de rejet finale	$15\,\mathrm{cm}$
NDT	Largeur de grille	$40\,\mathrm{cm}$
FGR	Rayon FPFH	$40\,\mathrm{cm}$
	Facteur de division	1.1
	Rayon de noyau minimum	$4\,\mathrm{cm}$
	Nombre de correspondances	$10\,000$
Super4PCS	Rayon LCP	$0.06\,\mathrm{cm}$
	Nombre de points conservés	1500
	Chevauchement estimé	0.5
SSFR	Erreur angulaire estimée (Δ_θ)	$0.5°$
	Largeur de classe avant sélection	$5\,\mathrm{cm}$
	Nombre de points maximum pour *mean-shift* (N_{max})	1000



(a) Avant recalage. **(b)** Après recalage.

Figure 3.14.: Exemple de recalage avec SSFR sur `DS1-H`. Le scan n° 4 (vert) est aligné sur le scan n° 3 (magenta).

On s'aperçoit que les méthodes NDT et ICP *point à point* présentent de faibles performances sur ce jeu de données. La méthode ICP *point à plan* atteint des précisions acceptables, cependant, les méthodes d'ICP échouent dans plus de 15 % des cas à cause du fort mouvement entre scans et du dynamisme de l'environnement. L'algorithme FGR semble moins performant que notre algorithme SSFR en matière de précision mais atteint tout de même un taux de réussite de 100 %. Les résultats de Super4PCS montrent de plus fortes erreurs que celles des autres algorithmes, ce qui confirme qu'elle est plus adaptée pour déterminer une transformation grossière initiale *a priori* . De plus, son temps de calcul est beaucoup plus important que celui des autres algorithmes si on cherche à obtenir une transformation précise. SSFR est la méthode qui obtient les meilleures performances sur l'erreur et le taux de réussite pour un temps de calcul très acceptable.

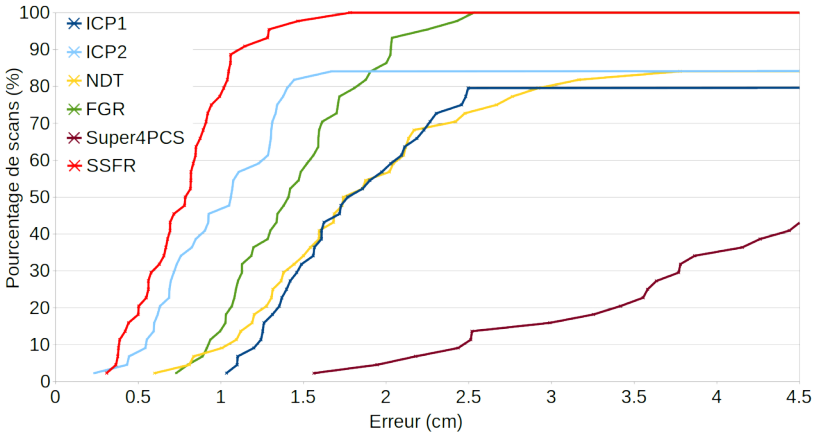


Figure 3.15.: Pourcentages de recalages pour lesquels l'erreur moyenne est inférieure à la valeur en abcisse. Les meilleurs résultats se situent en haut à gauche. Evaluation sur le jeu de données `DS1-H`. Comparaison avec les algorithmes : ICP1 (*point à point*), ICP2 (*point à plan*), NDT, FGR et Super4PCS.

Tableau 3.3.: Evaluation des méthodes sur le jeu de données `DS1-H`.

	Erreur moyenne (cm)	Taux de réussite (%)	Temps (s)
ICP1	1.71	79.5	13.5
ICP2	0.95	84.1	3.1
NDT	1.77	84.1	5.7
FGR	1.49	100	**2.4**
Super4PCS	4.9	86.4	65.0
SSFR	**0.80**	**100**	4.5

Afin d'étudier l'impact de la densité de points sur les résultats, nous sous-échantillonnons uniformément les nuages de points du jeu de données `DS1-H` avant application des algorithmes de recalage afin d'atteindre 5 niveaux de résolution différents. Notre méthode est comparée aux méthodes les plus performantes sur ce jeu de données, *c.-à-d.* ICP *point à plan* et FGR. Le rayon de noyau minimum pris en paramètre par la méthode FGR est ajusté à chaque type d'échantillonnage : il est défini comme la résolution moyenne des nuages de points (voir première colonne du tableau 3.4) afin d'obtenir les meilleurs résultats possibles. Δ_θ est définie à $2°$ afin de pouvoir traiter les densités les plus faibles.

Tableau 3.4.: Evaluation des algorithmes de recalage en fonction de la densité de points après sous-échantillonnage uniforme. Valeurs d'erreurs et de temps moyennées pour les 44 recalages du jeu de données `DS1-H`. TR : Taux de réussite.

	ICP *point à plan*			FGR			SSFR		
Résolution (cm)	TR (%)	Erreur (cm)	Temps (s)	TR (%)	Erreur (cm)	Temps (s)	TR (%)	Erreur (cm)	Temps (s)
4.0	84.1	0.95	2.9	100	1.46	1.5	**100**	**0.80**	5.9
6.5	79.5	1.03	2.1	100	1.96	1.1	**100**	**0.85**	3.0
9.8	77.3	1.03	0.5	100	2.96	1.0	**100**	**0.90**	1.2
12.9	81.8	1.06	0.3	86.4	4.35	1.1	**100**	**0.95**	0.6
20.4	56.8	1.31	1.4	40.9	7.6	1.6	**97.7**	**1.05**	0.2

On remarque que l'algorithme SSFR garde un bon comportement en cas de faible échantillonnage avec un taux de réussite de $98\ \%$ pour un échantillonnage avec une grille de largeur $30\,\mathrm{cm}$ et une erreur inférieure à $1.05\,\mathrm{cm}$ quel que soit l'échantillonnage. Cela s'explique par son aspect global en comparaison des autres méthodes qui sont locales. Le tableau 3.4 montre également que le temps de calcul de notre algorithme augmente rapidement avec le nombre de points (voir section 3.5). L'algorithme ICP obtient de bonnes précisions en cas de convergence mais son taux de

réussite s'affaiblit de manière significative. Les temps de calcul d'ICP dépendent du minimum local visé par l'algorithme pour chaque paire de scans et peuvent varier entre quelques dizaines de millisecondes jusqu'à quelques dizaines de secondes. Cette variabilité peut être problématique pour son intégration éventuelle dans une chaîne de traitements en temps réel.

3.4.4 Environnement intérieur complexe

Nous souhaitons à présent évaluer la méthode SSFR sur des données acquises dans des environnements plus complexes. Pour ce faire, une étude complète des critères d'évaluation décrits dans la section 3.4 est réalisée sur le jeu de données `DS2-L` acquis avec un dispositif Leica (voir annexe B.2). Les nuages de points sont premièrement échantillonnés uniformément pour obtenir une moyenne de $41\,000$ points par scan, ce qui correspond à une résolution moyenne de $7.0\,\mathrm{cm}$ par nuage de points. Le rayon de voisinage pour le calcul de normales est défini à $10\,\mathrm{cm}$. Les paramètres de notre algorithme sont résumés dans le tableau 3.5.

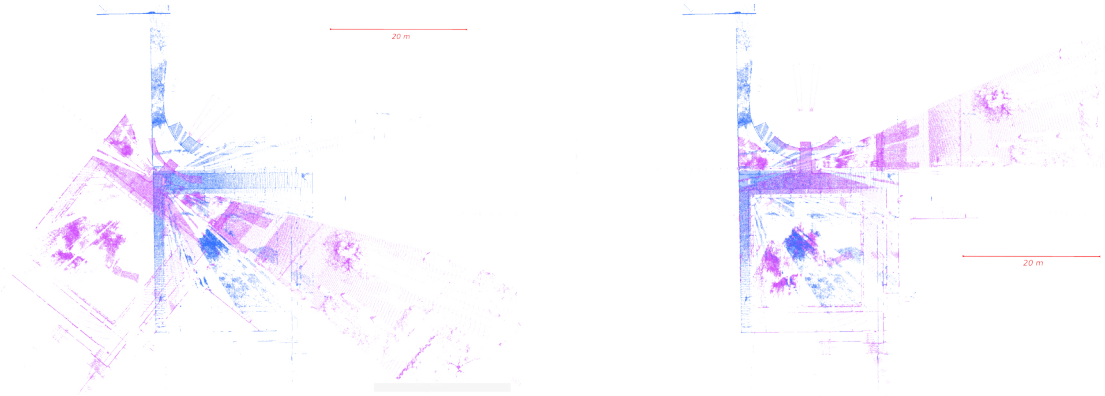
Tableau 3.5.: Paramétrage des algorithmes de recalage évalués sur le jeu de données `DS2-L`.

Algorithme	Paramètre	Valeur
FGR	Rayon FPFH	$1\,\mathrm{m}$
	Facteur de division	1.1
	Rayon de noyau minimum	$4\,\mathrm{cm}$
	Nombre de correspondances	$10\,000$
SSFR	Erreur angulaire estimée (Δ_θ)	$1°$
	Largeur de classe avant sélection	$5\,\mathrm{cm}$
	Nombre de points maximum N_{max} pour *mean-shift*	1500

Les autres méthodes mentionnées précédemment dans l'évaluation ont été testées sur ces données. Seule la méthode FGR parvient à un résultat de recalage convergeant vers la réalité pour deux paires de scans (*c.-à-d.* dont l'erreur moyenne est inférieure à $15\,\mathrm{cm}$). Cependant, même sur ces deux paires, l'erreur moyenne de recalage est de $7.12\,\mathrm{cm}$. Un exemple de recalage par notre algorithme est illustré sur la figure 3.16. Les résultats sont présentés dans le tableau 3.6.

Notre algorithme offre une bonne précision sur ces données. On s'aperçoit que l'erreur de rotation bien que faible peut entraîner une forte erreur moyenne entre points. La précision de l'algorithme est donc très dépendante de la précision de l'étape de rotation, *c.-à-d.* de la précision du calcul de normales, du parallélisme, de la courbure des murs réels et du nombre de points échantillonnés sur les plans.

On peut obtenir des résultats avec une faible perte de précision et un gain de temps significatif en sous-échantillonnant uniformément les données. Avec une résolution de $13.5\,\mathrm{cm}$, nous obtenons une précision de $2.4\,\mathrm{cm}$ et un temps de calcul moyen de $19\,\mathrm{s}$ par recalage.



(a) Avant recalage **(b)** Après recalage

Figure 3.16.: Exemple de recalage avec notre algorithme SSFR sur le jeu de données `DS2-L`. Le scan n° 1 (magenta) est aligné sur le scan n° 0 (bleu).

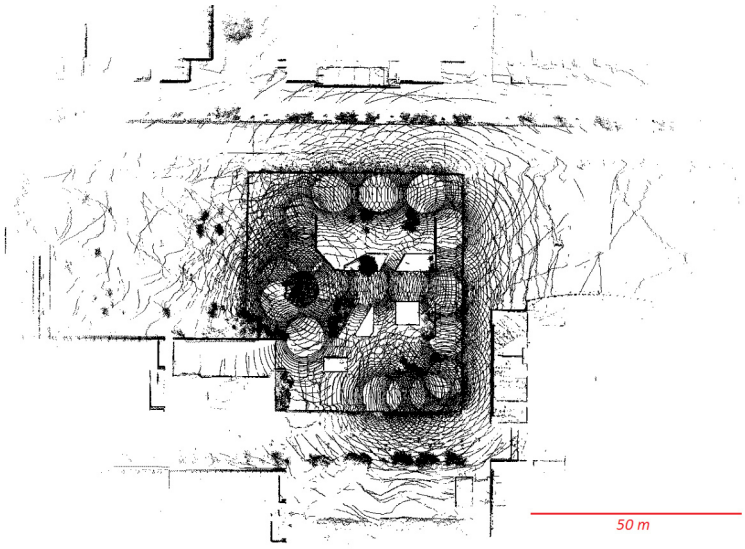
Tableau 3.6.: Evaluation de notre méthode SSFR sur le jeu de données `DS2-L`. La deuxième colonne correspond à l'erreur moyenne entre les points de la source S transformés par la vérité terrain et transformés par notre algorithme.

S_C	Erreur (cm)	Erreur de translation (cm)	Erreur de rotation (°)	Temps (s)
1_0	1.53	1.49	0.057	15.1
2_1	0.55	0.36	0.087	22.8
3_2	1.04	1.02	0.075	26.9
4_3	1.18	1.21	0.049	38.2
5_4	0.88	0.46	0.100	31.2
Moyenne	1.04	0.91	0.076	26.8

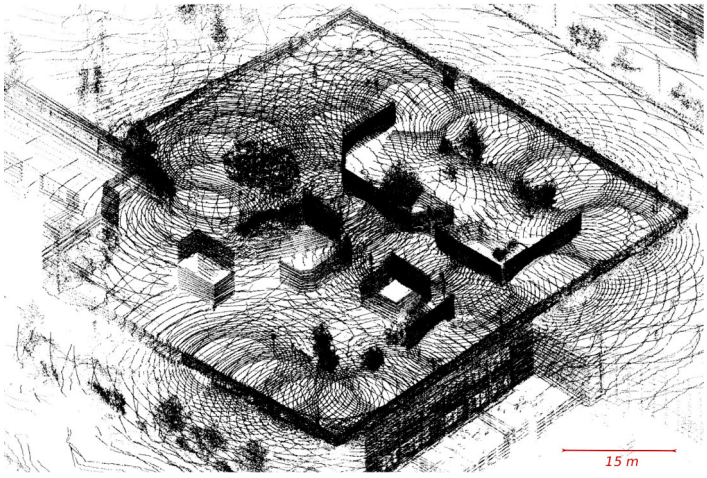
3.4.5 Contexte de navigation

L'algorithme est à présent évalué sur des nuages de points moins denses, avec une plus faible précision et issus d'une acquisition par un capteur Velodyne (jeu de données `DS3-V`, voir annexe B.3). L'environnement mesuré est représenté schématiquement figure 3.17. Le jeu de données contient 4100 scans acquis à une fréquence de $10\,\mathrm{Hz}$. Un scan sur 100 est extrait, nous obtenons donc 40 recalages à réaliser. Ce jeu de données est composé de scans acquis en mouvement, l'origine est donc mobile pendant l'acquisition ce qui ajoute une difficulté au travail de recalage. Des méthodes peuvent être appliquées afin de corriger les nuages de points avant recalage [Viv+13 ; ZS14] mais, dans cet exemple, nous n'avons utilisé aucune correction. Le résultat de recalage global par notre algorithme est montré figure 3.18.

Figure 3.17.: Reconstruction schématique du site PAVIN. Un dispositif Velodyne est monté sur un véhicule qui réalise un parcours. La fréquence d'acquisition est 10 Hz.



(a) Vue du dessus



(b) Vue de côté

Figure 3.18.: Site PAVIN après alignement des nuages du jeu de données `DS3-V`. Seuls 18 des 40 scans sont représentés pour une bonne visualisation.

Pour évaluer les résultats, la fermeture de boucle est utilisée. En effet, deux scans sont sélectionnés, un au début de la boucle et un à la fin. Puis, nous comparons le recalage direct de ces deux scans avec les recalages en série réalisés en utilisant les scans intermédiaires. Le premier recalage est complété par une brève étape d'ICP afin d'affiner le plus possible le résultat et d'obtenir une référence précise. Le rayon de voisinage pour le calcul de normales est défini à $70\,\mathrm{cm}$ pour prendre en compte les plans les plus éloignés dont l'échantillonnage est très peu dense. Le paramétrage des différents algorithmes est résumé dans le tableau 3.7. Les taux de réussites de ICP *point à plan*, FGR et de notre algorithme sont disponibles dans le tableau 3.8. Notre algorithme mène à une erreur moyenne point à point de $19.84\,\mathrm{cm}$ à la fermeture de boucle ce qui correspond à une erreur par scan de $0.99\,\mathrm{cm}$. Le temps moyen d'un recalage est $16\,\mathrm{s}$.

Tableau 3.7.: Paramétrage des algorithmes de recalage évalués sur le jeu de données `DS3-V`.

Algorithme	Paramètre	Valeur
ICP2	Distance de rejet initiale	$5\,\mathrm{m}$
	Distance de rejet finale	$2.5\,\mathrm{m}$
FGR	Rayon FPFH	$1.0\,\mathrm{m}$
	Facteur de division	1.1
	Rayon de noyau minimum	$10\,\mathrm{cm}$
	Nombre de correspondances	10 000
SSFR	Erreur angulaire estimée (Δ_θ)	1°
	Largeur de classe avant sélection	$5\,\mathrm{cm}$
	Nombre de points maximum N_{max} pour *mean-shift*	∞

Tableau 3.8.: Comparaison de l'algorithme SSFR avec les méthodes ICP *point à plan* et FGR sur `DS3-V`.

Méthode	Taux de réussite (%)
ICP *point à plan*	57.5
FGR	87.5
SSFR	**100**

3.4.6 Cas d'échec

Pour montrer les limitations de notre algorithme, nous testons le quatrième jeu de données DS4-S, (voir annexe B.4). Le scanner utilisé dans ce jeu de données a la particularité d'avoir une ouverture de 180° ce qui mène à de faibles zones de chevauchement. Sur la figure 3.19, une partie du jeu de données a été recalée. 88% des scans peuvent être recalés par SSFR. Un exemple de configuration que notre algorithme ne peut pas résoudre est représenté figure 3.20. Les deux scans n'ont pas été recalés car le chevauchement ne contient pas assez de points sur des murs communs. Plus précisément, le système est sous-contraint suivant une direction dans le plan horizontal.

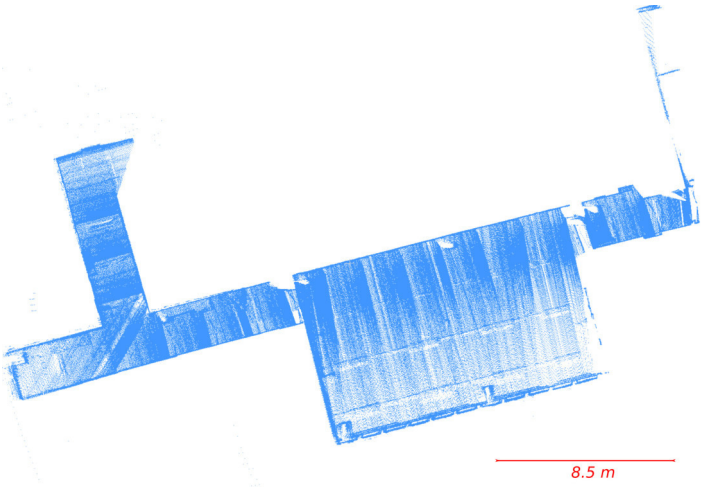


Figure 3.19.: Recalage de sept scans extraits du jeu de données DS4-S par notre algorithme.

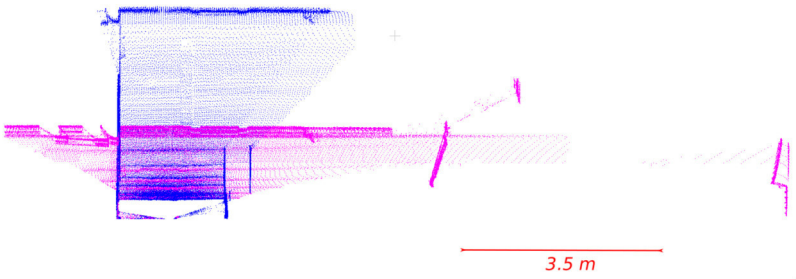


Figure 3.20.: Erreur de recalage par notre algorithme (SSFR) sur deux nuages de points extraits du jeu de données DS4-S. Une direction n'est pas représentée par des murs communs entre scans, ce qui cause une erreur de translation.

3.5 Discussion

Les résultats obtenus dans la section 3.4.1 montrent que notre algorithme est performant dans différents contextes d'acquisition. Dans cette section, nous discutons les avantages et les limites de notre méthode. Dans notre évaluation, nous avons mis en évidence certains de ses bénéfices :

- sa précision et sa vitesse (voir tableau 3.6),
- sa robustesse à de faibles densités de points (voir tableau 3.4),
- sa robustesse à de faibles chevauchements (voir section 3.4.3),
- sa robustesse à la complexité de l'environnement (voir section 3.4.4),
- sa robustesse au bruit (voir section 3.4.5),
- son invariance relativement au mouvement initial (voir section 3.4.2),
- la simplicité de son paramétrage (voir figure 3.10).

Néanmoins, on peut remarquer que le temps de calcul de notre algorithme augmente rapidement avec le nombre de points traités pour un paramétrage donné (voir tableau 3.4). Cette augmentation est notamment causée par le filtre de densité et par l'étape de *mean-shift*, elle peut donc être minimisée en ajustant le paramètre Δ_θ en fonction de la densité de points. De plus, si on considère les durées d'acquisition de scans LiDAR dans le contexte de la reconstruction de bâtiments (de quelques dizaines de secondes à quelques minutes), les temps de recalage paraissent compatibles avec une application en temps réel. En outre, le taux de réussite obtenu est le plus élevé des algorithmes comparés dans cette étude quel que soit le taux d'échantillonnage. On peut donc imaginer un alignement initial par notre algorithme après un fort sous-échantillonnage et un affinement par ICP afin d'atteindre de meilleures précisions en un temps limité. A noter que notre algorithme ne peut pas résoudre les recalages si trois murs communs non-parallèles ne peuvent pas être identifiés dans les deux nuages, là où des algorithmes locaux fondés sur les détails des scans peuvent parfois obtenir de bons résultats. Ces limitations mènent à des cas d'erreur dans le jeu de données `DS4-S` (7 cas d'erreur) dont un exemple est donné figure 3.20. Notre méthode est la seule capable de recaler tous les scans des trois premiers jeux de données présentés dans cette partie. L'erreur de recalage est faible (de l'ordre du centimètre) même en présence d'objets complexes et sur de larges surfaces bruitées. Le temps de calcul dépend essentiellement du nombre de points d'entrée et de la qualité des normales estimées. SSFR est assez rapide pour être intégré dans une chaîne de traitements d'acquisition et les mouvements entre scans ne sont pas limités.

3.6 Adaptation aux modèles

Nous proposons à présent l'adaptation de l'algorithme précédemment décrit comme traitement de nuages de points pour recaler les modèles de scans calculés dans le chapitre 2. En effet, l'étude de la partie précédente a permis d'obtenir un scan sous forme d'un ensemble de polygones connectés. Or, les caractéristiques de ces polygones peuvent être facilement injectées dans l'algorithme de recalage présenté dans cette partie afin de l'adapter à ces nouveaux modèles. Cette méthode est en cours d'implémentation et n'a pas été testée sur des données réelles.

3.6.1 Création de l'ensemble des transformations

Tout d'abord, l'ensemble des transformations doit être retrouvé tel que décrit dans la section 3.3. Les normales principales précédemment extraites de l'image gaussienne deviennent alors les normales de tous les polygones des nouveaux modèles. On peut éventuellement paramétrer une aire minimale de polygone pouvant être prise en compte pour limiter le nombre de rotations testées et donc le temps de calcul. Ensuite, l'étape de translation peut être adaptée en construisant les histogrammes à partir des aires des parties de polygones contenus dans chaque classe.

3.6.2 Sélection de la transformation

Le procédé le plus complexe à modifier est l'évaluation de la meilleure transformation. En effet, l'estimateur du chevauchement, précédemment fondé sur des critères de voisinage entre points, doit être adapté à un modèle sans point. Ceci peut être réalisé en deux temps. Tout d'abord, une étape de regroupement (*clustering*) peut mettre en commun les polygones de la source et de la cible selon l'orientation de leurs normales et leurs distances à l'origine. Puis, pour chaque paire de polygones dans les groupes précédemment trouvés, on peut définir un nouveau plan en calculant la moyenne entre les caractéristiques des plans supports (*c.-à-d.* la normale et la distance à l'origine) et projeter les polygones sur ce nouveau plan. Avec un calcul d'opérations logiques sur les polygones tel que proposé par la librairie CGAL par exemple, il est alors possible de retrouver les polygones d'intersection et de calculer leur aire. Le chevauchement peut être estimé comme la somme de ces aires pour toutes les paires. La transformation correspondant à l'estimateur de valeur maximale sera alors choisie.

3.7 Conclusion

Le recalage de nuages de points est une étape classique de la chaîne de traitements permettant de reconstruire l'intérieur d'un bâtiment. En effet, l'acquisition LiDAR étant centrée sur le point de vue du capteur, le recalage permet de mettre en relation différents scans afin d'obtenir un objet unique qui peut être traité par la suite. La diversité des capteurs et des modes d'acquisition encourage la recherche de nouveaux algorithmes plus généraux permettant d'automatiser un procédé aujourd'hui encore majoritairement manuel. Une grande partie des algorithmes existants est fondée sur des caractéristiques locales de scène. Dans ce chapitre, nous avons donc proposé une nouvelle approche plus globale nommée SSFR, permettant de recaler des nuages de points à chevauchement partiel, dans un contexte d'acquisition et de reconstruction de scènes structurées. Notre algorithme n'utilise pas de donnée odométrique ni de cible physique, conformément à l'objectif 1 défini section 3.2. Il est fondé sur des caractéristiques géométriques inhérentes aux environnements structurés et mène à des résultats fiables dans des conditions d'acquisition très diverses. L'approche globale de notre algorithme permet d'estimer un recalage quel que soit le mouvement

initial conformément à l'objectif 4 et assure un bon fonctionnement sur des scènes contenant peu de détails locaux conformément aux objectifs 2 et 5. Son processus non-itératif permet d'atteindre une bonne précision en un temps limité permettant ainsi d'envisager une implémentation temps réel pour reconstruire des édifices pendant l'acquisition, conformément à l'objectif 6. Son évaluation sur quatre jeux de données issus de différents systèmes d'acquisition LiDAR prouve que SSFR est robuste à de faibles chevauchements entre scans, à une faible densité de points et à un fort bruit de mesure, l'objectif 3 est donc atteint. Les objectifs 4 et 5 peuvent néanmoins être mitigés puisque notre approche globale requiert l'échantillonnage d'au moins trois plans non parallèles dans les scènes et certains cas peuvent ne pas être résolus. A noter que deux des jeux de données utilisés dans l'évaluation ont été acquis à l'Institut Pascal par notre équipe pour démontrer le bon fonctionnement de notre méthode.

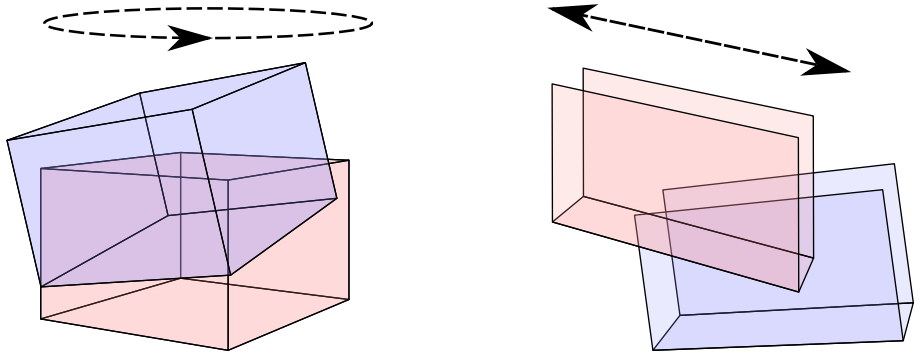
Afin d'intégrer le recalage à une chaîne de traitements de reconstruction plus complète, l'erreur de recalage doit être estimée. Cela permet notamment de fusionner la sortie de l'algorithme à d'autres données éventuelles de localisation ou de réaliser un ajustement plus global en cas de fermeture de boucle. L'estimation de cette erreur fera l'objet du chapitre suivant.

Estimation d'erreur : calcul de covariance

4

4.1 Problématique

Chaque algorithme de recalage produit une nouvelle pose, *c.-à-d.* un vecteur de 6 éléments (la translation sur les trois axes du repère global et les trois angles de rotation) avec son erreur associée. Cette erreur doit être estimée afin de pouvoir intégrer le recalage à un processus de navigation [Mon+02] ou de réaliser un ajustement global en cas de fermeture de boucle [LM97 ; Spr+09]. Notre objectif dans ce chapitre est donc de trouver un modèle d'estimation probabiliste permettant d'estimer l'erreur de recalage et un algorithme permettant d'ajuster ce modèle sur différentes configurations. Le modèle choisi doit pouvoir prendre en compte le bruit du nuage, l'anisotropie de l'échantillonnage, mais avant tout la géométrie et l'orientation de la scène. En effet, les causes d'erreurs les plus importantes sont les symétries de la scène ou l'effet « couloir » qui empêchent l'algorithme de résoudre le recalage selon certains degrés de liberté. Ces situations sont qualifiées de sous-contraintes. La figure 4.1 donne deux exemples de tels cas. Il est donc important de rendre compte de cette incertitude dans la distribution de probabilité de l'erreur de recalage.



(a) Incertitude de rotation : symétrie

(b) Incertitude de translation : effet « couloir »

Figure 4.1.: Exemple de configurations sous-contraintes. L'objet bleu est considéré comme la source et l'objet rose comme la cible. Les flèches indiquent un mouvement possible du recalage.

L'approche la plus commune est d'utiliser un modèle de distribution gaussienne 6D autour de la pose de recalage réelle afin de faciliter sa fusion avec d'autres données et son intégration à un filtre de Kalman. La difficulté est d'obtenir un ensemble de données d'erreurs suffisant pour pouvoir ajuster le modèle gaussien par une estimation de covariance. Dans le cas d'algorithmes locaux, une méthode usuelle est de déplacer légèrement le nuage source plusieurs fois et d'obtenir les poses de recalage relatives à ces poses initiales. On cherche alors à ajuster le modèle gaussien

sur les poses obtenues. Cette méthode met bien en évidence les incertitudes liées à la géométrie. Cependant, elle est longue et n'est applicable qu'aux algorithmes de recalage sensibles à la position initiale des scans. L'algorithme CELLO (*Covariance Estimation through Learned Likelihood Optimization*) [VB13] a été développé pour estimer des matrices de covariance par apprentissage sur un ensemble de recalages 2D associés à leur erreur réelle. LANDRY et coll. [Lan+19] proposent une extension 3D de cet algorithme en associant des matrices de covariance avec des paires de nuages de points à recaler. Cette extension a été testée sur des données réelles mais seule la pose initiale est prise en compte dans la détermination des covariances réelles.

Dans ce chapitre, nous allons présenter cet algorithme et l'évaluer sur un jeu de données synthétiques simplifié en faisant varier le niveau de bruit, l'échantillonnage, la densité de points et les poses initiales. Nous proposerons également une adaptation pour améliorer sa précision et la prise en compte des caractéristiques des scènes ayant une influence sur la forme de la distribution. Nous tenterons de répondre à la problématique suivante :

Problématique

Comment estimer la distribution de l'erreur d'un algorithme de recalage ?

Les contraintes auxquelles devra faire face l'estimateur sont les suivantes :

Contraintes

— Plusieurs algorithmes de recalage doivent pouvoir être évalués selon cet estimateur.

— Le niveau de bruit et l'échantillonnage peuvent varier selon les configurations de scans et doivent être pris en compte dans l'estimation.

— Les algorithmes de recalage ne convergent pas toujours vers la solution réelle.

— Les configurations sous-contraintes doivent être mises en évidence.

Dans un premier temps, nous présenterons l'algorithme CELLO et son implémentation dans le cadre de la navigation 3D, puis nous proposerons une adaptation de l'algorithme. Une évaluation synthétique sera présentée afin de déterminer les avantages et les inconvénients de cet algorithme et les pistes d'amélioration de son implémentation.

4.2 Etat de l'art

Nous étudions dans un premier temps l'estimation de pose afin de mettre en évidence le besoin d'obtenir une matrice de covariance fiable représentant l'erreur de l'algorithme de recalage dans le cadre de la navigation autonome. Dans un second temps, nous nous intéressons aux différentes méthodes d'estimation de covariance.

4.2.1 Estimation de pose

Le recalage peut également être perçu comme une mesure de la pose du scanner relativement à une pose initiale. Nous considérons ici un ensemble de T scans acquis successivement suivant les poses $X = \{X_1, \dots X_T\}$, où $X_t\,(t \in [1, T])$ est une variable aléatoire. Le modèle d'observation de poses est communément représenté par une chaîne de Markov (voir figure 4.2). En effet, les poses réelles $\{X\}$ sont estimées à partir des poses issues de l'observation (*c.-à-d.* du recalage dans notre cas) contenues dans $Z = \{Z_1, \dots Z_T\}$, où $Z_t\,(t \in [1, T])$ est également une variable aléatoire. Dans ce cadre, il existe une distribution de probabilité permettant de lier une pose avec la pose suivante : $P(X_{t+1}|X_t)$ et une distribution permettant de lier une observation à la pose réelle : $P(Z_t|X_t)$. La première est nommée « distribution de processus » et rend compte de la limitation du déplacement entre deux scans tandis que la seconde est nommée « distribution d'observation » et représente l'erreur induite par le processus de recalage utilisé. A partir de ces distributions, l'objectif est de déduire les états X selon les poses Z issues du recalage. Si les distributions de processus et d'observation sont connues, l'estimation peut avoir deux objectifs :

- estimer à un instant t une pose X_t en maximisant $P(X_t|Z_{1:T})$ avec $Z_{1:T} = (Z_1 \cap \dots \cap Z_T)$, ce qui permet d'estimer la position en temps réel,
- estimer après acquisition complète les poses $\{X_t\}_{t=1\dots t}$ qui maximisent $P(X_{1:T}|Z_{1:T})$ avec $X_{1:T} = (X_1 \cap \dots \cap X_T)$, c'est ce qui est fait dans le cadre de fermetures de boucles *a posteriori*.

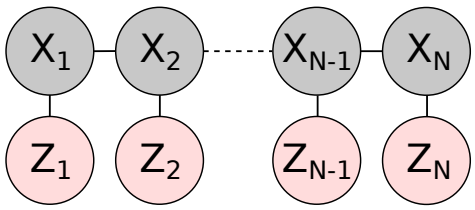


Figure 4.2.: Chaîne de Markov. $\{X_i\}_{i=1,\dots,N}$: états inconnus ; $\{Z_i\}_{i=1,\dots,N}$: observations.

Pour répondre au premier objectif, la distribution $P(X_t|Z_{1:T})$ peut être estimée à partir des distributions de processus et d'observation évaluées pour chaque pose par un algorithme itératif nommé *forward/backward*. Le filtre de Kalman est l'application de l'algorithme *forward/backward* au cas spécifique de distributions normales. La popularité de cet algorithme est due à son efficacité et sa facilité d'implémentation sur des données de grandes dimensions. De plus, différentes données odométriques peuvent être facilement intégrées à l'estimation. Le second objectif peut être atteint par l'algorithme de Viterbi [Vit06] si aucune fermeture de boucle n'est détectée.

Dans le cas contraire, des informations sont ajoutées entre les poses et l'algorithme de LU et MILIOS, nommé LUM, permet d'estimer des positions pour des distributions normales [LM97].

La difficulté de ce schéma réside dans l'ajustement des modèles de processus et d'observation. Le choix des modèles est critique et leur calibration est indispensable pour chaque contexte d'acquisition. L'algorithme d'espérance-maximisation (*Expectation-Maximization*) permet d'adapter les modèles choisis aux données d'observation acquises [Thr+98], son adaptation aux chaînes de Markov est nommé algorithme de *Baum-Welch*. Après une initialisation des paramètres des modèles, une première estimation des poses est réalisée. Puis, la probabilité d'obtenir ces poses est maximisée relativement aux paramètres. Ces deux étapes sont répétées itérativement jusqu'à convergence.

Dans cette étude, nous nous plaçons dans le cas de distributions de probabilités suivant une loi normale. Plus spécifiquement, nous définissons la variable aléatoire Z_t telle que :

$$\begin{cases} Z_t = HX_t + b_t^z \\ P(Z_t|X_t) = \mathcal{N}(HX_t, R)\,, \end{cases} \tag{4.1}$$

où b_t^z est le bruit gaussien, H est la fonction d'observation, HX_t est l'espérance et R la covariance du modèle gaussien. La variable aléatoire X_{t+1} est définie par :

$$\begin{cases} X_{t+1} = FX_t + b_t^x \\ P(X_{t+1}|X_t) = \mathcal{N}(FX_t, Q)\,, \end{cases} \tag{4.2}$$

où b_t^x est le bruit gaussien, F est la fonction de processus, FX_t est l'espérance et Q la covariance du modèle gaussien.

4.2.2 Estimation de covariance

Après étude du modèle de la chaîne de Markov, un premier obstacle apparaît dans l'estimation de pose par recalage. En effet, en pratique, la distribution de probabilité d'observation varie selon le recalage effectué, *c.-à-d.* que R n'est pas constant. Certains auteurs choisissent de corréler cette variation aux instants d'acquisition : $R_t = R(t)$ [Meh70]. Ils cherchent alors à maximiser $P(X_{1:T}, R_{1:T}|Z_{1:T})$. D'autres auteurs choisissent d'actualiser la covariance R_t en la supposant fixe sur une fenêtre de scans successifs autour de l'observation Z_t [Hu+03 ; Kim+09]. Cependant, les scans ne sont pas forcément acquis à intervalles de temps réguliers et peuvent être pris dans des scènes très différentes. Il apparaît donc plus adapté de lier la covariance à la pose : $R_t = R(X_t)$. STAKKELAND et coll. [Sta+09] choisissent de représenter les paramètres du modèle de distribution par des fonctions de l'état (dans notre cas la pose). Ces fonctions sont difficiles à déterminer dans le contexte de la navigation et demandent un lourd travail de paramétrage. ARAVKIN et BURKE [AB14] proposent de déterminer ces fonctions à l'aide d'une nouvelle optimisation mais perdent l'aspect récursif de l'algorithme de *forward/backward*. Le problème de ces méthodes est qu'en pratique, rien n'indique que la covariance soit corrélée dans l'espace, particulièrement lors de fortes transitions comme un changement de pièce.

La covariance R peut également être liée directement aux mesures, *c.-à-d.* aux nuages de points. Certains auteurs proposent un calcul de covariance à partir de la fonction de coût communément optimisée dans les algorithmes de recalage locaux de type ICP [BB01 ; BB03 ; Cen07]. En effet, pour ces algorithmes, la transformation est déduite de la minimisation de :

$$f_c = \sum_i ||Tp_i - m_i||^2 , \tag{4.3}$$

où T est la matrice de transformation, p_i est un point de la source et m_i est le point le plus proche de p_i dans la cible. Les associations $\{p_i, m_i\}$ sont recalculées à chaque itération. Après avoir estimé la transformation par optimisation et l'avoir appliquée au nuage source, BENGTSSON et BAERVELDT [BB01] linéarisent la minimisation de $f_c(T)$ et utilisent la hessienne de cette dernière afin de déduire la covariance associée aux paramètres de T. La hessienne peut être calculée directement si f_c est dérivable analytiquement. Sinon, une étape d'échantillonnage autour de la pose estimée peut être réalisée [BB01]. Cette méthode met bien en évidence les variations dues à l'environnement mais ne prend pas en compte l'effet du bruit du capteur ce qui mène à des résultats trop pessimistes. Pour pallier ce problème, CENSI [Cen07] intègre le bruit du capteur à leur modèle de covariance en utilisant le théorème des fonctions implicites et obtient des résultats significatifs en 2D. Cependant, l'extension 3D de cette méthode implémentée par MANOJ et coll. [Man+15] donne des résultats de covariance trop optimistes [Men+17]. En effet, cette méthode ne prend pas en compte les réassociations de points pendant la minimisation, elles ont en réalité un impact non-négligeable sur la fonction de coût même autour de la convergence [Lan+19].

Les méthodes précédentes dépendent de l'implémentation de l'algorithme de recalage et elles rencontrent beaucoup d'obstacles pour modéliser correctement le bruit du capteur, les systèmes sous-contraints et les défauts de l'algorithme. Il apparaît donc opportun de s'intéresser à des méthodes orientées données. Les méthodes *brute force* ou *Monte Carlo* sont les plus précises à ce jour. Elles consistent à associer une covariance à un scan donné (appelé S_0 par la suite) en estimant la covariance sur un ensemble de poses après recalage. En 2D, BENGTSSON et BAERVELDT [BB03] obtiennent l'ensemble de N poses en créant N scans à partir de S_0 et en les recalant sur S_0. Pour cela, ils créent une polyligne à partir des points de S_0. Puis, ils ré-échantillonnent la polyligne et déplacent le nouveau nuage autour de sa pose initiale. IVERSEN et coll. [Ive+17] étendent cette méthode en 3D pour un capteur de type Kinect, en ré-échantillonnant les objets grâce à une triangulation et en ajoutant un bruit spécifique au capteur. Ces méthodes obtiennent de bons résultats mais sont très lentes ce qui les rend inutilisables en navigation. De plus, en 3D, il est difficile d'obtenir un modèle de la surface sur lequel ré-échantillonner les points de manière fiable et rapide et, en pratique, les considérations d'échantillonnage et de bruit ne peuvent pas forcément être prises en compte. La covariance reflète alors uniquement la forme de l'environnement.

VEGA-BROWN et coll. [VB+13] introduisent une nouvelle méthode d'estimation de covariance en 2D nommée CELLO et fondée sur une étape d'apprentissage automatique permettant de tirer parti des avantages des algorithmes *brute force* tout en diminuant leurs temps de calcul. De plus, la phase d'entraînement permet d'intégrer des informations de bruit, d'échantillonnage et de géométrie. Pour un recalage, le principe est d'estimer une covariance inconnue en utilisant des erreurs d'estimation connues d'autres recalages. L'apprentissage permet de sélectionner les erreurs d'estimation de recalages connus, utiles à l'estimation d'une covariance d'un recalage inconnu selon les caractéristiques des scans étudiés. Cette méthode est étendue à la 3D par LANDRY et coll. [Lan+19]. Elle est décrite précisément dans les parties suivantes. VEGA-BROWN et ROY [VBR13] sont allés plus loin avec CELLO-EM dans les cas où les vérités terrains ne sont pas disponibles pour les données d'entraînement et où les erreurs sont donc inaccessibles. Dans cette extension, ils intègrent un algorithme d'espérance-maximisation à CELLO. Ils alternent une phase d'estimation des poses par l'algorithme *forward/backward* avec la phase d'apprentissage de CELLO. LIU [Liu18] préfère utiliser un réseau de neurones pour estimer des covariances en 2D à partir d'un jeu de données vérité terrain. La difficulté d'utiliser un réseau de neurones réside dans la conservation des caractéristiques d'une matrice de covariance, la covariance étant une matrice symétrique semi-définie positive. De plus, cette méthode n'est pas encore étendue en 3D.

Objectifs

1. Etudier les avantages, les défauts et les limites de l'algorithme CELLO-3D.
2. Evaluer son comportement sur un jeu de données simulées .
3. Trouver des descripteurs adaptés aux environnements intérieurs.

4.3 CELLO

4.3.1 CELLO-2D

Vue générale

L'algorithme CELLO a tout d'abord été développé et testé sur des données 2D. Les nuages de points sont alors acquis par une rotation d'un capteur LiDAR et sont contenus dans le plan $\{x, y\}$. Cet algorithme suppose qu'on dispose d'un ensemble S de N_S scans et de leurs poses réelles exactes dans un repère global. On peut donc calculer les erreurs de recalage avec un algorithme spécifique (ICP ou autre) de chaque scan sur d'autres scans cibles. On appelle ces erreurs $\{e_i\}_{i \in \{1 \dots N_S\}}$ avec i le numéro du scan.

Intuitivement, CELLO permet de déterminer la covariance R_j liée à un scan j inconnu en calculant une « covariance pondérée » à partir des erreurs de recalage $\{e_i\}_{i \in \{1 \dots N_S\}}$. Cette covariance pondérée est définie dans l'équation (4.4).

$$R_j^* = \frac{\sum_{i=1, i\neq j}^{N_s} k_{ij} * e_i e_i^T}{\sum_{i=1 \ldots i\neq j}^{N_s} k_{ij}} . \tag{4.4}$$

Les poids $\{k_{ij}\}_{i,j \in [1 \ldots N_s]}$ sont déterminés en fonction de la ressemblance des scans entre eux. Dans le cas où l'on souhaite calculer la covariance liée à un scan i, plus les scans i et j sont similaires, plus le poids k_{ij} doit être élevé pour que l'erreur e_j soit prise en compte dans le calcul.

Pour calculer cette covariance pondérée, les poids sont déterminés en attachant des descripteurs aux scans et en utilisant une métrique permettant de calculer un score de ressemblance entre deux ensembles de descripteurs. La métrique est apprise sur un ensemble de données d'entraînement. Le processus est résumé figure 4.3 : la covariance d'un scan inconnu peut être prédite à partir du descripteur du scan, d'un ensemble connu de descripteurs avec leurs erreurs associées et d'une métrique apprise au préalable. Les différentes étapes décrites ici sont détaillées dans les paragraphes suivants.

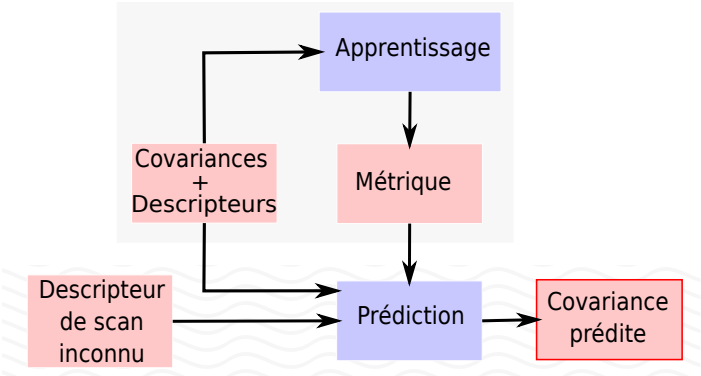


Figure 4.3.: Algorithme CELLO.

Définition des poids

Afin d'évaluer la similarité entre scans évoquée dans la section précédente pour calculer les poids, des descripteurs sont associés à chacun des scans et concaténés dans un vecteur Φ. Ces descripteurs ont pour objectif de compacter les informations utiles à la déduction de la covariance. Les poids sont définis à partir de ces descripteurs avec la formule 4.5 où M est la métrique permettant de pondérer les descripteurs et leurs associations. Elle permet notamment de décider quels sont les descripteurs ou les associations de descripteurs discriminants pour déterminer la covariance et quel est le seuil pour juger que deux descripteurs discriminants sont proches.

$$k_{ij} = \exp(-\Delta\Phi_{ij} M \Delta\Phi_{ij}^T) \qquad \text{avec} \qquad \Delta\Phi_{ij} = (\Phi_i - \Phi_j) \tag{4.5}$$

La matrice M étant difficile à définir manuellement, VEGA-BROWN et coll. [VB+13] passent par un apprentissage de ses valeurs grâce à des vérités terrain.

Apprentissage de M

Le but est alors de déterminer la métrique M adéquate afin d'évaluer la différence entre deux vecteurs de descripteurs. Les différentes étapes de l'apprentissage sont les suivantes :

1. On définit un ensemble d'entraînement constitué de scans et de leurs erreurs de recalage relatives.
2. On attribue des descripteurs à chaque scan permettant de compacter son information. Ces descripteurs doivent être les plus corrélés possible avec la covariance du scan *c.-à-d.* qu'ils doivent contenir des informations de bruit du capteur, de géométrie de la scène et d'échantillonnage.
3. On initialise la matrice M de façon à avoir le même poids pour tous les descripteurs.
4. Itérativement
   a) Prédiction : on calcule une covariance prédite R_i^* avec la moyenne pondérée définie équation (4.4).
   b) On maximise, en fonction de M, la densité de probabilité d'obtenir les erreurs de recalage e_i, sachant la covariance prédite pour chaque scan (voir équation (4.6)). Cette densité est la suivante :

$$P(e_i/R_i^*) = \prod_{i=1}^{N_s} \frac{1}{\sqrt{2\pi}|R_i^*|} \exp(-\frac{e_i^T R_i^{*-1} e_i}{2}) \,. \tag{4.6}$$

Pour maximiser la densité 4.6, on maximise la log-vraissemblance définie telle que :

$$\mathcal{L}(M) = -\frac{1}{2} \sum_{i=1}^{N_s} (\log(|R_i^*(M)|) + e_i^T R_i^*(M)^{-1} e_i) \,. \tag{4.7}$$

A noter que VEGA-BROWN et coll. recommandent d'utiliser un terme de régularisation pour éviter le sur-apprentissage. Ce terme est de la forme :

$$\mathcal{R} = \sum_{i=1}^{N_s} (\log(k_{i,j})) \,. \tag{4.8}$$

Il force la métrique à prendre en compte toutes les erreurs à disposition pour calculer une prédiction. Ce terme est particulièrement nécessaire quand les données d'entraînement sont peu nombreuses et non-homogènes. La fonction de coût finale est la suivante :

$$f_c(M) = (1-\alpha)\mathcal{L}(M) + \alpha\mathcal{R}(\mathcal{M}) \,, \tag{4.9}$$

α étant un paramètre à régler par l'utilisateur pour calibrer la part de régularisation dans l'optimisation.

Prédiction

Une fois la matrice M apprise, la covariance d'un scan inconnu peut être estimée à partir de son descripteur. Pour cela, les poids relatifs à ce descripteur sont recalculés et l'équation (4.4) est utilisée pour estimer la covariance.

4.3.2 CELLO-3D

CELLO a été adapté en 3D par LANDRY et coll. [Lan+19]. Pour cela, plusieurs modifications ont été nécessaires.

Problématique 3D

Dans cette extension, les auteurs cherchent à attribuer une covariance (sous forme de matrice 6×6) à une paire de scans et non plus à un scan unique. En effet, les descripteurs sont associés au nuage de points représentant la zone de chevauchement entre les deux scans à recaler. Ils ne dépendent pas de la pose initiale des scans mais de leur pose finale.

Descripteurs 3D

Afin de définir leurs descripteurs, LANDRY et coll. découpent l'environnement par une grille 3D. Des descripteurs sont calculés dans chaque cellule de cette grille. Les types des descripteurs sélectionnés par les auteurs sont :

- une valeur scalaire décrivant la planéité du nuage dans la cellule ;
- une valeur scalaire décrivant la cylindricité ;
- un histogramme de normales à 9 classes.

Puis les descripteurs de toutes les cellules sont concaténés dans le vecteur de descripteurs final.

Prédiction 3D

Afin d'augmenter le nombre de données de recalage et par conséquent d'erreurs e_i, les auteurs ont choisi, pour chaque scan, d'effectuer son recalage N_r fois suivant différentes poses initiales distribuées autour de la pose réelle. Pour chaque scan, on peut alors calculer la covariance statistique des résultats comme définie dans l'équation (4.10). Cette estimation sera utile par la suite. Ce procédé est directement lié aux méthodes *brute-force* décrites section 4.2.

$$R_i = \frac{\sum_{r=1}^{N_r} e_{i,r} e_{i,r}^T}{N_r} \tag{4.10}$$

Comme le poids ne dépend plus de la pose initiale des scans, les poids sont les mêmes pour toutes les erreurs de recalage sur une même paire de scans. La formule de prédiction de la covariance devient alors une moyenne pondérée des covariances R_i avec $i \, \epsilon \, [0, N_s]$ (voir équation (4.11)).

$$R_j^* = \frac{\sum_{i=1, i\neq j}^{N_s} \sum_{r=1}^{N_r} k_{ij} e_{i,r} e_{i,r}^T}{\sum_{i=1, i\neq j}^{N_s} \sum_{k=1}^{N_r} k_{ij}} = \frac{\sum_{i=1, i\neq j}^{N_s} k_{ij} N_r R_i}{\sum_{i=1, i\neq j}^{N_s} N_r k_{ij}} = \frac{\sum_{i=1, i\neq j}^{N_s} k_{ij} R_i}{\sum_{i=1, i\neq j}^{N_s} k_{ij}} \tag{4.11}$$

Optimisation 3D

A partir de (4.10), LANDRY et coll. modifient la formulation de l'optimisation pour l'apprentissage de la métrique. En effet, le deuxième terme de la log-ressemblance (4.7) peut être réécrit de la manière suivante :

$$\sum_{r=1}^{N_r} e_{i,r}^T R_i^{*-1} e_{i,r} = \sum_{r=1}^{N_r} trace(e_{i,r}^T R_i^{*-1} e_{i,r}) = trace(R_i^{*-1} \sum_{r=1}^{N_r} e_{i,r} e_{i,r}^T) = N_r trace(R_i^{*-1} R_i) \tag{4.12}$$

On en déduit la vraisemblance à minimiser :

$$\begin{aligned} \mathcal{L}(M) &= \sum_{i=1}^{N_s} \sum_{r=1}^{N_r} \left(\log(|R_i^*(M)|) + e_{i,r}^T R_i^*(M)^{-1} e_{i,r} \right) \\ &= N_r \sum_{i=1}^{N_s} \left(\log(|R_i^*(M)|) + trace(R_i^*(M)^{-1} R_i) \right) \end{aligned} \tag{4.13}$$

Inconvénients

Cette méthode suppose que l'algorithme de recalage a convergé vers la pose réelle puisque les descripteurs sont associés à la zone de chevauchement après recalage. L'algorithme ne peut donc pas détecter une mauvaise convergence. De plus, comme la majorité des méthodes, CELLO utilise des modèles gaussiens qui ne sont pas forcément adaptés aux environnements sous-contraints. Enfin, les données d'entraînement sont difficiles à obtenir. En effet, les possibilités d'échantillonnage, les modèles de bruit et les géométries de scène sont infinies et il est difficile d'obtenir un jeu de données d'entraînement représentatif même synthétiquement. La configuration de l'algorithme est complexe. En plus du paramétrage de l'optimisation (α et taux d'apprentissage), les descripteurs doivent être sélectionnés et chaque descripteur a ses propres paramètres (rayon de voisinage, taille des histogrammes, seuils, etc.). Enfin, seule la géométrie des scènes est prise en compte pour l'estimation des covariances d'apprentissage. Ces covariances sont définies en modifiant uniquement la pose initiale de la source et la distribution de ces nouvelles poses influe beaucoup sur la distribution des poses résultantes du recalage, notamment dans le cas de systèmes sous-contraints.

4.4 Adaptation proposée

4.4.1 Nouveau calcul des poids

Les auteurs de CELLO utilisent un vecteur concaténant les descripteurs qu'ils soient scalaires (le nombre de points par exemple) ou vectoriels (un histogramme par exemple). Les poids définis équation (4.5) sont donc calculés grâce à la métrique M en pondérant les descripteurs entre eux mais aussi entre chaque élément d'un même descripteur si celui-ci est vectoriel (entre les classes de l'histogramme de normales par exemple). La taille des descripteurs est donc liée linéairement au

nombre d'éléments de M à prédire. Plus le nombre d'éléments à prédire est grand, plus la quantité de données d'apprentissage doit être importante pour éviter les effets du sur-apprentissage. Or, comme nous l'avons vu précédemment, ces données sont difficiles à obtenir et le temps d'apprentissage peut devenir très long. En outre, cette corrélation n'a pas de fondement physique. Nous avons donc transformé le vecteur $\Delta\varphi$ pour que chaque élément soit une distance scalaire entre descripteurs. Cela permet notamment d'inclure des distances non euclidiennes comme décrit par la suite. La taille de la matrice M est ainsi considérablement réduite, l'occupation de la mémoire est alors plus faible, les temps de calculs sont également réduits et le risque de surapprentissage est limité.

4.4.2 Nouveaux descripteurs

VEGA-BROWN [VB13] affirme que les descripteurs utilisés ont une faible influence sur le résultat et que la métrique M permet d'extraire de manière fiable les caractéristiques nécessaires à l'estimation de la covariance. Cependant, LIU [Liu18] souligne l'importance du choix de ces descripteurs afin que tous les facteurs d'erreur soient pris en compte dans l'estimation de la covariance, *c.-à-d.* le bruit du capteur, l'échantillonnage, la géométrie et l'orientation des scènes. Nous avons choisi de créer de nouveaux descripteurs. Nous avons éliminé la grille mise en place par les auteurs afin de calculer des descripteurs pour chaque cellule (voir section 4.3.2) et nous calculons des descripteurs pour tout le nuage. Nous avons conservé l'histogramme de normales de 9 classes de la version de [VB13]. Nous avons ensuite ajouté un histogramme de points construit à partir d'une grille 3D de l'espace et un autre histogramme construit à partir d'un cylindre d'axe z. Les classes sont alors définies par un découpage angulaire du cylindre. Nous avons aussi utilisé une valeur d'évaluation du bruit sur le nuage en calculant la valeur médiane du rapport de valeurs propres locales suivant :

$$w = \frac{\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} \tag{4.14}$$

Nous avons également extrait la normale principale de chaque nuage. Cette normale correspond au plan contenant le plus de points dans la scène. La distance pour ce descripteur est définie comme la valeur absolue du produit scalaire entre deux descripteurs de ce type.

4.5 Evaluation

LANDRY et coll. réalisent leur évaluation sur des jeux de données réels en utilisant des procédés d'augmentation de données (*data augmentation*) pour rendre leur apprentissage plus robuste. Nous proposons ici une brève évaluation sur une base de données simulées afin de comprendre précisément le comportement de l'algorithme relativement au bruit du capteur, à l'orientation et à la géométrie de la scène. Pour l'étude de la géométrie, nous nous restreignons à la détection de l'effet « couloir ». Plus généralement, cette étude permet de mettre en évidence les caractéristiques majeures des scènes qui jouent sur la précision du recalage. Nous créons donc des

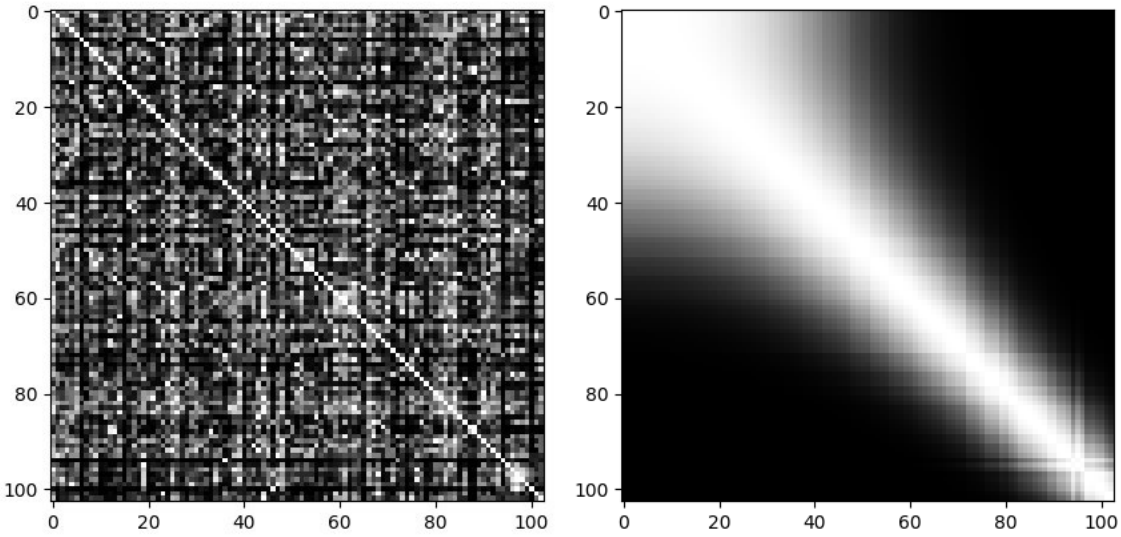
jeux de données synthétiques simulant des scènes d'intérieur simples, échantillonnées par des points. Des paires de nuages de points sont ainsi créées afin de simuler la zone de chevauchement. Le nuage correspondant à la source est alors déplacé d'une faible distance et le recalage est calculé par ICP. Cette opération est réalisée N_{rec} fois par paire et une covariance est calculée à partir des poses de recalage ainsi acquises. Comme outil visuel permettant de mettre en évidence le comportement de l'algorithme, nous utilisons la matrice de poids. Chaque ligne de cette matrice est associée à une paire de nuages de points. Ces lignes représentent les poids associés aux autres paires de nuages qui permettent d'estimer la covariance. Ainsi, on peut représenter cette matrice sous la forme d'une image dans laquelle la couleur du pixel représente le poids entre deux paires de nuages. Plus le poids est fort (blanc) et plus les paires sont considérées comme proches, *c.-à-d.* que leurs covariances réelles ont un impact fort dans leur prédiction respective. A noter que nous représentons les poids entre deux paires identiques en blanc (car leur similarité est parfaite), bien qu'en réalité ces poids ne soient pas utilisés dans l'algorithme. Dans cette partie, nous n'utilisons aucun terme de régularisation compte tenu de l'homogénéité des données.

4.5.1 Evaluation de l'entraînement

Dans un premier temps, nous souhaitons évaluer l'entraînement de l'algorithme et sa capacité à corréler les caractéristiques des scènes aux formes de covariances. Dans ce cadre, nous évaluons premièrement la prise en compte du bruit, deuxièmement, la prise en compte de l'orientation des couloirs et troisièmement, la prise en compte de la géométrie de la scène. A noter que l'échantillonnage n'est pas traité ici et pourrait faire l'étude d'une évaluation également.

Bruit du capteur

Tout d'abord, nous testons un ensemble de 100 paires de nuages de points échantillonnant un cube de dimensions $5 \times 5 \times 5$ m. Un bruit gaussien croissant relativement aux indices des paires est ajouté aux nuages de points. Les nuages de la première paire ne sont pas bruités et les nuages de la dernière paire contiennent un bruit gaussien d'écart type 2 cm. Les nuages contiennent 10000 points. Ce jeu de données est appelé $\mathcal{J}_{bruit}$ par la suite. Nous entraînons l'algorithme avec les paires de $\mathcal{J}_{bruit}$ et la matrice de poids résultante de l'entraînement est représentée figure 4.4. Sur cette figure, on remarque que la corrélation entre le bruit du capteur et la largeur de la distribution a bien été capturée par les descripteurs proposés dans notre adaptation : pour une paire de nuages de points donnée, les paires ayant un impact important sur l'estimation de la matrice de covariance relative sont celles qui ont un niveau de bruit similaire (correspondant aux colonnes proches de la diagonale). C'est pour cela qu'une diagonale blanche apparaît sur l'image. On remarque également que les descripteurs précédemment utilisés par LANDRY et coll. ne sont pas suffisants pour capturer cette caractéristique.

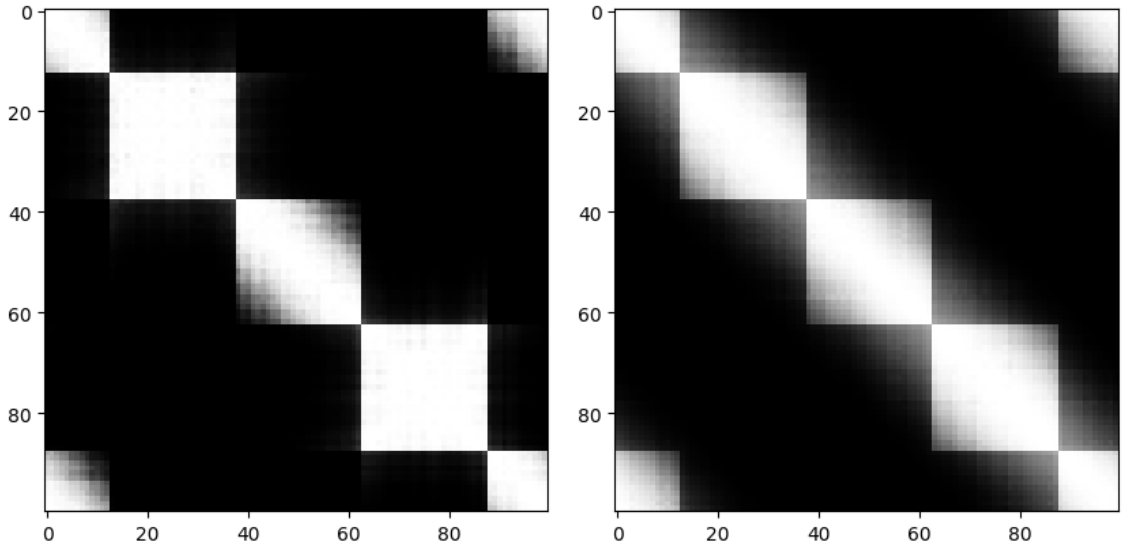


(a) CELLO 3D **(b)** Adaptation

Figure 4.4.: Matrices de poids finales après les entraînements de CELLO 3D par l'algorithme décrit par LANDRY et coll. **(a)** et par notre adaptation **(b)**. Les paires de nuages de points ont un bruit gaussien croissant. Pour une paire donnée, les poids les plus forts doivent être attribués aux paires ayant un niveau de bruit similaire.

Orientation

Ensuite, un nouvel ensemble de données synthétiques contenant 100 paires de nuages de points de 10000 points chacun est créé. Les points sont échantillonnés sur un objet de type « couloir » dont l'orientation varie suivant l'axe z. L'angle de l'orientation du couloir à l'axe x augmente de $0°$ à $180°$. Ce jeu de données est appelé $\mathcal{J}_{orient}$. Nous entraînons l'algorithme avec les paires de $\mathcal{J}_{orient}$ et la matrice de poids résultante de l'entraînement est représentée figure 4.5. La diagonale blanche qui apparaît à nouveau met en évidence que l'entraînement permet de corréler l'orientation des paires de nuages de points à la forme de la covariance. Cependant, on remarque la formation de groupes. Ces groupes sont liés aux classes de l'histogramme de normales utilisé comme descripteur. Les groupes blancs dans les coins haut droit et bas gauche sont dûs au fait que les couloirs orientés à $0°$ et à $180°$ sont proches et ont un fort impact mutuel sur le calcul de leurs covariances. Avec notre adaptation, ces groupes sont nuancés grâce au descripteur de normale principale et à l'histogramme d'occupation cylindrique qui permettent d'ajuster les poids. Avec la version de LANDRY et coll., bien que la voxelisation de l'espace permette de nuancer les liens entre les paires de certains groupes (comme le groupe central par exemple), aucun descripteur ne permet de corréler les paires proches entre groupes.

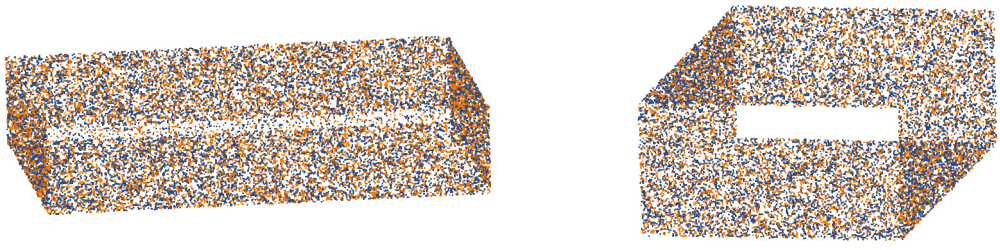


(a) CELLO-3D **(b)** Notre adaptation

Figure 4.5.: Matrices de poids finales après les entraînements de CELLO 3D par l'algorithme décrit par LANDRY et coll. **(a)** et par notre adaptation **(b)**. Les paires de nuages de points échantillonnent des couloirs orientés avec un angle croissant relativement à l'axe x. Pour une paire donnée, les poids les plus forts doivent être attribués aux paires ayant une orientation similaire, donc proche de la diagonale.

Géométrie

Un des critères de qualité majeur à tester est la détection de systèmes sous-contraints. Pour ce faire, un nouveau jeu de données nommé $\mathcal{J}_{complet}$, est testé. Il contient 1000 paires de nuages de points échantillonnés sur deux types d'objets : des parallélépipèdes et des couloirs infinis. Un exemple de chacun des objets est donné figure 4.6. Les deux nuages d'une paire sont échantillonnés sur le même objet mais le nombre de points et l'échantillonnage sont différents (voir tableau 4.1). Sur la figure 4.7, nous avons représenté la matrice de poids correspondant à l'entraînement sur 900 paires de nuages extraites de ce jeu de données.

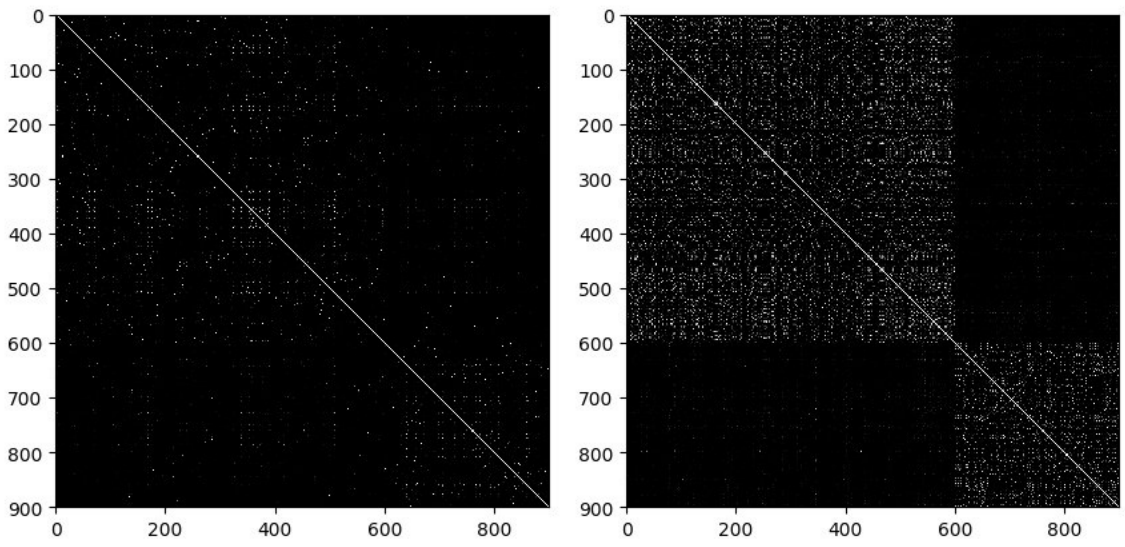


(a) **(b)**

Figure 4.6.: Exemple d'objets **(a)** « parallélépipède » et **(b)** « couloir » du jeu de données $\mathcal{J}_{complet}$. Le nuage source est représenté en bleu et le nuage cible en orange.

Tableau 4.1.: Caractéristiques du jeu de données synthétiques créé.

Nombre de paires	1000
Nombre de points par nuage	Loi uniforme : $\mathscr{U}(10000, 12000)$
Bruit (cm)	Loi uniforme : $\mathscr{U}(0, 0.02)$
Position des murs sur chaque axe horizontal	Loi uniforme : $\mathscr{U}(-10, -0.5)$ et $\mathscr{U}(0.5, 10)$
Position du sol et du plafond sur l'axe vertical	Loi uniforme : $\mathscr{U}(-2, -0.5)$ et $\mathscr{U}(0.5, 2)$
Transformation	Rotation aléatoire
Nombre de recalages par paire	300
Ratio de couloirs sur le nombre total de paires	$1/3$



(a) CELLO 3D **(b)** Adaptation

Figure 4.7.: Matrices de poids finales après les entraînements de CELLO 3D par l'algorithme décrit par LANDRY et coll. **(a)** et par notre adaptation **(b)**. Les deux premiers tiers des scans représentent des parallélépipèdes. Le tiers restant représente des couloirs infinis. On retrouve la formation de deux groupes et les similarités entre scans déterminées à partir de leurs descripteurs grâce à notre adaptation.

On remarque que, après convergence de l'algorithme proposé, les deux groupes sont bien séparés. Ainsi, dans le calcul de la covariance d'un objet « couloir », la covariance d'un parallélépipède aura un poids presque nul et n'interviendra donc pas et inversement. On remarque que la version proposée par LANDRY et coll. retrouve moins précisément ces groupes. Avec les descripteurs adéquats, l'algorithme est donc capable de séparer des groupes de recalages selon la géométrie des scènes recalées.

4.5.2 Evaluation de la prédiction

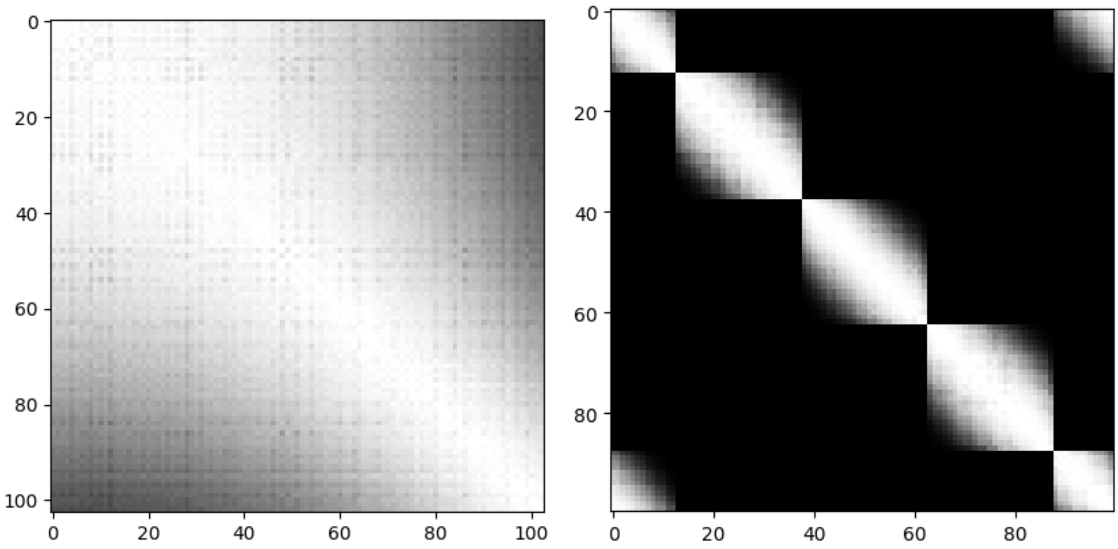
Le jeu de données $\mathcal{J}_{complet}$ décrit dans le tableau 4.1 est utilisé afin de réaliser un entraînement sur 900 paires et une validation sur 100 paires en conservant le ratio de parallélépipèdes et de couloirs. Afin d'estimer la distance entre les modèles de distributions prédits, nous avons choisi la divergence de Kullback-Leibler [KL51]. Cette métrique met en évidence la quantité d'information perdue avec la covariance prédite en comparaison de la covariance réelle. Les résultats sont présentés dans le tableau 4.2.

Tableau 4.2.: Evaluation de notre adaptation de la méthode CELLO3D sur un jeu de données synthétiques représentant des parallélépipèdes et des couloirs infinis échantillonnés par des points.

	CELLO 3D	Adaptation proposée
Kullback-Leibler - Parallélépipèdes	7.04	8.34
Kullback-Leibler - Couloirs	24.78	8.96
Temps d'apprentissage par itération (secondes)	16.6	10.5

A la lecture de ce tableau, on peut conclure que l'adaptation proposée est plus fiable pour prédire les covariances de systèmes sous-contraints en translation. Avec l'algorithme proposé par LANDRY et coll., bien que la divergence de Kullback-Leibler pour les nuages de points de type parallélépipède soit assez faible, elle est 3 à 4 fois plus importante pour les nuages de points de type « couloir ». Cela confirme que la séparation des groupes de l'apprentissage par géométrie manque de précision.

Afin de vérifier que les caractéristiques de bruit et d'orientation peuvent être mises en évidence par notre adaptation après l'entraînement sur le jeu de données $\mathcal{J}_{complet}$, nous testons les jeux de données $\mathcal{J}_{bruit}$ et $\mathcal{J}_{orient}$. Les matrices de poids relatives à $\mathcal{J}_{bruit}$ et $\mathcal{J}_{orient}$ sont présentées figure 4.8. La figure 4.8a montre que le bruit est pris en compte mais que les autres caractéristiques (géométrie et orientation principalement) compliquent sa mise en évidence. L'image est donc nettement moins contrastée que dans la figure 4.4. La figure 4.8b montre que l'orientation est bien prise en compte dans l'estimation mais, tout comme le bruit, les autres caractéristiques mitigent sa mise en évidence puisque les groupes dûs à la binarisation de l'histogramme réapparaissent clairement. La métrique M ne serait donc pas suffisante pour permettre de corréler avec précision les différents facteurs dont dépendent la forme et l'amplitude de la matrice de covariance.



(a) Bruit **(b)** Orientation

Figure 4.8.: Matrices de poids finales après entraînement par notre adaptation sur le jeu de données complet de différents jeux de données. **(a)** Evaluation sur le jeu de données $\mathcal{J}_{bruit}$ permettant d'évaluer la prise en compte du bruit du capteur. **(b)** Evaluation sur le jeu de données $\mathcal{J}_{orient}$ permettant d'évaluer la prise en compte de l'orientation d'un objet de type « couloir ».

4.6 Conclusion

Les algorithmes de recalage commettent une erreur d'estimation qu'il est important de pouvoir quantifier afin de corriger une suite de recalages par fermeture de boucle ou d'estimer une pose dans le cadre de la navigation robotique. Nous avons vu que la majorité des chaînes de traitement d'estimation de poses préfèrent utiliser des modèles gaussiens de par leur facilité d'utilisation et leur adaptabilité aux algorithmes d'estimation tels que le filtre de Kalman. Cependant, l'ajustement de ces modèles gaussiens peut s'avérer compliqué à cause des nombreux facteurs qui influencent la précision du recalage comme, par exemple, l'échantillonnage des scènes, le bruit induit par le capteur, la forme de la zone de chevauchement, la densité de points, la géométrie de la scène, etc. La majorité des algorithmes de l'état de l'art ne parviennent pas à prendre tous ces facteurs en compte. De plus, il apparaît difficile d'implémenter un algorithme valide pour différentes méthodes de recalage. Nous avons donc étudié l'algorithme CELLO qui propose de passer par un apprentissage automatique à partir de données d'erreur de recalage en associant des descripteurs aux scans de points. Cette approche peut ainsi être utilisée sur des données synthétiques ce qui permettrait de faire varier les différents facteurs d'erreur des algorithmes tout en conservant une vérité terrain fiable. De plus, elle serait applicable à tous les algorithmes de recalage. Cependant, elle requiert une large base de données et des descripteurs adaptés. Nous avons testé l'algorithme CELLO-3D sur des données synthétiques avec l'algorithme ICP afin de mettre en évidence son comportement dans différentes configuration conformément à l'objectif 2 fixé en introduction de ce chapitre. Son

fonctionnement étant dépendant des descripteurs associés aux nuages de points, nous avons sélectionné de nouveaux descripteurs à intégrer à l'algorithme pour améliorer son fonctionnement dans le cadre d'applications LiDAR en intérieur et répondre à l'objectif 3. Afin de ne pas alourdir l'apprentissage, nous avons également modifié la fonction de poids employée. Les résultats de notre évaluation montrent que l'algorithme est capable de séparer des groupes de paires de scans par apprentissage de leur covariance selon le bruit du capteur, la géométrie et l'orientation de leurs zones de chevauchement. Cependant, la métrique apprise par l'algorithme semble ne pas être suffisante pour corréler avec précision tous les descripteurs entre eux. L'évaluation a donc bien permis de répondre à l'objectif 1. La perspective majeure de ce travail est de prédire la covariance de données réelles avec l'entraînement présenté ici sur les données simulées.

Conclusion générale

Contributions

Au cours de ce travail, nous avons cherché à participer à l'automatisation et à l'amélioration de la chaîne de traitements allant de l'acquisition de nuages de points par un dispositif LiDAR à la reconstruction 3D d'intérieurs de bâtiments. En effet, actuellement, ces traitements sont majoritairement manuels. L'acquisition LiDAR dresse de nombreux obstacles à la reconstruction automatique (anisotropie, bruit, occultations, etc.) et les méthodes existantes manquent de précision. De plus, ces dernières sont souvent difficilement adaptables à l'acquisition de différents types de scènes structurées et à l'utilisation de différents scanners LiDAR.

Dans la première partie de cette thèse, nous avons étudié le processus de modélisation de scans de nuages de points. Les enjeux d'une telle modélisation sont multiples et, selon l'application, les critères de reconstruction peuvent fluctuer. L'état de l'art propose majoritairement des méthodes qui extrapolent les nuages de points d'entrée afin de combler les occultations et d'obtenir un objet directement manipulable par des spécialistes du bâtiment. Cependant, ces modèles sont fondés sur de fortes hypothèses de construction et présentent un grand nombre d'écarts avec la réalité des mesures. Nous avons alors fixé nos objectifs de reconstruction dont les points majeurs sont :

- une faible extrapolation des données de nuages de points,
- une représentation surfacique sous forme de polygones,
- la mise en évidence des zones d'occultation et d'ouverture.

Nous avons d'abord cherché un outil de modélisation de surfaces locales dans le but de l'intégrer à une modélisation globale. L'état de l'art de ce domaine de recherche est très dense, cependant, l'estimation de normales présente encore des limites. Les méthodes existantes se déclinent en quatre catégories : les ajustements de surfaces, les approches dites de tirages aléatoires, les procédures *a posteriori* et les méthodes fondées sur une triangulation du nuage. Ces algorithmes peinent à être performants à la fois sur des données non-bruitées et sur des données bruitées et/ou contenant des points aberrants. De plus, l'anisotropie de l'échantillonnage est un frein majeur à leur bon fonctionnement. En outre, les coins et les arêtes sont rarement pris en compte dans l'estimation, or la précision de détection de ces caractéristiques est déterminante pour parvenir à une modélisation globale fiable dans notre chaîne de traitements. Enfin, les temps de calcul des algorithmes les plus performants restent élevés et leur application à un contexte LiDAR sur des données de plusieurs millions de points peut s'avérer difficilement réalisable. Nous avons proposé une méthode d'estimation de normales rapide, qui prend en compte les discontinuités de courbure et dont la robustesse au bruit est adaptée au type de données acquises par un dispositif LiDAR. Cette méthode ajuste des plans locaux et utilise la théorie des M-estimateurs pour séparer le voisinage et éliminer les points aberrants. Une évaluation de notre méthode sur différents jeux de données a démontré son efficacité. Dans cette évaluation, nous avons comparé notre approche à sept méthodes de l'état de l'art mettant ainsi en évidence les avantages et inconvénients de chacune d'elles et l'adaptation de notre méthode au type de données considéré.

Ensuite, la modélisation globale d'un bâtiment a été étudiée. Les méthodes actuelles reposent principalement sur une détection de plans horizontaux et verticaux. Les outils de segmentation par plans tels que Hough, RANSAC ou la croissance de régions sont utilisés et adaptés à chaque méthode. De plus, une majorité d'approches passent par une construction d'un arrangement de lignes ou de plans afin de compléter les zones occultées et les surfaces manquées dans la détection. Ces méthodes tendent à extrapoler significativement le nuage de points afin d'obtenir une représentation étanche et robuste au bruit et aux occultations. Malheureusement, ces reconstructions ne sont pas toujours fidèles au nuage de points acquis et reposent sur de nombreuses hypothèses de construction, ce qui mène à une réduction de leur applicabilité sur des données réelles. Nous avons souhaité développer une nouvelle méthode, plus proche des données, qui permettrait de pallier les problèmes liés à l'anisotropie de l'échantillonnage tout en mettant en évidence les zones occultées et les ouvertures. Cette méthode repose sur un traitement conjoint du nuage de points et de l'image d'acquisition angulaire relative. Une détection de primitives planaires est suivie par une détection de segments de droites et enfin de coins, afin de former des polygones plans. Un étiquetage est réalisé en continu pendant ces étapes afin de conserver les informations d'occultation et d'ouverture. De plus, les objets non planaires ne sont pas reconstruits mais conservés sous forme de nuages de points. Notre méthode donne des résultats fiables même pour des objets fins et permet de rendre compte visuellement à l'utilisateur de la qualité du scan et des zones occultées.

La deuxième partie de cette thèse a porté sur l'étape de connexion des scans les uns avec les autres. En effet, le processus d'acquisition LiDAR impose de réaliser des mesures selon différents points de vue dans un bâtiment fournissant ainsi un ensemble de scans dans le repère du capteur. La mise en commun de ces scans dans un repère global requiert alors une information de position du scanner pour chaque point de vue. Cette position est difficile à obtenir avec des capteurs et peut alourdir la tâche d'acquisition. Nous avons donc exploré le domaine du recalage de nuages de points afin de déterminer les méthodes les plus adaptées au contexte d'intérieur de bâtiment. Les méthodes les plus employées reposent sur une approche locale mettant en correspondance des points des nuages relativement à leur voisinage. Un procédé itératif permet alors d'actualiser ces correspondances et de déplacer les nuages. Les approches locales sont difficilement utilisables en intérieur à cause de l'uniformité du voisinage d'une majorité de points, notamment ceux situés sur des plans. De plus, de nombreuses méthodes requièrent un mouvement faible entre scans ce qui peut compliquer la tâche d'acquisition. Nous avons proposé un nouvel algorithme global qui utilise la structure des scènes afin d'estimer séparément la rotation et la translation de la transformation rigide. Cette méthode a fait l'objet d'une publication dans le journal international *Remote Sensing* (MDPI) en 2017 [San+17b].

Enfin, l'erreur commise par la transformation doit être obtenue en vue d'ajuster globalement une suite de recalages lors de fermetures de boucles ou pour fusionner le résultat du recalage à un ensemble de données de localisation extérieures. Pourtant, les estimateurs proposés par l'état de l'art sont, soit trop restrictifs, soit trop permissifs et ne permettent pas de fusionner correctement les données collectées par le recalage. Parmi les pistes de recherches qui ont été explorées pour estimer l'erreur d'un recalage, une méthode récente de l'état de l'art nommée CELLO-3D et fondée sur l'apprentissage automatique, a été particulièrement développée dans ce travail. La phase d'apprentissage permet d'adapter l'estimation à différents algorithmes et environnements et permet en outre, une implémentation en temps réel. Cette approche repose sur la qualité de descripteurs 3D attachés aux nuages. Nous avons donc proposé une adaptation de ces descripteurs et du calcul de distance entre descripteurs afin d'améliorer la précision de l'estimation. L'évaluation de l'algorithme sur une base de données synthétiques permet de mettre en évidence ses performances dans des cas variés.

Perspectives

L'étude réalisée dans cette thèse a permis de mettre en évidence les difficultés rencontrées dans le processus de reconstruction d'un environnement intérieur à partir de données LiDAR. Les réponses proposées participent à l'amélioration de la chaîne de traitements complète et ouvrent la voie à de nombreux futurs travaux sur le sujet. Nous établissons ici une liste non exhaustive des travaux envisagés par la suite.

Court terme

Prédiction de l'erreur

A court terme, l'algorithme d'estimation de l'erreur de recalage (voir chapitre 4) devra être testé avec la méthode globale proposée chapitre 3. La base de données simulées sur laquelle s'appuie l'étude du chapitre 4 pourrait être largement étendue. En effet, celle-ci ne comporte qu'un système sous-contraint illustrant l'effet « couloir ». On pourrait créer beaucoup d'autres configurations comme des cylindres ou des plans. En outre, un plus grand nombre d'orientations des objets pourrait être intégré à l'apprentissage afin d'affiner la prédiction de la covariance d'un scan inconnu. Les données simulées pourraient également être obtenues à partir d'images de profondeur afin de mieux représenter l'échantillonnage des données réelles. Enfin, à partir de l'apprentissage sur des données simulées, il serait intéressant de tester la prédiction sur des données réelles. Les jeux de données décrits dans l'annexe B sont associés à des poses de vérité terrain et pourraient donc être utilisés pour évaluer la validité de la méthode.

Fusion des modèles de scans

Après avoir reconstruit les scans et les avoir recalés les uns sur les autres, une étape nécessaire à l'obtention d'un modèle complet serait la fusion des scans modélisés. En effet, comme expliqué en introduction de ce travail de thèse, la majorité des chaînes de traitement de reconstruction recalent les nuages de points avant d'opérer leur modélisation. Cependant, le lien avec les images de profondeur étant perdu, la segmentation du nuage est plus complexe et requiert plus de paramètres. Nous avons montré que des résultats précis pouvaient être obtenus en modélisant au préalable les scans et en alignant les modélisations *a posteriori* (voir chapitre 2). Dans le prolongement direct des travaux de cette thèse, l'étape finale de la chaîne de traitements consisterait donc à fusionner les polygones de deux scans alignés afin d'obtenir un résultat comparable aux méthodes de reconstruction de l'état de l'art [Cha+10 ; Och+16 ; Oes+14]. Pour ce faire, une première étape serait la mise en correspondance des polygones par paires. Ensuite, on peut envisager de moyenner les caractéristiques de leurs plans supports (*c.-à-d.* la normale et la distance à l'origine). Ces nouvelles caractéristiques mèneraient à une correction des segments du contour de chacun des polygones. Enfin, certains segments pourraient être fusionnés avant de rassembler les deux objets dans un nouveau polygone par une méthode telle que proposée par Rivero et Feito [RF00]. La difficulté de cette tâche résiderait alors dans la fusion des étiquetages des contours. Le résultat obtenu permettrait de minimiser les zones occultées et d'étendre la vue d'un scan à l'autre.

Comparaison des recalages

Nous avons proposé un algorithme de recalage pour des nuages de points 3D. Nous avons vu que cet algorithme était adaptable aux modèles reconstruits par la méthode présentée dans le chapitre 2. Les qualités des recalages entre nuages de points et entre scans modélisés pourraient être comparées. En effet, la modélisation proposée peut,

en théorie, permettre de se défaire du bruit du capteur sur la précision de localisation des plans et améliorer la qualité des normales en entrée de l'algorithme de recalage et donc améliorer la précision du recalage. De plus, les problèmes liés à l'anisotropie de l'échantillonnage sur l'évaluation de la transformation la plus adaptée pourraient être résolus. Cependant, une comparaison approfondie est nécessaire pour vérifier cette hypothèse.

Long terme

Diversification des surfaces

Peu d'auteurs proposent une reconstruction prenant en compte les murs courbes de l'environnement. Oesau et coll. [Oes+14] proposent une approche 2D à partir de la mise en commun de droites ajustées localement. Cette approche mène à une reconstruction des murs courbes par un ensemble de plans fins connectés. Cependant, la précision de cette reconstruction reste faible. A l'instar d'auteurs tels que Guennebaud et Gross [GG07], Ouyang et Yung [OY05] ou Cazals et Pouget [CP05], une diversification des modèles de surfaces locales à ajuster pourrait être envisagée. Plusieurs modèles pourraient être ajustés sur un voisinage et le meilleur modèle pourrait être sélectionné. La segmentation du nuage de points pourrait alors être modifiée afin de prendre en compte les nouveaux modèles. Oesau et coll. [Oes+14] ajustent deux modèles à un nuage de points 2D. Pour déterminer le meilleur modèle localement, ils utilisent une heuristique fondée sur l'accroissement du voisinage. Cependant, cette approche reste locale et pourrait complexifier une mise en commun dans le processus de segmentation. Une optimisation globale pourrait être appliquée à partir d'estimateurs robustes afin d'ajuster les modèles adéquats et d'exclure les modèles non adaptés. Cette optimisation pourrait prendre en compte à la fois la précision de l'ajustement local et la cohérence globale des voisinages. Une autre piste de recherche serait de détecter les surfaces globales à ajuster directement sur l'image $\{\varphi, \theta\}$ telle que décrite dans le chapitre 3.

Intégration dans des processus de SLAM

Une fois l'estimation de l'erreur d'alignement précisée, l'algorithme de recalage pourrait être intégré à un processus de SLAM [ZS14 ; Nüc+07]. Les problématiques de temps devraient être travaillées afin de rendre l'algorithme plus efficace. En outre, dans le cadre du SLAM à haute vitesse, le chevauchement des scans peut être faible. Une étude approfondie de la robustesse face à de tels chevauchements pourrait être envisagée. De plus, des déformations importantes des objets peuvent être causées par la vitesse de déplacement du capteur. Plus généralement, une comparaison des performances de notre algorithme avec les méthodes existantes selon la vitesse du scanner serait une contribution intéressante à l'état de l'art afin d'aider les utilisateurs à sélectionner les méthodes adéquates selon les besoins des contextes d'acquisition.

SLAM actif

Notre modélisation intègre des informations sémantiques (occultations, objets, ouvertures) qui pourraient aider au guidage d'un utilisateur ou d'un robot pendant l'acquisition des données. Le nouveau meilleur point de vue pour le scan d'une pièce pourrait notamment être recalculé à partir des informations d'occultation et d'ouvertures. La « perception active » est le domaine de recherche qui étudie le lien entre l'observation par des capteurs et l'action d'un agent (un robot par exemple) en réponse à cette observation. Le but de cette action est d'enrichir les informations déjà collectées [Baj+18]. Dans le cadre de l'acquisition LiDAR, la perception active est adaptée en SLAM actif (*active SLAM*). Les informations fournies par notre méthode pourraient donc être utilisées par une méthode de SLAM actif. Les méthodes de perception active actuellement intégrées dans le SLAM peuvent avoir plusieurs objectifs. Le premier est de réaliser des fermetures de boucles afin de corriger une erreur d'estimation devenue trop importante [KE13]. L'exploration de l'environnement est un second objectif [KE15]. Dans un environnement intérieur, cela consisterait à visiter toutes les pièces d'un bâtiment de manière efficace. En outre, dans le cadre de la reconstruction 3D, la minimisation des zones occultées est un enjeu majeur afin d'obtenir une modélisation de l'environnement étanche et proche des données. La majorité des méthodes de SLAM actif fonctionnent en définissant une métrique donnant un score au scan acquis et en maximisant ce score relativement au déplacement. Dans notre cas, bien qu'un score soit éventuellement calculable par estimation de l'aire des zones occultées et détection des ouvertures, la maximisation relativement au déplacement semble complexe. Dans le contexte de navigation en environnement intérieur, on peut imaginer deux phases. Premièrement, l'aire des occultations sur l'enveloppe d'une pièce serait minimisée, puis le déplacement serait orienté vers l'ouverture détectée pour explorer les pièces adjacentes. La première étape précédemment citée pourrait éventuellement être réalisée en minimisant la distance aux contours occultés précédemment détectés par l'algorithme de modélisation.

Autres contextes d'application de reconstruction 3D

Les outils développés dans cette thèse pourraient également être utilisés dans un cadre plus large de robotique mobile. En effet, il est possible d'intégrer l'algorithme de recalage à une chaîne de traitements permettant la reconstruction d'objets structurés échantillonnés par des points par exemple. Dans ce cadre, d'autres dispositifs d'acquisition peuvent être utilisés tels que des caméras de profondeur. L'algorithme de recalage devrait alors faire face à de fortes contraintes de temps de calcul. Pour aller plus loin, la reconstruction des scans pourrait permettre de maximiser la vue de l'objet, grâce à de la perception active, dans un objectif de reconnaissance notamment [Baj+18].

Notations

Général

$\mathscr{S}$	Nuage de points
$\mathbf{v}$	Vecteur colonne de trois éléments
$\bar{\mathbf{v}}$	Vecteur moyen
$\mathscr{A}$	Ensemble
$\#A$	Cardinal de l'ensemble A

. (4.15)

Chapitre I

N	Ensemble de normales
$S(\mathbf{p})$	Voisinage du point$\mathbf{p}$
$\hat{v}$	Vecteur initial (avant traitement)
v^*	Vecteur estimé
r	résidu
w	poids

. (4.16)

Chapitre II

N_X	Nombre d'éléments de X
π	Plan
$\mathscr{L}$	Ensemble de lignes
Δ	Ensemble de segments de droites
$\mathscr{C}$	Ensemble de coins
ℓ	ligne
δ	Segment de droite
c	coin

. (4.17)

Chapitre III

R	Rotation
T	Transformation

. (4.18)

Bibliographie

[AB14] A. Y. Aravkin et J. V. Burke. « Smoothing dynamic systems with state-dependent covariance matrices ». In : *53rd IEEE Conference on Decision and Control*. IEEE. 2014, p. 3382-3387 (cité à la page 124).

[AB99] N. Amenta et M. Bern. « Surface reconstruction by Voronoi filtering ». In : *Discrete and Computational Geometry* 22.4 (1999), p. 481-504 (cité à la page 24).

[Ale+03] M. Alexa, J. Behr, D. Cohen-Or et coll. « Computing and rendering point set surfaces ». In : *IEEE Transactions on Visualization and Computer Graphics* 9.1 (2003), p. 3-15 (cité à la page 16).

[All+07] P. Alliez, D. Cohen-Steiner, Y. Tong et M. Desbrun. « Voronoi-based variational reconstruction of unoriented point sets ». In : *Proceedings of the fifth Eurographics symposium on Geometry processing* (2007), p. 39-48 (cité à la page 24).

[Aud19] W. Audureau. « Notre-Dame de Paris : les reconstitutions en 3D peuvent aider à la reconstruction ». In : *Le Monde* (avr. 2019) (cité à la page 1).

[Avr+10] H. Avron, A. Sharf, C. Greif et D. Cohen-Or. « ℓ_1-Sparse reconstruction of sharp point set surfaces ». In : *ACM Transactions on Graphics* (oct. 2010), p. 1-12 (cité à la page 22).

[Baj+18] R. Bajcsy, Y. Aloimonos et J. K. Tsotsos. « Revisiting active perception ». In : *Autonomous Robots* 42.2 (2018), p. 177-196. arXiv : 1603.02729 (cité à la page 144).

[Bar84] A. Barr. « Global and local deformations of solid primitives ». In : *Proc. Siggraph* (1984), p. 21-30 (cité à la page 20).

[BB01] O. Bengtsson et A. J. Baerveldt. « Localization in changing environments - Estimation of a covariance matrix for the IDC algorithm ». In : *IEEE International Conference on Intelligent Robots and Systems* 4 (2001), p. 1931-1937 (cité à la page 125).

[BB03] O. Bengtsson et A. J. Baerveldt. « Robot localization based on scan-matching - Estimating the covariance matrix for the IDC algorithm ». In : *Robotics and Autonomous Systems* 44.1 (2003), p. 29-40 (cité à la page 125).

[BB10] A. Budroni et J. Boehm. « Automated 3D Reconstruction of Interiors from Point Clouds Automated 3D Reconstruction of ». In : *International Journal of Architectural Computing* 08.01 (2010), p. 55-74 (cités aux pages 57, 58, 60).

[BJ88] P. Besl et R. Jain. « Segmentation through variable-order surface fitting ». In : *IEEE Trans. Pattern Anal. Mach. Intell.* 10.2 (1988), p. 167-192 (cités aux pages 55, 57).

[BM12] A. Boulch et R. Marlet. « Fast normal estimation for point clouds with sharp features using a robust randomized Hough transform ». In : *Computer Graphics Forum* 31.5 (2012), p. 1765-1774 (cités aux pages 13, 18, 35, 36).

[BM16] A. Boulch et R. Marlet. « Deep learning for robust normal estimation in unstructured point clouds ». In : *Computer Graphics Forum* 35.5 (2016), p. 281-290 (cités aux pages 12, 18, 35, 51).

[BM92] P. Besl et N. McKay. « A Method for Registration of 3-D Shapes ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), p. 239-256 (cités aux pages 85, 87, 89, 90).

[Bor+11] D. Borrmann, J. Elseberg, K. Lingemann et A. Nüchter. « The 3D Hough Transform for plane detection in point clouds : A review and a new accumulator design ». In : *3D Research* 2.2 (2011), p. 1-13 (cité à la page 55).

[Bou+14] A. Boulch, M. d. L. Gorce et R. Marlet. « Piecewise-planar 3D reconstruction with edge and corner regularization ». In : *Eurographics Symposium on Geometry Processing* 33.5 (2014), p. 55-64 (cités aux pages 57, 59, 61, 64).

[BR05] F. Bretar et M. Roux. « Hybrid image segmentation using LiDAR 3D planar primitives ». In : *ISPRS Proceedings. Workshop Laser scanning, the Netherlands* 78 (sept. 2005), p. 12-14 (cité à la page 56).

[Bro84] P. Brou. « Using the Gaussian Image to Find the Orientation of Objects ». In : *The International Journal of Robotics Research* 3.4 (1984), p. 89-125 (cité à la page 88).

[Cad+16] C. Cadena, L. Carlone, H. Carrillo et coll. « Past, present, and future of simultaneous localization and mapping : towards the robust-perception age ». In : *IEEE Transactions on Robotics* 32.6 (2016), p. 1-27 (cités aux pages 85, 91).

[CC08] C. Chen et K. Cheng. « A sharpness-dependent filter for recovering sharp features in repaired 3D mesh models ». In : *IEEE Transactions on Visualization and Computer Graphics* 14.1 (jan. 2008), p. 200-212 (cité à la page 24).

[CC09] A. Censi et S. Carpin. « HSM3D : Feature-Less Global 6DOF Scan-Matching in the Hough/Radon Domain ». In : *Robotics and Automation* (mai 2009), p. 3899-3906 (cité à la page 55).

[Cen07] A. CENSI. « An accurate closed-form estimate of ICP'S covariance ». In : *Proceedings - IEEE International Conference on Robotics and Automation* May 2007 (2007), p. 3167-3172 (cité à la page 125).

[Cha+10] A. L. CHAUVE, P. LABATUT et J. P. PONS. « Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data ». In : *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), p. 1261-1268 (cités aux pages 12, 57, 59, 61, 142).

[Cha+92] G. CHAMPLEBOUX, S. LAVALLEE, R. SZELISKI et L. BRUNIE. « From accurate range imaging sensor calibration to accurate model-based 3D object localization ». In : *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Juin 1992, p. 83-89 (cité à la page 90).

[Che+19] H. CHEN, J. HUANG, O. REMIL et coll. « Structure-guided shape-preserving mesh texture smoothing via joint low-rank matrix recovery ». In : *Computer-Aided Design* 115 (2019), p. 122-134 (cité à la page 23).

[CM92] Y. CHEN et G. MEDIONI. « Object Modeling by Registration of Multiple Range Images ». In : *Image Vision Comput.* 10 (jan. 1992), p. 145-155 (cités aux pages 87, 89, 90).

[CO18] E. CHE et M. J. OLSEN. « Multi-scan segmentation of terrestrial laser scanning data based on normal variation analysis ». In : *ISPRS Journal of Photogrammetry and Remote Sensing* 143 (2018), p. 233-248 (cité à la page 24).

[CP05] F. CAZALS et M. POUGET. « Estimating differential quantities using polynomial fitting of osculating jets ». In : *Computer Aided Geometric Design* 22.2 (2005), p. 121-146 (cités aux pages 14, 15, 35, 143).

[CY99] J. M. COUGHLAN et A. L. YUILLE. « Manhattan world : Compass direction from a single image by bayesian inference ». In : *Proceedings of the Seventh IEEE International Conference on Computer Vision*. T. 2. IEEE. 1999, p. 941-947 (cités aux pages 4, 57).

[Cze+18] T. CZERNIAWSKI, B. SANKARAN, M. NAHANGI, C. HAAS et F. LEITE. « 6D DBSCAN-based segmentation of building point clouds for planar object classification ». In : *Automation in Construction* 88 (2018), p. 44-58 (cité à la page 56).

[DG10] J.-E. DESCHAUD et F. GOULETTE. « A fast and accurate plane detection algorithm for large noisy point clouds using filtered normals and voxel growing ». In : *3D Data Processing, Visualization, and Transmission* (mai 2010) (cité à la page 57).

[DH71] R. O. DUDA et P. E. HART. « Use of the Hough Transformation to Detect Lines and Curves in Pictures ». In : *Technical Note, Artificial Intelligence Center, SRI International* 36 (1971) (cité à la page 55).

[Dro+08] A. DROR, M. NILOY J et C.-o. DANIEL. « 4-Points Congruent Sets for Robust Pairwise Surface Registration ». In : *ACM Transactions on Graphics (SIG- GRAPH)* 27.3 (août 2008), p. 10 (cité à la page 92).

[DS06] T. Dey et J. Sun. « Normal and feature approximations from noisy point clouds ». In : *Foundations of Software Technology and Theoretical Computer Science* 4337 (2006), p. 21-32 (cité à la page 24).

[Els+10] J. Elseberg, D. Borrmann, K. Lingemann et A. Nüchter. « Non-Rigid Registration and Rectification of 3D Laser Scans ». In : *International Conference on Intelligent Robots and Systems*. Oct. 2010, p. 1546-1552 (cité à la page 166).

[Epa69] V. Epanechnikov. « Nonparametric estimation of a multidimensional probability density ». In : *Theory of Probability and its Applications* 14.1 (1969), p. 153-158 (cité à la page 96).

[Fis+81] M. Fischler, R. Bolles et A. Bruckstein. « Random Sample Consensus : A paradigm for model fitting with application to image analysis and automated cartography ». In : *Communications of the ACM* 24.6 (1981), p. 381-395 (cités aux pages 56, 92).

[Fit+97] A. Fitzgibbon, D. Eggert et R. Fisher. « High-level model acquisition from range images ». In : *Computer-Aided Design* 29.4 (1997), p. 321-330 (cité à la page 57).

[Fit03] A. W. Fitzgibbon. « Robust registration of 2D and 3D point sets ». In : *Image and vision computing* 21.13-14 (2003), p. 1145-1153 (cité à la page 90).

[Fle+03] S. Fleishman, I. Drori et D. Cohen-Or. « Bilateral mesh denoising ». In : *ACM Trans. Graph.* 22.3 (juil. 2003), p. 950-953 (cité à la page 20).

[Fle+05] S. Fleishman, D. Cohen-Or et C. Silva. « Robust moving least-squares fitting with sharp features ». In : *ACM SIG-GRAPH* 24.3 (2005), p. 544-552 (cités aux pages 13, 15-17, 21).

[GA05] A. Gruen et D. Akca. « Least squares 3D surface and curve matching ». In : *ISPRS Journal of Photogrammetry and Remote Sensing* 59.3 (2005), p. 151-174 (cités aux pages 89, 90).

[Gel+03] N. Gelfand, L. Ikemoto, S. Rusinkiewicz et M. Levoy. « Geometrically stable sampling for the ICP algorithm ». In : *International Conference on 3-D Digital Imaging and Modeling*. Oct. 2003, p. 260-267 (cité à la page 90).

[GG07] G. Guennebaud et M. Gross. « Algebraic point set surfaces ». In : *ACM Transactions on Graphics* 26.3 (2007), p. 23 (cités aux pages 14, 143).

[Gou71] H. Gouraud. « Continuous shading of curved surfaces ». In : *IEEE Transactions on Computers* C.20 (1971), p. 623-629 (cité à la page 18).

[Gue+18] P. Guerrero, Y. Kleiman, M. Ovsjanikov et N. J. Mitra. « Learning local shape properties from raw point clouds ». In : *Computer Graphics Forum* 37.2 (2018), p. 75-85 (cités aux pages 19, 35).

[Han+13] J. Han, N. Perng et Y. Lin. « Feature Conjugation for Intensity Coded LiDAR Point Clouds ». In : *IEEE Geoscience and Remote Sensing Letters* 139.3 (2013), p. 135-142 (cité à la page 91).

[He+14] R. He, B. Hu, X. Yuan et L. Wang. *Robust recognition via information theoretic learning*. 2014 (cité à la page 25).

[Hic19] N. Hichri. « Bim bang : why bim and 3D mockups are key factors for AEC ». In : *it3D* (2019), p. 44-48 (cités aux pages 1, 5).

[HK87] R. Hoffman et J. A. K. « Segmentation and classification of range images ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 9.5 (1987), p. 608-620 (cité à la page 14).

[Hok] Hokuyo. *Hokuyo website*. www.hokuyo-aut.jp (cité à la page 165).

[Hol+15] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu et S. Behnke. « Registration with the Point Cloud Library : A Modular Framework for Aligning in 3-D ». In : *IEEE Robotics Automation Magazine* 22.4 (déc. 2015), p. 110-124 (cité à la page 91).

[Hon+15] S. Hong, J. Jung, S. Kim et coll. « Semi-automated approach to indoor mapping for 3D as-built building information modeling ». In : *Computers, Environment and Urban Systems* 51 (2015), p. 34-46 (cité à la page 56).

[Hop+92] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald et W. Stuetzle. « Surface reconstruction from unorganized points ». In : *ACM SIGGRAPH Computer Graphics* 26.2 (1992) (cités aux pages 13, 14, 35, 94).

[Hor+88] B. Horn, H. Hilden et S. Negahdaripour. « Closed-Form Solution of Absolute Orientation using Orthonormal Matrices ». In : *Journal of the Optical Society of America A* 5 (juil. 1988), p. 1127-1135 (cité à la page 90).

[Hor84] B. Horn. « Extended Gaussian images ». In : *Proceedings of the IEEE* 72.12 (déc. 1984), p. 1671-1686 (cité à la page 88).

[HS13] L. He et S. Schaefer. « Mesh denoising via L0 minimization ». In : *ACM Trans. Graph.* 32.4 (juil. 2013), p. 1-8 (cité à la page 23).

[HS88] C. Harris et M. Stephens. « A combined corner and edge detector ». In : *In Proc. of Fourth Alvey Vision Conference*. 1988, p. 147-151 (cité à la page 91).

[Hu+03] C. Hu, W. Chen, Y. Chen, D. Liu et coll. « Adaptive Kalman filtering for vehicle navigation ». In : *Journal of Global Positioning Systems* 2.1 (2003), p. 42-47 (cité à la page 124).

[Hua+13] H. Huang, S. Wu, M. Gong et coll. « Edge-aware point set resampling ». In : *ACM Trans. Graph.* 32.1 (jan. 2013), p. 1-12 (cités aux pages 13, 16, 24, 35).

[Hub11] P. J. Huber. « Robust statistics ». In : *International Encyclopedia of Statistical Science* (2011). Sous la dir. de M. Lovric, p. 1248-1251 (cités aux pages 17, 53).

[Ive+17] T. M. Iversen, A. G. Buch et D. Kraft. « Prediction of ICP pose uncertainties using Monte Carlo simulation with synthetic depth images ». In : *IEEE International Conference on Intelligent Robots and Systems* 2017-September (2017), p. 4640-4647 (cité à la page 125).

[JH99] A. E. JOHNSON et M. HEBERT. « Using spin images for efficient object recognition in cluttered 3D scenes ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 21.5 (1999), p. 433-449 (cité à la page 91).

[Jin+05] S. JIN, R. R. LEWIS et D. WEST. « A Comparison of algorithms for vertex normal computation ». In : *The Visual Computer* 21.1–2 (2005), p. 71-82 (cité à la page 18).

[Jon+03] T. JONES, F. DURAND et M. DESBRUN. « Non-iterative, feature-preserving mesh smoothing ». In : *ACM Trans. Graph.* 22.3 (juil. 2003), p. 943-949 (cité à la page 20).

[Jon+04] T. JONES, F. DURAND et M. ZWICKER. « Normal improvement for point rendering ». In : *Computer Graphics Forum* 24.4 (2004), p. 53-56 (cité à la page 20).

[Kaw+14] K. KAWASHIMA, S. KANAI et H. DATE. « As-built modeling of piping system from terrestrial laser-scanned point clouds using normal-based region growing ». In : *Journal of Computational Design and Engineering* 1.1 (2014), p. 13-26 (cité à la page 11).

[KDV14] K. KHOSHELHAM et L. DÍAZ-VILARIÑO. « 3D modelling of interior spaces : Learning the language of indoor architecture ». In : *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives* 40.5 (2014), p. 321-326 (cité à la page 53).

[KE13] A. KIM et R. M. EUSTICE. « Perception-driven navigation : Active visual SLAM for robotic area coverage ». In : *Proceedings - IEEE International Conference on Robotics and Automation* (2013), p. 3196-3203 (cité à la page 144).

[KE15] A. KIM et R. M. EUSTICE. « Active visual SLAM for robotic area coverage : Theory and experiment ». In : *International Journal of Robotics Research* 34.4-5 (2015), p. 457-475 (cité à la page 144).

[Kel+19] J. R. KELLNER, J. ARMSTON, M. BIRRER et coll. « New Opportunities for Forest Remote Sensing Through Ultra-High-Density Drone Lidar ». In : *Surveys in Geophysics* 40.4 (juil. 2019), p. 959-977 (cité à la page 1).

[KI91] S. B. KANG et K. IKEUCHI. « Determining 3-D object pose using the complex extended Gaussian image ». In : *Computer Vision and Pattern Recognition (CVPR)*. 1991, p. 580-585 (cités aux pages 88, 94).

[Kim+09] K. H. KIM, J. G. LEE et C. G. PARK. « Adaptive two-stage extended Kalman filter for a fault-tolerant INS-GPS loosely coupled system ». In : *IEEE Transactions on Aerospace and Electronic Systems* 45.1 (2009), p. 125-137 (cité à la page 124).

[Kir+91] N. KIRYATI, Y. ELDAR et A. BRUCKSTEIN. « A Probabilistic Hough Transform ». In : *Pattern Recognition* 24.4 (1991), p. 303-316 (cité à la page 55).

[KL51] S. KULLBACK et R. A. LEIBLER. « On information and sufficiency ». In : *The annals of mathematical statistics* 22.1 (1951), p. 79-86 (cité à la page 136).

[Lac+19] J. LACONTE, S.-P. DESCHENES, M. LABUSSIERE et F. POMERLEAU. « Lidar Measurement Bias Estimation via Return Waveform Modelling in a Context of 3D Mapping ». In : *International Conference on Robotics and Automation* (2019), p. 8100-8106 (cité à la page 3).

[Lai+15] R. LAING, M. LEON, J. ISAACS et D. GEORGIEV. « Scan to BIM : the development of a clear workflow for the incorporation of point clouds within a BIM environment ». In : *Archimag* 1 (2015), p. 279-289 (cité à la page 51).

[Lan+19] D. LANDRY, F. POMERLEAU et P. GIGUERE. « CELLO-3D : Estimating the Covariance of ICP in the Real World ». In : *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, p. 8190-8196 (cités aux pages 86, 122, 125, 126, 129-136).

[Lei] LEICA. *Leica website*. w3.leica-geosystems.com (cité à la page 165).

[Lip+03] Y. LIPMAN, DANIELCOHEN-OR, D. LEVIN et H. TAL-EZER. « Mesh-independent surface interpolation ». In : *Geometric Modeling for Scientific Visualization* (2003). Sous la dir. de SPINGER-VERLAG, p. 37-49 (cité à la page 16).

[Lip+07] Y. LIPMAN, D. COHEN-OR, D. LEVIN et H. TAL-EZER. « Parameterization-free projection for geometry reconstruction ». In : *ACM Transactions on Graphics* 26.3 (2007), p. 6 (cité à la page 16).

[Liu18] K. Y. LIU. « Learning Gaussisan noise models from high-dimensional sensor data with deep neural networks ». Thèse de doct. Massachusetts Institute of Technology, 2018 (cités aux pages 126, 131).

[LM97] F. LU et E. MILIOS. « Globally consistent range scan alignment for environment mapping ». In : *Autonomous robots* 4.4 (1997), p. 333-349 (cités aux pages 121, 124).

[Low04] D. G. LOWE. « Distinctive Image Features from Scale-Invariant Keypoints ». In : *International Journal of Computer Vision* 60.2 (nov. 2004), p. 91-110 (cité à la page 91).

[Mac+17] H. MACHER, T. LANDES et P. GRUSSENMEYER. « From Point Clouds to Building Information Models : 3D Semi-Automatic Reconstruction of Indoors of Existing Buildings ». In : *Applied Sciences* 7.10 (2017), p. 1030 (cités aux pages 4, 51, 53, 54, 56, 58, 59).

[Mag+07] M. MAGNUSSON, A. LILIENTHAL et T. DUCKETT. « Scan registration for autonomous mining vehicles using 3D-NDT ». In : *Journal of Field Robotics* 24.10 (2007), p. 803-827 (cité à la page 92).

[Mag09] M. MAGNUSSON. « The Three-Dimensional Normal-Distributions Transform — an Efficient Representation for Registration, Surface Analysis, and Loop Detection ». Thèse de doct. Orebro University, 2009, p. 220 (cité à la page 92).

[Mak+06] A. MAKADIA, A. PATTERSON et K. DANIILIDIS. « Fully Automatic Registration of 3D Point Clouds ». In : *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. T. 1. Juin 2006, p. 1297-1304 (cité à la page 89).

[Man+15] P. S. MANOJ, L. BINGBING, Y. RUI et W. LIN. « A closed-form estimate of 3D ICP covariance ». In : *Proceedings of the 14th IAPR International Conference on Machine Vision Applications, MVA 2015* 3 (2015), p. 526-529 (cité à la page 125).

[Med+03] B. MEDEROS, L. VELHO et L. H. de FIGUEIREDO. « Robust smoothing of noisy point clouds ». In : *SIAM Conference on Geometric Design and Computing* (2003) (cités aux pages 17, 25).

[Meh70] R. MEHRA. « On the identification of variances and adaptive Kalman filtering ». In : *IEEE Transactions on automatic control* 15.2 (1970), p. 175-184 (cité à la page 124).

[Mel+14] N. MELLADO, D. AIGER et N. J. MITRA. « Super 4PCS fast global pointcloud registration via smart indexing ». In : *Computer Graphics Forum* 33.5 (2014), p. 205-215 (cités aux pages 92, 107).

[Men+17] E. MENDES, P. KOCH, S. LACROIX et coll. « ICP-based pose-graph SLAM To cite this version : HAL Id : hal-01522248 ICP-Based Pose-Graph SLAM ». In : (2017), p. 195-200 (cité à la page 125).

[Mit+03] N. J. MITRA, A. NGUYEN et L. GUIBAS. « Estimating surface normals in noisy point cloud data ». In : *Nineteenth Annual Symposium on Computational Geometry* (2003), p. 322-328 (cité à la page 14).

[Mon+02] M. MONTEMERLO, S. THRUN, D. KOLLER, B. WEGBREIT et coll. « FastSLAM : A factored solution to the simultaneous localization and mapping problem ». In : *Aaai/iaai* 593598 (2002) (cité à la page 121).

[Mér+11] Q. MÉRIGOT, M. OVSJANIKOV et L. J. GUIBAS. « Voronoi-based curvature and feature estimation from point clouds ». In : *IEEE Transactions on Visualization and Computer Graphics* 17.6 (2011), p. 743-756 (cités aux pages 24, 35).

[Nan+12] L. NAN, K. XIE et A. SHARF. « A search-classify approach for cluttered indoor scene understanding ». In : *ACM Transactions on Graphics* 31.6 (2012), p. 1-10 (cité à la page 11).

[NL17] A. NÜCHTER et K. LINGEMANN. *Robotic 3D Scan Repository*. kos.informatik.uos.de/3Dscans/. 2017 (cité à la page 166).

[Nüc+07] A. NÜCHTER, K. LINGEMANN, J. HERTZBERG et H. SURMANN. « 6D SLAM-3D mapping outdoor environments ». In : *Journal of Field Robotics* 24.8-9 (2007), p. 699-722 (cités aux pages 90, 143).

[Och+16] S. OCHMANN, R. VOCK, R. WESSEL et R. KLEIN. « Automatic reconstruction of parametric building models from indoor point clouds ». In : *Computers and Graphics (Pergamon)* 54.5 (2016), p. 94-103 (cités aux pages 51, 56, 60, 61, 142).

[Oes+14] S. Oesau, F. Lafarge et P. Alliez. « Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut ». In : *ISPRS Journal of Photogrammetry and Remote Sensing* 90 (2014), p. 68-82 (cités aux pages 53, 54, 58, 60, 61, 142, 143).

[OY05] D. Ouyang et F. H. Yung. « On the normal vector estimation for point cloud data from smooth surfaces ». In : *Computer Aided Design* 37.10 (2005), p. 1071-1079 (cités aux pages 14, 143).

[Pat+09] K. Pathak, N. Vaskevicius et A. Birk. « Uncertainty analysis for optimum plane extraction from noisy 3D range-sensor point-clouds ». In : *Intelligent Service Robotics* 3.1 (2009), p. 37-48 (cité à la page 92).

[Pau+02] M. Pauly, M. Gross et L. Kobbelt. « Efficient simplification of point-sampled surfaces ». In : *IEEE conference on visualization* (2002), p. 163-170 (cité à la page 16).

[PB10] K. Pathak et A. Birk. « Fast Registration Based on Noisy Planes with Unknown Correspondences for 3D Mapping ». In : *IEEE Transactions on Robotics* 26.3 (2010), p. 424-441 (cité à la page 92).

[Pom+12] F. Pomerleau, M. Liu, F. Colas et R. Siegwart. « Challenging data sets for point cloud registration algorithms ». In : *The International Journal of Robotics Research* 31.14 (déc. 2012), p. 1705-1711 (cités aux pages 46, 165).

[Pom+13] F. Pomerleau, F. Colas, R. Siegwart et S. Magnenat. « Comparing ICP Variants on Real-World Data Sets ». In : *Autonomous Robots* 34.3 (fév. 2013), p. 133-148 (cité à la page 89).

[Pom+15] F. Pomerleau, F. Colas et R. Siegwart. « A Review of Point Cloud Registration Algorithms for Mobile Robotics ». In : *Foundations and Trends in Robotics* 4.1 (2015), p. 1-104 (cité à la page 89).

[Pop+08] J. Poppinga, N. Vaskevicius, A. Birk et K. Pathak. « Fast plane detection and polygonalization in noisy 3D range images ». In : *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS* (2008), p. 3378-3383 (cité à la page 57).

[PV06] S. Pu et G. Vosselman. « Automatic extraction of building features from terrestrial laser scanning ». In : *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 36.5 (2006), p. 25-27 (cité à la page 57).

[PY09] C. Poullis et S. You. « Automatic reconstruction of cities from remote sensor data ». In : *CVPR* (2009), p. 2775-2782 (cité à la page 57).

[RF00] M. Rivero et F. R. Feito. « Boolean operations on general planar polygons ». In : *Computers & Graphics* 24.6 (2000), p. 881-896 (cité à la page 142).

[Rus+09] R. B. Rusu, N. Blodow et M. Beetz. « Fast Point Feature Histograms (FPFH) for 3D registration ». In : *IEEE International Conference on Robotics and Automation* (2009), p. 3212-3217 (cité à la page 91).

[San+17a] J. SANCHEZ, F. DENIS, P. CHECCHIN, F. DUPONT et L. TRASSOUDAINE. *3D point cloud repository*. 2017 (cité à la page 46).

[San+17b] J. SANCHEZ, F. DENIS, P. CHECCHIN, F. DUPONT et L. TRASSOUDAINE. « Global registration of 3D LiDAR point clouds based on scene features : application to structured environments ». In : *Remote Sensing, MDPI* 9.10 (2017) (cité à la page 141).

[SB97] S. M. SMITH et J. M. BRADY. « SUSAN. A New Approach to Low Level Image Processing ». In : *International Journal of Computer Vision* 23.1 (mai 1997), p. 45-78 (cité à la page 91).

[Sch+07] R. SCHNABEL, R. WAHL et R. KLEIN. « Efficient RANSAC for point-cloud shape detection ». In : *Computer Graphics Forum* 26.2 (2007), p. 214-226 (cité à la page 55).

[Seg+09] A SEGAL, D HAEHNEL et S THRUN. « Generalized-ICP ». In : *Robotics : Science and Systems* 5 (2009), p. 168-176 (cité à la page 90).

[SHR17] O. SORKINE-HORNUNG et M. RABINOVICH. *Least-Squares Rigid Motion using SVD*. Rapp. tech. ETH Zurich, Department of Computer Science, 2017 (cité à la page 97).

[SIC] SICK. *SICK website*. www.sick.com (cité à la page 166).

[Spr+09] J. SPRICKERHOF, A. NÜCHTER, K. LINGEMANN et J. HERTZBERG. « An Explicit Loop Closing Technique for 6D SLAM. » In : *ECMR*. 2009, p. 229-234 (cité à la page 121).

[Sta+09] M. STAKKELAND, O. OVERREIN, E. F. BREKKE et O. HALLINGSTAD. « Tracking of targets with state dependent measurement errors using recursive BLUE filters ». In : *2009 12th International Conference on Information Fusion*. IEEE. 2009, p. 2052-2061 (cité à la page 124).

[Ste+11] B. STEDER, R. B. RUSU, K. KONOLIGE et W. BURGARD. « Point Feature Extraction on 3D Range Scans Taking into Account Object Boundaries ». In : *in proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2011 (cité à la page 91).

[Sun+07] X. SUN, P. ROSIN, R. MARTIN et F. LANGBEIN. « Fast and effective feature-preserving mesh denoising ». In : *Transactions on Visualization and Computer Graphics, IEEE* 13 (sept. 2007), p. 925-938 (cités aux pages 19, 23).

[Sun+15] Y. SUN, S. SCHAEFER et W. WANG. « Denoising point sets via L0 minimization ». In : *Computer Aided Geometric Design* 35.C (mai 2015), p. 2-15 (cité à la page 23).

[SZ12] V. SANCHEZ et A. ZAKHOR. « Planar 3D modeling of building interiors from point cloud data ». In : *2012 19th IEEE International Conference on Image Processing*. Oct. 2012, p. 1777-1780 (cités aux pages 11, 54, 56, 59).

[Tau01] G. TAUBIN. « Linear anisotropic mesh filters ». In : *RC22213 Computer Science* (nov. 2001), p. 110-51 (cité à la page 23).

[TB15] C. THOMSON et J. BOEHM. « Automatic geometry generation from point clouds for BIM ». In : *Remote Sensing* 7.9 (2015), p. 11753-11775 (cités aux pages 56, 59).

[Tex18] B. TEXIER. « Syrie : des Français au secours du patrimoine en péril ». In : *Archimag* (juin 2018) (cité à la page 1).

[Thr+98] S. THRUN, W. BURGARD et D. FOX. « A probabilistic approach to concurrent mapping and localization for mobile robots ». In : *Autonomous Robots* 5.3-4 (1998), p. 253-271 (cité à la page 124).

[TK+07] F. TARSHA-KURDI, T LANDES et P GRUSSENMEYER. « Hough-Transform and Extended Ransac Algorithms for Automatic Detection of 3D Building Roof Planes From Lidar Data ». In : *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007* XXXVI.1 (2007), p. 407-412 (cité à la page 56).

[TM98] C. TOMASI et R. MANDUCHI. « Bilateral filtering for gray and color images ». In : *International Conference on Computer Vision (ICCV '98* (1998), p. 839-846 (cité à la page 19).

[Tom+10] F. TOMBARI, S. SALTI et L. DI STEFANO. « Unique Signatures of Histograms for Local Surface Description ». In : *European Conference on Computer Vision (ECCV)*. 2010, p. 356-369 (cité à la page 91).

[TZ00] P. H. TORR et A. ZISSERMAN. « Mlesac : A new robust estimator with application to estimating image geometry ». In : *Computer Vision and Image Understanding* 78.1 (2000), p. 138-156 (cité à la page 56).

[TZ98] P. H. TORR et A. ZISSERMAN. « Robust computation and parametrization of multiple view relations ». In : *ICCV* 78.1 (1998), p. 727-732 (cité à la page 56).

[Van+12] C. A. VANEGAS, D. G. ALIAGA et B. BENES. « Automatic extraction of Manhattan-world building masses from 3D laser range scans ». In : *IEEE transactions on visualization and computer graphics* 18.10 (2012), p. 1627-1637 (cité à la page 58).

[VB+13] W. VEGA-BROWN, A. BACHRACH, A. BRY, J. KELLY et N. ROY. « CELLO : A fast algorithm for Covariance Estimation ». In : *Proceedings - IEEE International Conference on Robotics and Automation* Icra (2013), p. 3160-3167 (cités aux pages 126-128).

[VB13] W. VEGA-BROWN. « Predictive parameter estimation for Bayesian filtering ». Thèse de doct. Massachusetts Institute of Technology, 2013 (cités aux pages 86, 122, 131).

[VBR13] W. VEGA-BROWN et N. ROY. « CELLO-EM : Adaptive sensor models without ground truth ». In : *IEEE International Conference on Intelligent Robots and Systems* (2013), p. 1907-1914 (cité à la page 126).

[Vel] VELODYNE. *Velodyne website*. www.velodynelidar.com (cité à la page 165).

[Vit06] A. J. VITERBI. « A personal history of the Viterbi algorithm ». In : *IEEE Signal Processing Magazine* 23.4 (2006), p. 120-142 (cité à la page 123).

[Viv+12] D. VIVET, P. CHECCHIN et R. CHAPUIS. « Odométrie radar par analyse de la distorsion-Applications à la navigation de véhicules terrestres et nautiques ». In : *Traitement du Signal* 29.3-5 (2012), p. 205-228 (cité à la page 3).

[Viv+13] D. VIVET, P. CHECCHIN et R. CHAPUIS. « Localization and Mapping Using Only a Rotating FMCW Radar Sensor ». In : *Sensors* 13.4 (2013), p. 4527-4552 (cité à la page 113).

[Wan+16] F. WANG, Y. YE, X. HU et J. SHAN. « Point cloud registration by combining shape and intensity contexts ». In : *9th IAPR Workshop on Pattern Recognition in Remote Sensing (PRRS)* 139.3 (2016), p. 1-6 (cité à la page 91).

[Wei+18] M. WEI, J. HUANG, X. XIE et coll. « Mesh denoising guided by patch normal co-filtering via kernel low-rank recovery ». In : *IEEE Transactions on Visualization and Computer Graphics* (2018) (cité à la page 23).

[Wil79] S. C. WILLIAM. « Robust locally weighted regression and smoothing scatterplots ». In : *Journal of the American Statistical Association* (1979), p. 859-836 (cité à la page 18).

[Xio+13] X. XIONG, A. ADAN, B. AKINCI et D. HUBER. « Automatic creation of semantically rich 3D building models from laser scanner data ». In : *Automation in Construction* 31 (2013), p. 325-337 (cités aux pages 54, 57).

[Xu+11] L. XU, C. LU, Y. XU et J. JIA. « Image smoothing via L0 gradient minimization ». In : *ACM Trans. Graph.* 30.6 (2011), p. 1-12 (cité à la page 23).

[Xu+90] L. XU, E. OJA et K. P. « A new Curve DetectionMethod : Randomized Hough Transform (RHT) ». In : *Pattern Recognition Letters* 11 (1990), p. 331-338 (cité à la page 56).

[Yag+02] H. YAGOU, Y. OHTAKE et A. BELYAEV. « Mesh smoothing via mean and median filtering applied to face normals ». In : *Geometric modeling and processing. Theory and applications* (juil. 2002), p. 124-131 (cités aux pages 19, 24).

[Yag+03] H. YAGOU, Y. OHTAKE et A. BELYAEV. « Mesh denoising via iterative alpha-trimming and nonlinear diffusion of normals with automatic thresholding ». In : *Computer Graphics International, IEEE* (jan. 2003), p. 28-33 (cité à la page 19).

[Yan+13] J. YANG, H. LI et Y. JIA. « Go-ICP : Solving 3D registration efficiently and globally optimally ». In : *International Conference on Computer Vision (ICCV)*. 2013, p. 1457-1464 (cité à la page 90).

[YE04] S. YUZHONG et B. K. E. « Fuzzy vector median-based surface smoothing ». In : *Computer Graphics International, IEEE* 10.3 (mai 2004), p. 252-265 (cité à la page 19).

[YJ94] N. YLÄ-JÄÄSKI Kiryati. « Adaptive Termination of Voting in the Probabilistic Circular Hough Transform ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16.9 (1994) (cité à la page 55).

[YL99] M. YANG et E. LEE. « Segmentation of measured point data using a parametric quadric surface approximation ». In : *Computer Aided Design* 31.7 (1999), p. 449-457 (cité à la page 14).

[Zha+18] J. ZHANG, J. CAO, X. LIU et coll. « Multi-normal estimation via pair consistency voting ». In : *IEEE Transactions on Visualization and Computer Graphics* (2018), p. 1077-2626 (cités aux pages 21, 35, 36).

[Zha94] Z. ZHANG. « Iterative point matching for registration of free-form curves and surfaces ». In : *International Journal of Computer Vision* 13.2 (1994), p. 119-152 (cités aux pages 85, 87, 89, 90).

[Zhe+18] Y. ZHENG, G. LI, X. XU, S. WU et Y. NIE. « Rolling normal filtering for point clouds ». In : *Computer Aided Geometric Design* 62 (2018), p. 16-28 (cités aux pages 20, 21).

[Zho+16] Q.-y. ZHOU, J. PARK et V. K. B. « Fast Global Registration ». In : *European Conference on Computer Vision (ECCV)*. T. 2. Oct. 2016, p. 766-782 (cités aux pages 85, 91).

[Zho09] Y. ZHONG. « Intrinsic shape signatures : A shape descriptor for 3D object recognition ». In : *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. Sept. 2009, p. 689-696 (cité à la page 91).

[ZS14] J. ZHANG et S. SINGH. « LOAM : Lidar Odometry and Mapping in Real-time. » In : *Robotics : Science and Systems*. T. 2. 2014, p. 9 (cités aux pages 91, 113, 143).

[Zhe+11] Y. ZHENG, H. FU, O. K. AU et C. TAI. « Bilateral normal filtering for mesh denoising ». In : *IEEE Transactions on Visualization and Computer Graphics* 17.10 (2011), p. 1521-1530 (cités aux pages 19, 20, 22-24).

[Özt+09] A. ÖZTIRELI, G. GUENNEBAUD et M.GROSS. « Feature preserving point set surfaces based on non-linear kernel regression ». In : *Proceedings of Eurographics* 28.2 (2009), p. 493-501 (cité à la page 21).

Annexes

Test de connectivité

A

On considère une ligne ℓ_l étiquetée comme interaction entre plans ($\boldsymbol{\pi}^{\ell_l} = \left\{\pi_1^{\ell_l}, \pi_2^{\ell_l}\right\}$). Nous cherchons les pixels qui appartiendraient à la droite d'intersection des plans $\pi_1^{\ell_l}$ et $\pi_2^{\ell_l}$. On rappelle que ce groupe de pixels est nommé $Q^{\ell_l *}$. Pour chaque angle ϕ (ou θ) correspondant à une ligne (resp. colonne) de l'image I, l'angle θ (resp. ϕ) est calculé tel qu'un point correspondant à la paire (ϕ, θ) appartienne à la droite d'intersection entre $\pi_1^{\ell_l}$ et $\pi_2^{\ell_l}$.

$\mathbf{p}_{\phi,\theta}$ est le point 3D correspondant aux angles ϕ et θ tel que :

$$\mathbf{p}_{\phi,\theta} := d_p \left(\cos(\phi)\sin(\theta), \sin(\phi)\sin(\theta), cos(\theta)\right) \ , \tag{A.1}$$

d_p étant la distance de $\mathbf{p}_{\phi,\theta}$ à l'origine du capteur. Si $\mathbf{p}_{\phi,\theta}$ appartient à la droite d'intersection théorique, il respecte donc le système d'équations suivant :

$$\begin{cases} \mathbf{p}_{\phi,\theta} \cdot \mathbf{n}_1 = d_1 \\ \mathbf{p}_{\phi,\theta} \cdot \mathbf{n}_2 = d_2 \end{cases} \tag{A.2}$$

Comme la distance d_p n'est pas recherchée dans un premier temps, on peut contracter les deux équations et on obtient :

$$\frac{\mathbf{n}_1^x \cos(\phi)\sin(\theta) + \mathbf{n}_1^y \sin(\phi)\sin(\theta) + \mathbf{n}_1^z \cos(\theta)}{\mathbf{n}_2^x \cos(\phi)\sin(\theta) + \mathbf{n}_2^y \sin(\phi)\sin(\theta) + \mathbf{n}_2^z \cos(\theta)} = \frac{d_1}{d_2} \tag{A.3}$$

Nous appelons D le rapport $\frac{d_1}{d_2}$.

— Si on fixe l'angle ϕ et qu'on cherche l'angle θ correspondant, l'équation (A.3) peut se réécrire de la façon suivante :

$$A\sin(\theta) = B\cos(\theta) \tag{A.4}$$

avec :

$$\begin{cases} A = \mathbf{n}_1^x \cos(\phi) + \mathbf{n}_1^y \sin(\phi) - D(\mathbf{n}_2^x \cos(\phi) + \mathbf{n}_2^y \sin(\phi)) \\ B = D\mathbf{n}_2^z - \mathbf{n}_1^z \end{cases} \tag{A.5}$$

L'équation (A.4) admet deux solutions :

$$\begin{cases} \theta_1 = \arctan(\frac{B}{A}) \\ \theta_2 = \arctan(\frac{B}{A}) - \pi \end{cases} ; \tag{A.6}$$

— Si on fixe l'angle θ et qu'on cherche l'angle ϕ correspondant, l'équation A.3 peut se réécrire de la façon suivante :

$$E\cos(\phi) + F\sin(\phi) = G \tag{A.7}$$

avec :

$$\begin{cases} E = \mathbf{n}_1^x \sin(\theta) - D\mathbf{n}_2^x \sin(\theta) \\ F = \mathbf{n}_1^y \sin(\theta) - D\mathbf{n}_2^y \sin(\theta) \\ G = D\mathbf{n}_2^z \cos(\theta) - \mathbf{n}_1^z \cos(\theta) \end{cases} \tag{A.8}$$

L'équation (A.7) admet deux solutions :

$$\begin{cases} \phi_1 = \arccos(\frac{G}{\sqrt{(E^2+F^2)}}) + \arctan 2(F, E) \\ \phi_2 = \arctan 2(F, E) - \arccos(\frac{G}{\sqrt{(E^2+F^2)}}) \end{cases} . \tag{A.9}$$

Par la suite, on sélectionne les solutions qui respectent la contrainte impliquée par d_p dans le système (A.2) et on en déduit les pixels qui forment $Q^{\ell_l *}$.

Données LiDAR

B

B.1 Jeu de données 1 : Hokuyo (DS1-H)

Le premier jeu de données est appelé « apartment » et est disponible en ligne sur le site de l'ASL [Pom+12]. Les 45 scans constituant ce jeu de données ont été acquis par l'institut fédéral suisse de technologie à Zürich pour tester la robustesse d'algorithmes pendant l'exploration d'un environnement dynamique. En effet, certains objets sont déplacés entre les acquisitions. Le capteur Hokuyo utilisé fournit une bonne précision ($\pm$ 3 cm) et une cartographie précise de la scène pourrait être reconstruite à partir des données. Son champ de détection atteint 30 m. Son angle d'ouverture est 270°. Une documentation plus détaillée est disponible sur le site du constructeur [Hok]. Le capteur est monté sur un équipement mobile permettant d'obtenir des scans 3D et un dispositif *théodolite* est utilisé pour estimer les poses *vérité terrain* de manière fiable.

B.2 Jeu de données 2 : Leica (DS2-L)

Le second jeu de données a été acquis à l'Institut Pascal de Clermont-Ferrand pour travailler sur des données d'intérieur avec une complexité plus importante. Ces données ont été acquises avec un scanner Leica qui permet d'obtenir une très bonne précision ($\pm$ 3 mm) pour des points situés jusqu'à 120 m du capteur et un angle d'ouverture de 360° horizontalement et 270° verticalement. Ce scanner est recommandé pour obtenir une reconstruction de scène détaillée. Une documentation plus détaillée est disponible sur le site du constructeur [Lei]. Ce jeu de données contient six scans numérotés de 0 à 5 d'un bâtiment de deux étages avec de nombreuses fenêtres. Cet environnement peut être qualifié de complexe car il contient des éléments non-structurés tels que des arbres, des murs courbes, des barrières, des colonnes et des escaliers. De plus, chaque scène peut mesurer jusqu'à 65 m d'une extrémité à l'autre. Les poses *vérité terrain* ont été acquises en utilisant des cibles physiques.

B.3 Jeu de données 3 : Velodyne (DS3-V)

Le troisième jeu de données a été constitué à l'Institut Pascal grâce à un capteur Velodyne HDL-32 qui fournit des points de mesure avec une ouverture angulaire de 360° horizontalement et 40° verticalement, il lance 32 rayons lasers et tourne autour d'un axe vertical. Ce type de capteur est fréquemment utilisé en robotique, dans des applications de navigation. Il fournit des points 3D à une fréquence élevée (10 Hz) à une distance pouvant atteindre 100 m. Sa précision typique est de $\pm$ 2 cm. Une documentation plus détaillée est disponible sur le site du constructeur [Vel]. La scène extérieure est un parcours structuré : le site « PAVIN » (cf. Fig. 3.17). Le capteur est

monté sur un véhicule mobile et les scans sont acquis en mouvement. Quarante scans sont extraits de l'ensemble de données (1 scan sur 100 dans l'ordre chronologique d'acquisition). Ce jeu de données est compliqué à traiter car les points sont mesurés à de grandes distances ce qui induit un fort bruit de mesure.

B.4 Jeu de données 4 : Sick (DS4-S)

Le dernier jeu de données a été obtenu à l'université de Osnabrück [Els+10] sur le site [NL17]. Un capteur SICK LMS-200 monté sur un robot a été utilisé avec un angle d'ouverture de 180° horizontalement, et 100° verticalement et sa précision typique est $\pm$ 1.5 cm. Une documentation plus détaillée est disponible sur le site du constructeur [SIC]. Ce jeu de données est particulièrement difficile à traiter en raison du fort bruit du capteur et de son faible angle d'ouverture.

Tableau B.1.: Description des jeux de données. La colonne « Nombre de points » correspond au nombre de points moyen par scan. La colonne « Taille maximale » indique la distance maximale entre les points les plus éloignés. La colonne « Rés » correspond à la résolution moyenne des scans. La colonne « Mouv. » qualifie le mouvement entre les scans. La dernière colonne spécifie si la vérité terrain entre les transformations est disponible.

	Capteur	Nombre de scans	Nombre de points	Taille (m)	Rés. (cm)	Mouv.	Trans.
DS1-H	Hokuyo UTM-30LX	45	365 000	11	0.61	Faible	Oui
DS2-L	Leica P20	6	9×10^6	65	0.35	Grand	Oui
DS3-V	Velodyne HDL-32E	41	56 000	120	7.70	Faible	Non
DS4-S	SICK LMS-200	63	81 600	25	8.20	Faible	Non