Reconstruction de modèles 3D photoréalistes de façades à partir de données image et laser terrestre

Jérôme Demantke

UNIVERSITÉ PARIS-EST

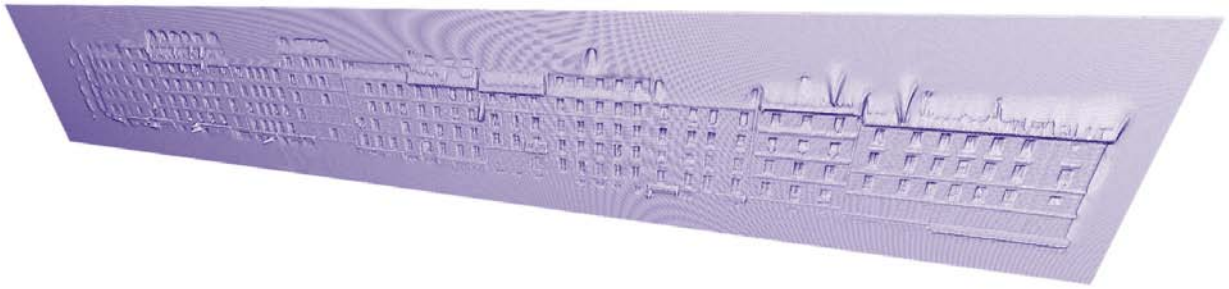
École Doctorale **MSTIC**

Mathématiques et **S**ciences et **T**echnologies de l'**I**nformation et de la **C**ommunication-

Thèse de doctorat en sciences et technologies de l'information géographique

Jérôme **DEMANTKÉ**

RECONSTRUCTION DE MODÈLES 3D PHOTORÉALISTES DE FAÇADES À PARTIR DE DONNÉES IMAGE ET LASER TERRESTRE



IGN
INSTITUT NATIONAL
DE L'INFORMATION
GÉOGRAPHIQUE
ET FORESTIÈRE

Jury

Nicole **VINCENT** ..... Présidente
Paul **CHECCHIN** ..... Rapporteur
George **VOSSELMAN** ..... Rapporteur
Jean-Emmanuel **DESCHAUD** ..... Examinateur
Nicolas **PAPARODITIS** ..... Directeur
Bruno **VALLET** ..... Encadrant

Laboratoire MATIS, IGN

73, avenue de Paris 94165 Saint-Mandé, Cedex

Remerciements / Acknowledgements

Tout d'abord, je souhaite remercier les membres du jury,
merci à Nicole Vincent d'avoir présidé le comité de thèse, je lui suis également redevable pour le master de qualité dont j'ai bénéficié à Paris Descartes
I would like to address my acknowledgments to George Vosselman for his analysis of this work and his relevant remarks.
Merci à Paul Checchin pour avoir examiné ce travail de thèse en profondeur.
Merci à Jean-Emmanuel Deschaud pour ses questions pertinentes.

Merci à L'IGN et au laboratoire MATIS d'avoir permis cette thèse.
Merci à Nicolas Paparoditis dont j'admire l'enthousiasme et l'énergie qu'il consacre au laboratoire.
Merci à Bruno de m'avoir encadré, je lui suis reconnaissant pour les discussions, les conseils et la liberté qu'il m'a donné. Merci de m'avoir fait découvrir les manifolds, Dempster-Shafer, mais aussi le kart et le ski nautique.

Merci à Nicolas David et Clément Mallet, la crème-de-la-crème de l'encadrement de stage. Je pense en particulier aux conseils aiguisés de Nicolas en matière de programmation ou de bibliographie, et à l'enthousiasme communicatif de Clément (pour les séparateurs à vaste marge et le foot en couloir) qui ré-oxygène fréquemment le laboratoire.

Comme à tous les autres membres du Laboratoire, je leur témoigne de ma plus sincère amitié, grâce à eux, j'ai eu un réel plaisir à aller travailler, mais aussi partager les repas, les pauses café, les fermetures de l'IGN, les pots de thèse, le sport à midi, le décathlon IGN, les voyages, et tant de choses encore, le tout dans une ambiance chaleureuse, teintée d'humour et de largesse d'esprit.

Ainsi, je remercie Jean-Pascal, Fabrice et Bertrand pour leur soutien psychologique et technique et leurs grandes qualités humaines.
Merci à Fabien, Mathieu et Narut qui ne se sont jamais départis de leur bonne humeur, même lors d'un triathlon en eau glacée et boueuse.
Merci aux forces tranquilles du labo, aux nouveaux arrivants et à ceux que j'ai croisé :
merci à Isabelle, Mélanie, Corina, Nesrine, Lijun, Valérie, Abdalbassir, Adrien, Alexandre et Alexandre, Antonio, Athanasios, Arnaud, Bahman, Chu, Emmanuel, Jean-Christophe, Jean-Pierre, Lâmân, Lionel, Martin, Mathieu, Murat, Neelanjan, Nicolas, Olivier, Quoc, Rémi, Sébastien, Shawn et Tuan.

Merci à Marie-Claude qui est un peu une maman pour les grands enfants que nous sommes au MATIS.
Merci à Sylvie Cach pour sa disponibilité.

Merci aux architectes ; Didier & Anne et Christophe & Romina qui se sont intéressés à mon travail.
Merci à mes amis et en particulier Clément pour les discussions scientifiques et tout le reste.

Merci à ma belle famille si accueillante.
Merci à ma famille, mes parents, mes frères et ma soeur sur qui je peux toujours compter.
Un grand merci à Adeline, mon rayon de soleil quotidien.

Enfin j'ai une pensée pour ceux qui m'ont quitté pendant cette thèse.
Philippe & Monique Génin, Gianni & Lucia Drago, Pierre Bertière.
Chacun à votre manière, vous m'avez beaucoup appris et apporté.

"Il faut toujours dire ce que l'on voit. Surtout il faut toujours, ce qui est plus difficile, voir ce que l'on voit."
Charles Péguy

à Ethan

Reconstruction de modèles 3d photoréalistes de façades à partir de données image et laser terrestre

On souhaite détecter puis modéliser les façades de bâtiments à partir des données acquises par le véhicule de numérisation mobile de l'ign, le Stéréopolis. Il s'agit de trouver une représentation géométrique des façades adaptée aux données (signal lidar/laser et images optiques). La méthode doit être automatique et rendre possible la modélisation d'un grand nombre de façades afin de contribuer à la production de maquettes numériques de villes. Les verrous techniques proviennent de l'acquisition mobile en environnement urbain non contrôlé (géoréférencement du véhicule, densité variable de points lidar...), ils proviennent du signal lidar, issu d'une technologie relativement récente et pour lequel le processus de traitement n'est pas encore consensuel : faut-il exploiter ou non la géométrie capteur? Enfin, la quantité de données pose le problème du passage à l'échelle. Afin d'analyser la géométrie des nuages de points 3D lidar, nous avons proposé des attributs décrivant pour chaque point la forme de l'environnement local (linéaire-1D, planaire-2D ou volumique-3D). Les plans principaux des façades sont extraits automatiquement des données lidar grâce à un algorithme streamé de détection de rectangles verticaux. Nous avons développé deux modèles qui sont initialisés par ces rectangles. Une grille irrégulière dont chaque case, parallèle au plan principal peut avancer ou reculer. Une grille déformable qui est "poussée par les rayons lasers jusqu'aux point lasers". Enfin, nous avons montré comment la grille déformable peut être rendue cohérente avec les images optiques en alignant les discontinuités géométriques de la grille avec des discontinuités radiométriques des images.

Reconstruction of photorealistic 3D models of facades from terrestrial images and laser data

One wishes to detect and model building façades from data acquired by the ign mobile scanning vehicle, the Stereopolis. It is a question of finding a geometric representation of facades appropriate to the data (lidar/laser signal and optical images). The method should be automatic and enable the modeling of a large number of facades to help the production of digital city models. Technical obstacles come from the mobile acquisition in uncontrolled urban environment (vehicle georeferencing, variable lidar point density...), they come from the lidar signal, retrieved from a relatively new technology for which the process is not yet consensus : does one operates into sensor geometry or not? Finally, the amount of data raises the problem of scaling up. To analyze the geometry of lidar 3D point clouds, we proposed attributes describing for each point the shape of the local surroundings (linear-1D, planar-2D or volume-3D). The facade main planes are automatically extracted from lidar data through a streamed detection algorithm of vertical rectangles. We developed two models that are initialized by these rectangles. An irregular grid in which each cell, parallel to the main plane can move forward or backward. A deformable grid which is "pushed by the laser beams toward the laser points". Finally, we showed how the deformable grid can be made consistent with the optical images aligning the geometric discontinuities of the grid with radiometric discontinuities of the images.

Mots clés

Bâtiment, Urbain, Façade, Détection, Modélisation, Terrestre, Mobile, Numérisation Laser, Cartographie Mobile, LIDAR, Nuage de Points, La Géométrie, Voisinage Adaptif, Sélection d'Échelle, Dimensionnalité, Rectangles Verticaux, Grille Irrégulière, Déformable Grille 2.5D, Système de Coordonnées Prismatique, Fusion LIDAR Images

Key words

Building, Urban, Facade, Detection, Modeling, Terrestrial, Mobile, Laser Scanning, Mobile Mapping, LIDAR, Point Cloud, Geometry, Adaptive Neighborhood, Scale Selection, Dimensionality, Vertical Rectangles, Irregular Grid, 2.5D Deformable Grid, Prismatic Coordinate System, Fusing LIDAR Images

Résumé substantiel

Contents

0.1 Contexte : la modélisation urbaine

Le laboratoire MATIS développe depuis de nombreuses années des recherches sur la modélisation 3D automatique des bâtiments en milieu urbain. Ces "villes en 3D" répondent à plusieurs besoins [1]:

Voir et comprendre Visualiser simplement la ville pour mieux comprendre les sujets traités. La maquette 3D numérique se substitue à la traditionnelle maquette physique.

Analyser La maquette 3D devient un outil d'analyse par le biais d'un interfaçage avec d'autres types de simulateurs. Elle facilite les études d'impact dans les différents domaines de l'environnement : (bruit, pollutions, inondations,...) ou de la sécurité (Incendies, accidents urbains, interventions policières ou militaires...). Elle peut être utilisée pour la prévention des risques et pour la gestion des crises : permettre aux services de sécurité civile de rapidement visualiser les accès in 3D.

Concevoir et voir les projets d'urbanisme et d'architecture Comparer les projets de façon objective et claire pour prendre les bonnes décisions. Les applications sont nombreuses : Instruction des permis de construire, concours d'architecture, grands aménagements urbains, éclairage public...

Communiquer Présenter et expliquer les décisions prises. Par exemple lors de sessions publiques. Montrer les stratégies urbaines en visualisant la ville d'hier, d'aujourd'hui et de demain. Montrer et valoriser la ville sur les sites grands public (GoogleEarth, Geoportail). Développer le eTourisme.

Echanger Intégrer les données techniques dans les systèmes d'informations géographique (SIG) dans les cartes 3D et inversement. Partager les informations 3D avec les différents services et les entreprises qui en ont besoin : implantation d'antennes relais pour téléphones portables, passage de conduites, mobilier urbain, ...

L'objectif est de satisfaire au mieux ces différents besoins, à l'aide de modélisations précises, complètes et fonctionnelles (répondant aux besoins de l'utilisateur). L'amélioration technologique des systèmes d'acquisition permet d'obtenir des données plus précises et en plus grande quantité, mais parallèlement, les algorithmes doivent gagner en efficacité et s'adapter aux nouveaux problèmes que posent par exemple la quantité des données acquises. La classification introduite dans [2] décrit les différents niveaux de détail (level of detail, LoD).

LoD0 modèle de région, modèle de terrain 2.5D.
Précision attendue : plusieurs mètres
Capteurs concernés : Images optiques verticales ($\simeq$ 50 cm)
Laser aéroporté (un point pour 2 ou 3 m^2)
Radar interférométrique

LoD1 modèle de ville ou de site, modèle en "blocs" sans structure de toit.
Précision attendue : métrique
Capteurs concernés : Images optiques verticales ($\simeq$ 50 cm)
Laser aéroporté (1 à 2 points / m^2)

LoD2: modèle de ville ou de site, texturé, avec des structures de toit différenciées.
Précision attendue : décimétrique
Capteurs concernés : Images optiques verticales ($\simeq$ 10 cm)
Images optiques obliques
Laser aéroporté (plusieurs points / m^2)

LoD3 modèle de ville ou de site, modèle architectural.

LoD4 modèle d'intérieur. modèle architectural dans lequel on peut se déplacer.

De LoD2 à LoD3

L'outil actuel de production de modèles 3D de bâtiments IGN, appelée Bati3D se base uniquement sur des données aériennes. Les bâtiments sont modélisés par des polyèdres (LoD2) et les façades sont donc des rectangle. Les textures des façades sont parfois de mauvaise qualité, en effet, bien que les façades soient toujours vues par l'avion, l'angle de vue peut être rasant (au pire 6^o), ce qui produit des textures très étirées, comme on le voit sur la figure 1.
De nouveaux systèmes d'acquisition sont apparus: les systèmes aéroportés avec une prise de vue oblique et les véhicules d'acquisition mobile. Ces systèmes permettent d'obtenir de nouveaux points de vue sur la ville et notamment les façades. Cette thèse se focalise sur la reconstruction des façades texturées (LoD3), à partir de données acquises par le Stéréopolis. Le Stéréopolis est le véhicule de numérisation mobile terrestre développé par le laboratoire MATIS (IGN) depuis quelques années, il image les rues à l'aide de caméras optiques et de dispositifs de numérisation lidar. La vue des façades depuis la rue est cruciale, car c'est le point de vue "humain", il permet de voir les devantures de magasins, les numéros et noms de rues, les portes, les fenêtres... Il perd en précision vers le haut des bâtiments, ce qui explique la nécessité des prises de vues obliques qui font le "relais" entre la vue aérienne verticale et la vue depuis la rue. Cependant, le travaux de cette thèse utilisent exclusivement les données terrestres, la fusion de données constituant un sujet de recherche à part entière. En outre, le point de vue terrestre est pour le moment celui qui permet d'approcher au plus les façades.

0.2 Pour quelles applications?

Les données brutes acquises par le Stéréopolis durant un tour de quelque centaines de mètres sont très volumineuse. Ce sont des millier d'images et des milliards de points 3D laser. Naviguer dans de telles données n'est pas simple ni intuitif. L'information est morcelée et peut être redondante entre plusieurs images, les nuages de points sont éparses et peuvent contenir des problèmes de géoréférencement. Détecter et modéliser géométriquement les façades permet de compiler ces données dans un modèle plus facilement exploitable, qui servira à améliorer le modèle Bati3D, ils pourront aussi être intégrés dans la plateforme de navigation immersive iTowns. (iTowns permet de naviguer virtuellement dans une acquisition du Stéréopolis depuis internet. L'objectif est, à terme d'intégrer les différents éléments du mobilier urbain et de permettre une interaction avec eux.) La modélisation des façades permet aussi d'extraire des informations (empreinte au sol des bâtiments, hauteur, taille, nombre de fenêtres...) ces modèles peuvent aussi être comparés entre eux, utilisés tels quel pour les besoins de films ou de jeux vidéo, ou servir à la génération procédurale de modèles de bâtiments.

(a) Bati-3D ign

(b) Aerial view

(c) Building = Polyhedron

Figure 1: Bati-3D, Paris

0.3 Sujet

L'objectif est de modéliser et de texturer automatiquement les façades urbaines à partir des données Stéréopolis (imagerie visible ou thermique terrestre, points laser). Cette reconstruction devra couvrir 2 aspects:

Géométrie: il s'agit de trouver le meilleur type de représentation géométrique possible pour une façade, afin d'obtenir le meilleur compromis généricité/robustesse possible lors de la reconstruction. Cette reconstruction devra exploiter au mieux la représentation choisie et le type de données afin de produire un modèle cohérent avec celles-ci.

Cohérence avec les images: l'objectif est de générer des modèles cohérent avec les images pour l'application texturation (la texturation à proprement parler ne fait pas partie de la thèse).

0.4 Verrous techniques

Données

Les données lidar utilisés sont incomplètes: elles sont acquises depuis la rue, et les voitures, les arbres, ou des bâtiments peuvent cacher partiellement les façades. De plus, par nature, les nuages de points fournissent une information incomplète car ponctuelle. Se pose le problème de savoir comment pallier le manque d'information entre les points.
Ces données peuvent aussi contenir des erreurs de géoréférencement; est-il possible de proposer une méthode capable de traiter des données mal géoréférencées?
La fusion des données image et laser pose différents problèmes : Les données laser sont plus fiables et de nature géométriques, alors que les données images sont plus précises et de nature radiométrique. Il faut faire interagir ces deux types de donnée en exploitant au mieux les spécificités de chacune.

Volume de données et automatisation

Les algorithmes proposés doivent permettre de traiter de manière automatique le volume de données acquis par le Stéréopolis en un temps raisonnable.

Modélisation

Les façades sont des éléments très structurés et géométriques (murs plans, fenêtres rectangulaires, répétition des motifs...) Cependant, il y a une grande variabilité et on trouve aussi de nombreux objets "inclassables": plantes sur les balcons, volet ouvert, ornement... Le problème est donc de trouver un modèle qui permette d'appréhender tous les cas différents (généricité) mais qui soit malgré tout capable de fournir un modèle robuste aux perturbations.

Par ailleurs, il faut trouver un modèle adapté, d'une part aux données en entrée, et d'autre part aux objets (les façades) que l'on souhaite modéliser. Il faut donc trouver un modèle qui permette la meilleure intégration de ces aspects techniques/"bas-niveau" et sémantiques/"haut-niveau".

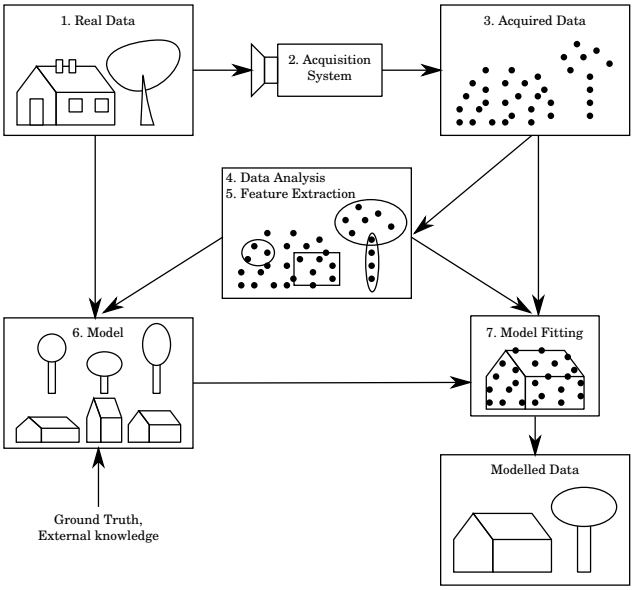


Figure 2: Vue d'ensemble des étapes de modélisation. La scène réelle (1) est échantillonnée par un capteur (2). Les données données obtenues (3) peuvent être analysées (4), on peut en extraire de l'information, des primitives géométriques (5). Cette analyse, ainsi que la connaissance des données réelles nous aide à imaginer un modèle adapté (6). Il faut ensuite faire correspondre ce modèle avec les données (7) pour enfin obtenir la modélisation.

0.5 De l'acquisition à la modélisation

La figure 2 montre les différentes étapes de modélisation. Un système d'acquisition permet d'extraire de l'information des données réelles sous forme d'un signal. Notre objectif est d'utiliser au mieux ces données acquises pour modéliser les données réelles. Pour cela, une étape d'analyse et d'observation des données réelles ainsi que des données acquises doit aboutir à un modèle susceptible de représenter au mieux les données. Il faut enfin faire correspondre le modèle avec les données pour obtenir la modélisation.
Les données acquises fournissent une vision partielle et inexacte de la réalité. Il faut faire attention que l'objectif final est une modélisation des données réelles et non des données acquises. Cependant,les données acquises sont notre seule moyen de connaître les données réelles, à l'exception de connaissances a priori sur les objets à modéliser.

0.6 Positionnement de la thèse

On peut trouver d'autres thèses sur la modélisation de façade comme les trois suivantes [3], [4] et [5]. dont les principale étapes de modélisation sont décrites ici.

Pu [3] : Une modélisation fondée sur la connaissance.

- Extraction de régions planes dans les nuages de points.
- Sémantisation de chaque région.
- Faire correspondre les régions avec une forme appropriée (polyèdre ou B-spline).
- Intégration des images : On fait correspondre les lignes détectées dans les images avec les bords du modèle géométrique.
- Modèle texturé de façon semi-automatique avec les images.

Boulaassal [4] : Une modélisation vectorielle.

- Réduction le bruit de mesure. suppression des points faux et indésirables.
- Segmentation du nuage de points en régions planes. RANSAC+Croissance de région
- Extraction des contours des régions planes.
- Calculs d'intersection entre les contours.
- Obtention d'un modèle vectoriel.

Deschaud [5] : Une modélisation hybride sous forme de primitives pour les surfaces planes et d'un maillage pour le reste du modèle.

- décimation du nuage de points en entrée : sélection des points qui maximisent une approximation de la courbure locale.
- Calcul des normales en chaque point.
- Débruitage des nuages de points.
- Segmentation en zones planes et non planes : croissance de voxels.
- Triangulation des zones non planes.
- Colorisation et texturation.

La méthode proposée est différente car nous abordons le problème de détection automatique des façades dans les nuages de points, nous avons aussi tenté de contourner l'étape de segmentation en régions planes qui est coûteuse en temps de calcul. De plus, nous nous sommes orienté vers une approche qui tente de tirer parti d'informations autres que la position des échos lidar, à savoir le temps d'acquisition de chaque point, la position du capteur, des rayons laser... La thèse proposée s'oriente donc plus vers des questions méthodologiques de traitement des données lidar et de passage à l'échelle.

Méthode proposée

- Descripteurs géométriques locaux, en particulier un indice de verticalité qui permet de favoriser les points 3D appartenant à une surface plane et verticale dans l'étape suivante de détection des façades.
- Détection streamée de rectangles verticaux supposés correspondre aux plans principaux des façades à l'aide d'un algorithme de type RANSAC.
- Deux modélisations proposées initialisés grâce aux rectangles verticaux :
  -Grille irrégulière : détection des discontinuités verticales et horizontales et attribution d'une profondeur à chaque case de la grille.
  -Grille déformable : surface 2.5D initialisée par le rectangle vertical et se déformant vers les points

0.7 Organisation de la thèse

Analyse

Les données lidar sont décrites dans le chapitre 1 avec leurs spécificités et les problèmes associés à leur traitement. Une méthode générale pour analyser la géométrie locale dans les nuages de points est proposée dans le chapitre 2. Les problèmes de passage à l'échelle sont évoqués dans le chapitre 3 et on explique pourquoi on favorise une approche qui traite des buffers temporels de l'acquisition.

Détection

Dans le chapitre 4, la méthode pour extraire les rectangles de façade est développée.

Modélisation

Les chapitres suivants traitent de la modélisation des façades. Le chapitre 5 propose une modélisation sémantique à l'aide de grilles irrégulières. On étudie les méthodes pour relier les points laser en topologie capteur dans le chapitre 6. Le chapitre 7 propose une modélisation photo-consistante grâce à une grille déformable qui est initialisée le long du rectangle estimé avec la méthode du chapitre 4 et est en quelque sorte *"poussée par les rayons lasers jusqu'aux point lasers"*. Enfin le chapitre 8 détaille la méthode pour faire correspondre les discontinuités de la grille déformable avec les discontinuités détectées dans les images optiques.

0.8 Principales contributions

Nous avons défini des descripteurs de dimensionalité en formalisant des descripteurs géométriques locaux classique. nous en avons dérivé deux nouveaux attributs. Une valeur d'entropie évaluant la pertinence des descripteurs de dimensionnalité qui permet de sélectionner automatiquement, pour chaque point, une taille de voisinage optimale pour la description "dimensionnelle". Un valeur de verticalité dont on se sert pour détecter les façade et qui s'avère efficace pour la classification de scènes urbaines. Nous avons proposé une méthode simple et efficace de détection de rectangles verticaux adaptée aux données lidar mobiles terrestres. En particulier, nous avons mis en évidence l'intérêt de pondérer la sélection aléatoire des points dans ransac selon un critère de pertinence, ici, le critère de verticalité. Le modèle de grille irrégulière nous a permis de mettre en évidence des algorithmes efficaces de traitement des points lidar, comme la détection des discontinuités horizontales et verticales. Nous avons soulevé la question de l'utilisation d'informations autres que la position des échos lidar, et mis en évidence certaines limites des nuages de points désorganisés, en particulier pour la reconstruction de surfaces. Nous avons alors montré l'intérêt d'utiliser la géométrie capteur grâce à un théorème qui encourage à mailler les surfaces selon l'ordre d'acquisition des points. Un modèle de grille déformable 2.5D a été proposé. Il permet de modéliser les façades en s'adaptant à l'orientation des rayons laser et de la façade grâce à un système de coordonnées original baptisé "système de coordonnées prismatique". Enfin nous avons exploré la possibilité de rendre cette grille cohérente avec les images optiques acquises en même temps.

0.9 Résumé des chapitres

Chapitre 1

Dans ce chapitre, nous avons décrit les données lidar et mis en avant les forces et les faiblesses de ces données. Les données acquises par les véhicules d'acquisition mobile dans les environnements urbains semblent complexe à exploiter.
les principaux problèmes sont les suivants : En plus d'être de nature **éparses** (points 3D) et échantillonnées de manière **inhomogène**, les données sont nécessairement **incomplètes** : de nombreux paramètres ne peuvent être contrôlés comme la vitesse et la trajectoire du véhicule. Les capteurs embarqués sont contraints de capturer l'information à partir du point de vue de la rue, le long de la trajectoire du véhicule. Par exemple, les volumes des bâtiments ne peuvent être appréhendés en entier. On doit s'accommoder de morceaux de façades, partiellement masqués par des objets au premier plan tels que les voitures, les arbres ou d'autres façades. Les données sont donc incomplètes, mais elles peuvent aussi être redondantes si le trajet d'acquisition contient des boucles, les erreurs de géoréférencement peuvent créer des **incohérences entre ces données redondantes**. Les données Lidar sont obtenues par des systèmes qui mettent en jeux différentes technologies. Les erreurs de chaque technologie ont différentes amplitudes. Le géoréférencement du véhicule est à l'origine des plus grandes imprécisions, c'est pourquoi on souligne le risque d'avoir des incohérences entre les données redondantes mais acquises à différents moments de l'acquisition et donc géoréférencées différemment.
Il est difficile de comprendre la répartition des points, et pourtant, la position des échos est la seule information disponible pour deviner la géométrie de la scène numérisée. On est par exemple tenté de se fier à la densité de points comme indice de confiance, pourtant cette densité peut être due aux conditions d'acquisition (véhicule arrêté à un feu rouge). **En fait, la position des échos est uniquement due -aux erreurs de mesure près- à la géométrie de la scène, alors que la densité de points dépend en grande partie des conditions d'acquisition.**
Lorsque les échos sont exprimés dans le système de coordonnées τ, Θ, R, (τ : temps d'acquisition, Θ : angle de tir vertical, R : distance au capteur), on observe une très grande régularité selon τ, Θ, en effet, les pulse laser sont émis avec une fréquence régulière. En revanche, R dépend uniquement de la distance aux objets. Ce système de coordonnées met aussi en évidence la densité de points quasi surfacique (surface τ, Θ).
Le système de coordonnées τ, Θ, R est indépendant de la position du capteur. ce qui est intéressant si les données sont mal géoréférencées. De plus, ces coordonnées sont rapide à obtenir car elles découlent directement du processus d'acquisition et peuvent fournir des hypothèses pour la reconstruction de surface. En effet, les surfaces détectées peuvent être approximées par un maillage τ, Θ. Cependant ce maillage peut se replier sur lui-même à cause de rebroussements du balayage laser.
Notre conclusion est que, même si cela amène de nouveaux problèmes, il est dommage d'utiliser exclusivement les points 3D. La géométrie capteur, et en particulier le système de coordonnées τ, Θ, R permet de retrouver une structure dans le nuage de points et de séparer les informations qui viennent des conditions d'acquisition (τ, Θ) de celles qui viennent des données numérisées (R).

Chapitre 2

L'objectif de ce chapitre était de fournir une méthode simple, générique et automatique pour décrire la géométrie autour de chaque écho lidar. Les deux questions auxquelles nous avons tenté de répondre sont :

- Trouver des attributs géométriques génériques pour décrire tout nuage de point.
- Trouver une méthode automatique pour choisir automatiquement le voisinage adapté à chaque point.

Pour cela, des attributs sont dérivés de l'approche classique de "tensor voting". Les attributs sont calculés dans un voisinage sphérique autour de chaque écho. Ils décrivent la dimensionnalité (1D, 2D ou 3D) du voisinage en fonction que la répartition des échos dans le voisinage est plutôt linéaire (1D), planaire (2D) ou volumique (3D). La taille de voisinage la plus appropriée est sélectionnée dans un intervalle de tailles potentielles grâce à l'attribut d'entropie E_f.
Il n'est pas nécessaire d'avoir de connaissances à priori sur la répartition des points, la densité ou le motif du balayage laser. Cependant, on a vu que d'une part, la description géométrique obtenue permet de

déduire ces différentes caractéristiques, et d'autre part, une connaissance de ces caractéristiques permet de bien borner les tailles de voisinage afin d'obtenir un résultat qui décrive la géométrie des objets scannés et qui soit autant que possible indépendant de la configuration d'acquisition. Autrement dit, l'algorithme peut être lancé une première fois afin d'analyser les caractéristiques du nuage de point et d'affiner les paramètres pour lancer l'algorithme une seconde fois.
Les avantages d'un simple sous-échantillonnage spatial sont aussi mis en avant pour s'abstraire de la répartition de points induite par la configuration d'acquisition.

Chapitre 3

Le passage à l'échelle impose certaines contraintes dans la conception des algorithmes. En particulier, il est nécessaire de découper les nuages de points afin d'appliquer les traitements sur des blocs plus petits. Différents choix sont envisageables, (Semantic, Spatial/Dimensionnel: temporal (1d), espace (3D), ou espace-temps (4D)). Dans notre contexte, la méthode la plus judicieuse semble être de conserver la structure originelle des nuages de points, en effet, les points sont enregistrés dans l'ordre d'acquisition et sont donc organisés naturellement selon la dimension temporelle. C'est donc la méthode la plus directe, mais elle présente aussi d'autre avantages. Un intervalle de temps correspond à un segment de trajectoire du véhicule durant l'acquisition. La position du capteur est donc bornée dans l'espace-temps, ainsi les points acquis durant cet intervalle sont toujours proches temporellement et souvent proches spatialement. Traiter les points par buffers temporels est donc une méthode

- rapide
- adaptée aux environnement changeants : non mélange de points appartenant à différentes époques.
- adaptée aux géoréférencement du véhicule : la dérive du géoréférencement est progressive et varie donc peu sur un court intervalle de temps.

C'est le choix retenu pour la détection des façades en sous forme de rectangles verticaux dans le chapitre 4. Cela permet de surmonter les problèmes du volume de données et du géoréférencement.

Chapitre 4

Nous avons présenté un algorithme streamé de détection de rectangles verticaux à partir des données lidar acquises par le Stéréopolis. Un RANSAC modifié est appliqué sur des buffers (avec recouvrements) de points 3D acquis durant un même intervalle de temps. Le descripteur de verticalité (chapitre 2) est utilisé pour favoriser les points appartenant à des zones planes et verticales. Les morceaux de façade (rectangles verticaux) sont extraits puis fusionnés selon un critère de distance et un critère de recouvrement. Les rectangles fusionnés les plus pertinents sont conservées. La construction du graphe de fusion est quadratique en nombre de morceaux, mais ce nombre est négligeable par rapport au nombre de points. Les régions verticales planaires ont montré leur utilité dans la géolocalisation fine : La dérive du véhicule peut être détectée grâce au décalage entre les rectangles qui correspondent à la même façade. les rectangles détectés sont utilisés pour initialiser les modèles de façade proposés dans cette thèse, à savoir une modélisation sémantique avec des grilles irrégulières dans le chapitre chapitre 5 et une grille déformable 2.5D dans le chapitre 7.

Chapitre 5

Nous avons proposé un modèle de façade à l'aide de grilles irrégulières. Ce modèle est calculé à partir d'un nuage de points. Au préalable, un rectangle vertical correspondant au plan principal de la façade a été détecté. Seuls les points inclus dans ce rectangle et suffisamment proches du plan principal sont pris en compte. Le rectangle est découpé en une grille irrégulière à l'aide de droites horizontales et verticales placées au niveau des principales discontinuités géométriques. Pour ce faire, on accumule les points horizontalement et verticalement, puis on calcule la variation de profondeur des points par rapport au plan principal. Les discontinuités sont des maxima de variation de profondeur. Ces discontinuités permettent de découper le rectangle en un ensemble de rectangles/cases. Pour chaque case, on a ensuite cherché la profondeur optimale en fonction des points se projetant dedans. On a préféré borner le nombre total de

profondeurs possibles, afin de permettre par exemple que toutes les cases qui contiennent une partie d'un même mur puissent avoir la même profondeur. Guidé par ce choix, un algorithme de discrétisation des profondeurs à été proposé. C'est une variante des "k-means" qui trouve automatiquement le nombre optimal k de profondeurs. Pour cela, on tente de minimiser à la fois un terme d'attache au données et le nombre k. Il reste ensuite à associer une de ces k profondeurs à chaque case. Le modèle obtenu est une grille irrégulières dont chaque case en en retrait ou en avant par rapport au plan principal. Ce modèle suppose donc que les façades sont constituées d'éléments rectangulaires parallèles entre eux. D'autres approches ont été étudiées pour déterminer la profondeur de chaque case, notamment un algorithme de graph-cut.

Chapitre 6

Nous avons démontré que pour reconstruire une surface continue à partir d'un signal lidar, mailler les points adjacents en topologie capteur (maillage capteur) est une solution appropriée car elle prend naturellement en compte des contraintes logiques imposées par les rayons laser. Cependant, la géométrie de la scène scannée se résume rarement à une simple surface continue.
Si le capteur est en face de la surface, on peut avoir confiance dans la continuité entre les échos adjacent, alors que si le capteur voit la surface avec un angle rasant, il peut exister des cavités entre deux échos adjacents qui contiennent une portion cachée de la surface et qui contredisent la continuité entre échos adjacents. L'angle d'incidence fournit donc un moyen simple de mesurer la fiabilité du lidar dans la détection de surfaces continues.
Bien que la continuité entre les échos adjacents ne soit pas fiable à cent pourcent, la surface continue fournie par le maillage capteur a toujours un sens physique : c'est l'interface entre ce qui est vu et non vu par le capteur. Ce maillage est donc intéressant à utiliser mais le principal obstacle est qu'il peut se replier sur lui-même si le balayage laser rebrousse chemin.

Chapitre 7

On a présenté une approche pour reconstruire la géométrie des façades à partir des données lidar. On souhaitait des surfaces cohérentes avec les données, et en particulier les images optiques acquises en même temps. La structure algorithmique proposée permet de reconstruire des grilles 2.5D dans un système de coordonnées 3D choisi par l'utilisateur. La grille est initialisée le long du plan principal de la façade, puis elle se déplace itérativement vers les échos dans une seule dimension (orthogonale au plan principal pour un système de coordonnées cartésien). Nous avons défini un système de coordonnées adapté au point de vue du capteur : le système de coordonnée prismatique qui est une généralisation du système de coordonnées cartésien. Il permet de tirer parti de la géométrie capteur tout en étant topologiquement cohérent avec l'espace 3D Les grilles obtenues sont adaptées à la navigation immersive et d'autres applications qui nécessitent des modèles 3D compacts et détaillés. Elles pourraient aussi être utilisées pour analyser la structure grammaticale des façades ou l'extraction de fenêtres.

Chapitre 8

On texture les grilles 2.5D avec les images optiques acquises en même temps. Dans ce chapitre, on s'intéresse à améliorer la texturation de la grille déformable par des images optiques. Quand on projette les images sur la grille, les endroits les plus sensibles aux imprécisions du modèle géométrique sont les discontinuités au niveau des bords d'objets. Si les discontinuités géométriques ne correspondent pas tout à fait aux discontinuités radiométriques dans les images optiques, on peut avoir des phénomènes de bavure particulièrement désagréables visuellement. On a donc proposé une méthode pour faire correspondre les discontinuités géométriques avec les discontinuités radiométriques (moins fiables mais plus précises). Les discontinuités géométriques sont extraites dans la grille, Les discontinuités radiométriques sont extraites dans les images. Ce sont des segments 2D qui sont projetés sur la grille afin d'obtenir des segments 3D. Ces segments sont appariés avec les discontinuités géométriques. On ne conserve que ceux qui sont suffisamment proches d'une discontinuité géométrique. Les segments image conservées sont utilisés pour recalculer la grille déformable : On relance la déformation itérative de la grille, mais cette fois, la contrainte de lissage est réduite au niveau des segments image, on autorise ainsi des déchirures plus fortes à l'endroit précis des discontinuités image.

Les résultats de la texturation sont satisfaisants car on peut vérifier la cohérence entre les données lidar et image optique. Cependant, la précision du modèle texturé n'est pas homogène: d'une part il y a une différence de résolution entre le modèle géométrique issu des données lidar et les données images : la texture image est plus résolue que la surface 3D. Et d'autre part on subit les limitations d'une approche "point de vue unique": la géométrie et la texture apparaissent d'autant plus imprécises qu'on s'éloigne de ce point de vue. On ne peut pas inventer de l'information et il paraît donc difficile d'améliorer les résultats à partir ces seules données. En revanche une fusion avec d'autres jeux de données pourrait être imaginée.

Contents

Introduction

0.10 Context: urban modeling

The MATIS laboratory developed researchs over many years on the automatic 3D modeling of buildings in urban areas. These "3D" cities serve several purposes [1]:

Visualize and understand Simply view the city to better understand the subjects. 3D Digital model replaces the traditional physical model.

Analyze The 3D model is an analytical tool through an interfacing with other simulator types. It facilitates impact studies in different areas of the environment: (noise, pollution, floods...) or safety (Fire, urban accidents, police or military interventions...), It can be usefull for risk prevention and management crises : allow civil security services to quickly view access in 3D.

Design and visualize the urban and architectural projects. Compare projects objectively and clearly to make the right decisions. The applications are numerous : Instruction building permits, architectural competitions, large urban, street lighting, ...

Communicate Present and explain decisions. For example during public sessions. Show urban strategies by visualizing the city of yesterday, today and tomorrow. Demonstrate and promote the city on major public sites (GoogleEarth, Geoportail). Develop eTourism.

Exchange Integrate technical data in information systems (GIS) maps in 3D and vice versa. Share 3D information with the various services and companies who need it : antennas for implementation mobile phones, crossing lines , street furniture, ...

The objective is to best meet these needs, using specific models, complete and functional (meeting the needs of the user). Technological improvement of acquisition systems provides more accurate data and in greater quantity, but at the same time algorithms need to increase efficiency and adapt to new problems posed for example by the increasing amount of acquired data. Classification introduced in [2] describes the different levels of detail (LoD).

LoD0 Regional model – 2.5D Digital Terrain Model
Expected accuracy : several meters
Involved sensors : Vertical optical images ($\simeq$ 50 cm)
Airborne Laser (one point per 2 or 3 m^2)
Radar interferometry

LoD1 City / Site model – "block model" without roof structures
Expected accuracy : meter
Involved sensors : Vertical optical images ($\simeq$ 50 cm)
Airborne Laser (1 to 2 points / m^2)

LoD2 City / Site model – textured, differentiated roof structures
Expected accuracy : decimeter
Involved sensors : Vertical optical images ($\simeq$ 10 cm)
Obliques optical images
Airborne Laser (several points / m^2)

(a) Bati-3d ign

(b) Aerial view

(c) Building = Polyhedron

Figure 3: Bati-3d, Paris

LoD3 City / Site model – detailed architecture model

LoD4 Interior model – "walkable" architecture models

From LoD2 to LoD3

The current IGN production tool of 3D building models called Bati3D is solely based on aerial data. The buildings are modeled as polyhedra (LoD2) and facades are therefore rectangles. The quality of facade textures is sometimes poor. Indeed, although facaces are always seen by the aircraft, the viewing angle can be grazing (at worst 6^o) which produces very stretched textures, as shown in figure 3.

Recently, new acquisition systems have emerged: airborne systems with oblique viewpoint and terrestrial mobile acquisition vehicles. These systems provide new perspectives on the city and particularly on the facades. This thesis focuses on the reconstruction of textured facades (LoD3) from data acquired by the Stereopolis. The Stereopolis is the mobile mapping system developed by the MATIS laboratory (IGN) for some years, it scans the streets using optical cameras and scanning lidar devices. The facade view from the street is crucial because it is the "human" point of view, it allows to see shop fronts, street numbers and street names, doors, windows... precision is lost towards the top of buildings, which explains the necessity of acquiring aerial oblique views which may act as an intermediary between the vertical aerial view and the view from the street. However, the work of this thesis uses only terrestrial data, data fusion constitutes a research topic in its own right. In addition, the terrestrial viewpoint is the one that allows at this time, to get the closest possible to the facades.

0.11 For which applications?

The raw data acquired by the Stereopolis during a tour of a few hundred meters is very large. These are thousands of images and billions of 3D lidar points. To navigate into such data is not simple or intuitive. The information is redundant and can be fragmented between multiple images, point clouds are sparse and may contain georeferencing problems. To detect facade and model 3D geometry can help to compile this data in a more easily exploitable model, that could improve the Bati3D model and could also be integrated into the immersive navigation platform iTowns (iTowns allows to navigate virtually in a Stereopolis acquisition from Internet. The goal is eventually to integrate the different elements of street furniture and allow interaction with them.) Information can be extracted from facade models (footprint of buildings, height , size, number of windows ...) and these models can also be compared to each other, used as is for the purpose of movies or video games, or for procedural generation of synthetic building models.

0.12 Topic

The objective is to automatically model urban facades from Stereopolis data (visible or thermal imaging and terrestrial laser points). This reconstruction will cover two aspects:

Geometry It is a question of finding the best possible type of geometric representation for a facade in order to obtain the best generic/robustness compromise as possible in the reconstruction. This reconstruction should fully exploit the representation chosen and type of data to produce a coherent model therewith.

Consistency with images The generated model has to be consistent with the images for texture mapping purposes (texture mapping itself is not part of the thesis).

0.13 Technical obstacles

Data

Input lidar data is incomplete: it is acquired from the street, and cars, trees, or buildings may partially hide the facades. In addition, by nature, point clouds provide incomplete because sparse information. This raises the question about how to overcome the lack of information between points.
The data may also contain georeferencing errors; is it possible to propose a method able to handle poorly georeferenced data?
The fusion of image and laser data poses different problems: Laser data is more reliable and geometrical by nature, while image data is more precise but radiometric. We need both types of data to interact and to exploit the best characteristics of each.

Volume of data and automation

The proposed algorithms should allow to automatically process the volume data acquired by the Stereopolis in a reasonable time.

Modeling

Facades are highly structured and geometric (flat walls, rectangular windows, repeating patterns...). However, there is a great variability and there are also many " unclassifiable" objects: plants on balconies, open shutters, ornaments... The problem is to find a model that enables to understand all the different cases (generic) but which is nevertheless able to provide a robust model against disruptions.
Furthermore, we must find a suitable model, firstly to the input data, secondly to the objects (facades) that we aim to model. We must find a model that enables the better integration of these technical/"low-level" and semantic/"high-level" aspects.

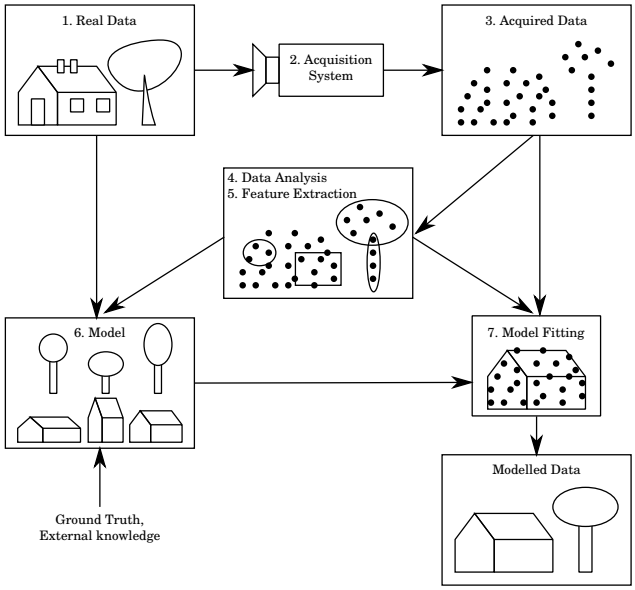


Figure 4: Overview of modeling steps. The actual scene (1) is sampled by a sensor (2). Acquired data (3) can be analyzed (4), one can extract information, geometric primitives (5). This analysis, as well as knowledge of the actual data helps us to imagine a suitable model (6). One then matches the model with the data (7) to finally get the output modelled data.

0.14 From acquisition to modeling

Figure 4 shows a diagram with the different modeling stages. An acqusition system retrieves information of the actual data as a signal. Our goal is to make the best use of these acquired data to model the actual data. For this, an analysis and observation step of the actual data as well as the acquired data should lead to a model that can well represent the data. Finally, one matches the model with the data to obtain the numerical modeling.
The acquired data provides a partial and inaccurate view of reality. Be careful that the ultimate goal is a numerical modeling of real data and not of the acquired data. However, the acquired data are our only way to know the actual data, with the exception of a priori knowledge about the objects we want to model.

0.15 Positioning of the thesis

One can find other theses on the modeling front as the following three [3] [4] and [5]. whose main modeling steps are described here.

Pu [3] : Knowledge based reconstruction.

- Extracts planar features from terrestrial laser point clouds.
- Determines the semantic meaning of each feature.
- Features fitted to some appropriate shapes such as polygons and B-spline surfaces.
- Integration of imagery : Image lines are matched with model edges which are generated from laser data.
- Images are semi-automatically mapped to the models as textures.

Boulaassal [4] : Vectorial modeling.

- Noise reduction. Outlier removal.
- Segmentation of the PCD into planar regions. RANSAC+Region growing.
- Edge extraction of planar regions.
- Finding intersections between edges.
- Obtaining the vectorial model.

Deschaud [5] : Hybrid modeling as primitives for flat surfaces and a mesh for the remainder of the model.

- Input point cloud decimation: selection of points that maximize an approximation of the local curvature.
- Normal calculation at each point.
- Denoising point clouds.
- Segmentation into planar and non-planar areas: voxels growing.
- Triangulation of non-planar areas.
- Colorizing and texturing.

The proposed method is different because we address the problem of automatic detection of facades in the point cloud, we also tried to bypass the step of segmentation in planar regions which is expensive in computation time. In addition, we oriented towards an approach that tries to take advantage of more information than the position of the lidar echoes, namely the acquisition time of each point, the position of the sensor and the laser beams... The proposed thesis has been moving in the direction of methodological issues in lidar data processing and scaling up.

Proposed Method

- local geometric descriptors, in particular a vertical index that can promote the 3D points that belong to a vertical flat area in the facade detection step.
- Streamed Detection of vertical rectangles supposed to correspond to the main facade planes using a RANSAC-type algorithm.
- Two proposed models initialized with the vertical rectangles:
  -Irregular Grid : Detection of horizontal and vertical discontinuities and assign a depth to each grid cell.
  -Deformable Grid : 2.5D surface initiated by the vertical rectangle and deformed towards the 3D points.

0.16 Structure of the thesis

Analysis

The lidar data is described in Chapter 1 with its specificities and processing problems. A general method to analyze the local geometry in point clouds is given in Chapter 2. The problem of scaling up is discussed in Chapter 3 and we explain why we favor an approach with temporal buffers.

Facade detection

In Chapter 4, the method to extract the facade rectangles is developed.

Facade modeling

The following chapters deal with facade modeling. Chapter 5 provides a semantic modeling with irregular grids. We study methods to connect 3D laser points in sensor topology in Chapter 6. Chapter 7 offers a photo-consistent modeling thanks to a deformable grid which is initialized along a rectangle estimated in Chapter 4 and is somehow *"driven by the laser beams towards the 3D points"*. Finally, chapter 8 details the method to match the discontinuities of the deformable grid with discontinuities detected in the optical images.

0.17 Main contributions

We defined dimensionality descriptors formalizing classical local geometric descriptors. we derived two new attributes. An entropy value evaluating the relevance of the dimensionality descriptors that allows to automatically select, for each point, an optimal neighborhood size for the "dimensional" description. A verticality value which is used to detect facades and which is effective for urban scene classification. We proposed a simple and effective method for detecting vertical rectangles adapted to mobile terrestrial lidar data. In particular, we highlighted the interest of weighting points for random selection in ransac according to a relevance criterion, here, the verticality criterion. The irregular grid model has allowed us to identify effective algorithms to process lidar points, such as the detection of horizontal and vertical discontinuities. We raised the issue of using more information than the lidar echoe locations, and highlighted some limitations of unorganized point clouds, especially for surface reconstruction. We then showed the value of using the sensor geometry by a theorem that encourages to mesh surfaces following the acquisition order of the lidar points. A deformable 2.5D grid was proposed. It allows to model facades adapting to the orientations of the laser beams and the facade with a original coordinate system called "prismatic coordinate system". Finally we explored the possibility of making this coherent grid with optical images acquired at the same time.

Chapter 1

Description of the lidar data

Contents

In this chapter, we present the lidar data. The lidar principle is briefly explained, then we present some characteristics of this signal and derived data types, In particular, the point cloud data (PCD) that contain punctual locations of lidar echoes. The different techniques involved in the echo location are listed. We detail the geolocation step that converts echo coordinates from sensor reference frame to absolute reference frame. We show how this step is particularly difficult and important in the case of mobile acquisition systems. Some specificities of lidar PCD are then described. We try to understand the echo distribution. The sparse aspect of such data is stressed, and we investigate which other information could be provided by the acquisition configuration knowledge and by lidar signal. Finally, conclusions are drawn.

1.1 Capturing the geometry with reflected light

Lidar technology

The operating principle of lidar systems is detailed in the thesis of Mallet [6], you can also find a description in French in the thesis of Boulaassal [4]. Remote sensing by laser or lidar, ("light detection and ranging") is an optical measurement that consists of a transmitter which emits a laser beam to a target and a sensor that collects the emitted beam after reflection on the target. The analysis of the returning beam, gives informations about the target location. In particular, the round trip time of the beam allows to deduce the distance between the sensor and the target (as the speed of light is known).

Return signal: the full-waveform

As shown in figure 1.1, the return signal gives the light intensity in function of time. An intensity peak corresponds to the beam return after the round trip to the target. It indicates when the beam hit the target and we deduce the laser "echo" location on this target. Please note that thereafter, we indifferently

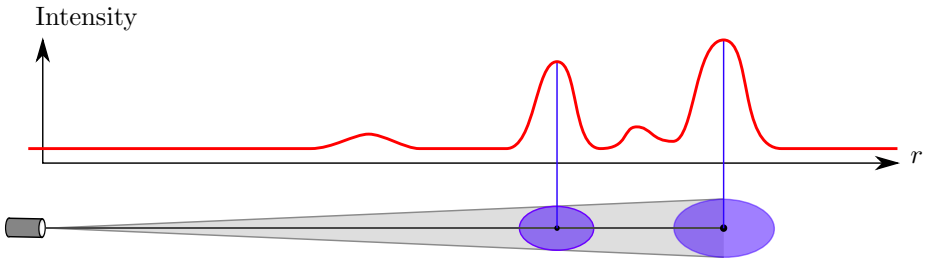


Figure 1.1: Return Signal: the full-waveform. An echo corresponds to a maximum of returning signal intensity. Two echoes are represented.

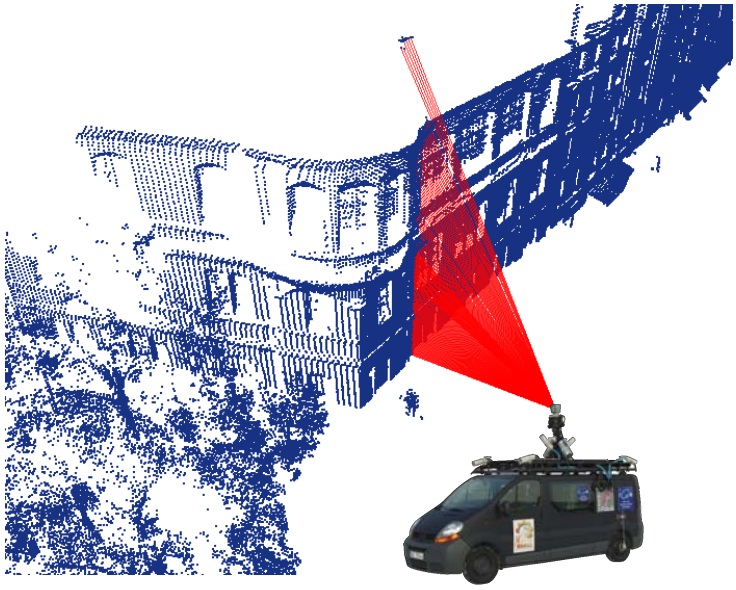


Figure 1.2: Lidar echoes (blue) acquired by the Streopolis. The lidar beams of a sweep line are displayed in red. (The car is not to scale).

use the terms "point" and "echo". The shape of the return signal called "full-waveform" can be more complex than a simple Dirac. The larger the laser footprint is, the more the response is spread over time, leading to flattened peak. The laser may also be reflected many times which produces several echoes and several peaks. The full-waveform analysis is the topic of [6]. This allows a detailed analysis of the signal that leads to precise results.

Point cloud data (PCD)

However, the signal is mostly processed on line and the output is a point cloud data (PCD). Indeed, lidar systems are often able to operate with a high frequency and they allow to acquire not one but many points. For example the RIEGL used on the Stereopolis allows to acquire 5 million points per second. The 3D echoes are recorded, they are sorted according to the acqusition order: the xyz coordinates, GPS acquisition time, intensity value and beam angle are stored in a file.

Lidar Sweep

The lidar sweep is performed in order to optimize the scene sampling. The beams should be scattered homogeneously in all the directions. However the high frequency acquisition often produces a higher echo density along the scanline. Several sweeping patterns exist, but in the context of this thesis, we consider that the sweep plane is perpendicular to the vehicle trajectory as in figure 1.2.

1.2 Obtaining the absolute location of echoes

1.2.1 Static and mobile acquisition procedures

Although the lidar principle remains the same, several acquisition procedures can be distinguished, depending on whether the lidar device is static or mounted on a mobile platform, and whether the acquisition is terrestrial or aerial.

Stationary Terrestrial Laser Scanning (STLS) : The lidar device is static, it can be installed on a tripod. The whole device rotates horizontally (360°) while the laser sweeps vertically.

Mobile Terrestrial Laser Scanning (MTLS) : The lidar device is embedded in a vehicle such as a car. The Mobile Mapping Systems (MMS) can acquire data on a larger scale by moving through the streets of a city for example.

Aerial Laser Scanning (ALS) : The lidar device is airborne. The laser often sweeps perpendicularly to the flying object trajectory. Such systems provide data from the sky and can cover extensive landscapes.

1.2.2 Sensor geolocation

The lidar system provides the echo locations relative to the sensor: the orientation is known mechanically with the shooting angle and the distance is measured thanks to the return signal. If an absolute location of the echoes is needed, a geolocation step is then necessary. It therefore takes two independent steps to obtain this absolute location:

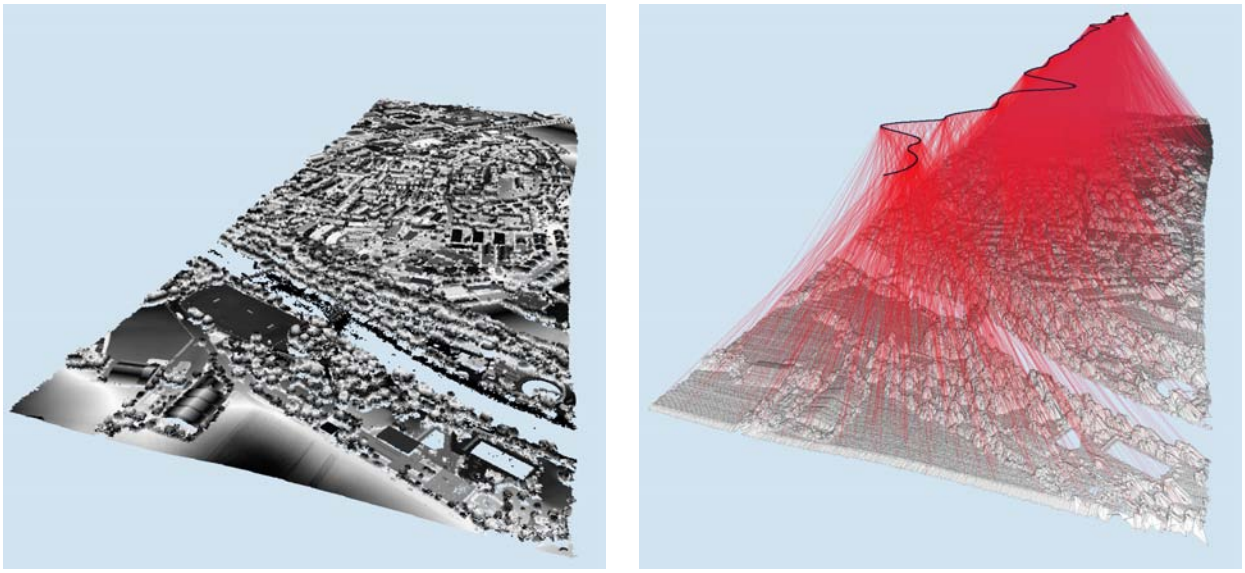
1. Positioning the echo relative to the sensor (lidar measurement).
2. Positioning the sensor relative to the world (geolocation).

In STLS, the second step consists of defining the geolocation of a single stationary spot. To obtain the absolute position of the echoes, the single rotation and translation is applied to all the echoes. **The two steps are independent**: a geolocation error may shift or rotate the position of the whole PCD, but it does not induce relative positioning error between the echoes. Such a PCD from stationary acquisition is said to be "rigid" because the relative position between echoes is reliable and is not affected by the georeferencing step.
On the contrary, in MTLS and ALS, the georeferencing step can induce relative positioning errors between echoes. The sensor is moving during the acquisition, and at each time, the sensor geolocation has to be found again. As a consequence, the geolocation of each echo may refer to a different sensor geolocation. The sequence of these successive sensor positions draws the path traveled by the vehicle during the acquisition. Pictorially, we try to georeference the full trajectory of the vehicle. The mobile laser scans are "non-rigid" because they may be distorted by this trajectory estimation.
In any case, the position of the echoes is more reliable in the sensor reference frame than in the absolute reference frame because the georeferencing step is added. In the next section we will see how the "local" sensor coordinates are converted into the global coordinates.

1.3 Converting sensor coordinates (τ, Θ, R) into real space coordinates (x, y, z)

1.3.1 Sensor topology

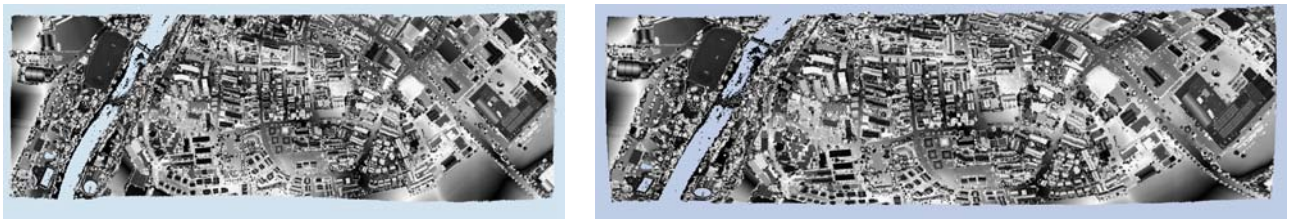
We consider the following acquisition configuration: The laser scans perpendicularly to the trajectory, with a vertical angle Θ. This is often the airborne configuration and the RIEGL lidar of the Stereopolis is mounted this way, as explained in [7]. Such acquisition configuration have been used to acquire the Vaihingen Dataset [8], [9]. The PCD, the sensor displacement and the laser beams are shown in figure 1.3 and 1.2.



(a) PCD, altitude in black and white)

(b) Laser Beams and trajectory

Figure 1.3: Vaihingen Dataset.



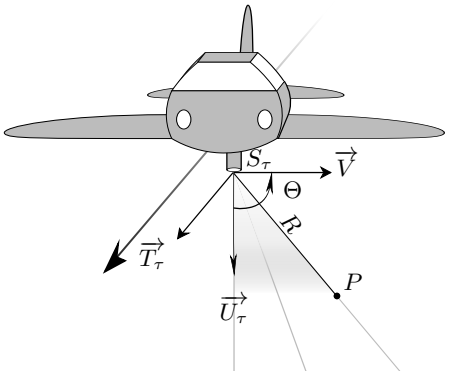
(a) Top View

(b) Points projected in the (τ, Θ) space

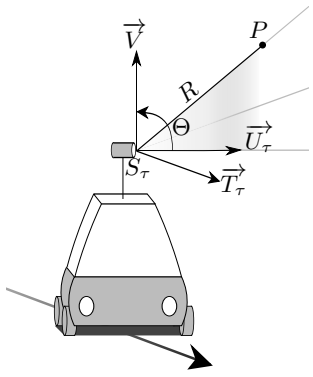
Figure 1.4: The dataset in (x, y, z) space (a) and (τ, Θ) space (b). In Aerial Datasets, the sensor is almost orthogonal to the scene, the projection deformation is thus low.

Let $P(x, y, z)$ an echo acquired at τ time,

$$P(x, y, z) = S_\tau + R\, cos\Theta\, \overrightarrow{U_\tau} + R\, sin\Theta\, \overrightarrow{V} \tag{1.1}$$



S_τ is the position of the sensor at τ.
R is the distance between S_τ and P.
Θ is the angle between the horizontal plane and the laser beam.
V is a vertical unit vector.
$U_\tau = T_\tau \times V$, with T_τ the unit vector of the vehicle displacement at τ.



Each term has specific features that imply various accuracies and value distributions as shown in table 1.1. Some terms depend on the geolocation system, and some others only depend on the lidar system. The coordinate system (τ, R, Θ) is special because it represents the sensor point of view and is independent of the geolocation system. As displayed in figure 1.4, the dataset in the (x, y, z) space and the (τ, Θ) space looks similar because the trajectory is almost straight and far from the acquired scene.

	S_τ	R	Θ
Accuracy depends on	Vehicle Positioning System	Lidar Measurement	Mechanical Measurement
Accuracy is	-	+	+++
Density depends on	Vehicle Trajectory and Speed	Scanned Objects Distance	Lidar Firing Frequency
Distribution Regularity ?	If Constant Speed	If Smooth Surfaces	Yes

Table 1.1: Accuracy and Distribution of S_τ, R and Θ.

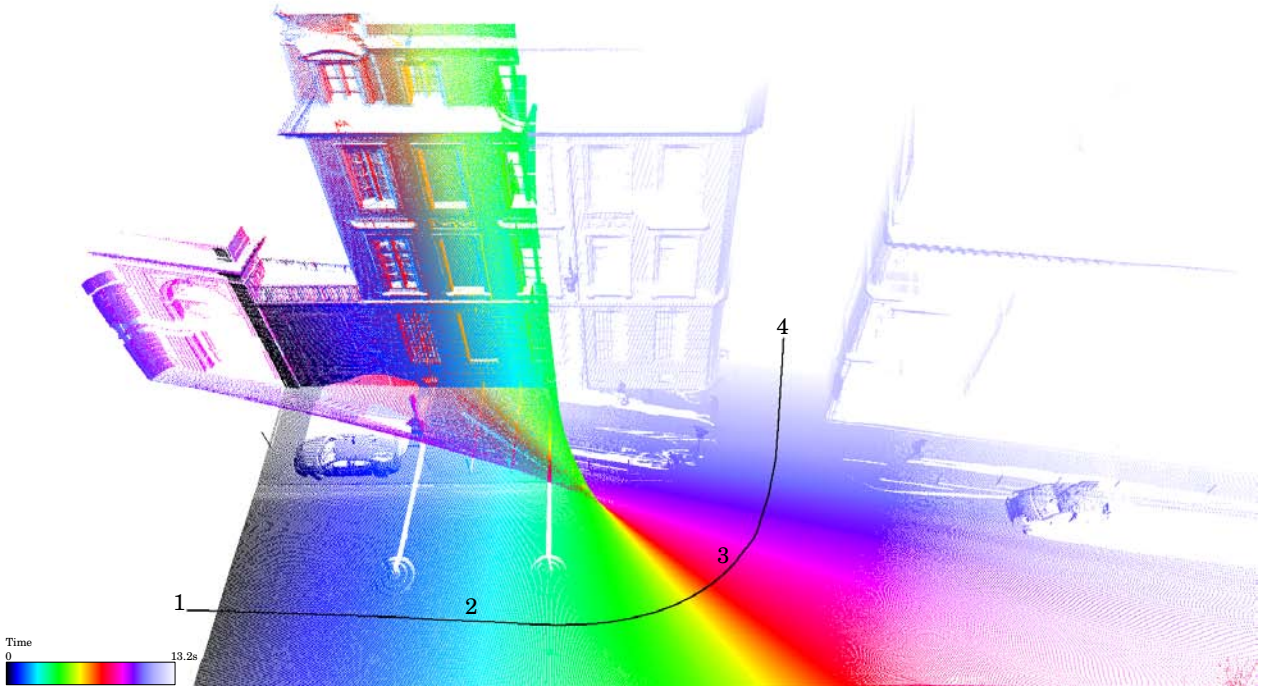


Figure 1.5: This PCD had been acquired while the Stereopolis performed a right angle turn. The trajectory is displayed in black, and we noted four locations that also noted in all the following pictures of this acquisition. We thus call it "1-2-3-4". The points are colored according to their acquisition time.

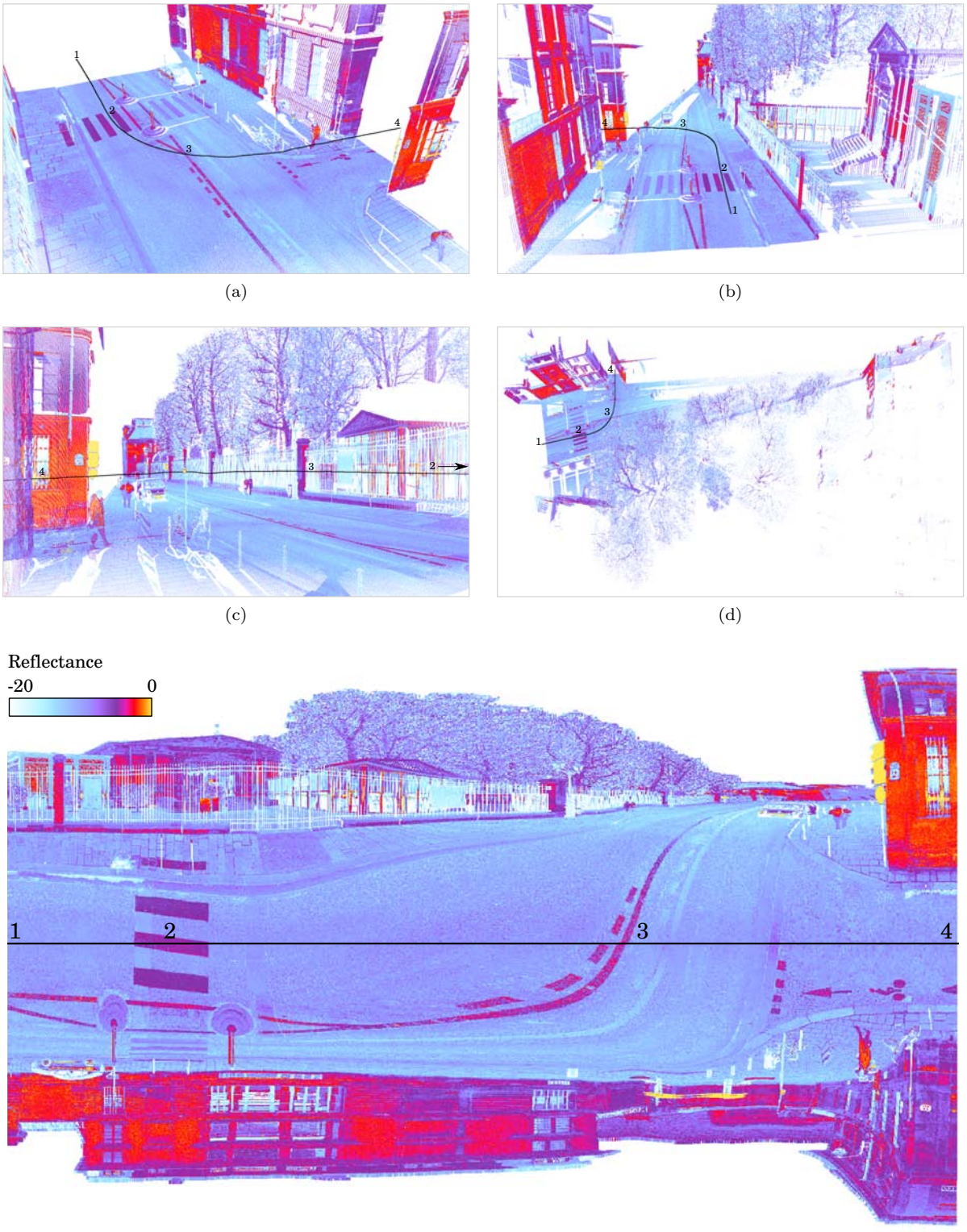
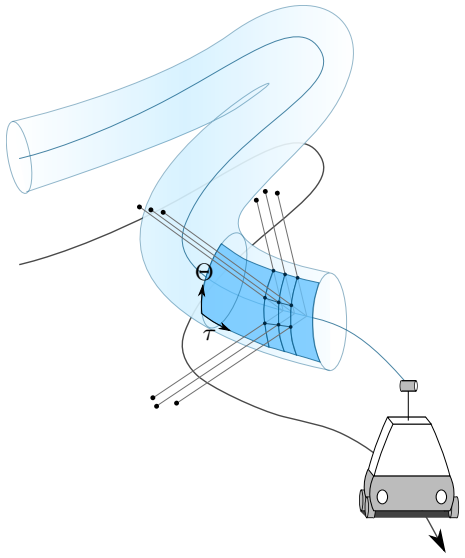


Figure 1.6: Various viewing angles (a-b-c-d), and sensor projection (e) for the "1-2-3-4" dataset, colored according to the reflectance. The trajectory is straight in the (τ, Θ) projection, and the point cloud is homogeneously dense, while the point density may strongly vary in the (x, y, z) space. Some shadows (null density) appear behind the people and bikes (c). One can see the scanlines on a facade (a). The point density is lower on the road part far from the trajectory (d). The point distribution is scattered in the trees (d) in such case, the (τ, Θ) point-connectivity has no sense.

1.3.2 (τ, Θ) space

The lidar sweep produces pulses at regular intervals of time τ and firing angles Θ. Hence, echo distribution in (τ, Θ) space is very homogeneous and structured. Each echo can be linked with the next point acquired, (τ connectivity) and with the point on the next scanline that has the same angle (Θ connectivity). It may be helpful to work with the images because they do not suffer from the problems of geolocation. In these images, the points are neighbors according to the τ, Θ connectivity that do not necessarily corresponds to a spatial neighborhood, ie to neighboring points in such image can be far in the (x, y, z) space.

Compared to ALS, (fig:1.4), in MTLS acquisition, the behaviors of τ and Θ are less easy to understand. In the figure 1.5, the PCD had been acquired while the Stereopolis performed a right angle turn. The acquisition time τ is displayed for each echo. The corresponding (τ, Θ) image is strongly distorted (fig:1.6),
The virtual tube displayed on the right is an iso-surface (iso-R) The blue rectangle mapped on it is an example of (τ, Θ) image. Assuming That All the echoes are at the same distance from the sensor, they would belong to a iso-R (tube). the echo distribution would then be very homogeneous along this iso-R. Actually, the R variation disperses the echoes and destroys this regularity, as explained in figure 1.6.



The point density is roughly uniform in the (τ, Θ) space. But it is no longer homogeneous in the (x, y, z) space. In the next chapter we will see the problems raised by this variable point density.

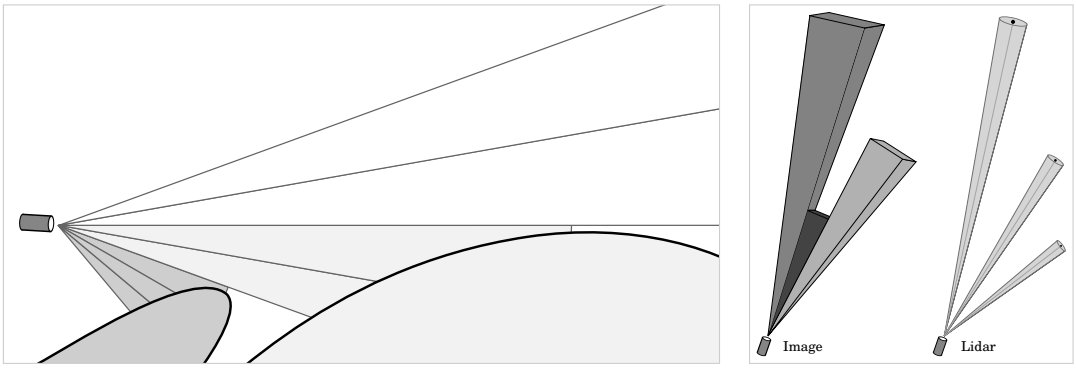
1.4 What lidar systems can tell us in addition to echo locations?

The point density depends both on the acquisition configuration and on the scanned objects, While the sensor remoteness induces a density reduction in PCD, in image data it causes an increase in the area that is projected onto a pixel. This mechanism preserves the continuity -the sensor connectivity- in image structure.

1.4.1 From τ, Θ image to sparse point cloud, the loss of continuity

Comparison between image and point cloud data

If we draw a comparison between the echoes and the image pixels (fig 1.7), the further the sensor is, the sparser the echoes are: the sensor remoteness deletes the echo connectivity in PCD. On the contrary, the image pixels remain related, regardless the distances of objects from the sensor. This comes from the sensor layout: a card is covered by sensor cells. The photons arriving on the map are collected by the cell they encounter. Each cell integrates the photons that come from its visibility cone. A pixel value is the result of this integration. The visibility cones partition the 3D space, There is no gap between the visibility cones of neighbor cells, the neighbor pixels correspond to neighbor areas in the 3D space. It is interesting to note that the sensor remoteness implies a density reduction in PCD, while in images, it implies an integration of photons that come from a larger area. The surface of this area projected on a single pixel grows as $(R\tan(\delta\theta))^2$, with R the distance to the sensor and $\delta\theta$ the visibility cone angle. An image remains therefore a continuous surface, whatever the distance to the objects. The neighbor pixels always correspond to neighbor areas, but the surface of these areas varies from pixel to pixel. The lidar PCD are sparse, because the beams footprints are not overlapping. The neighborhood relations between echoes are also more



(a) (b)

Figure 1.7: Sensor remoteness: comparison between point cloud and image data.
(a) Photography: Each sensor cell integrates the photons that come from its visibility cone.
(b) There is a continuity between image pixels, because the visibility cones partition the 3D space, while the lidar point clouds are sparse, because the beams footprints are not overlapping.

complex to draw, because the sensor location and orientation vary from echo to echo, while the pixels are acquired in the same time by a rigid system of sensor cells.

Failing to keep sensor connectivity between echoes, we will analyze the density variation according to the sensor location.

The causes of density variation

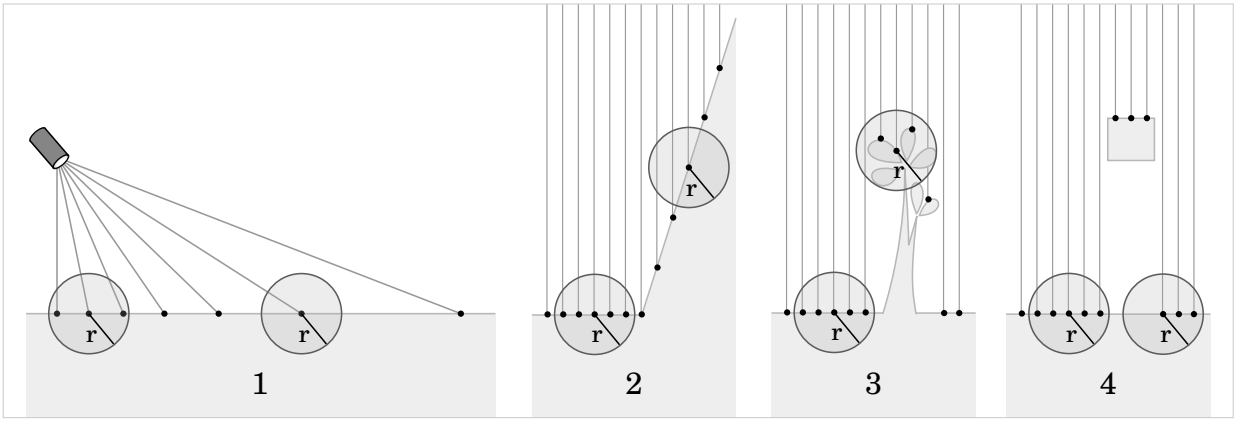
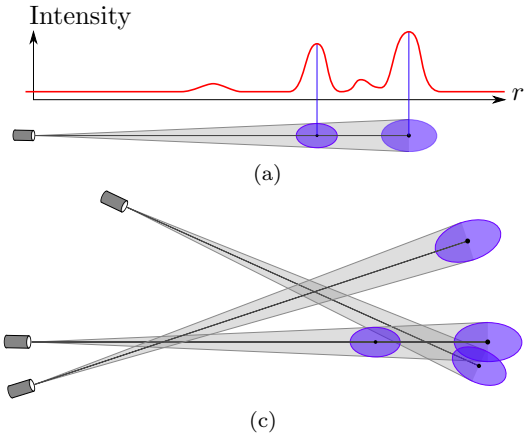


Figure 1.8: Causes of density variation. Point density depends on the sensor distance (1), and this distance depends on the surface geometry (2). Density is also decreased in porous areas (3) or by occluders (4).

The acquisition configuration (vehicle trajectory, shooting frequency, shooting angles...) gives many clues to understand the echo distribution. Indeed, for a given echo, if the sensor location (S_τ) and the shooting angle (Θ) are known, the lidar beam can be positioned in space, and as the echo cannot be elsewhere than along the beam, the only missing information is R (sensor-echo distance). This is also the only information which depends on the scanned object: the scanned object determines where the beam path is stopped... while the beam path itself is determined by the acquisition configuration. In sensor coordinate system, the acquisition configuration is responsible for the echo distribution in two dimensions (τ and Θ), while the scanned objects scatter echoes in the third dimension (R).



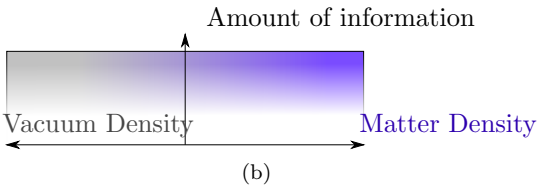


Figure 1.9: Matter, vacuum and amount of information.
(a) An echo corresponds to a maximum of returning signal intensity.
(b) The matter density can be measured according to the lidar echoes that indicates the presence of matter. The areas with no echoes are not necessarily empty of matter, they are actually empty of information. The laser beams go through the empty areas, we can thus measure the vacuum density according to the beams.
(c) Lidar beams and echoes represented by a visibility cone and an ellipsoid.

Unpredictable density

One might think that the knowledge of the acquisition configuration could allow to estimate the echo density as a function of R. The echo density actually decreases with the sensor remoteness (when $R \nearrow$), but as illustrated in figure 1.8, the relation between the echo density and R is more complex and depends not only on the acquisition configuration (viewing angle, scanning frequency ...) but also on the scanned object geometry. If the geometry is simple, as a flat surface, the density is easy to predict. However, if the density is lower than expected, many causes are possible: complex geometry, occluders, semi-transparent or porous objects, tree foliage...
In sum, the echo distribution depends on the scene geometry that can be arbitrarily complex, providing sparse PCD with a variable echo density. The point density cannot be used as it is, as a confidence measure of the data. It is correlated with acquisition conditions, especially the distance to the sensor. The point density can be inferred thanks to the coordinate system τ, Θ, R for a given geometric modeling.

1.4.2 Beam carving: detecting empty areas.

Contrary to image data that preserves pixel connectivity in a continuous surface, the sensor connectivity is lost in PCD while the laser beams encounter objects at various distances and the echoes are scattered in 3D space. Moreover, in mobile laser scanning, the sensor displacements makes the sensor connectivity more complex and less useful to understand the true connectivity between echoes in 3D space. In summary, PCD provide punctual information with no obvious connectivity structure.

How to guess the geometry between points

When one wants to estimate the geometry of a scanned scene through a PCD, the question is to guess the missing information between points.
- Do the points belong to the same surface? Or else, is there an empty gap between them?
- Which points are connected and which are not?
- And if some points are connected, how to interpolate a surface between them?
The issue of connecting the points or not is often tackled thanks to some assumptions:

"The close points are connected."

"The points belonging to the same plane are connected."

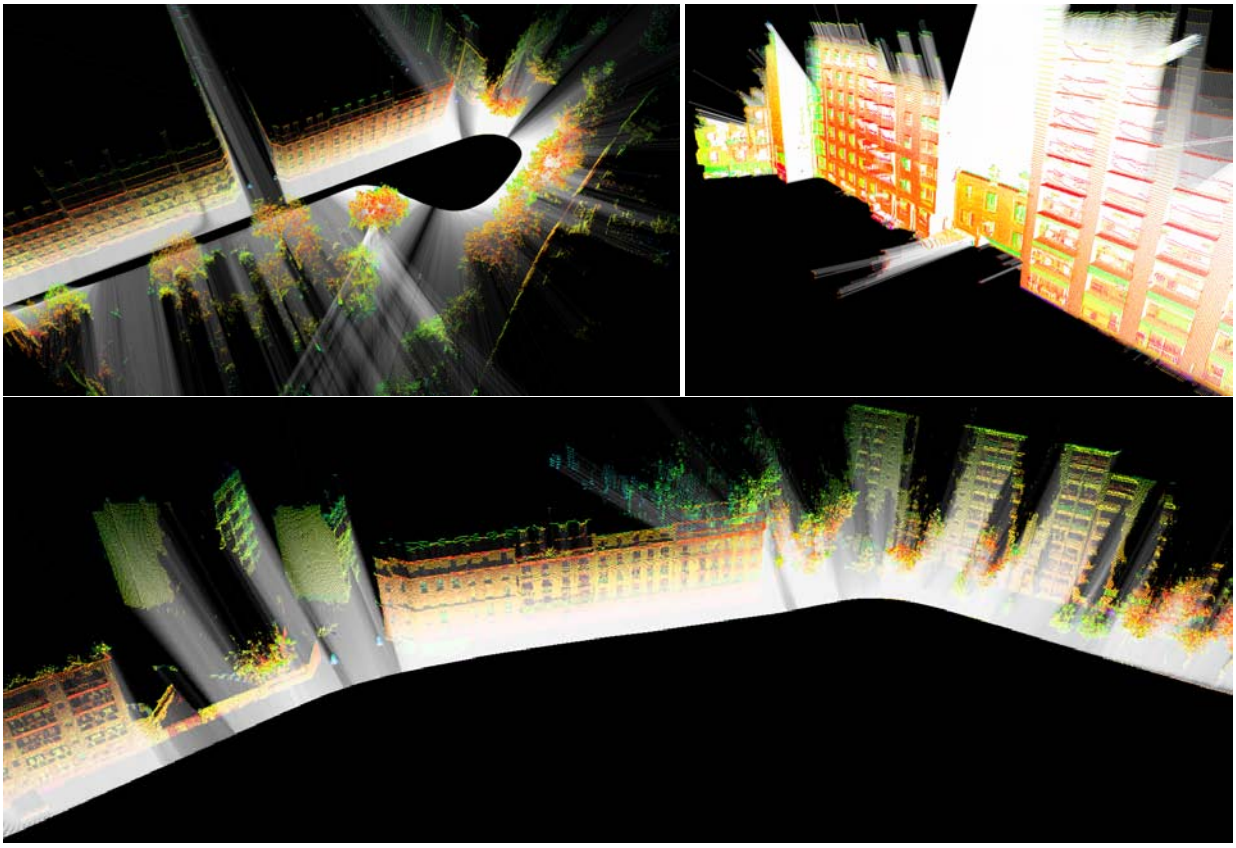


Figure 1.10: Lidar echoes colored according to their intensities and lidar beams (white).
Top-Left: During this acquisition, the lidar swept on one side, and we see that the vehicle did a U-turn.
Top-Right: We see the data as if we where inside the buildings. The beams are mostly stopped by the facade walls, but some go through the windows.
Bottom: The facades are masked by foreground objects such as trees or other facades. These occlusions create shadows on the facade walls. These areas empty of echoes are difficult to interpret: are they holes or masked areas? The lidar beams lift a part of the ambiguity: If the beams go through the area, this is a hole, else, this is a hidden area and there is no information about it.

However, these geometric assumptions are not always satisfied. If there is no information between points, there is no way to guess the true geometry between points (unless one has a priori on the scene, or for example if one has detected a known object).

Beams indicate the empty areas

An echo corresponds to a returning signal intensity peak and is usually stored as a 3D point, but, lidar measure contains more information than the echo locations. The cone formed by the laser beam delimit the area explored by the sensor, we call it "visibility cone". It corresponds to an empty area, and is truncated by the encountered objects. If the laser beams are displayed (fig: 1.10), the data appears more dense than if only the points are displayed.

As we have seen, point clouds are difficult to handle because we do not know what happens between the points. In a simple PCD, the absence of information is confused with the presence of vacuum. Thanks to the beams, we can clarify the problem by identifying the empty areas, thereby restraining the possible area for surface reconstruction.

A more complex way to model the lidar signal

In order to reduce the lack of information between echoes we can try to better exploit the signal: the lidar signal can be converted into more complex geometrical primitives (fig: 1.9). The actual lidar echo is located into an uncertainty ellipsoid which size depends both on the laser footprint and on the measure accuracy. The laser beams could be modeled by cones. A geometric model would be based on volumes containing an homogeneous matter density, as sketched in figure 1.9, with empty cones and full ellipsoids. But such a modeling implies to calculate the unions and the intersections of cones and ellipsoids, which is very complex, and would result in complex output shapes.
However, it appears that the two types of information are provided: the beams indicate the empty areas while the echoes indicate the presence of matter.

The lidar systems measure the impermeability to light

The presence of matter and the presence of vacuum are incompatible. Unless we take into account the areas semi-permeable to laser beams such as tree foliage or window glass. These areas that contain both echoes and beams, both matter and vacuum. We could then suggest an impermeability measure. Indeed, vacuum quantization and matter quantization, can cooperate in this measure: Impermeability is decreased along the beam paths while echoes vote for impermeability.

Quantizing the acquired information

The beams materialize the space portion explored by the sensor. Beyond the echoes, there is no information anymore. It is possible to determine if any space volume has been explored or not, and to count the number of beams that passed through and the number of echoes that lie into. Hence, it is possible to measure the amount of information acquired in any space volume. On the one hand, this allows to associate a reliability score to any geometrical modeling. For instance, a surface reconstructed thanks to many echoes is more reliable than a surface interpolated between only three echoes. On the other hand, this allows to distinguish between an unexplored area that contains no information and an area pierced by laser beams that indicate the presence of vacuum.
In summary, two types of quantities can be measured.

- Amount of vacuum amount of matter, impermeability measure.
- Amount of information.

Quantify the amount of information could help to manage more clearly the holes in lidar data. In addition, it may facilitate data fusion giving priority to the dataset that contains more information, or the more reliable information

1.5 Conclusion

Main obstacles

Processing lidar data acquired from MMS in urban environment seems challenging. In addition to be inherently sparse, and unevenly sampled, the data is necessarily incomplete: many parameters cannot be controlled such as the vehicle speed or trajectory, the on-board sensors are constrained to capture information from the street point of view, along the vehicle trajectory. For instance, the building volumes cannot be apprehended in their wholeness. We have to deal with facade pieces, partially hidden by occluders, such as cars, trees or other facades. The data is therefore incomplete, but can be redundant if some loops are performed during the acquisition. However, the redundant data may be inconsistent because of some georeferencing errors.

What to keep in mind

The lidar data are provided by systems that require many different technologies. Errors from each technology have different magnitude. Georeferencing errors are currently the most important. Taking into account this diversity in the data processing can yield better results.
The lidar data can take several forms (PCD, full-waveform ...). PCD are simple to process and compact but suffer from heterogeneous point density, while the full-waveform takes up more space in memory and is geometrically more complex, but contains more information. It seems important to think about the choice of the data type. Anyway, it is possible to use other information provided by the lidar systems such as the sensor location for each echo, and the acquisition order, that are simple features, light in memory and that bring valuable geometric information.
The coordinate system τ, Θ, R is independent to the sensor geolocation. It can be useful if the dataset contains self-registration errors. In addition, these coordinates are rapid to obtain as they relate to the acquisition process and they provide hypothesis for surface reconstruction. However, the "sensor geometry" is not always topologically consistent with the true geometry of scanned objects.

Chapter 2

Local Geometry Analysis of Unorganized Lidar Point Clouds

Contents

This chapter is an extension of the work presented in [10].
In this chapter, we study methods that automatically characterize the local geometry in unorganized lidar point clouds. We exclusively use the geometrical information contained into the 3D coordinates of the echoes. Nor the echo intensities, nor the sensor locations and the echo number are taken into account. We aim at a method as neutral as possible to provide low level features that describe the local geometry around each 3D point. The proposed approach computes 3D structure tensors in spherical neighborhoods of various sizes. Some "dimensionality" descriptors are derived that are combinations of the 3D structure tensor eigenvalues, they indicate whether point distribution in the neighborhood is more linear (1D), planar (2D) or volumetric (3D).
In section 2.3 we describe the "dimensionality" descriptors. In section 2.4 we investigate the way to retrieve an optimal neighborhood size for each point.

2.1 Introduction

Point cloud data from airborne and terrestrial devices provide a direct geometrical description of the 3D space. Such information is reliable, of high accuracy but spatially irregular and not dense. However, the underlying structures and objects may be detected among sets of close 3D points. The local geometry is estimated by the distribution of points in the neighborhood. Finding the best neighborhood for each point is a main issue for a large variety of common processes: data down sampling, template fitting, feature detection and computation, interpolation, registration, segmentation, or modeling purposes. The notion of neighborhood and its fundamental properties are fully described in [11].
The neighbors of a lidar point are traditionally retrieved by finding the k nearest neighbors or all the points included in a small restricted environment (sphere or cylinder) centered on the point of interest. The main problem stems from the fact that the k and environment radius values are usually :

- heuristically chosen,
- and assumed to be constant for the whole point cloud, instead of being guided by the data.

This does not ensure that all these neighbors belong to the same object as the current point. Therefore, its local description may be biased when including several distinct structures, and provides erroneous feature descriptors. Moreover, the relative variation in the spatial extent of geometrical structures is ignored. For aerial datasets, problems will occur at the borders between objects and for objects which size is inferior or close to the neighborhood size. For terrestrial datasets, in addition, the point density may significantly fluctuate due to foreground object occlusion, dependence on distance and relative orientation of the objects [12], leading to data sparseness or irregular sampling.
This chapter aims at proposing a methodology to find the optimal neighborhood radius for each 3D point on a lidar point cloud. An "optimal" neighborhood is defined as the largest set of spatially close points that belong to the same object as the point of interest. The inclusion of points lying on different surfaces is prohibited. The context of the study is rather general: in order to be applicable both on terrestrial static (TLS) and terrestrial mobile (MMS), and airborne (ALS) datasets, the method is simply based on the point location, without requiring knowledge on intensity, echo number or full-waveforms. The topology resulting from the sequential acquisition of the data is also considered to be lost ("unorganized" point cloud), preventing the adoption of specific scan line grouping methods [13]. Furthermore, the process is designed out of the scope of any application, even if the final goal is indeed to be beneficial to any application requiring a correct local description around each 3D point.
The problem of scale selection has been mainly tackled for surface reconstruction and feature extraction of scanned opaque objects. Several approaches have therefore been developed for noisy point clouds and

irregular sampling issues. Most of them are surface-based *i.e.*, they try to fit a curve or a surface of some form to the 3D point cloud [14]. Finding the optimal group that well represents the local geometrical properties is performed using indicators such as the normal and/or the curvature [15, 16, 17, 18]. For instance, it is retrieved by minimizing the upper bound on angular error between the true normal and the estimated one. Starting for the minimal possible subset around the point of interest, the neighborhood is iteratively increased until the angular variance reaches a predefined threshold [19]. Such works have been theoretically improved by Lalonde et al. [20], no more requiring knowledge on the data distribution, and applied to mobile mapping datasets. An alternative work, based of the expression of the positional uncertainty, is introduced in [21] for processing TLS datasets. However, these methods are effective for smoothly varying surfaces and may not be adapted to real anthropic surfaces acquired with various kinds of lidar systems. Furthermore, the model-based assumption does not hold when dealing with objects without predefined shapes (*e.g.*, vegetated areas) or with noise stemming for relief high frequencies (*e.g.*, chimneys and facades for ALS data or pedestrians and points inside buildings for TLS data). Consequently, in our context, a more suitable solution is to directly compute shape features [22, 18], *i.e.*, low-level primitives that may capture the variability of natural environments.
The proposed methodology is developed in Section 2.2. The shape features are described in Section 2.3. The computation of the optimal neighborhood radius embedded in a multi-scale framework is proposed in Section 2.4. Results on various laser scanning datasets are presented in Section 2.6, some possible applications are proposed in Section 2.7, and conclusions are drawn in Section 2.8.

2.1.1 Datasets

In order to assess the relevance of the proposed approach for various point densities, point distributions and points of view, three kinds of lidar datasets are tested: airborne, terrestrial static, and acquired with a mobile mapping system (named ALS, TLS, and MMS, respectively).

ALS:

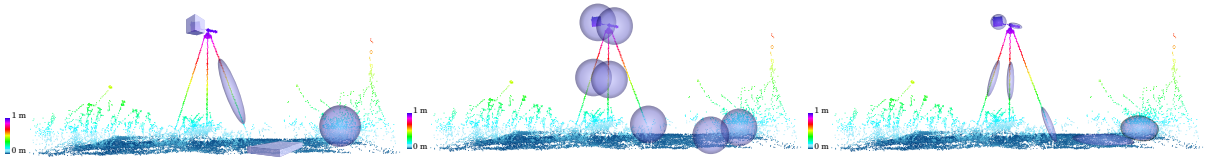
The algorithm was tested on four airborne datasets. The first one (ALS_G) has been acquired over Biberach (Germany), covering both residential and industrial areas as well as a city center with small buildings (point density of 5 pts/m^2). The second dataset (ALS_R) features a ground truth and concerns a residential area in Russia, with 5 pts/m^2 [23]. The third one covers the dense city center of Marseille (France), with high buildings, and thus sparse points on the building facades (ALS_F). Three parallel strips are present: the point density therefore varies between 2 and 4 pts/m^2 (for one strip and for the overlapping areas, respectively). Finally the fourth is an open dataset of Toronto (Canada) that contains very high skyscrapers and a crane (ALS_C).

TLS:

The terrestrial scans acquired over the Agia Sanmarina church (Greece) have been processed [21]. The dataset first offers a large variety of structures of various sizes as well as sparse vegetation on the ground. Furthermore, the point density significantly varies with the orientation of the surfaces with respect to the scanner position (TLS_G).
We also display the results on a dataset acquired by ourselves in the ign courtyard (TLS_F).

MMS:

Datasets over two urban areas (France and United States, respectively MMS_F and MMS_U) from distinct mobile mapping systems have been processed [24]. Such datasets also include man-made objects of various sizes and shapes, with varying point densities. The two specificities of MMS datasets are (1) gaps in the point cloud due to the occlusion of foreground objects, and (2) vertical privileged directions in the point clouds due to the sequential acquisition by lines. MMS_U is manually labeled with numerous object classes. MMS_F was acquired by the Stereopois.



(a) Various neighborhood shapes (b) Fixed Neighborhood (c) Adaptive Neighborhood

Figure 2.1: The neighborhood can be fixed (same shape and size for all the points) or adaptive.

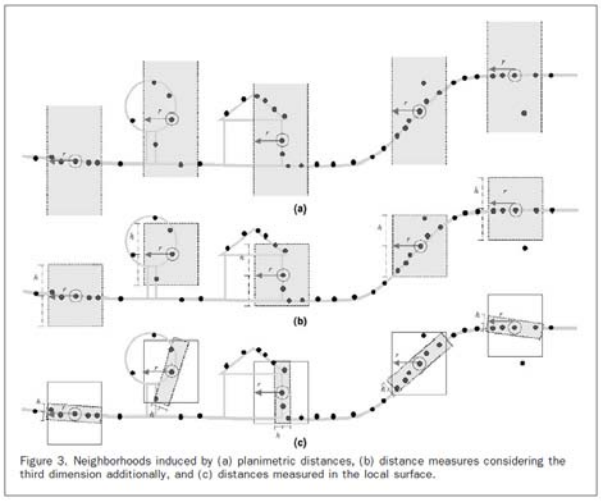


Figure 3. Neighborhoods induced by (a) planimetric distances, (b) distance measures considering the third dimension additionally, and (c) distances measured in the local surface.

Figure 2.2: An adaptive cylindrical neighborhood is used by [11] for aerial datasets.

2.1.2 Neighborhood choices

We aim at a local analysis, so we focus on the points contained in a neighborhood around each point. Some possible neighborhoods choices are presented.

In order to perform a **local** analysis, the geometrical descriptors are computed on subset of surrounding points. The results depends on this neighborhood choice.

Neighborhood Shape : Cylindrical or Spherical?

Cylindrical neighborhoods are suitable for a surface or 2.5D analysis, the point distribution can also be analyzed in a preferred direction, while isotropic spherical neighborhoods provide a more neutral analysis in 3D space.

The choice of the points that are included into the neighborhood depends on the neighborhood shape. An adaptive cylindrical neighborhood is used by [11] for aerial datasets. The neighborhood is initialized by an infinite vertical cylinder centered on each point, then, the cylinder orientation and height are adapted to the estimated surface. This adaptive neighborhood highlights the fact that the cylinder orientation have to be chosen. The cylinder orientation allows to include more or less points in the neighborhood along a privileged direction : the direction of the cylinder axis (often z axis) is specifically considered. In [11], The initial cylinder axis direction is z, the adapted cylinder is then oriented along the normal of a local plane estimation. In both cases, the cylinder is oriented according to the normal of an assumed surface: the elevation axis is orthogonal to the horizontal plane that approximates the earth surface. This adaptive neighborhood system can be seen as a coarse-to-fine approach to retrieve the local surface from a rough horizontal plane estimation. Indeed, using a cylindrical neighborhood leads to perform a surfacing or 2.5D analysis. Aerial datasets are often considered as 2.5D; there are three reasons for this:

1. As the plane flies over the scene, only the top of the objects are detected, and the acquired data corresponds to a surface that overlays the scene (except with semi-permeable elements as foliage).
2. The z coordinate range is lower than the ranges of x and z.

3. Some applications such as landcover classification aim to provide a 2D data analysis.

For these reasons, the point distribution is mainly horizontal, and the elevation is sometimes considered as a feature, more than a spatial coordinate. The infinite vertical cylinder neighborhood is equivalent to a 2D circular neighborhood and it corresponds to this horizontal cartographic point of view. It can be used to evaluate the variation of z and quantify the penetration into the vegetation. And also to detect objects that behave differently depending on z, as trees (trunk and foliage). The cylindrical neighborhood may also be adapted to the estimated surface, one could say "projected" on it. Under some assumptions of continuous smooth surface, the cylindrical neighborhood is shown to be more precise compared to the spherical neighborhood for surface curvature estimation [25].
However, we aim to provide a 3D geometrical description. For this purpose, the spherical neighborhood is preferred: isotropy and rotation invariance are ensured such that the computed shape descriptors are not biased by the shape of the neighborhood. Furthermore, r is the single parameter to be optimized.

Fixed vs Adaptive

An adaptive neighborhood avoids object mixing, but the feature comparison between different points has then less sense because they are performed on neighborhoods of different sizes or/and cardinalities.
The neighborhoods can be the same for all the points (fixed size and shape), or adapted to the context (fig:2.1). The adaptive neighborhood focuses on the only object that contains the point. It allows to avoid object mixing and may provide a more precise geometry description. An adaptive neighborhood is dynamic: a first step (that requires an initial neighborhood) is needed to analyze the geometry and deduce the adapted shape. Two or more steps are thus necessary to adapt the neighborhood. A problem of adaptive neighborhoods is that the results for each point are less comparable because the neighborhoods are not equivalent. The ideal would be to compare neighborhoods that have the same shape, size and number of points. The constant size neighborhoods provide a geometry description at a given scale. This seems to be the more rational choice to compare the geometrical behaviors. However, as the point density varies, the number of points in such neighborhoods may vary from one (if the point is an outlier), to n points. For instance, in the MMS datasets there is often a greater point density at the bottom of the facade (close to the sensor) than at the top (far from the sensor). This may produce various results while the facade geometry remains unchanged from the bottom to the top. At the opposite, the "k-nearest neighbors" neighborhoods allow to maintain a constant number of points, but the size varies. A spatial pruning may help to obtain more comparable neighborhoods, as explained in section 2.5.3. The "point density" issue is more detailed in chapter 1.

2.2 Proposed Method

2.2.1 Description

Our methodology aims at finding the optimal neighborhood radius for each lidar point, working directly and exclusively in the 3D domain, without relying on surface descriptors (such as normals) or structures (such as triangulations or polygonal meshes). It is composed of two main steps:

1. Computation of three dimensionality features for each point, between predefined minimal and maximal neighborhood scale. These features describe the distribution of the points in 3D space, and more exactly, the matching between the local point cloud and each of the three dimensionalities (linear, planar or volumetric).

2. Scale selection: retrieval of the neighborhood radius for which one dimensionality is most dominant over the two others.

The three dimensionality features (a_{1D}-a_{2D}-a_{3D}) are computed exhaustively, at each point and for each acceptable neighborhood scale, from the local covariance matrix. An isotropic spherical neighborhood, centered on the point of interest, is adopted for this purpose. These low-level features, as well as the automatic set up of the radius lower and upper bounds are described in Section 2.3.

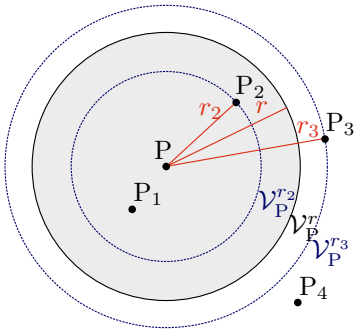


Figure 2.3: Neighborhood denomination.
The neighborhood $\mathcal{V}_P^r$ of a point P at scale r is defined as the set of points P_k verifying :
$P_k \in \mathcal{V}_P^r \Leftrightarrow \|P - P_k\| \leq r$.

Then, the optimal radius is retrieved by comparing the behaviors of these three features between the minimal and maximal acceptable radius. Two *radius-selection* criteria are tested to evaluate each scale and find the most relevant value. This multi-scale analysis is performed in order to capture variation in shape when aggregating points for an object distinct from the object of interest (edge effect or outliers). It is also useful in case of significant density variation and lack of support data for gathering points over a large volume while ensuring the conservation of the prevailing dimensionality.
Finally, our method presents three interesting characteristics:

- Definition of a confidence index of the saliency of one dimensionality over the two other ones.
- Multi-scale analysis and automatic set up of the bounding scales.
- Labeling of each point according to its privileged dimensionality, providing an interesting basis for segmentation and classification algorithms.

2.3 Shape Features

In this section, we describe the shape features computed in a spherical neighborhood around each lidar echo. The 3D structure tensor gives the average behavior of the points around the centroid. It indicates how the point distribution is stretched or squeezed along three orthogonal directions/dimensions. We derive "dimensionality" features to quantify whether the points are rather distributed in one, two or three orthogonal directions in the neighborhood. and a labeling indicating the most predominant behavior (fig 2.7).

1D : linear distribution

2D : planar distribution

3D : scatter distribution

2.3.1 Notation

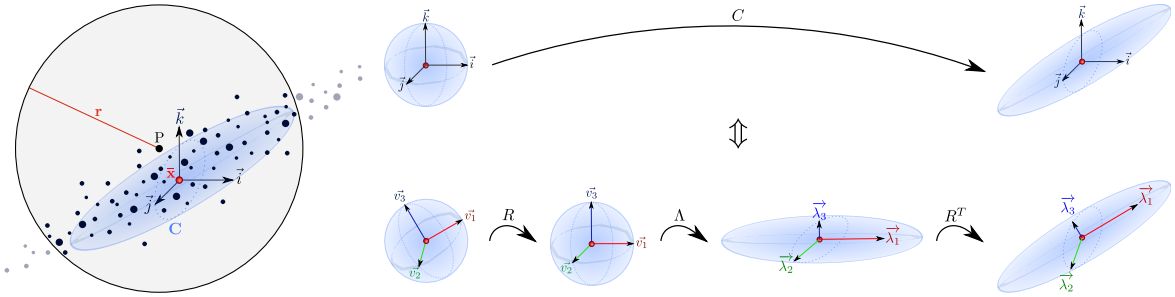
The neighborhood $\mathcal{V}_P^r$ of a point P at scale r is defined as the set of points P_k verifying :

$$P_k \in \mathcal{V}_P^r \Leftrightarrow \|P - P_k\| \leq r. \tag{2.1}$$

The proximity order of the points from P is noted k. P_1 is the closest point to P and $P \Leftrightarrow P_0$. See fig 2.3.

2.3.2 Principal Component Analysis and 3D Structure tensors

A classical approach consists in performing a Principal Component Analysis (PCA) of the 3D coordinates of $\mathcal{V}_P^r$ [18, 26]. This statistical analysis uses the first and second moments of $\mathcal{V}_P^r$, and results in three orthogonal vectors centered on the centroid of the neighborhood. The PCA synthesizes the distribution of points along the three dimensions [27], and thus models the principal directions and magnitudes of



(a) The shape of $\mathcal{V}_{\mathrm{P}}^{r}$ is represented by an oriented ellipsoid.

(b) The tensor function $\mathbf{C}$ is decomposed into a rotation $\mathbf{R}$ and a scaling $\sqrt{\mathbf{\Lambda}}$.

Figure 2.4: An ellipsoid shape to estimate point distribution in the neighborhood.

variation of the point distribution around the center of gravity. These magnitudes are combined to provide shape descriptors for each of the three dimensions. More advanced features based on harmonics or spin images [28, 29] are not necessary since the segmentation task, which indeed requires contextual knowledge, is not tackled in this paper.

Let $\mathbf{x}_i = \begin{pmatrix} x_i & y_i & z_i \end{pmatrix}^T$ and $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1,n} \mathbf{x}_i$ the center of gravity of the n lidar points of $\mathcal{V}_{\mathrm{P}}^{r}$.

Given $\mathbf{M} = \begin{pmatrix} \mathbf{x}_1 - \bar{\mathbf{x}} & \dots & \mathbf{x}_n - \bar{\mathbf{x}} \end{pmatrix}^T$, the 3D structure tensor is defined by $\mathbf{C} = \frac{1}{n}\mathbf{M}^T\,\mathbf{M}$. Since $\mathbf{C}$ is a symmetric positive definite matrix, an eigenvalue decomposition exists and can be expressed as $\mathbf{C} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^T$, where $\mathbf{R}$ is a rotation matrix, and $\mathbf{\Lambda}$ a diagonal, positive definite matrix, known as *eigenvector* and *eigenvalue* matrices, respectively (fig:2.4). The eigenvalues are positive and ordered so that $\lambda_1 \geq \lambda_2 \geq \lambda_3 > 0$. $\forall j \in [1,3]$, $\sigma_j = \sqrt{\lambda_j}$, denotes the standard deviation along the corresponding eigenvector $\overrightarrow{v_j}$. Thus, the PCA allows to retrieve the three principal directions of $\mathcal{V}_{\mathrm{P}}^{r}$, and the eigenvalues provide their magnitude. The average distance, all around the center of gravity, can also be modeled by a surface. The shape of $\mathcal{V}_{\mathrm{P}}^{r}$ is then represented by an oriented ellipsoid. The orientation and the size informations are divided between $\mathbf{R}$ and $\mathbf{\Lambda}$: $\mathbf{R}$ turns the canonical basis into the orthonormal basis ($\overrightarrow{v_1}$,$\overrightarrow{v_2}$,$\overrightarrow{v_3}$) and $\sqrt{\mathbf{\Lambda}}$ transforms the unit sphere to an ellipsoid (σ_1, σ_2 and σ_3 being the lengths of the semi-axes). As enhanced in Figure 2.6 and for instance in [26], such an ellipsoid reveals the linear, planar or volumetric behavior of the neighborhood *i.e.*, whether the point set is spread in one, two or three dimensions (blue, gray, and green ellipsoids in Figure 2.6, respectively).

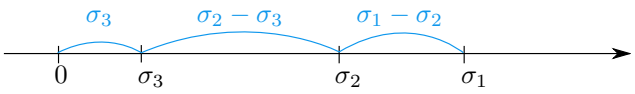
2.3.3 Smoothed geometry

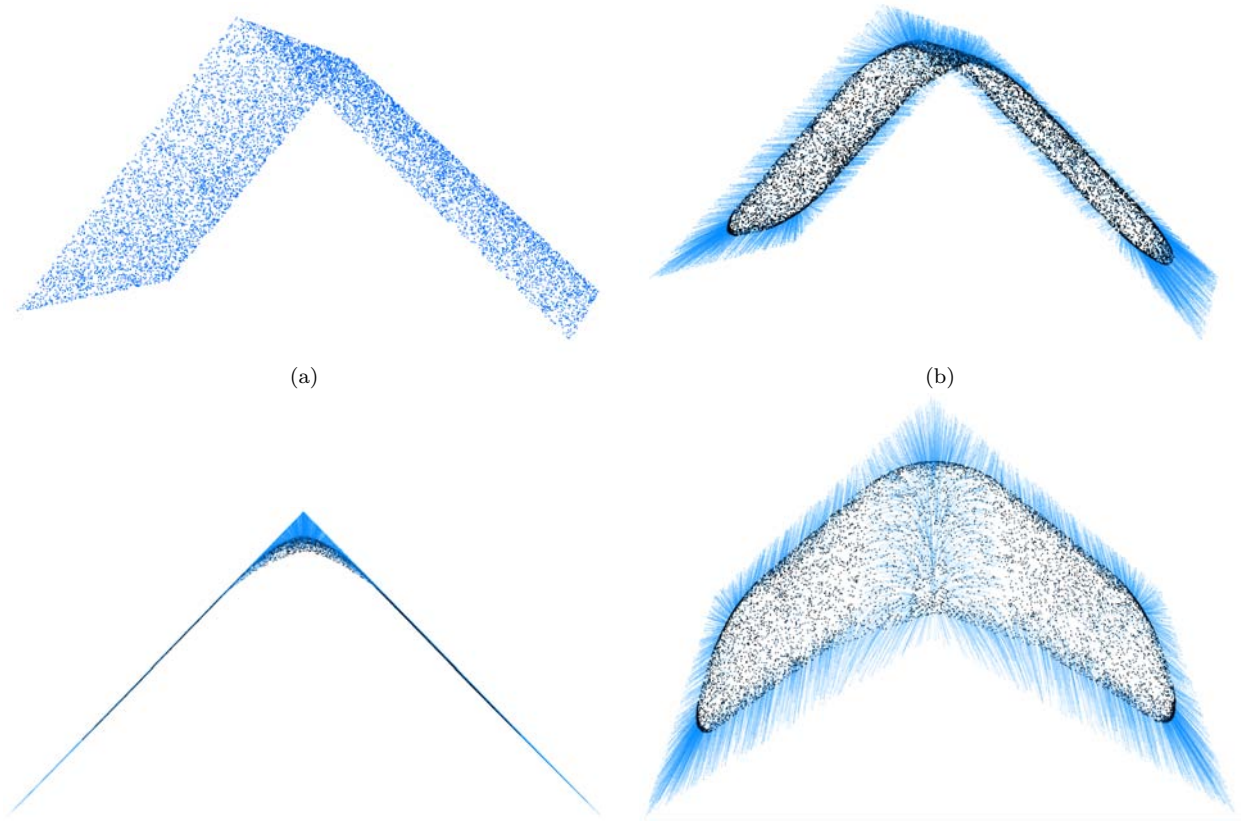
The geometrical analysis provided by the structure tensors have some limitations that we will try to exhibit thanks to a synthetic data set (fig 2.5). The centroid is calculated for the 1024 nearest neighbors of each point. The centroid set is more compact than the point set and the ridge is smoothed. The PCA describes the behavior of the points that lie in a sphere that is centered on the point, but the eigenvectors and the ellipsoid are centered on the centroid. The geometric description therefore "moves" towards the centroid. In conclusion, this method provides smoothed results that badly describes sharp edges.

2.3.4 Dimensionality features and labeling

Various geometrical features can be derived from the eigenvalues. Several indicators have already been proposed [30, 31], and the following ones have been selected (Figure 2.9) to describe the linear ($a_{1\mathrm{D}}$), planar ($a_{2\mathrm{D}}$), and scatter ($a_{3\mathrm{D}}$) behaviors within $\mathcal{V}_{\mathrm{P}}^{r}$:

$$a_{3\mathrm{D}} = \frac{\sigma_3}{\mu}, \quad a_{2\mathrm{D}} = \frac{\sigma_2 - \sigma_3}{\mu}, \quad a_{1\mathrm{D}} = \frac{\sigma_1 - \sigma_2}{\mu},$$





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

Figure 2.5: Centroid displacement. This synthetic dataset contains 10000 points randomly scattered in a rectangle bent at right angle (a). The centroid is calculated for the 1024 nearest neighbors of each point. In (b), (c) and (d), the displacements between each point (blue) and its centroid (black) are displayed.

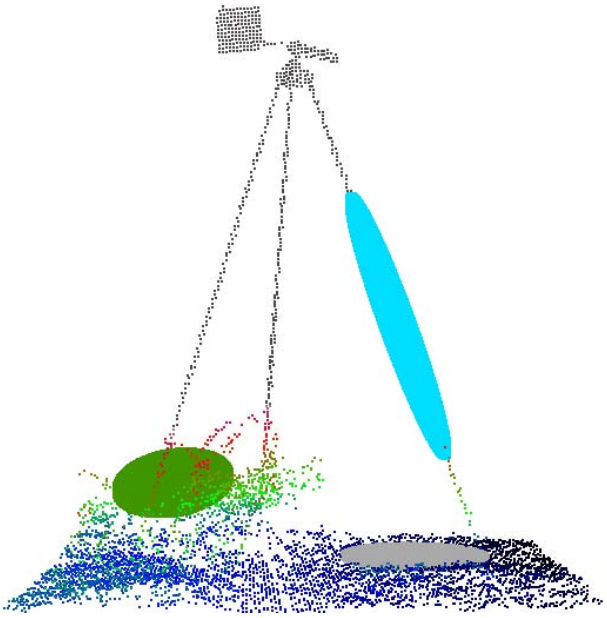


Figure 2.6: Three examples of ellipsoids (blue, gray and green) computed over three areas of interest of distinct dimensionalities for the TLS_G dataset (a tripod over a low and sparse vegetation – height colored).

where μ is the normalization coefficient. Both choices $\mu = \sigma_1$ and $\mu = \sum_{d=1,3} \sigma_d$ are conceivable. But with $\mu = \sum_{d=1,3} \sigma_d$, the values are not equally bounded. For instance, by remembering that $0 \leq \sigma_3 \leq \sigma_2 \leq \sigma_1$, $a_{3\mathrm{D}} \leq 1/3$ while $a_{2\mathrm{D}} \leq 1/2$. In order to extend intervals to $[0, 1]$, some coefficients are applied in [32] and they obtain

$$A_{1\mathrm{D}} = \frac{\sigma_1 - \sigma_2}{\sum_{d=1,3} \sigma_d}, \quad A_{2\mathrm{D}} = 2\frac{\sigma_2 - \sigma_3}{\sum_{d=1,3} \sigma_d}, \quad A_{3\mathrm{D}} = 3\frac{\sigma_3}{\sum_{d=1,3} \sigma_d},$$

We preferred $\mu = \sigma_1$, because it directly implies $a_{1\mathrm{D}}$, $a_{2\mathrm{D}}$, $a_{3\mathrm{D}} \in [0, 1]$ and because the features are more comparable (especially $A_{3\mathrm{D}}$ that is often smaller than $A_{1\mathrm{D}}$ and $A_{2\mathrm{D}}$). In conclusion, we employ the following formulas:

$$a_{1\mathrm{D}} = \frac{\sigma_1 - \sigma_2}{\sigma_1}$$

$$a_{2\mathrm{D}} = \frac{\sigma_2 - \sigma_3}{\sigma_1}$$

$$a_{3\mathrm{D}} = \frac{\sigma_3}{\sigma_1}$$

The dimensionality labeling (1D, 2D or 3D) of $\mathcal{V}^r_{\mathrm{P}}$ is defined by:

$$\mathrm{d}^*(\mathcal{V}^r_{\mathrm{P}}) = \underset{d \in [1,3]}{\arg\max}[a_{_{\mathrm{dD}}}]. \tag{2.2}$$

If $\sigma_1 \gg \sigma_2, \sigma_3 \simeq 0$, $a_{1\mathrm{D}}$ will be greater than the the two others so that the dimensionality labeling $d^*(\mathcal{V}^r_P)$ results to 1. Contrariwise, if $\sigma_1 , \sigma_2 \gg \sigma_3 \simeq 0$, $a_{2\mathrm{D}}$ *i.e.*, the planar behavior will prevail. At last, $\sigma_1 \simeq \sigma_2 \simeq \sigma_3$ implies $d^*(\mathcal{V}^r_P) = 3$.
As $a_{1\mathrm{D}}$, $a_{2\mathrm{D}}$, $a_{3\mathrm{D}} \in [0, 1]$ and $a_{1\mathrm{D}} + a_{2\mathrm{D}} + a_{3\mathrm{D}} = 1$, the three features can be considered as the probabilities of each point to be labeled as 1D, 2D, or 3D. It will help us to select the most appropriate neighborhood size by finding which radius favors the most one dimensionality (see Equation 2.3).

2.3.5 Derive a normal vector from the structure tensor

In flat areas, a plane estimation is provided by the structure tensor. $\overrightarrow{v_1}$ and $\overrightarrow{v_2}$ belong to this plane, and $\overrightarrow{v_3}$, the third eigen vector is orthogonal to it. $\overrightarrow{v_3}$ is thereby a normal vector. The main weakness of $\overrightarrow{v_3}$ as

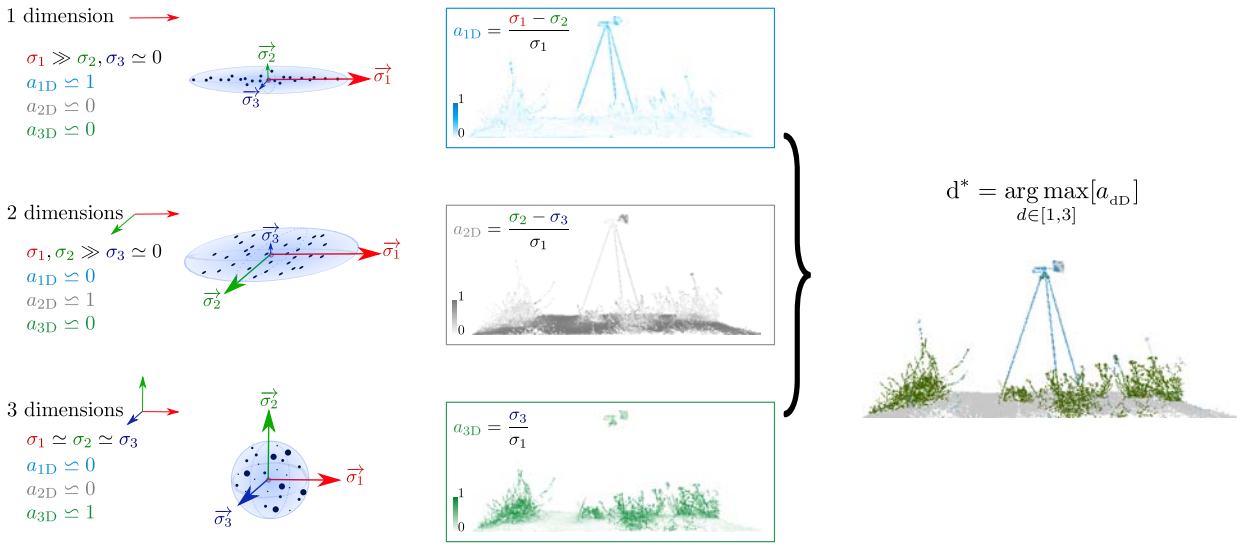


Figure 2.7: From structure tensors to dimensionality features and labellings.

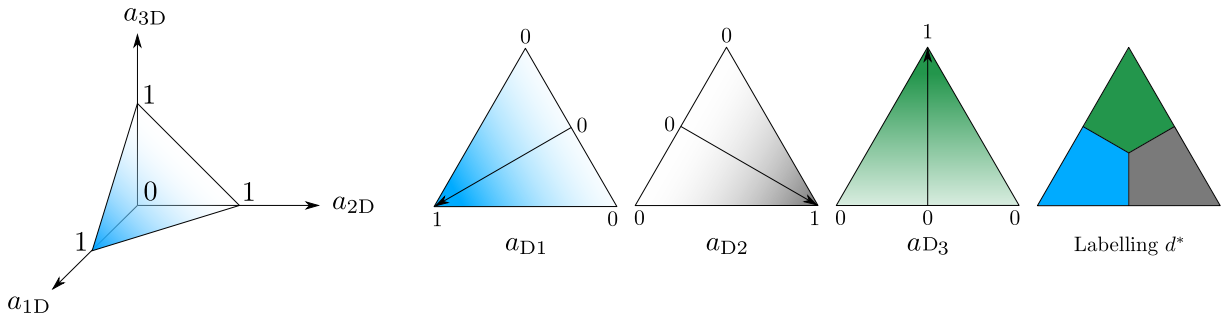


Figure 2.8: Dimensionality labeling d^* displayed in the dimensionality space.

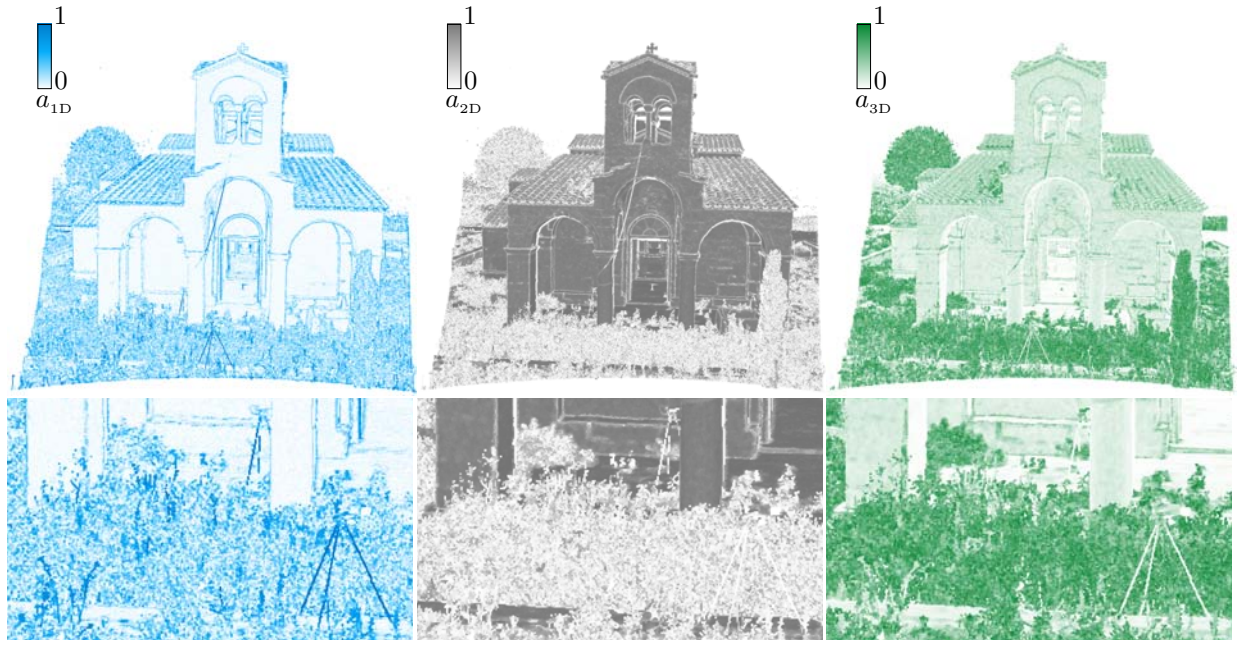


Figure 2.9: Behaviors of the three dimensionality features (TLS_G dataset).

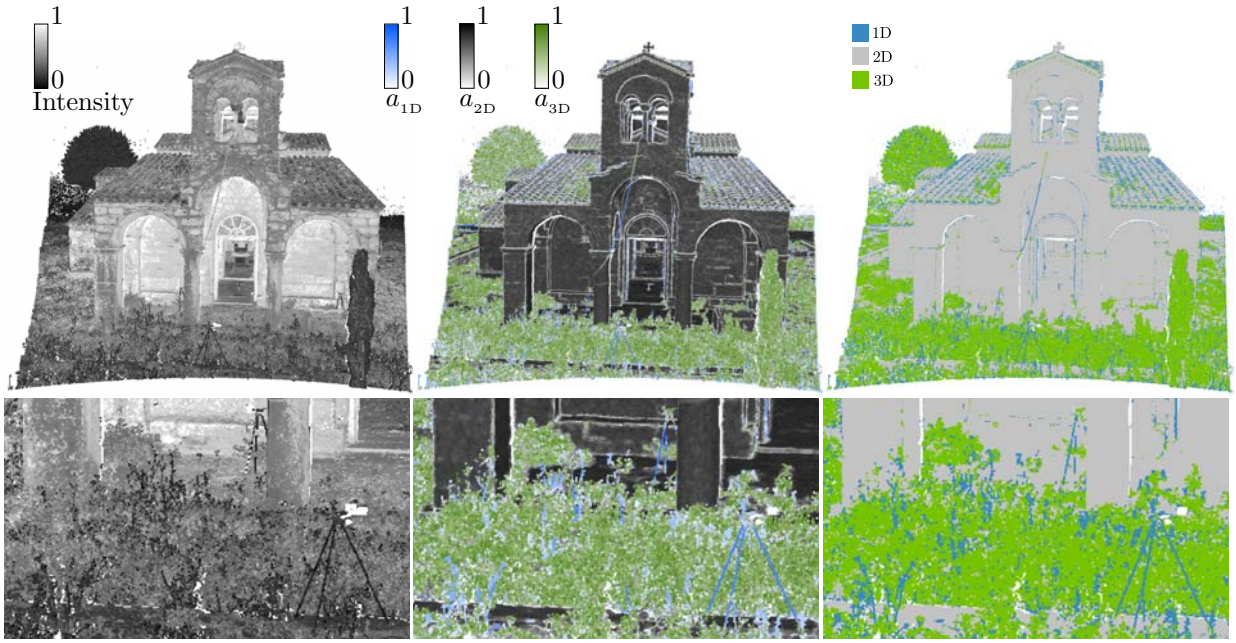


Figure 2.10: Intensity, dimensionality features and labeling (TLS_G dataset).

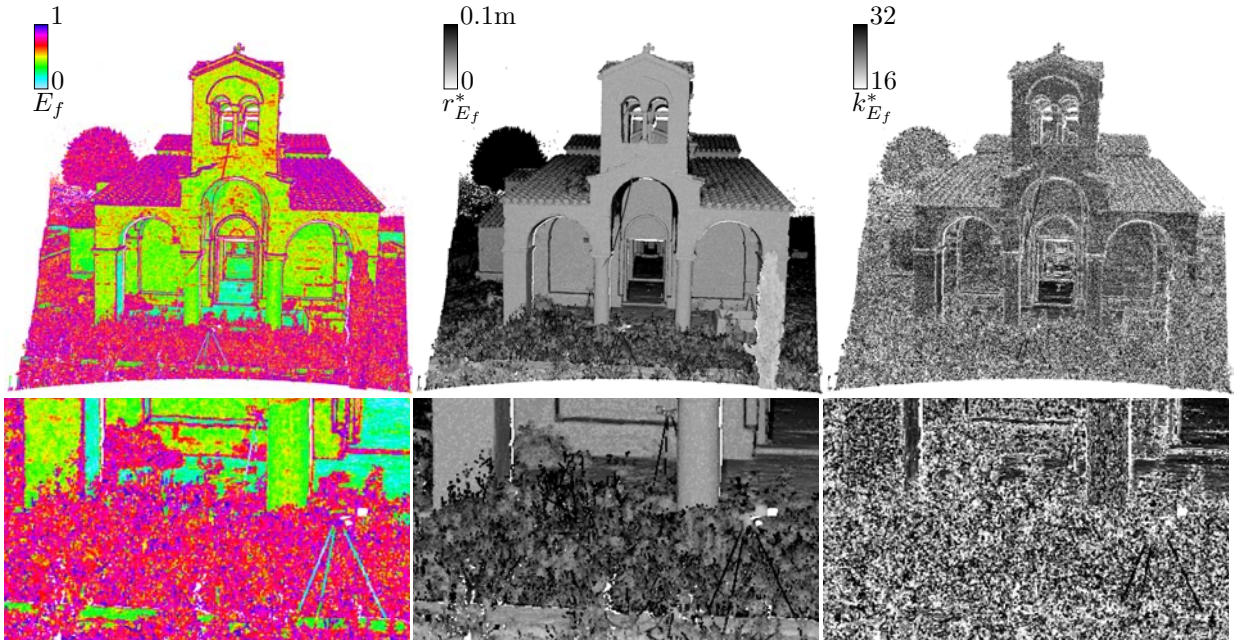


Figure 2.11: Entropy feature, optimal neighborhood radius and optimal number of neighbors (TLS_G dataset).

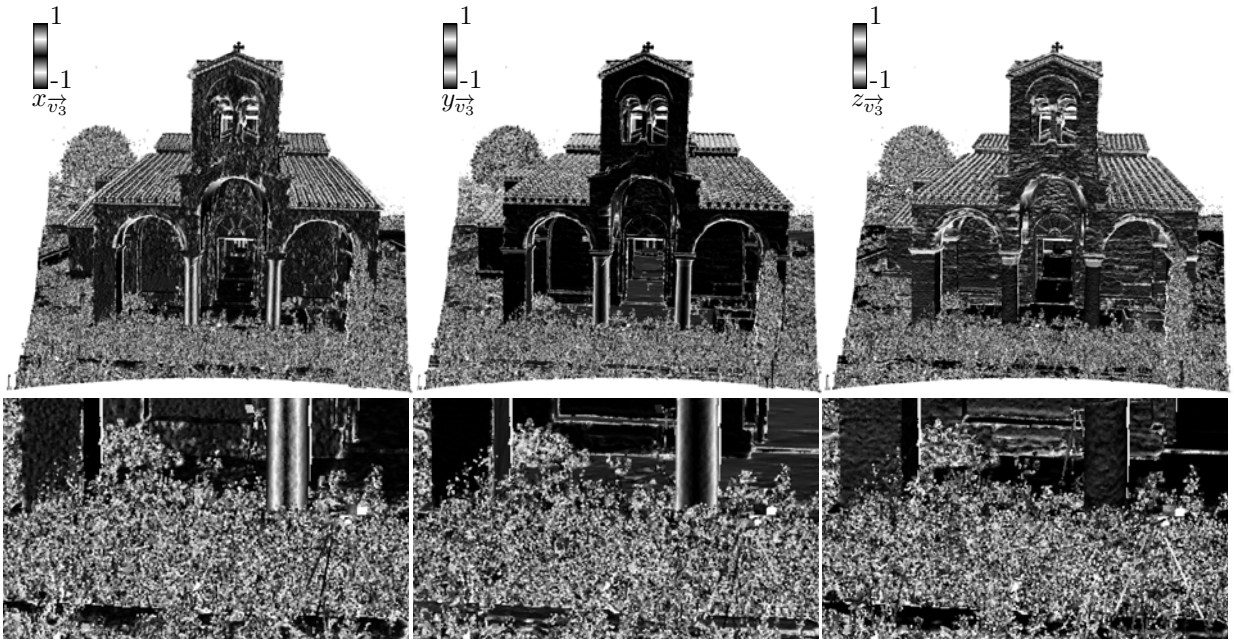


Figure 2.12: Normal vector (TLS_G dataset).

normal vector is the noise-sensitivity: the plane outliers included in the neighborhood alter the plane estimation. However, in the next section (2.4), an optimal neighborhood size selection is proposed according to the dimensionality features, that may favor the flattest neighborhood. As will be explained, the size selection do not favor the flat neighborhood in every cases, but -roughly- only if the area is labeled as 2D. As shown in figure, the normals seem correct along the walls and are noisy in the bush. Anyway normals do not make sense in non-surface areas.
The proposed method allows to directly derive normal vectors that are maybe not optimal, but reliable, especially in planar areas.

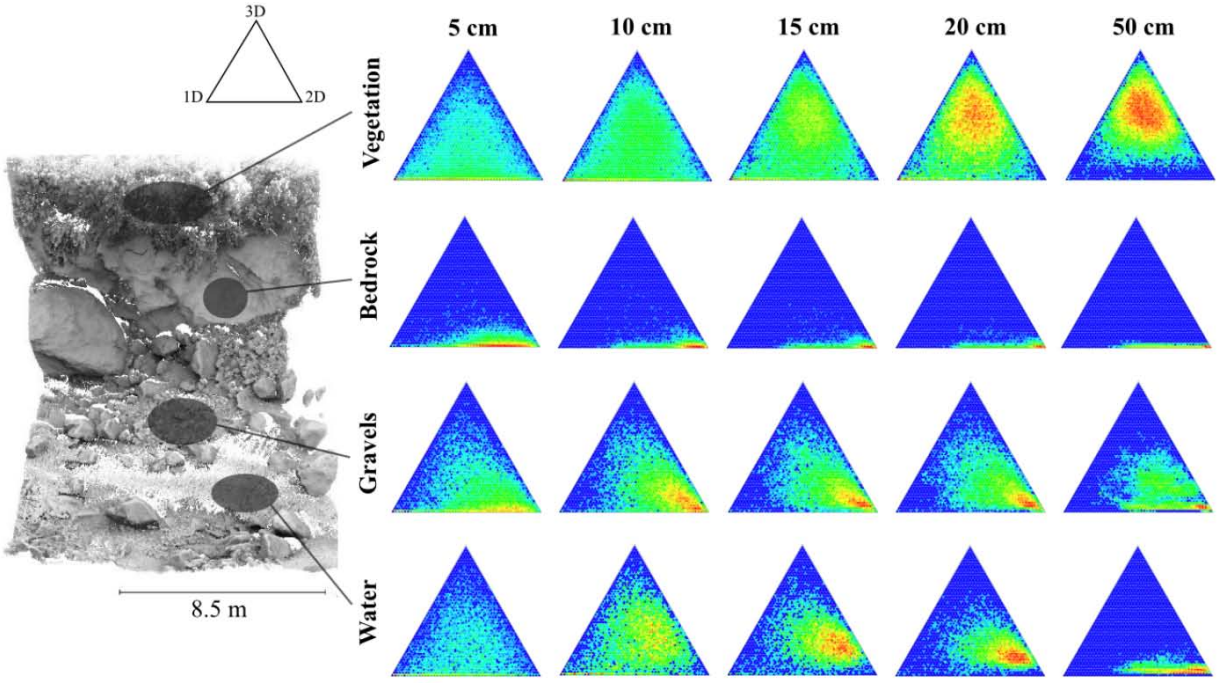
Normal Orientation

The sensor location indicates the orientation of the reflected surface, and thus the orientation of the estimated normal to this surface. As shown on figure 2.13, the extremal cases are when the normal ($\overrightarrow{n}$) to the surface is detected with a grazing angle, the normal is then orthogonal to the beam ($\overrightarrow{b}$) (highest and lowest arrows). For any detected surface, the inequality $\overrightarrow{n}.\overrightarrow{b} \leq 0$ is therefore ensured, which enables to choose the normal orientation ($\Leftrightarrow$ normal sign).

Figure 2.13: The Far Side of the Moon: If all the beams have the same direction, the sensor can only see a half of a spherical object.

2.3.6 Dimensionality features at several scales

The features can be calculated for several neighborhood sizes as in [33] where an approach very similar to ours allows to quantify the dimensionality (1D,2D or 3D) of a neighborhood (fig 2.14). This multi-scale



Left : excerpt from the point cloud of fig. 1. Right: dimensionality diagrams for the four main classes of a mountain river environment at scales ranging from 5 to 50 cm. Colors from blue to red correspond to the density of points from the training dataset and characterize the degree of clustering around a given dimensionality.

Figure 2.14: Multiscale analysis from the paper [33]

attributes are used as inputs for a classification in geomorphology (rivers, coastal environments, cliffs,...). This classification is considered as efficient especially in separating vegetation from the rest of the data. The geometric properties thus depend on the neighborhood size, and the dimensionalities calculated at several sizes are not redundant, on the contrary they improve classification. However, we assume the existence of an optimal size for the dimensionality calculation. In fact, this is not contradictory with a multi-scale framework: If the dimensionalities are computed at several octaves, the optimal size can be retrieved in between the size bounds of each octave. Such multi-scale framework has not been investigated. In the following section, we focus on an automatic way to obtain the optimal size, with no a priori knowledge about the echoes distribution and the scanned objects.

2.4 Scale selection

2.4.1 Optimal neighborhood radius

In order to find a optimal neighborhood size, A *radius-selection* criterion have been developed, namely the entropy feature E_f.

As presented in Section 2.2, the dimensionality features are computed for increasing radius values between a lower bound and an upper bound ($r_{\min}$ and $r_{\max}$, respectively). Their set up is presented in Section 2.5. They represent the minimal and maximal acceptable neighborhood radius according to the area of interest and the sensor. The $[r_{\min}, r_{\max}]$ space has been sampled in 16 values, which is a suitable trade-off between tuning accuracy and computing time. Since the radius of interest is usually closer to $r_{\min}$ than to $r_{\max}$, the r values are not linearly increased but with a square factor. This allows to have more samples near the radius of interest and less when reaching the maximal values.
A *radius-selection* criterion have been developed, namely the entropy feature E_f.

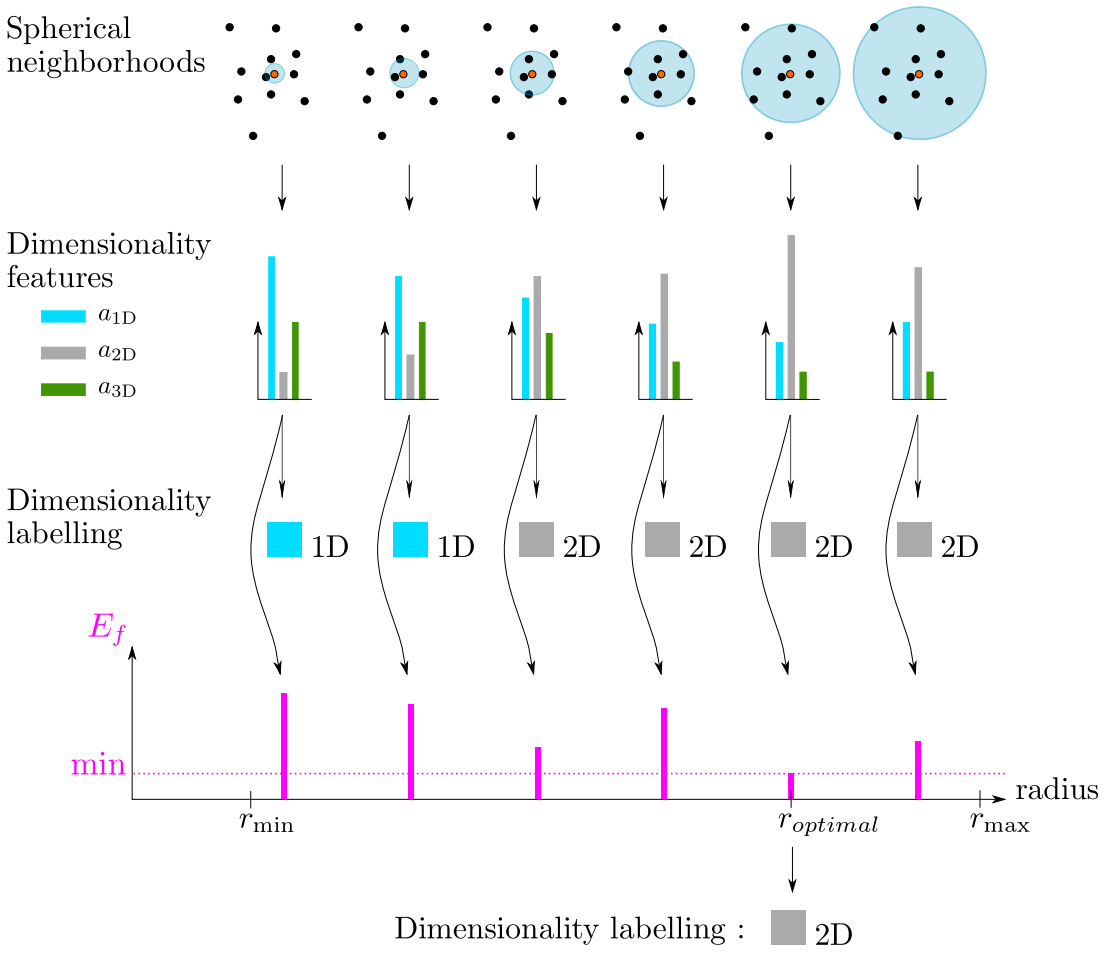


Figure 2.15: Overview of the scale selection method.

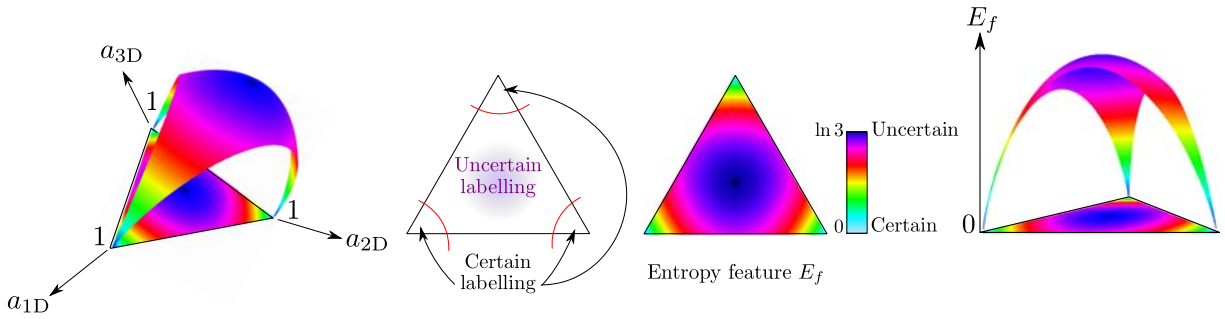


Figure 2.16: The entropy value is displayed in the dimensionalities space.

2.4.2 How to define a *radius-selection* criterion?

The problem is that in the absence of ground truth, or a clearly identified primitive such as a plane or a straight line, there is no way to verify which neighborhood is best. The neighborhoods have to be compared together in order to select the more relevant.
In [34], the ellipsoids are progressively merged. The merging process continues or stops according to a criterion formulated as a threshold on a distance measure, defined on the merge candidate ellipsoids. Two criteria are proposed, based on a *density-distance* and a *shape-distance*. Optimizing the density implies to be as close as possible to the points (to the data). It is thus a data criterion, as the variance used in [35]. If the ellipsoids are merged according to the *shape-distance*, distance to the data is minimized, the merging tries to reduce the redundancy, and thus, to output a simpler model. For instance, the flat ellipsoids of the ground may be merged into a unique "ground ellipsoid". This modeling may loose some details of the terrain relief (move away from the data), but it decreases the number of ellipsoids. The *shape-distance* is thus a MDL criterion (Minimum Length Description).
In the merging process, the ellipsoids are compared by pair, and there is not the notion of "most relevant ellipsoid" in the *shape-distance*. We tested density criteria that were not convincing, because the density is too dependent on the neighborhood size, moreover, density and dimensionality are also dependent.
As we are not able to qualify the geometrical features, we decided to design a criterion that asses the adequacy with the proposed description. This is the purpose of the entropy Feature E_f.

2.4.3 Entropy Feature E_f

a measure of unpredictability is given by the Shannon entropy of the discrete probability distribution $\{a_{1D}, a_{2D}, a_{3D}\}$:

$$E_f(\mathcal{V}_P^r) = -a_{1D}\ln(a_{1D}) - a_{2D}\ln(a_{2D}) - a_{3D}\ln(a_{3D}). \tag{2.3}$$

Figure 2.16 gives the spatial representation of this entropy value.
The lower $E_f(\mathcal{V}_P^r)$ is, the more one dimensionality prevails over the two other ones.
The relevance of scale selection is demonstrated in Figure 2.17.
This criterion allows to define an optimal radius $r^*_{E_f}$ that minimizes $E_f(\mathcal{V}_P^r)$ in the $[r_{\min}, r_{\max}]$ space :

$$r^*_{E_f} = \underset{r\in[r_{\min},\, r_{\max}]}{\arg\min}\ E_f(\mathcal{V}_P^r). \tag{2.4}$$

A dimensionality labeling is then provided by : $d^*(\mathcal{V}_P^{r^*_{E_f}})$.

2.5 Bounding scales

The optimal neighborhood radius is retrieved between predefined lower and upper bounds. They depend on various characteristics, and are therefore specific to each dataset.

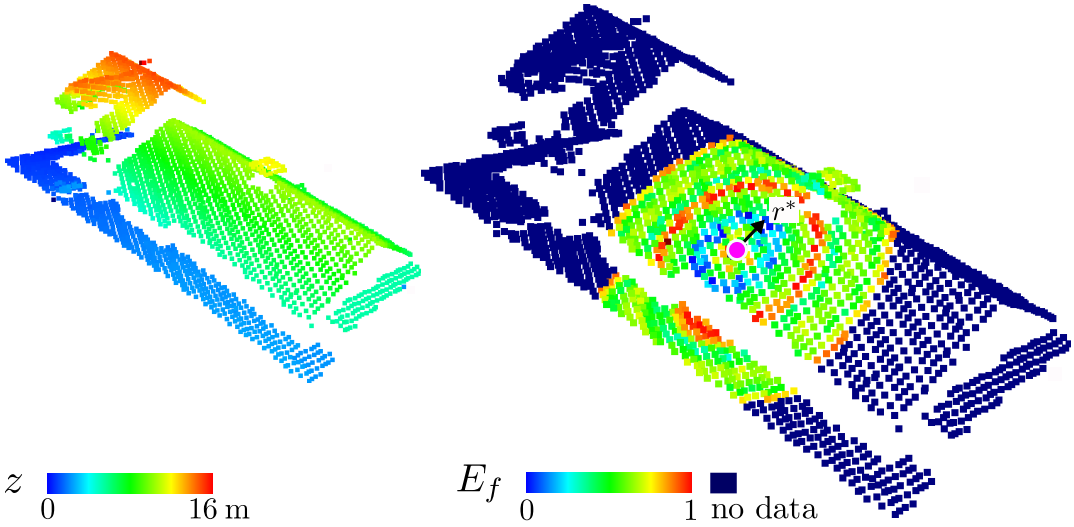


Figure 2.17: Illustration of the relevance of the entropy feature for scale selection (building roof in ALS_G). **Left**: 3D point cloud (height colored). One can see the two main roof parts and a central superstructure. **Right**: 3D point cloud colored with E_f computed for one point of interest (pink dot). The color of each neighbor P_k corresponds to the E_f value, computed on the smallest neighborhood containing P_k. The entropy first decreases until the neighborhood reaches the edge of the roof (blue circle – optimal size). Then, the entropy increases and becomes maximum when the neighborhood gathers distinct objects (red circle), here the ground and a chimney.

2.5.1 Lower Bound

The choice of the lower bound is driven by (1) the noise level; (2) the scan sampling; (3) the scene geometry; and (4) computational constraints :

(1) The noise level

Local scattering may appear for linear or planar surfaces. It may stem from sensor noise or unfavorable geometrical configurations (*e.g.*, laser beam nearly parallel to a surface). Shape features are then biased and the point cloud may locally be erroneously labeled as "volumetric". The knowledge of the intensity of such noise is a prerequisite for solving this issue. Alternatively, the minimum size can be estimated applying the methods mentioned in Section 2.1 for this purpose.

(2) The scan sampling

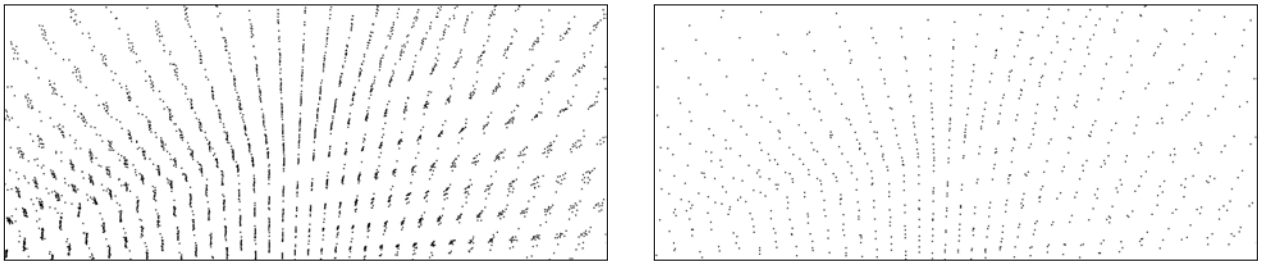
The laser beam deflection system or the kind of mapping device may create point clouds with very irregular landscape sampling (low values in one direction and larger values in the orthogonal one). This is particularly true for MMS since datasets are acquired line by line with forward motion of the device. To cope with this issue, the $r_{\min}$ value is increased until $\mathcal{V}^r_{\mathrm{P}}$ blends points belonging to at least two different scan lines.
In fact, there are two causes of the scan sampling.
Survey specifications: The point density, the overlapping ratio between stripes...
Scan pattern that depends on the technology (polygon mirror or optical fiber).

(3) The scene geometry

As on facades in ALS or cross-roads facades in MMS, where the points are scattered, the point sampling is due to the scene geometry.

(4) Computational constraints

A minimal number of points is required for the PCA. We consider relevant statistics start with 10 points. If the two first above-mentioned issues are not problematic, the point density directly provides the lower



(a) (b)

Figure 2.18: Pruning of the Lucerne dataset: A point is kept in every 0.4m^3 voxel. Original distribution (a) and sub-sampled (b).

bound.

2.5.2 Upper Bound

The selection of the upper bound is not critical and can be tuned with the knowledge of the lower frequencies of the relief *i.e.*, the size of the largest objects in the scene. For TLS and MMS datasets, this corresponds to facades (typically 3 m) whereas for ALS data, ground regions and large buildings are involved. As E_f remains almost constant for these large planar areas, a cut-off value around 5 m works well, even if in practice 3-4 m are sufficient.

2.5.3 Spatial Pruning

The spatial pruning may appear as the "easy way" to bypass point density problem. Indeed, spatial pruning is easy and fast to perform, and the resulting point set is lighter. But it allows to become independent of the lidar sampling and thus to obtain a more homogeneous point distribution. At the lowest scales, point distribution depends on acquisition (noise, lidar sweeping method...), and do not reflect the "true" geometry. As explained previously, we prefer to avoid too small neighborhoods thanks to lower bounds. Another way to abstract from the noise level is the spatial pruning. We use the following algorithm:

Spatial Pruning

Inputs : the PCD, the voxel width w.

Voxelize the Space : A data structure is created to accumulate the points in the voxels. For this purpose, we use a C++ map. In order to associate a key to each voxel, we build an Hilbert space-filling curve with a resolution level equal to w. The curve index can then be computed for each point, indicating in which voxel the point lies. Using such a map allows to create only the voxels that are not empty.

For each not empty voxel, the centroid is computed and the closest point to the centroid is kept.

The point density is bounded by a maximum value of w^{-3} (1 point per voxel). This upper bound is a security:

- If a "fixed radius" neighborhood is used, the number of points per neighborhood is limited. This is not the case in the original PCD, where a neighborhood (in a high point density area) can contain an arbitrarily large number of points. Such "crowded" neighborhoods are time consuming and it is useful to avoid its.
- If a "k-nearest neighbors" neighborhood is used, the same high point density areas lead to arbitrarily small neighborhood sizes in the original PCD. It is also useful to avoid its, because the strong size variations make the neighborhoods comparisons less significant.

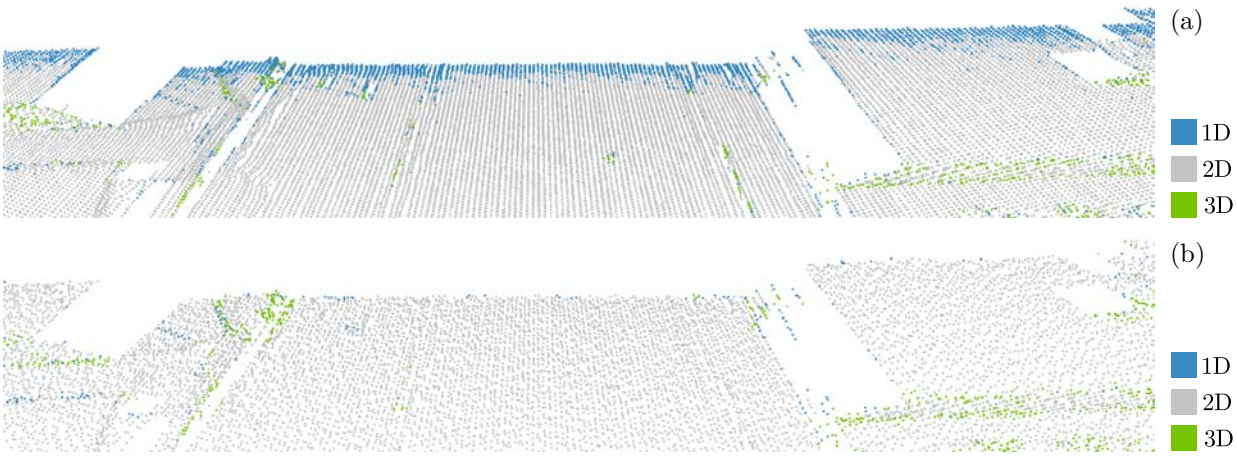


Figure 2.19: Dimensionality labellings in ALS_C: comparison between the dataset before (top) and after a spatial pruning (bottom). The spatial pruning kept one point per cubic meter. In the original dataset, the point distribution highlights the scan lines that are labeled in 1D. The spatial pruning allows to abstract from this acquisition-dependent distribution and the points are labeled in 2D.

The neighborhood size varies less whether with "fixed radius" or "k-nearest neighbors". The more homogeneous density reinforces the correlation between the neighborhood size and its number of points. The choice of the neighborhood type is thus less important.
The spatial pruning is a simple way to avoid many problems caused by density variation. Even if it induces a loss of accuracy, it seems to be useful in many situations:

A fast PCD analysis. The pruning allows to control the number of points and the computation time.

Initialization. For the same reasons, some algorithms can be initialized with a pruned PCD, and then enhanced with the original one.

Computations at large scales. At large scales, the neighborhoods contain numerous points and are redundant with the neighborhoods of nearby points. In fact, all the points are not useful anymore, that is why a pruning is appropriate. However, we may want to continue to store the results for all points of the original PCD. For this purpose, it is possible to work with two PCD: the original one and the pruned. The neighborhoods are built around each original point, but the spatial requests are performed in the pruned PCD. Hence, a "pruned" neighborhood is built for each original point.

2.5.4 Precision vs Robustness

Selecting an optimal radius is also a trade off between precision and robustness.
This section does not offer some techniques to bound the radius search space. We only highlight the fact that there is an underlying trade off between precision and robustness. Enhancing one of these criteria is often at the expense of the other. This is a reason explaining the difficulty to retrieve an optimal radius: two incompatible criteria are involved. However, the scale bounds may ensure minimum scores of precision and robustness.

Precision is the smallest significant scale. It increases if the calculations are made locally, and therefore, if the radius decreases. The maximum of precision is reached just above the scale below which the model no longer describes the data (the noise level for instance).

Robustness is the confidence that can be placed in the signal analysis. It therefore increases with the number of points. the maximum of robustness is reached when all the points of the same shape are included into the neighborhood, and before including points from other shapes.

These observations allow to redefine the lower bound as the maximum expected precision, and the upper bound as

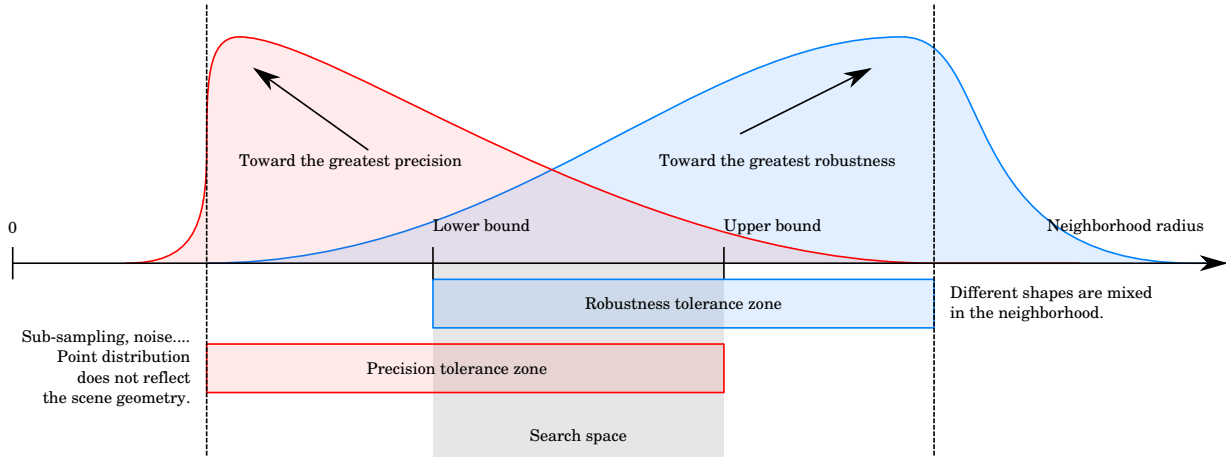


Figure 2.20: Trade off between precision and robustness. The search space of the optimal radius can be defined as the intersection between a "precision tolerance zone" and a "robustness tolerance zone".

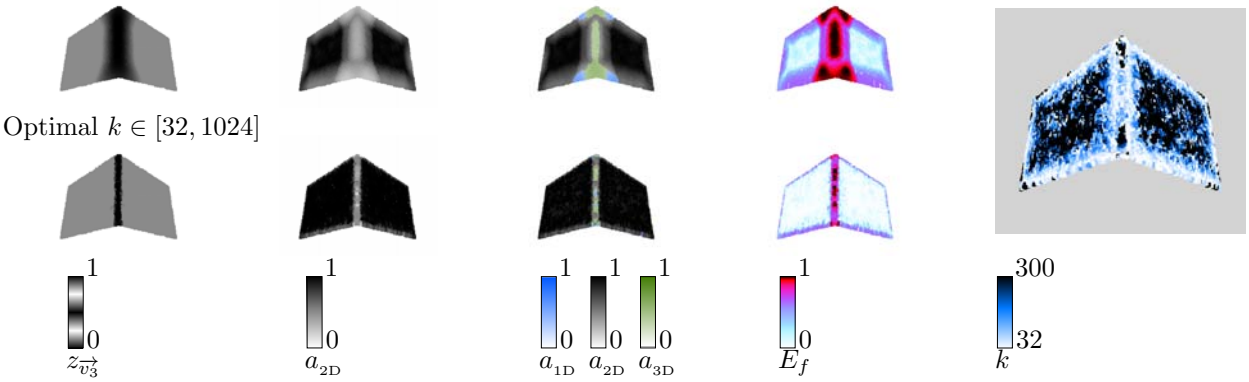


Figure 2.21: Comparison between constant and optimal neighborhood size on a synthetic dataset. The results are sharper with the optimal neighborhood: Most of the points are labeled in 2D, except on the ridge and the edges.

The lower bound is a trade off between the maximum precision, i.e. the smallest r, and the minimum robustness, i.e. the smallest k.

The upper bound is a trade off between the minimum precision, i.e. the greatest r, and the maximum robustness, i.e. the greatest k.

We thereby obtain four bounds for the search space of r^*, as sketched in section 2.20.

2.6 Results

2.6.1 Scale selection

Synthetic dataset

In figure 2.21, a comparison is drawn between constant and optimal neighborhood sizes on a synthetic dataset. The dataset contains 10000 points randomly scattered in a rectangle bent at right angle. the attributes are computed for a constant number of neighbors $k = 1024$ (top row) and for an optimal size $32 \leq k \leq 1024$ that minimizes E_f (bottom row). The plane detection is more sharper with the optimal neighborhood as we see with the normal feature $z_{\overrightarrow{v_3}}$ and a_{2D}. The optimal value k (gray background) is noisy, despite the fact that the data is synthetic and the surfaces perfectly plane. Actually, these plane surfaces lead to the noisy result: a_{2D} is great whatever the setting of k. E_f is then low for all k, and the

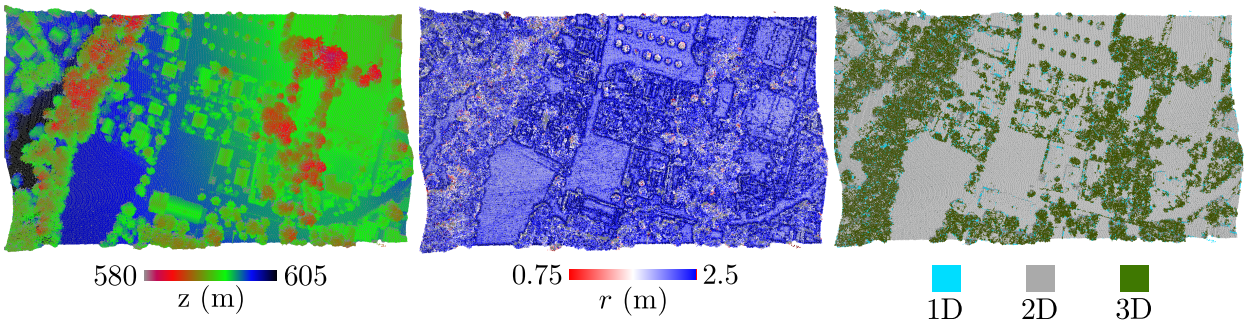


Figure 2.22: Radius selection (r^*) and dimensionality labeling for ALS_G dataset.

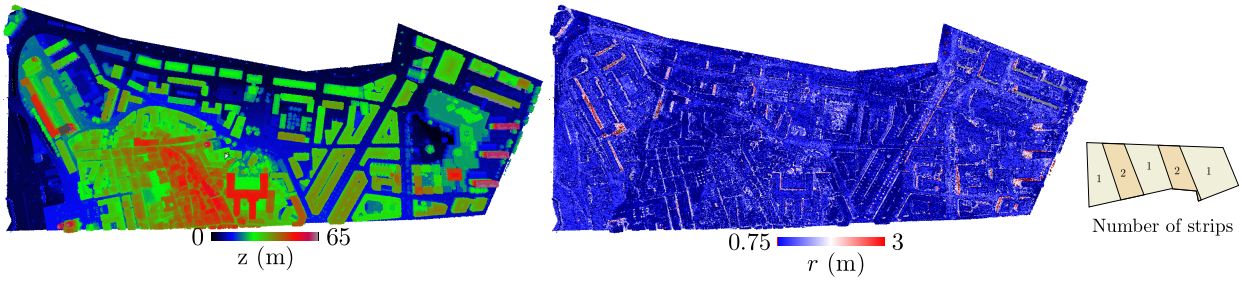


Figure 2.23: Radius selection (r^*) and dimensionality labeling for ALS_F dataset.

optimal size selection is more random. However, when the points are close to the ridge and the edges, the optimal size is most often small. The result is then sharper than with the constant size, this also ensures that the normals stay orthogonal to the plane.

Real datasets

The six datasets have been processed with the proposed approach. Scale selection results are displayed in Figures 2.11, 2.22, 2.23 and 2.24, using the E_f criterion. The observed behaviors are those theoretically expected, which confirms the effectiveness of our multi-scale analysis using E_f:
Planar areas exhibit high radius values for ALS and TLS datasets (see Figures 2.11, 2.22 and 2.23). This is particularly true for the TLS dataset since the multi-scale analysis allows to overtake the noise level.
Lowest scales correspond to border areas, allowing to retrieve building and vegetation edges for ALS datasets, windows in MMS datasets (Figures 2.24), or small architectural elements in TLS.
The algorithm allows to select high values when no predominant dimensionality is found for small and medium radius values. A sufficient number of 3D points is collected to retrieve a coherent dimensionality : 1D for wires and poles (Figures 2.24), 2D for building facades in ALS datasets (*e.g.*, highest values in Figure 2.23 which correspond to a few number of points lying on the facades of high buildings) or 3D for vegetated areas (all datasets).
The method deals well with varying point densities. Such variation may come from the number of overlapping strips for ALS datasets (a lower radius size is necessary when two overlapping strips exist for an area of interest, see Figure 2.23), from occlusions or from the fluctuating incidence angle of the laser beam on the surfaces for TLS or MMS datasets (Figure 2.24).

Object mixing

In figure 2.21, It is observed that the 3D label the ridge is the result of mixing points from two planes. The object mixing is also observed in figure 2.27 even if the optimal radius selection is performed. If many lines belong to the same plane as in in figure 2.27(1), the plane is detected instead of the lines. At the between the crane trunk and the crane arm, the mixing of these linear structures is labeled 3D. This labeling is not

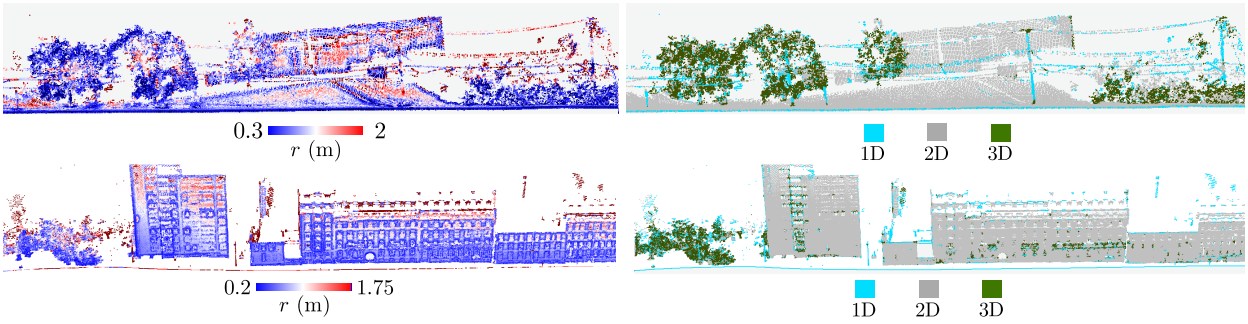


Figure 2.24: Radius selection (r^*) and dimensionality labeling for MMS_U dataset.

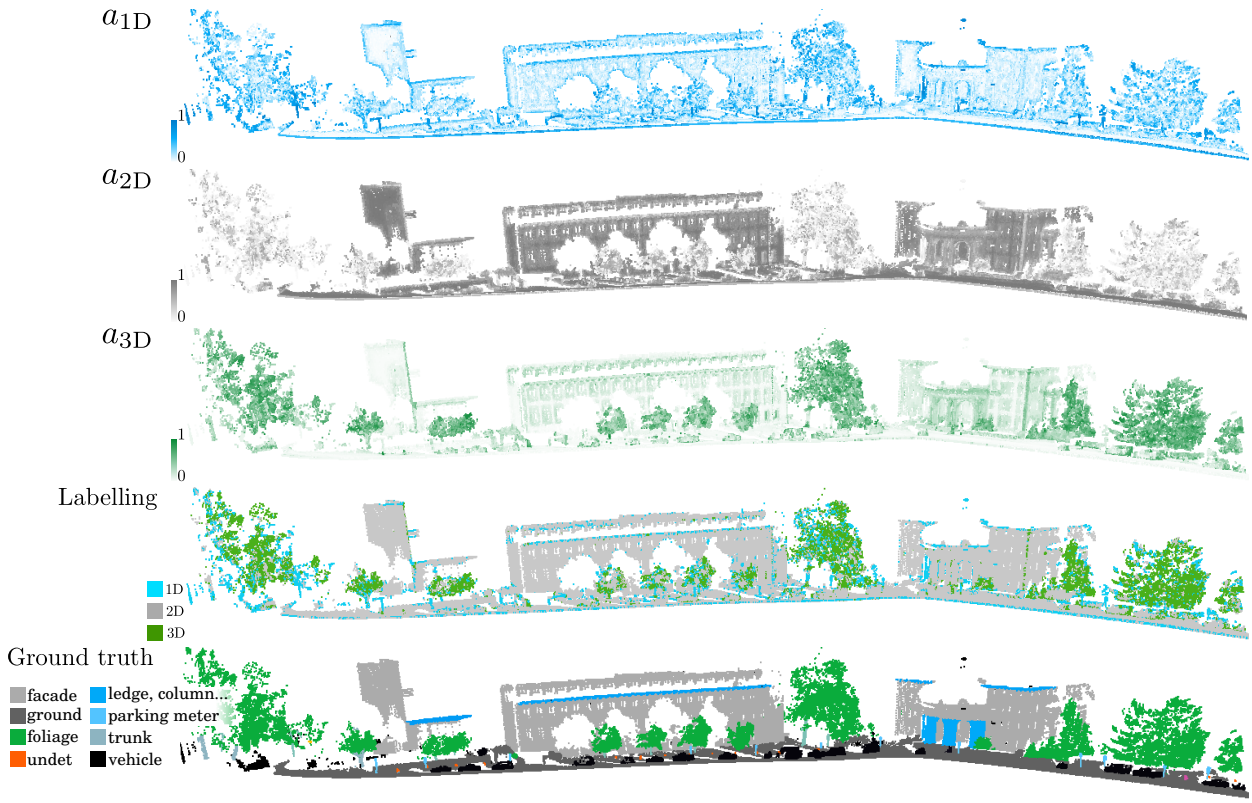


Figure 2.25: Dimensionality features and labellings compared with a manual labeling of the Oakland dataset. One can check visually the consistency between our automatic labeling and many semantic objects such as the ledges and columns (1D), the facades (2D) and the foliage (3D).

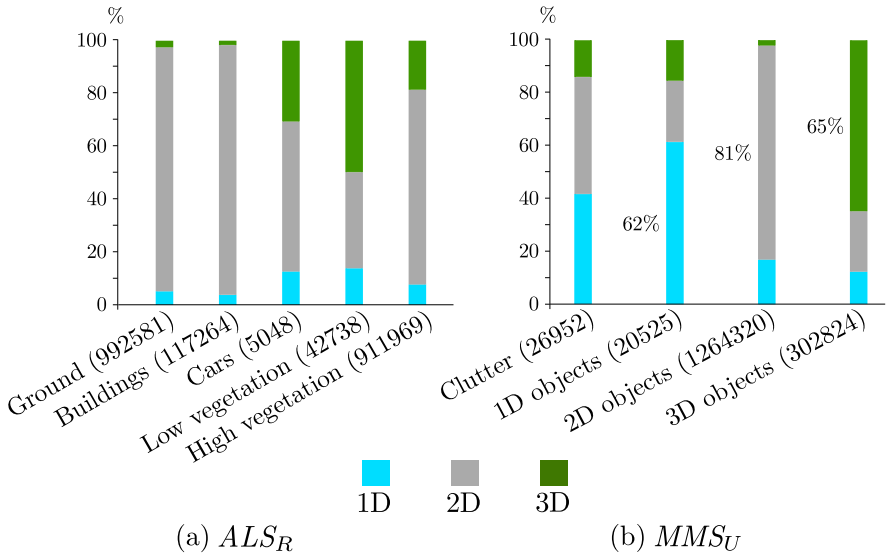


Figure 2.26: Dimensionality labeling statistics for two datasets available with ground truth.
(a) ALS_R dataset: the five main classes have been conserved.
(b) MMS_U dataset: the 59 classes have been condensed into four classes according to their shape (1D: wires, poles, trunks etc. – 2D: wall, door, facade etc. – 3D: foliage, grass etc. – Clutter: objects without specific shapes). The number of points for each class is indicated inside brackets.

wrong, as it corresponds to the actual behavior of the points in these areas, but this is not what we expected. Two situations can be identified.

Multi-scale: the geometry varies from scale to scale, as the lines that belong to a plane. This case has also been encountered with scan lines and high-voltage lines. One solution would be to store, not one, but many optimal radii and the associated dimensionalities. Indeed, there is only one E_f minimum, but several local E_f minima. A multi-scale analysis could be considered, it is interesting to keep in mind that the shapes at large scales are the fusion of shapes at smaller scales.

Shape edges and intersections: In these limit cases, we would like to extend the shape properties to the edges or across a junction with another shape. To extend the properties of a shape imply to identify this shape as more relevant than the others. One way would be to select certain relevant shapes and to extend their properties to the points belonging to this shape. This is possible with region growing methods. One other way would be a point selection in the neighborhood (RANSAC) in order to avoid the object mixing. However, the scale selection is already a kind of point selection, because many point sets are tested (the neighborhoods of various radii).

To our opinion, the proposed method provides a geometrical description that is not the best one in many cases (plane estimation, edges...). If the application indicates the searched primitives, such as planes, the method can be tuned, or another method can be performed in order to extract the plane. However, if we just want to analyze the geometry, nothing indicates if a plane is more relevant at the roof ridge than the proposed 3D geometric description. A more complex or oriented geometrical analysis may be biased. That is why we think that an advantage of this method is to be simple and easy to understand. This is thereby a good tool for many application.

The effectiveness of the scale selection process can be assessed by observing the predominant dimensionality that has been retrieved (1D, 2D or 3D ?) for various objects of interest. Comparisons with manually labeled ALS and MMS data have been performed (cf. Figure 2.25 and 2.26). For both datasets, planar objects are correctly retrieved, with few errors corresponding to building edges or small urban items, especially for ALS data. Besides, objects with one privileged dimension such as poles, wires or trunks are mainly labeled as 1D (MMS_U dataset, in Figure 2.26b). However, since these objects are in fact cylindrical with a small width (*e.g.*, trunks or traffic lights), they may look planar or volumetric for medium-sized neighborhoods.

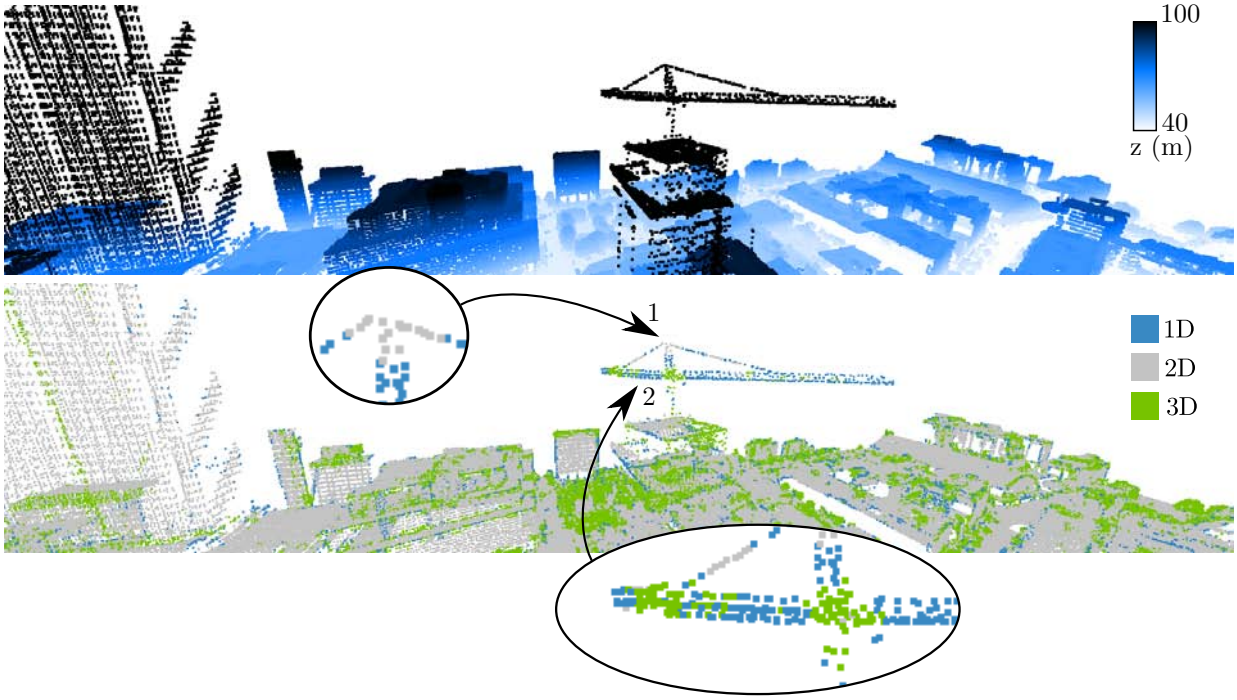


Figure 2.27: Radius selection (r^*) and dimensionality labeling for ALS_C. Some unexpected behavior on the crane :
(1) The cables are labeled 2D instead of 1D.
(2) The junction between the trunk and the arm is labeled 3D instead of 1D. The mixing of several objects leads to a label of greater dimension.

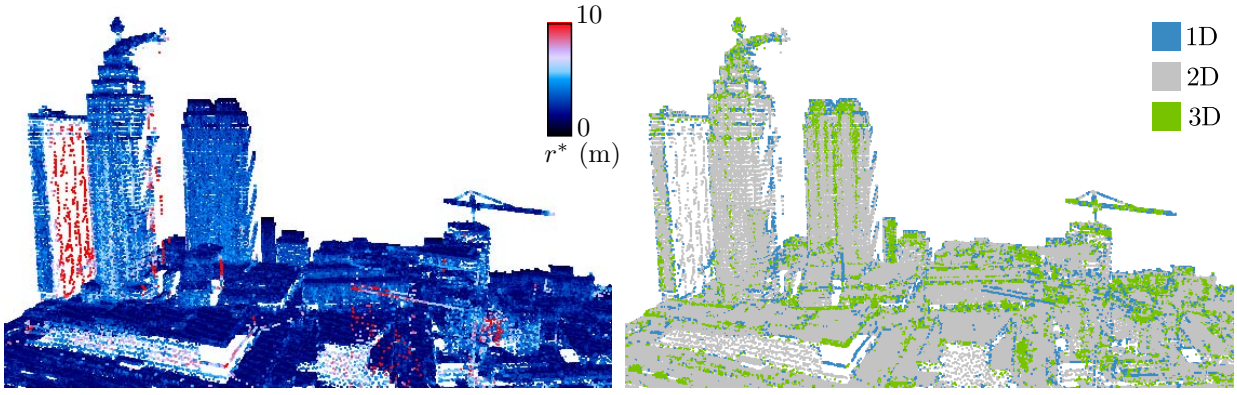


Figure 2.28: Radius selection (r^*) and dimensionality labeling for ALS_C. The facades are well detected as flat areas even if the point density is very low. The optimal radius is then great (red).

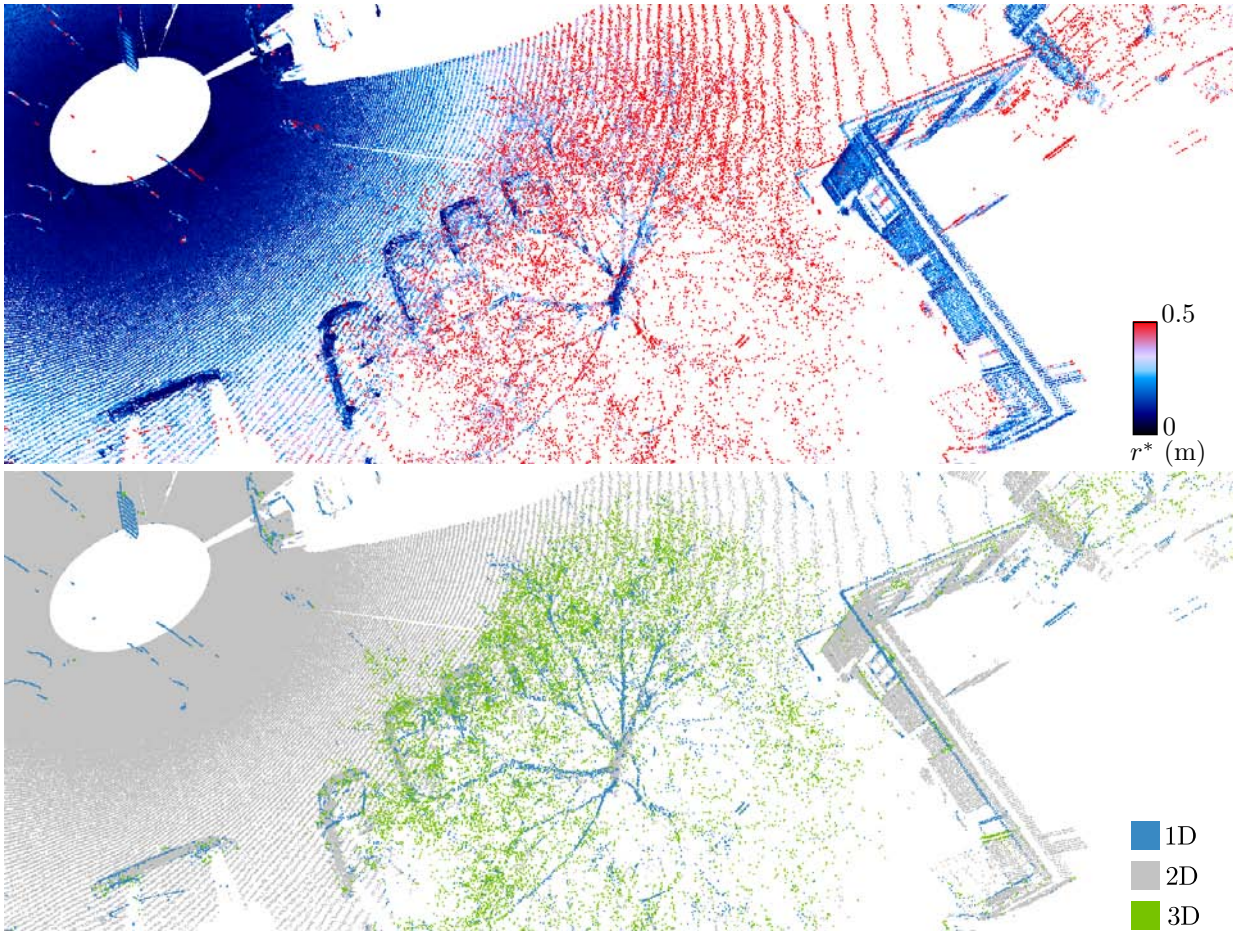


Figure 2.29: Radius selection (r^*) and dimensionality labeling for TLS_F. The optimal radius increases from the sensor location (empty circle at top-left) toward the remote areas at right, and the ground is still labeled 2D despite the scanning pattern arranged in lines. The tree labeling separates the trunk (2D), the branches (1D) and the foliage (3D). The optimal radius also behaves differently : small on the trunk, less small on the branches and much greater in the foliage.

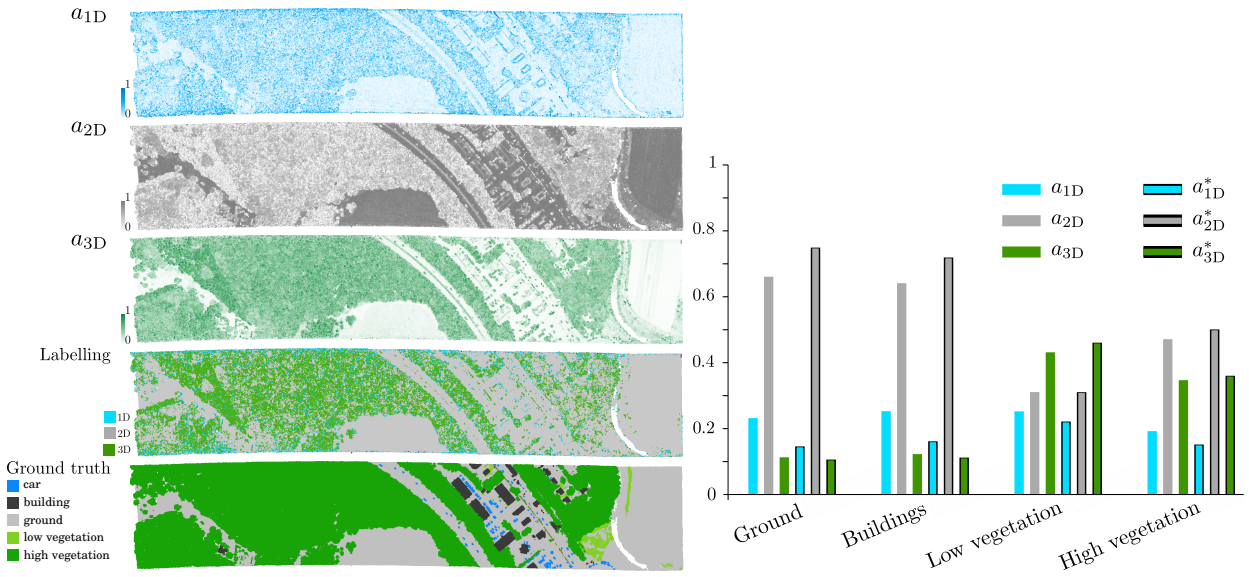


Figure 2.30: Improvements in shape feature computation using an optimal neighborhood radius per point. a_{dD} and a^*_{dD} are the shape feature for dimensionality d, computed with neighborhoods of constant size and adaptive size, respectively. The constant neighborhood size corresponds to the 30 nearest neighbors of each point.

Conversely, volumetric objects such as trees may locally look planar. This happens for large objects low point densities or when no multiple scatterings are found with vegetated areas. Such phenomena is limited for genuine 3D acquisitions (TLS or MMS datasets), whereas it is clearly enhanced for 2.5D data such as for the ALS_R data (Figure 2.26a). High vegetation areas are principally labeled as *planar*, which is due to the dense canopy cover. Nevertheless, as displayed in Figure 2.22, such effect almost disappears when 3D points are acquired within tree canopies.

2.6.2 Comparisons with constant neighborhood size

The adaptive size strategy allows to retrieve a correct number of points to estimate the local dimensionality of the point set. Such strategy may be compared with the results achieved with neighborhoods of fixed size for a whole dataset.
Firstly, the three shape features are computed with both strategies for four classes of interest of the ALS_R dataset (Figure 2.30). One can see that the planar behavior of *ground* and *building* points is improved with the adaptive size. Besides, for vegetated areas, the volumetric behavior is slightly enhanced, however this happens in conjunction with an increase of the planar behavior (dense canopies). Secondly, in addition to improved shape features, Figure 2.31 shows that the privileged dimensionality is also better retrieved with the adaptive strategy (MMS_F dataset). This is particularly true for trees (1D→3D), small building facades elements (1D→2D), and tree trunks (2D→1D). Furthermore, since the geometrical analysis is less affected by the 3D point distribution, the labeling procedure is far less noisy.

2.7 Dimensionality features as a toolbox.

In this section, some possible usages of the dimensionality features are detailed.

2.7.1 Derivation of horizontality and verticality scores

The local analysis that have been developed allows to derive simple descriptors. We propose here an *Horizontality* and a *Verticality* score. we assume that an horizontal area is a flat and horizontal surface. As well, a vertical area is assumed to be a flat and vertical surface. For each point p_i, a normal vector is

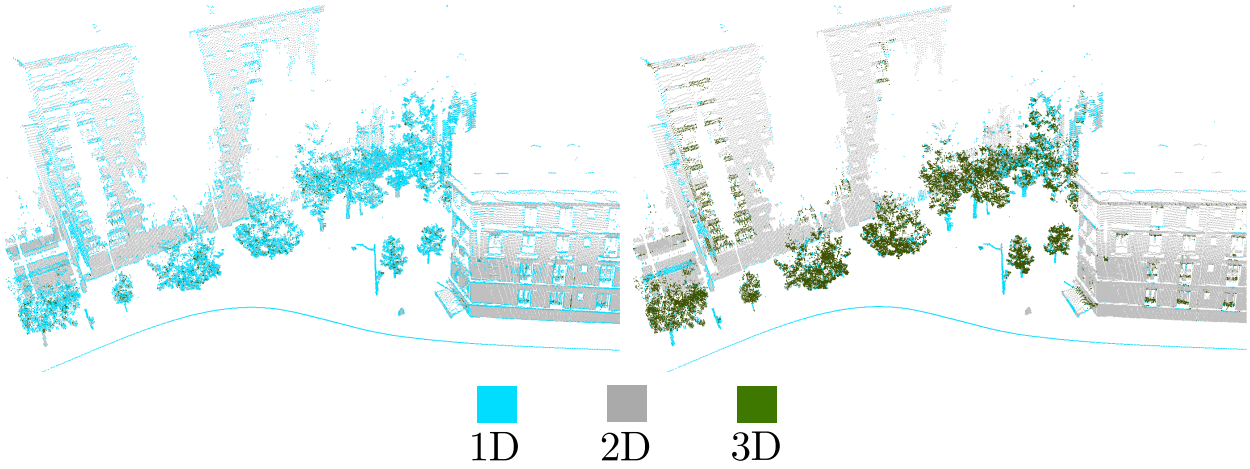
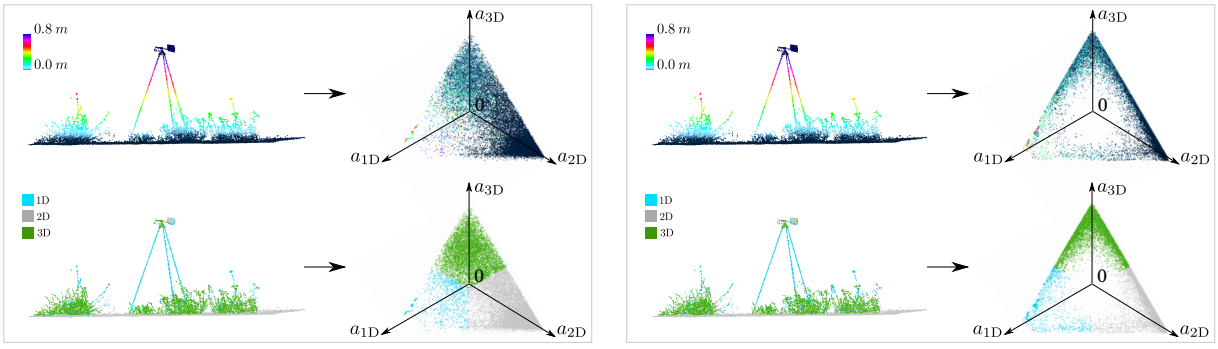


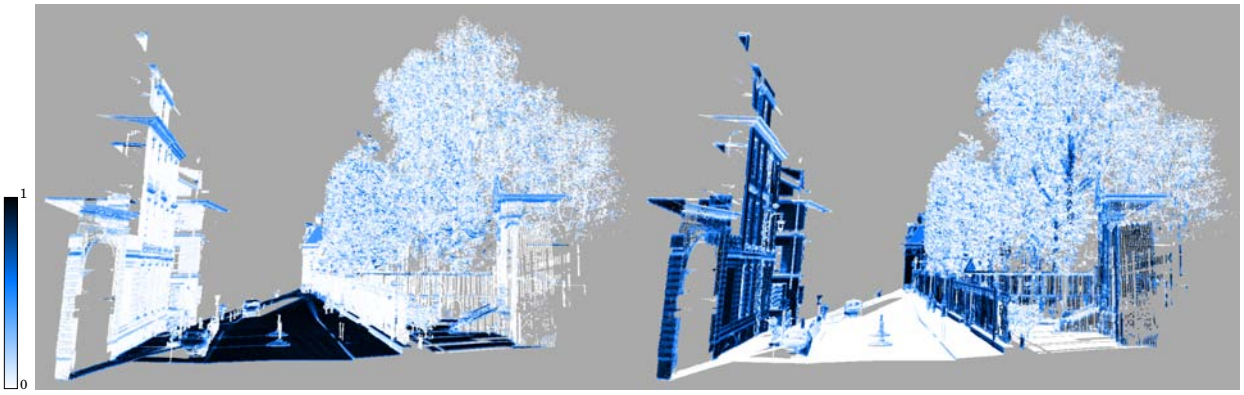
Figure 2.31: Dimensionality labeling for the MMS_F dataset using a constant neighborhood size for the whole point cloud (**left**), and the optimal value per point selected with E_f (**right**). The constant neighborhood size corresponds to a 1 m radius.



(a) Constant radius

(b) Optimal radius

Figure 2.32: Constant vs optimal radius. In dimensionality space (triangles), points are evenly distributed in a neighborhood of constant size (a) when they are pushed toward the corners by the scale selection method (b): minimize E_f means avoiding the uncertainty area in the middle of the triangle (E_f max).



(a) *Horizontality* (b) *Verticality*

Figure 2.33: *Horizontality*, and *Verticality* scores computed for a MMS dataset.

provided by $\overrightarrow{v_3}$ and the and the flatness is measured by a_{2D}. The scores are thereby defined:

$$Horizontality = a_{2D}|z_{\overrightarrow{v_3}}| \qquad Verticality = a_{2D}(1 - |z_{\overrightarrow{v_3}}|)$$

In figure 2.33, they are computed on a MMS dataset. All flat and horizontal surfaces: road, sidewalks, but also car roofs and hoods obtain a high *Horizontality* score. As well, all flat and vertical surfaces: facade walls, tree trunks, poles or streetlights obtain a high *Verticality* score.
Such descriptors may also be useful for ground or facade extraction. Even if many other objects are detected, they provide a simple and powerful way to filter points. The relevance of the *Verticality* feature has been assessed for the semantic interpretation of 3D point cloud data in [36]. A classifier-independent ranking procedure made it second among 21 features.

2.7.2 Using dimensionalities in various applications

The dimensionality features and labellings have been tested and used in the MATIS laboratory for different applications.

Tree extraction in MMS data

The data is segmented according to the labels: poles:1D, facade:2D, foliage:3D and an attribute of cylindricity allows to detect the tree trunks. The trees are then detected and individualized [37].

Registration of lidar PCD

The Iterative Closest Point (ICP) algorithm is improved by using the features calculated from the structure tensors, according to our radius-selection method. Two features are well-suited for registration : E_f and the omnivariance (product of σ_1, σ_2 and σ_3) both highlight salient shapes as the building corners and edges [38].

Segmentation

In annex, a *mean shift* segmentation is proposed.

2.8 Conclusion

The objective of this chapter was to provide a simple, general and automatic method to provide a description of the geometry around each lidar echo. For this purpose, some features are derived from a classical "tensor voting" approach. The features are computed in a spherical neighborhood around each echo. They describe the dimensionality (1D, 2D or 3D) of the distribution of the echoes included in such a

neighborhood. The most appropriate neighborhood size is automatically selected among a range of potential sizes according to the entropy feature E_f.
Results on distinct lidar point clouds acquired with airborne, terrestrial and mobile mapping systems under various conditions showed the effectiveness of the proposed method. No a priori knowledge or assumption on the point cloud distribution, density, laser scan pattern, object size or underlying structures was required to achieve such results. Several experiments demonstrated how favorably our method performed compared to constant neighborhood sizes, and how relevant such approach is. Finally, the prevailing dimensionality and the scale of interest for the underlying object are found for each point, providing interesting cues for many potential subsequent segmentation or classification algorithms.
To conclude, we would like to stress a characteristic of the dimensionality descriptors: they are understandable and meaningful. To our opinion, this is a strong advantage from the user perspective: their behaviors are predictable, allowing to derive tools for specific applications.

In the remainder of this thesis...

In chapter 3 we investigate the ways to optimize the computation of these descriptors for large amounts of data. For MMS data, we explain our choice to perform a streamed computation. In chapter 4 we detect the facades according to the *Verticality* score.

Chapter 3

Scaling up

Contents

Automatic algorithms performed on a great amount of data require an adapted strategy to browse into the data and to isolate consistent subsets in order to perform algorithms locally. One can exploit the raw data organization, but the data can also be restructured.
In the following, such choices are presented and discussed. Especially the different ways the data can be organized or browsed:

- Semantic,
- Spatial/Dimensional: temporal (1D), Three-dimensional space (3D), or Space-time (4D).

The semantic partitioning corresponds to an ideal segmentation of the dataset, in accordance with the scanned objects. Such approach implies a knowledge about the content of the data. But the dataset can also be structured or browsed without any a priori knowledge, according to one or several dimensions (time, x, y, z...). The dimensions have to be chosen when building a space partitioning data structures (octree, kd-tree...) that speeds up the access to close points.
We finally explain why a temporal partitioning is preferred for facade detection according to our dataset.

3.1 Data structure: how to split the data?

3.1.1 Semantic partitioning

Processing a voluminous PCD rarely requires to handle the whole dataset in the same time. In many applications we would like to handle each PCD subset associated with a single scanned object independently. For this purpose we need a partitioning that perfectly matches with the scanned objects, i.e. all echoes of a single object are clustered in a single partition. This way, each object/partition could be handled independently, and the topological relationships between objects could be explored through a

graph linking the partitions. Note that this object partitioning is not unique. Regarding our facade building example, according to the chosen scale, the objects can be:

- the building bricks,
- the floors and the roofs,
- the entire buildings
- or the building blocks.

Leading to as much partitionings as possible scale levels. A multiscale partitioning is feasible, combining all the "scale-level" partitionings. It consists in a tree structure where each node is an object, the child nodes are the included sub-objects, and the leaves are the 3D points. This tree structure allows to efficiently retrieve the corresponding 3D points of every node-object. We believe that such a semantic tree would be very useful, facilitating many applications in urban scene analysis. This "Holy Grail" could be achieved with the combined efforts of several algorithms that detect each object type independently, but not only: even if the objects are often spatially isolated from the others, many cases remain ambiguous. The objects are often mixed, hidden or sub-sampled, in addition there are always some objects that do not fit any category. Rules and constraints must be used to reduce the choices and to achieve a good semantic tree. In conclusion, we cannot expect to obtain a semantic tree thanks to a low-level approach. The plane primitive detection that we propose in the next chapter cannot rely on such a partitioning. Conversely, the detected primitives could contribute to construct the semantic tree.

3.1.2 Spatial data structures

Splitting and Sorting

Partitioning the data is helpful in two optimization ways:

Memory: handling the whole dataset is rarely necessary and it is memory consuming. Processing blocks is more efficient, and often equivalent.

Time: the spatial requests according to one or several dimensions, are faster if the data is structured in accordance with these dimensions: the spatial data structures (SDS) such as octree or kd-tree allows to speed up the requests in O(log n). A "semantic tree" could also fulfill this role.

The role of the data structure is to connect elements that are geographically close. This is what allows faster access to neighbors of an element instead of searching throughout the entire data. The memory constraint imposes to split the data while the time constraint imposes to connect the different data components together. These requirements are almost opposite. The idea behind the tree structure is to separate the components that are distant (spatial tree) or that belong to different objects (semantic tree). Sometimes, the tree is not built on the whole dataset, a partitioning is then performed at two scale levels:

1. Data is split into large pieces : semantic "City blocks" [39] or horizontal grid squares for aerial data [40].
2. A dedicated spatial data structure (octree, kd-tree,...) is then constructed for each piece.

It should be noted that there are problems on the edge when cutting blocks, while kd/octree only accelerate queries without affecting the calculation. Indeed, the objects to the edges of a block may be cut. How to maintain the integrity of these objects? It is necessary to leave a margin, and therefore work on overlapping blocks.

Dimensional partitioning

The spatial trees are often three-dimensional (xyz): the 3D space is split into voxels (octree) or into boxes that contain the same number of points (k-d tree).

3.1.3 Echoes are time-ordered in the raw data

Besides their spatial organization in the 3D space, lidar echoes acquired by the Stereopolis have also a time coordinate. By the way, they are stored in the order they was acquired: the data is natively sorted in chronological order. Linearly browsing through the raw data has therefore a physical sense! With a temporal partitioning, two scans of the same object acquired at different times are not processed together. It could be seen as a drawback but we see that as an advantage because urban areas are dynamic environments, and many things might have changed between two scans of the same place: cars and pedestrians might have moved, windows and doors might have been opened or closed... All this advocates to consider two scans of the same place as different objects, which simply lie at the same geographic location. Obviously, these objects should be associated (but not assimilated) for registration or change detection purposes, but not for primitive extraction or object detection. Finally, lidar echoes are space-time points. We did not investigate a 4D partitioning, the proposed algorithm is performed on temporal buffers. Some other arguments explained below encouraged us to follow this direction.

3.2 Along the trajectory

The vehicle trajectory constitutes the backbone of an acquisition. Indeed, all the acquired data and especially the location measures in the sensor reference frames are tied to it, in the sense that it provides the sensor geolocations.

3.2.1 Trajectory: definition

We define the trajectory as the successive sensor locations during the acquisition. The trajectory is a 4D element that maps each acquisition time τ with a sensor location. The trajectory is thus equivalent to the function S_τ defined in chapter 1, section 1.3.

3.2.2 Vehicle geolocation

Obtaining the vehicle geolocation is challenging: MMS are usually equipped with a Global Navigation Satellite System (GNSS). In satellite denied environments (urban canyon, tunnel...) the GNSS is relayed by an inertial navigation system (INS) in order to ensure a continuous geolocation. However, the INS may drift which leads to errors of a few tenths of a centimeter. This is a relatively great error ($\simeq 1$ m) beside the positioning error echo-sensor ($\simeq 0.01$ m). Moreover these errors in the trajectory estimation affect differently each echo. As can be suggested by an analogy with a spine, each laser beam is rigidly fixed to the sensor position, but the sensor positions are uncertain and may move, causing the displacement of the associated beams and echoes (fig 3.1). In other words, the error in trajectory geolocation is propagated in the echo geolocations. This error varies along the trajectory, and it can also affect differently each echo. For example, an angular variation causes displacements that are all the more important, if the echoes are remote. As well, if there are actually georeferencing errors, the distance between the echoes is not reliable and it is difficult to correct these biases without knowing the estimated trajectory. In mobile laser scans, the relative location between echoes thus depends on the georeferencing process.

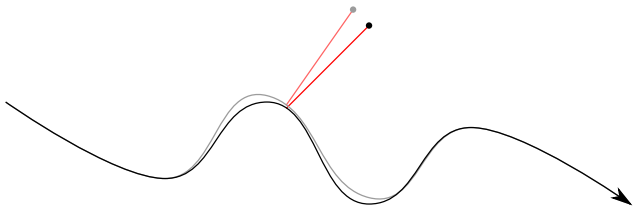


Figure 3.1: Georeferencing error propagation. The estimated trajectory may be deformed, involving the displacements of the laser beams rigidly fixed to it. A small error in trajectory estimation (black and gray lines) can imply a greater error in the echo location.

3.2.3 Relative position errors between echoes

The mobile laser scans are "non-rigid" because they may be distorted by the trajectory estimation. If the georeferencing of the vehicle is inaccurate, leading to a wrong trajectory estimation, the relative position between the echoes is not reliable. However, a local consistency is maintained in the PCD, and this is actually a temporal locality: the relative position error between two echoes acquired in a short time interval is ensured to be low and the drift of the navigation system is progressive. The data is consistent over a time interval, but it can be shifted gradually during the acquisition. Such a shift can be highlighted when the vehicle passes several times at the same place and comparing these successive acquisitions as in figure 3.2. It is clear that the echoes detected the same objects, but the georeferencing causes a shift. It is not possible to handle the whole dataset because the inconsistent echoes from the different epochs are mixed. However, such redundancy in the data can be used to self-register the PCD.

3.2.4 Self-registration

We will not discuss the methods of self-registration and other techniques to improve or correct geolocation but you can refer to the relating literature [41] [42]. The self-registration allows to register two parts (or more) of the same acquisition that correspond to the same place. As the geolocation error is supposed to be smooth along the trajectory, the deformation that register the two parts can be propagated along the trajectory. The registration is strongly related to the trajectory estimation. In fact, the output is nothing but a good trajectory estimation: assuming that the positioning sensor-echo is ok, the trajectory contains the sensor position at each instant and allows to replace the echoes in the absolute reference frame.
If no self-registration is performed, it is not possible to handle the whole dataset but it remains possible to handle temporal sections of the PCD.

3.3 Computing the local geometrical features along the trajectory

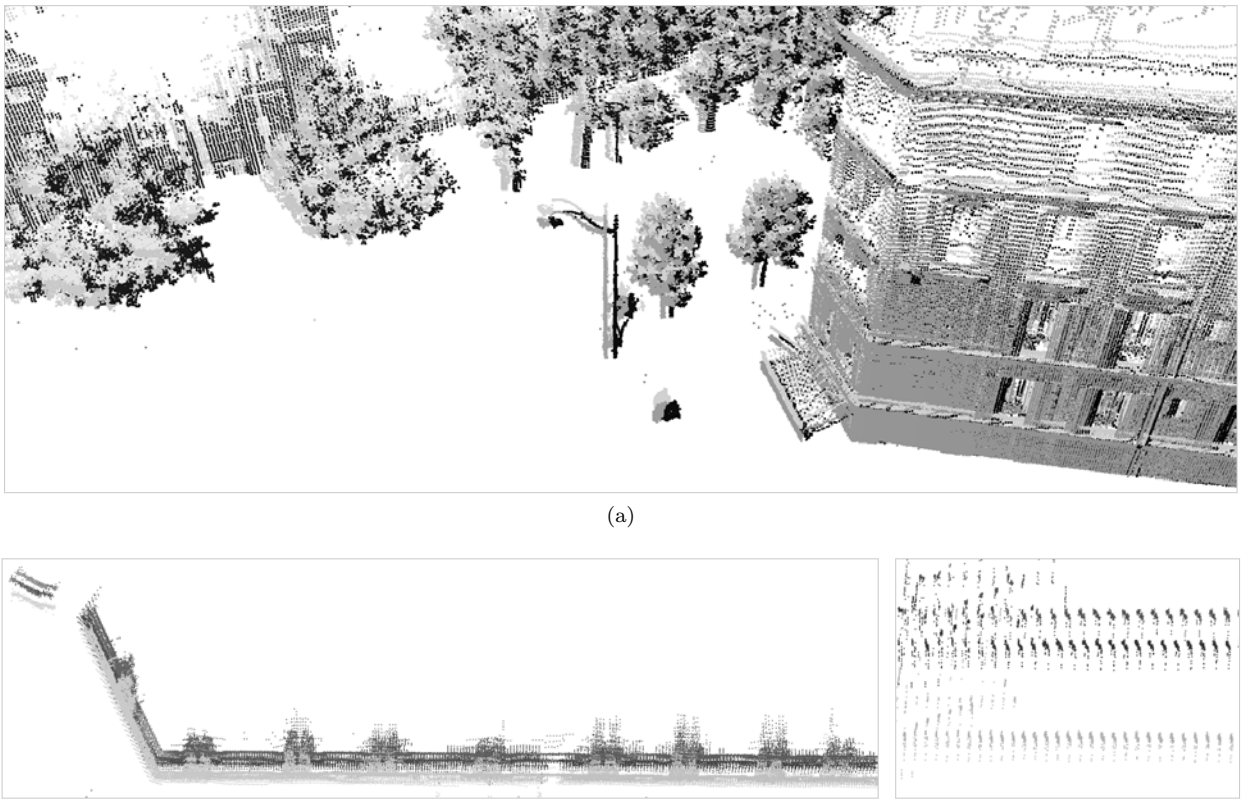
As shown in figure 3.3, the computation of local features are not optimal when performed on the whole dataset. using a temporal buffer gives better results and is also faster.

3.4 Conclusion

For MMS data, the local geometrical features computation can be streamed, and it is even preferable if the data is not self-registered.
Scaling up imposes some constraints in the algorithm design. In particular, it is necessary to cut the point clouds to apply processings on smaller blocks. Different choices are possible (Semantic, Spatial/Dimensional : temporal (1D) , Three-dimensional space (3D), or Space-time (4D)). In our context, the most sensible approach seems to be to keep the original structure of the point cloud, Indeed , the points are stored in the order of acquisition and are naturally organized according to the temporal dimension. This is the most direct method, but it has also other benefits. A time interval corresponding to a segment of the vehicle path during acquisition. The sensor location is thus bounded in space-time, and points acquired during this interval are always close temporally and often close spatially. Processing temporal buffers is a method

- fast
- adapted to the changing environment : no mixing of points from different epochs.
- suited to the vehicle georeferencing : georeferencing drift is gradual and therefore varies little over a short time interval.

This is the choice made for the facade detection as vertical rectangles in chapter 4. This overcomes the problems of data volume and georeferencing.



(a)

(b) (c)

Figure 3.2: Shift between echoes acquired at three different times (three shades of gray).
(a) During the same acquisition this street side have been scanned three times (three shades of gray). The data is shifted between the three epochs.
(b) and (c) Horizontal projection of the points in the xy-plane. The points lie on a facade. One can see the facade footprint repeated three times. The more scatter areas correspond to the window columns. At right, the zoom highlights the echo distribution (c). The spotted distribution is caused by the vertical lidar sweep. that concentrate the echoes along vertical scan lines on the facade walls.

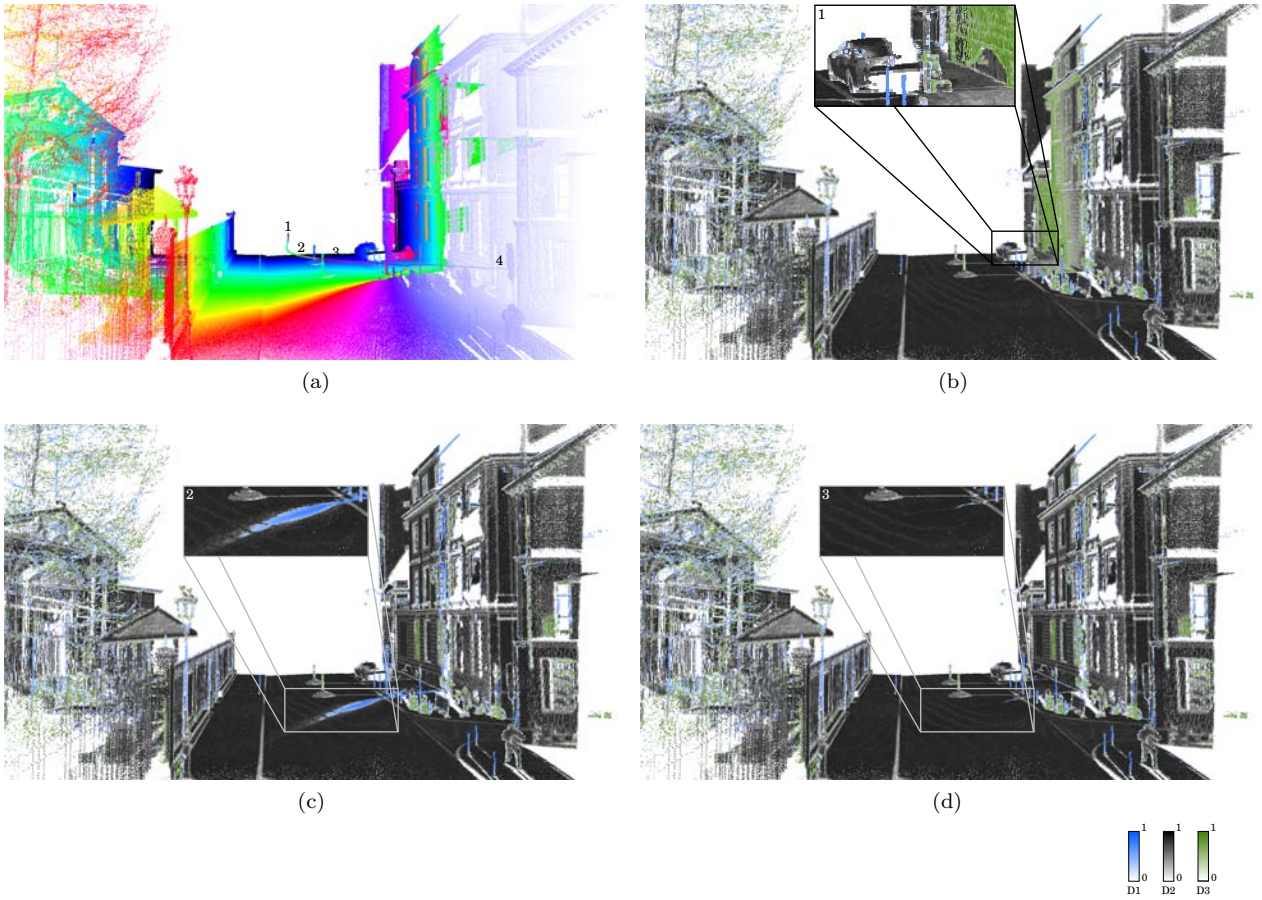


Figure 3.3: Dimensionalities computed on the dataset "1-2-3-4" (Presented in chapter 1, figure 1.5).
(a) GPS Time.
(b) Dimensionalities computed on the whole dataset. A facade is 3D (green) except for the area that corresponds to the car shadow (zoom 1) where there is only one scan pass, and the dimensionality is more 2D (gray). The 3D labeling of the rest of the facade thus corresponds to a registration shift.
(c) Dimensionalities computed with a streamed buffer of 10 000 points. This buffer size is too low: there is an artifact on the road (zoom 2). The labeling is 1D not because of the road shape but because of the buffer shape.
(d) Dimensionalities computed with a streamed buffer of 100 000 points. This buffer size seems suitable: There is neither registration problems, nor artifacts (zoom 3).

Chapter 4

Streamed Vertical Rectangle Detection

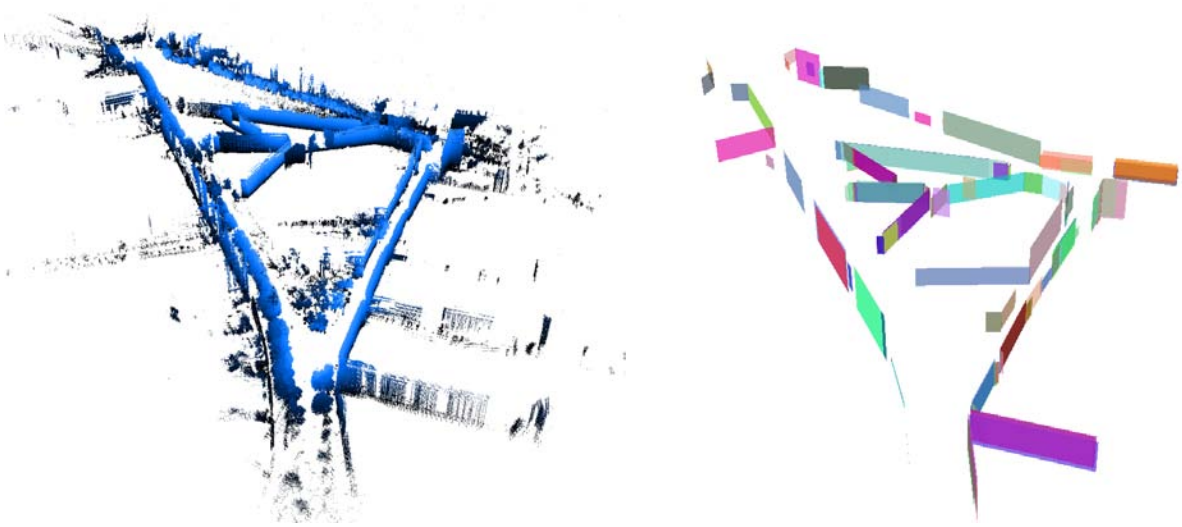
Contents

This chapter is an extension of the work presented in [43].
The high level of detail of data collected by mobile lidar mapping systems allows for fine geometrical modeling of urban environment. This high level of detail makes the amounts of data required to model an entire city huge. Consequently, efficient methods are needed to detect the main scanned structures in order to split the modeling of a whole city into smaller parts (blocks, building, facades,...) Whereas numerous works focus on the fine reconstruction of certain types of urban objects (facades, trees, signalization,...) the detection of such objects in large amounts of data remains quite unexplored, making these methods hardly scalable. In this paper, we aim to automate a necessary step in fine facade modeling over large areas: detecting the main facade rectangles present in the scene. This seemingly simple task faces the scaling problem and other difficulties that often leads to perform it in a semi-automatic way, by pre-selecting manually areas containing each facade or resorting to a cadastral database [44], a 3D model [45] or aerial lidar data [46]. However, automation of this treatment is necessary to enable the modeling of large-scale urban scenes.

4.1 Introduction

4.1.1 The dataset

The dataset used for this study was acquired by the Stereopolis in a dense urban area by a mobile mapping system and consists of 3 loops over a 300m trajectory. The Lidar is a RIEGL fixed on the roof of the vehicle. Scanning axis is vertical and each sweep records 201 echos. Beam angles with horizontal plane are between 0°and 80°(fig 1.2). The dataset is displayed in figure 4.1(a) and contains approximately 10 million points. As the scan is performed at constant rate, and the mobile mapping vehicle is moving at variable speed (it even stops at traffic lights), the point density along the trajectory is very variable. The simplest way to deal with this issue is to filter out points to ensure a reasonable maximum density.



(a) Whole dataset (b) Detected rectangles

Figure 4.1: Whole view of the dataset used to test the method (a), and the detected rectangles (b).

4.1.2 Plane detection

Primitive detection (rectangle, planes) is a widely studied problem, and many efficient methods have been proposed and improved. Our aim is to find the most appropriate method to solve our problem. Common primitive detection methods such as Hough or RANSAC do not scale well to large datasets as their complexity is more than linear. This problem can be addressed in two (non exclusive) ways:

1. Improve the performance of the detection method.

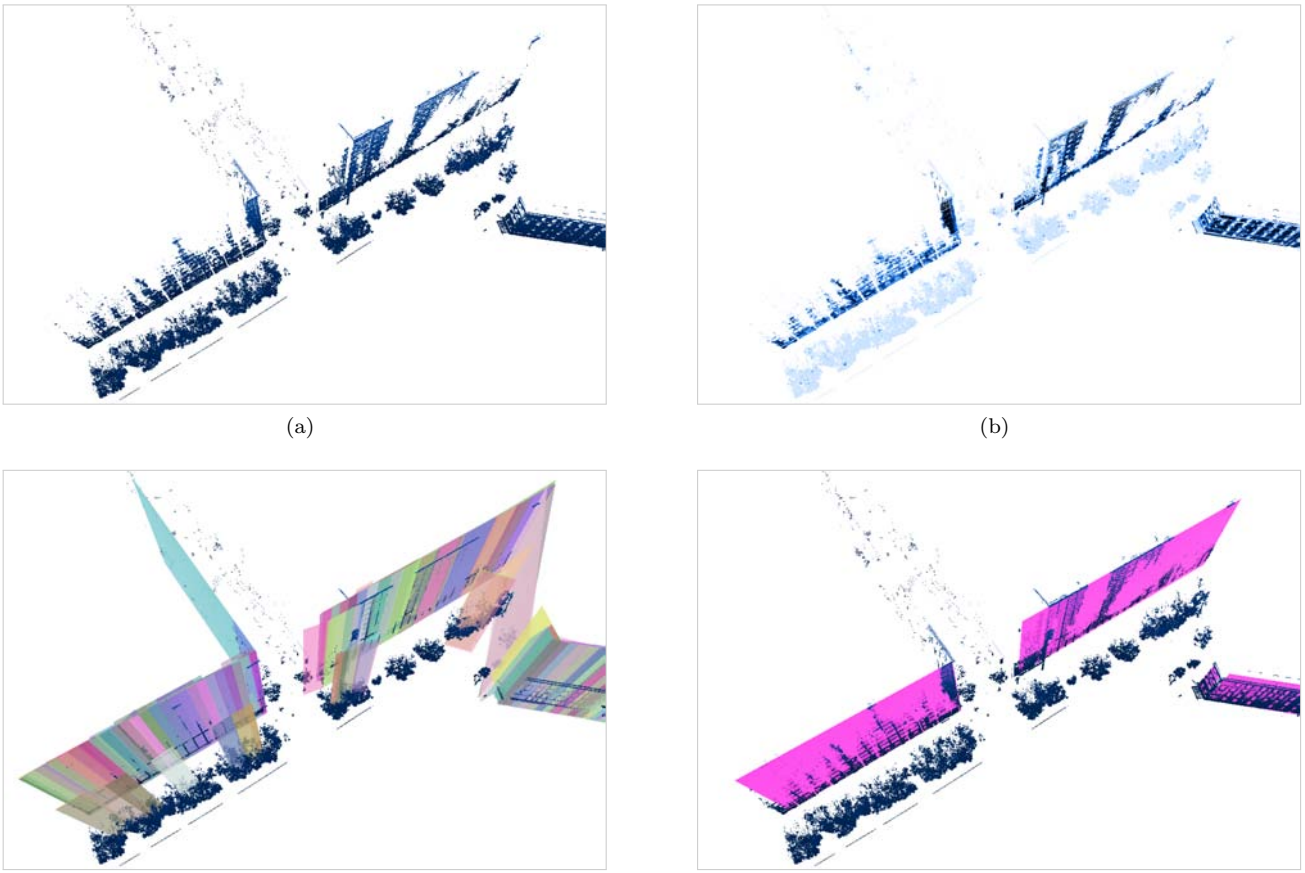
2. Partition the data into smaller blocks.

Several authors have proposed methodologies to improve the performance of facade detection methods. For instance, [44] proposes to accelerate the Hough transform algorithm by thresholding the accumulation in parameter space with an upper bound above which plane hypotheses are directly accepted. Other methods take advantage of repeated structures in facades in order to filter and consolidate point clouds ([47] and [48]). In the proposed method, point filtering is performed in a probabilistic way with RANSAC : we increase the probability that RANSAC selects points belonging to vertical planes (which contain many points). More precisely, the probability to select a point is calculated based on local geometrical features (see section 4.2.1).

The problem of rectangle detection in ALS data has been studied in [49], Flat areas with no holes are extracted in the whole PCD by a RANSAC algorithm. The best rectangle is then fitted thanks to a novel method: the rectangle rotates into the extracted flat area in order to find the best orientation and size. After plane detection, the modeling consistency have to be controlled. Only the planes that belong to the surface of scanned objects have to be kept, and these plane pieces have to intersect well together (surface continuity).

But if the dataset is not too voluminous and complex, one can use a greedy algorithm: a plane arrangements allow to retrieve the object volumes among a set of intersecting planes.

In our context, the plane detection cannot be performed in one time on the whole dataset, first because we aim to deal with a large amount of data and we look for a streamed workflow, second because, in our acquisition, the path contains loops, and the data acquired at the same place at several passages are not consistent with each other: there is a spatial shift caused by the vehicle geolocation inaccuracy (see chapter 3). Although a self-registration could be done before the plane detection, we prefer to develop a method that could be streamed or even performed in real-time.



(a) (b)

(c) (d)

Figure 4.2:
(a) Input data : a lidar point cloud.
(b) Probability that a point belongs to a flat vertical area.
(c) Line segment soup (colored rectangles).
(d) Output : detected rectangles (pink).

4.2 Proposed Method

The proposed method is divided into three steps:

1. **Weighting points :** Local geometrical features are computed on each point. This step allows to weight 3D points with a probability that they belong to a facade.
2. **Finding line segments :** The journey of the vehicle is replayed. At regular intervals, a weighted RANSAC is performed on points accumulated in a buffer. The buffer contains points acquired around the current position. The line segments with sufficient scores are kept. At the end of this step, a line segment soup is obtained.
3. **Merging line segments :** The line segments computed with RANSAC are connected together according to a distance criterion ($C_{\mathcal{D}}$) and an overlap criterion ($C_{\mathcal{O}}$). The connectivity is checked on each pair of line segments. Then, a graph is drawn, linking the connected line segments. Finally, the connected component are extracted, connected line segments are merged and the resulting facades are filtered.

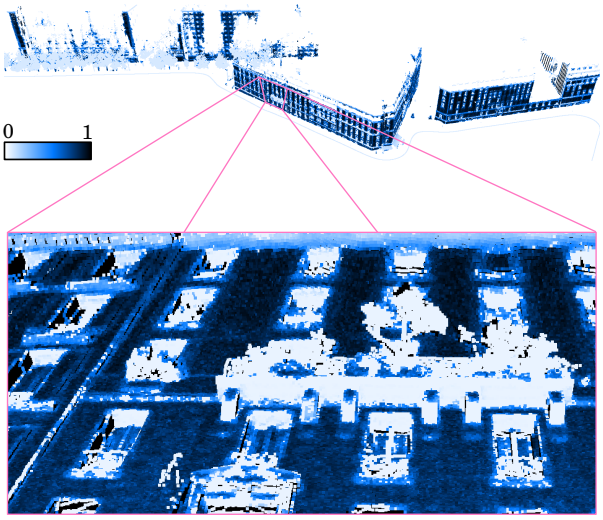
Data processing are illustrated from the input point cloud to the output detected rectangles in figure 4.2. We will now describe these three steps in detail.

4.2.1 Weighting Points

The first step of the algorithm is an analysis of the local geometry on the whole dataset. We use the *Verticality* score defined in chapter 2 to define a probability P_f that a 3D point belongs to a flat vertical area (see figure at right).

$$P_f = Verticality$$

This step can be performed in stream, and as shown in chapter 3 it is even preferable if the data is not self-registered. The points belonging to a local vertical plane have a higher probability to belong also to a vertical wall. The probability P_f is thus intended to favor points according to the local geometric analysis. Hence, a central idea of this chapter is to combine a small-scale analysis with a primitive detection on a larger scale. This will be done by exploiting the probability P_f in two different manners in a RANSAC algorithm.



Probabilities that each 3D point belongs to a flat vertical area. $P_f \in [0, 1]$. Values are stronger on facades, but also on large tree trunks.

4.2.2 Finding Line Segments

Facade detection in 2D: To tackle the problem of facade detection in laser scans, some assumptions are commonly made [50]:

- A facade is roughly planar
- The main plane is supposed vertical

Even if this verticality assumption is strong, it is verified in most cases and allows to reduce the problem of facade detection in 3D to the simpler problem of segment detection in 2D. Hence, 3D points are usually accumulated in horizontal pixel maps [39], or planes [51], in which lines are searched instead of planes. The lines may be found using the Hough transform or RANSAC. Both methods are adequate for detecting simple primitives in noisy data. **Multiple planes detection:** Using RANSAC, problems appears when several planes have to be detected. This difficulty can be overpassed by modifying the algorithm : a minimum description length criterion is used to estimate the model parameters [52] or an order constraint favors some plane orientations [4]. In our approach, RANSAC is performed on overlapping blocks and many plane hypotheses are found and then compared to keep the most relevant. We will now search for vertical planes in the data based on a RANSAC algorithm that we modified by exploiting P_f in both point selection process and primitive score computation. Moreover, the RANSAC will not be performed on the whole data but on overlapping blocks.
First, we note that the facade planes are vertical, we do not need to look for planes in 3D but simply for lines in a projection of the points in the horizontal plane. If a 2D line reaches a sufficient score, it is kept. It is then converted into a line segment according to the inlier points : inliers that are farthest of each other are chosen to bound the line segment.
Streamed detection: RANSAC is not performed at once on the full data, but on overlapping point buffers. A point buffer will be associated to the trajectory interval of the vehicle while acquiring these points. If we call L_{traj} the length of the trajectory and $s \in [0, L_{traj}]$ the curvilinear coordinate of the vehicle center position along the trajectory, we can define the k^{th} buffer as the points acquired while s lies in the interval:

$$[kL_{gap}, kL_{gap} + L_{buffer}]$$

where L_{gap} is the distance between two successive interval lower bounds and L_{buffer} the buffer length (fig 4.3). Hence, the overlap between two successive buffers is equal to $L_{overlap} = L_{buffer} - L_{gap}$ and buffers are

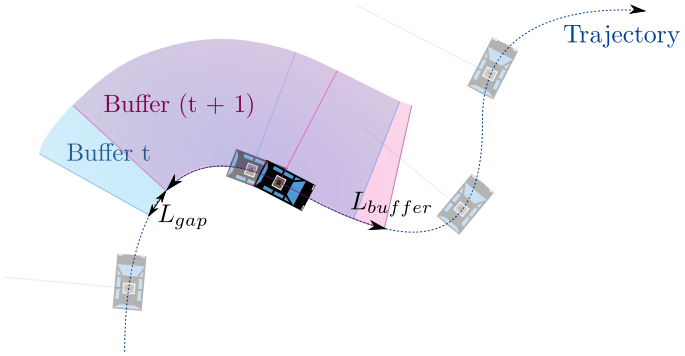


Figure 4.3: Sketch of two point buffers, according to a virtual position of the vehicle t and the next one (t + 1) shifted with L_{gap}.

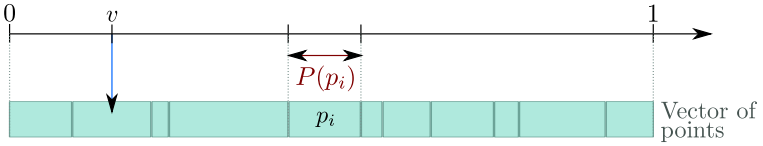


Figure 4.4: Illustration of the point selection with non uniform probability.

overlapping if $L_{buffer} > L_{gap}$. In our experiments, we have chosen $L_{gap} = 2,5$ m and $L_{buffer} = 10$ m which induces a buffer overlap of 7.5 m = 75%. Such a large overlap increases robustness at the cost of a slight increases in computation time.

Weighted RANSAC: On each buffer, a weighted RANSAC is performed. RANSAC iterates two steps: random selection of two points to define a 2D line and computation of a line score. The line with the best score after a certain number of iteration is returned. We introduce the probability P_f in each step:

1. The probability to select a point p_i among all points is $P(p_i) = \dfrac{P_f(p_i)}{\sum P_f}$. To implement this, we compute the sums:

$$S_i = \sum_{k=1}^{i} P(p_k) / \sum_{k=1}^{n} P(p_k)$$

   then to select the points, we use a uniform sample $u \in [0, 1]$ and select the point p_i for which $u \in [S_i, S_{i+1}]$ (see fig 4.4).

2. Whereas in RANSAC, the score of a line is its number of inliers, we compute the line score by adding individual inlier scores taking into account P_f and also the coherence between the estimated normal $\overrightarrow{e_{3p_i}}$ of the point neighborhood and the normal of the line $\overrightarrow{n_{\mathcal{L}}}$:

$$Score(p_i) = P_f(p_i) \times |\overrightarrow{e_{3p_i}}.\overrightarrow{n_{\mathcal{L}}}|$$

$$Score(\mathcal{L}) = \sum_{d(p_i,\mathcal{L})<d_{max}} Score(p_i)$$

   where $d(p_i, \mathcal{L})$ is the orthogonal distance between a point p_i and the line $\mathcal{L}$.

Injecting P_f in the point selection allows to find the most pertinent lines much faster. Injecting it in the score ensures that the detected lines are really along physical planes (walls).
The graph 4.5 shows that using the P_f suppresses noise for the plane detection.

Distance weighting: RANSAC is very sensible to the inlier threshold. In particular points near the threshold distance will be randomly classed as inliers or outliers if their distance is slightly above or

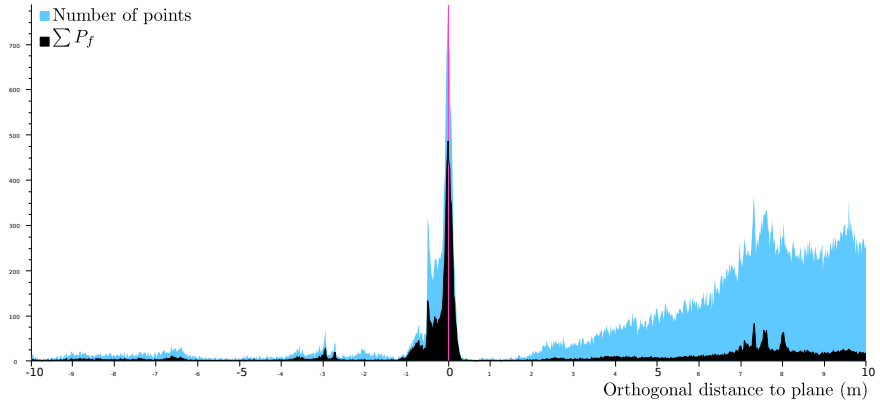


Figure 4.5: Point distribution around the estimated plane: histogram of the signed orthogonal distances to the plane (blue). And accumulation of the P_f (black).

Figure 4.6: Both $Rect(d)$ and $G(d)$ can be used to add the scores of each point to the score of a line, depending on their orthogonal distance d to this line. $Rect(d)$ is parameterized by the maximal distance for inliers d_{max} and $G(d)$ is parameterized with σ.

beneath the threshold. Summing $Score(p_i)$ over all inliers is equivalent to summing $Rect(d_i) \times Score(p_i)$ over all points in the buffer where

$$Rect(d) = \begin{cases} 1 & \text{if} \quad d \leq d_{max} \\ 0 & \text{if} \quad d > d_{max} \end{cases}$$

The problem comes from the fact that $Rect(d)$ is not continuous near d_{max}. We propose to make the score continuous by replacing $Rect$ with a Gaussian function G (cf figure 4.6).
Finally, each line is restricted to the smallest segment containing all the inliers. The method has been tested with $\sigma = 0.1$ m, 0.5 m and 1 m. The best results were obtained for $\sigma = 0.1$ m which allows for the best precision. The output of this step is a soup of all the line segments provided by RANSAC on all buffers. In the next step, we will merge the segments of this soup that match the same line.

4.2.3 Merging Line Segments

The line segments found in the previous step can be arbitrarily cut by the buffer bounds, The aim of this step is to merge the segments that potentially belong to the same line. Thereby, the detected facade footprints will correspond to the line segments obtained after the merging individual segments. The connectivity of every pair of 2D line segments is evaluated with a distance criterion (equation 4.1) and an overlap criterion (equation 4.2) that we will now detail for two line segments $[OV]$ and $[AB]$.

Distance Criterion:

Let $\overrightarrow{u}$ be a unit vector and $\overrightarrow{n}$ a normal vector of $[OV]$. We define

$$\mathcal{D}_{[OV]}([AB]) = \frac{|\overrightarrow{OA}.\overrightarrow{n}| + |\overrightarrow{OB}.\overrightarrow{n}|}{2}$$

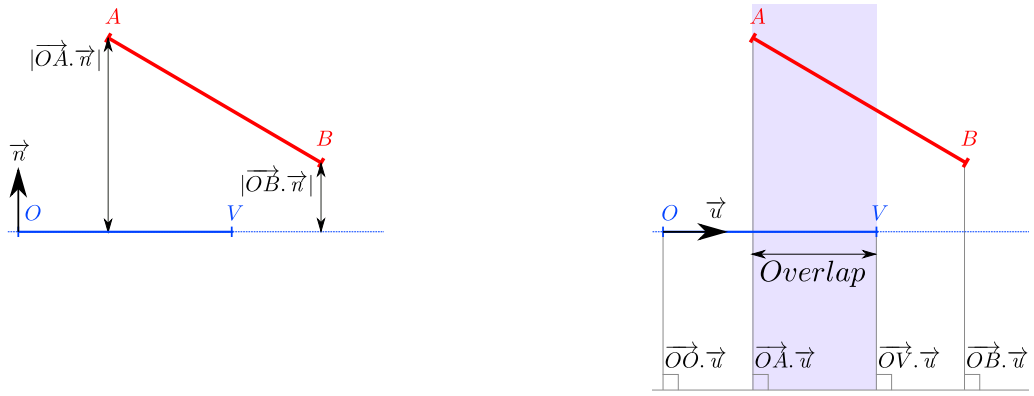


Figure 4.7: The distance of $[AB]$ to $[OV]$ (fig a) and the overlap between $[OV]$ and $[AB]$ projected on (OV) (fig b).

the mean of orthogonal distances of A and B to (OV) (fig 4.7.a): In order to obtain a symmetric function, we define:

$$\mathcal{D}([OV],[AB]) = \frac{\mathcal{D}_{[OV]}([AB]) + \mathcal{D}_{[AB]}([OV])}{2}$$

Finally the symmetric distance criterion writes:

$$C_{\mathcal{D}}([OV],[AB]) \equiv \mathcal{D}([OV],[AB]) < r\,\sigma, \quad r \geq 1 \tag{4.1}$$

Two line segments satisfy this criterion if the distance is lower than a maximal tolerance $r\,\sigma$ where σ is the sigma of G (fig 4.6). Empirically, we fixed it to $5\,\sigma$. If two line segments are along the same line, their orthogonal distance $\mathcal{D}$ is zero, even if they are distant while being on the same line. This is why we also need to measure the overlap between the segments.

Overlap Criterion:

Assuming $\overrightarrow{OV}.\overrightarrow{AB} \geq 0$, let:

$$\mathcal{O}_{[OV]}([AB]) = \min(\overrightarrow{OV}.\overrightarrow{u}, \overrightarrow{OB}.\overrightarrow{u}) - \max(\overrightarrow{OO}.\overrightarrow{u}, \overrightarrow{OA}.\overrightarrow{u})$$

the overlapping part of the projection of $[AB]$ on (OV) (fig 4.7.b). This value is positive if the projection of $[AB]$ is effectively overlapping $[OV]$. Once again, a symmetrized overlap value is given by :

$$\mathcal{O}([OV],[AB]) = \frac{\mathcal{O}_{[OV]}([AB]) + \mathcal{O}_{[AB]}([OV])}{2}$$

Finally the overlap criterion writes:

$$C_{\mathcal{O}}([OV],[AB]) \equiv \mathcal{O}([OV],[AB]) > s\,L_{overlap}, \quad s \in]0,1[\tag{4.2}$$

Empirically, we set the minimum overlap $(s\,L_{overlap})$
to $2.5 \times L_{overlap}$.

Merge:

To merge the segments globally, we create a merge graph where the nodes are the segments, and an edge between two nodes means that the corresponding segments should be merged (they satisfy both criteria). Each connected component of this graph correspond to a set of segments to be merged together. For each connected components with sufficiently high score (sum of individual inliers scores) an unique plane is estimated (least squares fit) from all the inliers of the segments to merge. The condition to keep connected component is a threshold on the sum of inlier scores. This means that a very large number of poor inliers

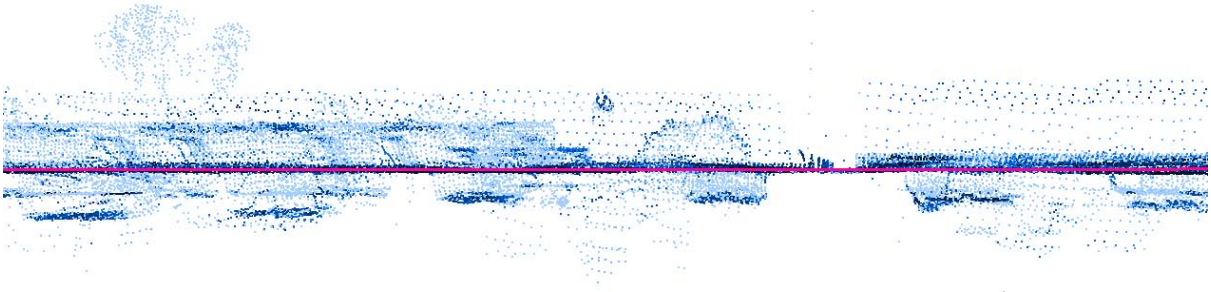


Figure 4.8: Top view of a facade. Detected line (pink).

can be turned into a facade, whereas a small number of good inliers on a narrow facade could be rejected. Instead, average score could be chosen, favoring small vertical flat areas. But the sum yields better results because the facades differ from other urban objects by their size.
As explained in chapter 3, it may happen that the relative point location is good, while the absolute geolocation is less accurate. In such a case, especially if the vehicle performed some loops during the acquisition, some segments extracted at different acquisition times could be merged, whereas this is a geolocation error that brought them closer. For this reason, one may willing to not perform a global merge.

- It is possible to check a temporal constraint when the graph edges are built, in order to avoid edges that link segment too distant in time.
- It is also possible to build the graph dynamically, adding and removing segments as the temporal buffer moves.

Rectangle Delimitation:

The plane obtained by this merging procedure should be delimited to form a facade rectangle (fig 4.1(b)). We developed this methodology as a focusing step which enables to isolate blocks of points corresponding to individual facades from a large amount of data, such that one can simply choose the smallest vertical rectangle containing all inliers.
The choice of rectangle delimitation however depends on the application, in particular if a topology between the rectangles is required. Recovering this facade topology is complex: facade rectangles have to be consistent with the 3D volumes of underlying scene. Topology can be refined by computing intersections between rectangles or deleting areas with low point density. Clues to detect facade bounds are provided by images, where the sky-building limit is protruding and continuous [53], contribution of aerial data can also facilitate this task by exploiting another point of view.

4.3 Results

The algorithm detects most facades from the scanned scene with a high precision, orthogonal distance of inliers to planes reveals visually the relevance of the results (fig 4.8 and 4.9). A few over and under-detection problems were encountered: aligned trees have trunks that present a locally flat and vertical shape so the plane passing through the alignment will have a good score at the expense of the facade behind. This problem of tree rows can be circumvented by extracting trees with another algorithm such as that presented in [37]. Conversely, facades too highly occluded by trees or with a direction too orthogonal to the trajectory have too few points to be correctly detected.
We encountered no example of over-segmentation in our test area: all the segments corresponding to the same facade were always merged. Concerning under-segmentation, it is natural in the case of urban scenes as adjacent facades often share the same plane, so they cannot be distinguished based on our method. To separate coplanar facades, other merging criteria should be used such as the discontinuities in facade heights [39], exploiting the alignments of fine features such as windows [54] or analyzing accumulations of the vertical image gradient [55].

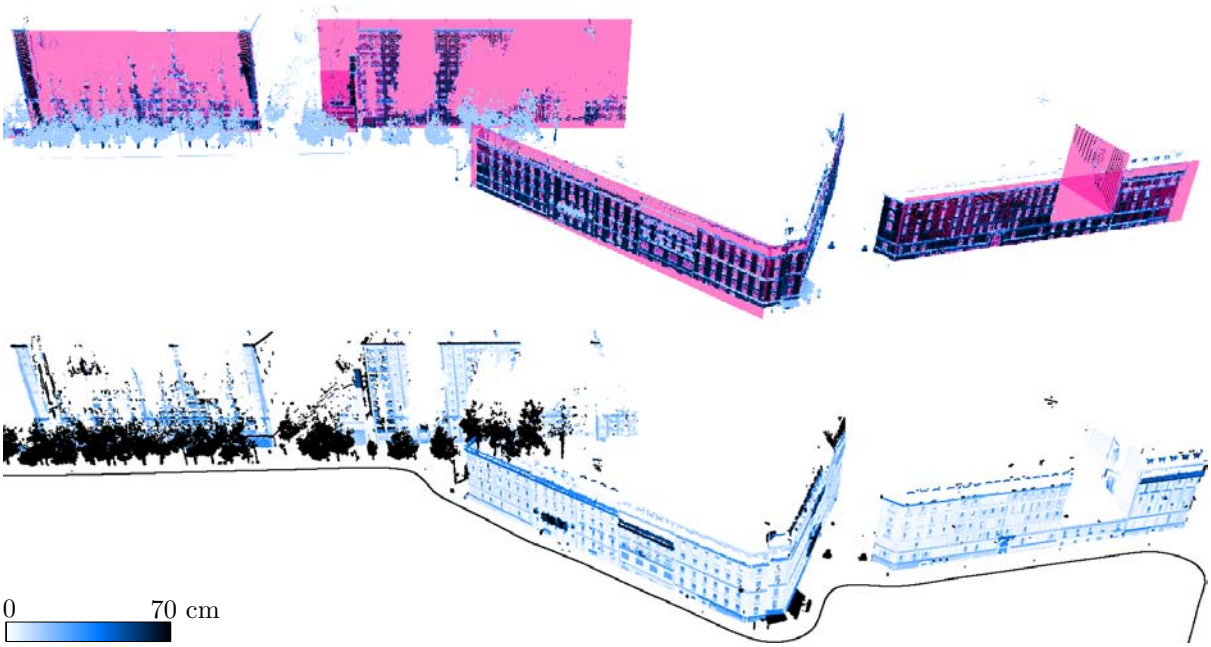


Figure 4.9: Detected rectangles (pink) overlaid on point cloud and orthogonal distance to the closest plane.

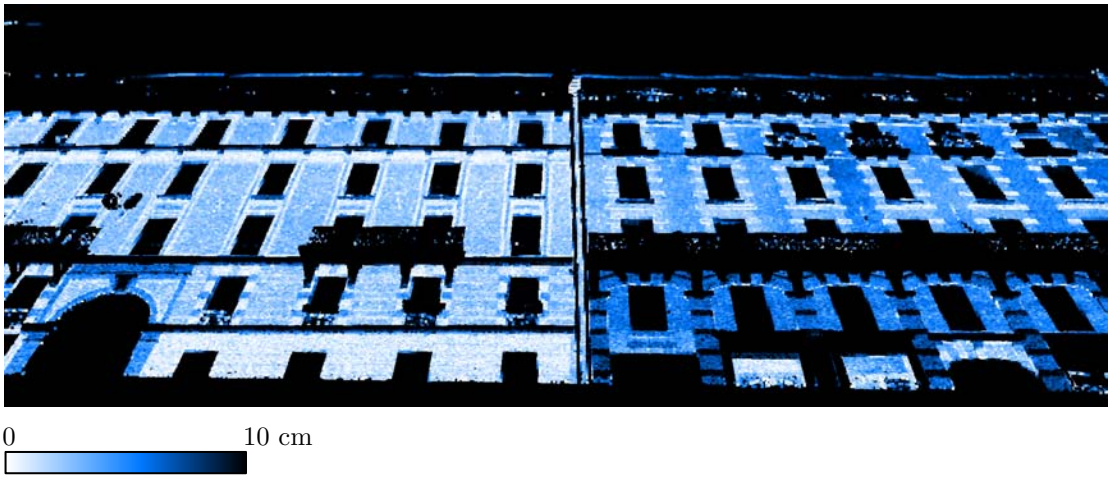


Figure 4.10: Orthogonal distance to the closest plane. The accuracy of our approach is depending on σ (fig 4.6). Here, one can see two different facades. They are oriented along the same direction, but they are slightly shifted : the right one is darker. Both have been matched with the same main plane because this shift is lower than $5\,\sigma$ (equation 4.1).

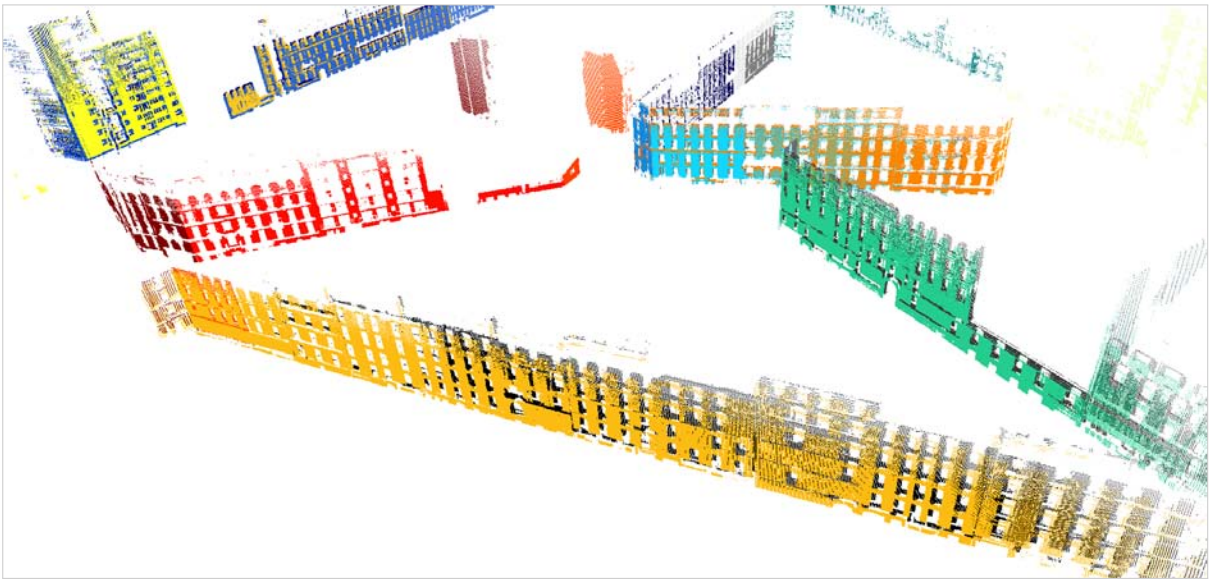


Figure 4.11: Point cloud data without outliers. Inlier points of a detected rectangle have the same color. It happens one facade to be swept several times. Then, a shift due to georeferencing problems may appear between detected rectangles, as the yellow and the dark ones at the forefront.

4.4 Conclusions

We presented a streamed vertical rectangle detection algorithm which automates facade database production from terrestrial laser scans. This algorithm overcomes the volume of data and georeferencing problems, and provides an initial analysis of urban scenes. A modified RANSAC is performed on overlapping buffers of 3D points acquired during the same time interval. Facade parts are thus extracted from the datasets in linear time (in number of 3D points) and constant memory complexity. Facade parts are then merged and the most relevant facade planes are kept. The construction of the merge graph is quadratic in the number of segments, but this number is negligible compared to the number of points.
The vertical planar regions have proved their benefit in fine localization [56] (fig 4.11). The vehicle drift can be detected thanks to shifted rectangles that correspond to the same facade, then, rectangle matching could enable registration refinement. In [41], the rectangles are fitted with the facade rectangles of the bati-3d model in order to perform a non-rigid registration.
In this thesis, the detected rectangles are used to initialize facade models. Two approaches have been tested : a semantic modeling with irregular grids (chapter 5) and a deformable 2.5D grid (chapters 7 and 8).

Chapter 5

Semantic Modeling : Irregular Grid

Contents

In this chapter, we consider the geometric reconstruction of facades with a model that takes into account the facade structure. The chosen model is a grid that cuts the facade according to the main horizontal and vertical geometric discontinuities. The facade is divided into rectangles that may contain a window, a door, a part of the wall, or any other item. In a second step, we try to estimate the depth of each rectangle, and to merge neighboring rectangles that match the same element.
The input data is only PCD, but we present a state of the art of methods that use either PCD or images. The proposed algorithm requires a plane estimation of the main facade wall. For this purpose, we use the work of the chapter 4 that is automatic and does not individualize facades of buildings if they are perfectly aligned. Thus, the input data is not necessarily the facade of a single building. However, the methods of this chapter may help to individualize the facades of a building row.

5.1 Introduction

5.1.1 Urban modeling : complexity behind simplicity

Urban elements offer a wide variety of shapes, sizes and materials. Urban scenes are thus heterogeneous, but paradoxically, very structured. The relative locations, the sizes, the shapes and the other object properties are, most of the time, ruled by many human conventions. For instance, the cars are supposed to be on the road, to have four wheels and to be less than x meter high. Detecting urban elements should be a textbook case: the models are well known and parametrized. But this apparent simplicity hides more complexity and the task is not as simple as it sounds. This paradox is due to our perception of things: As a human being, I know these rules and I perceive them more strongly as I use them to recognize the elements and to understand the situation. In addition, the complexity is combinatorial: as well as associated toy building bricks produce an infinity of buildings, the urban unit rules may produce an infinity of urban scenes. The facades are a typical case of ruled and structured urban elements, composed of finite building

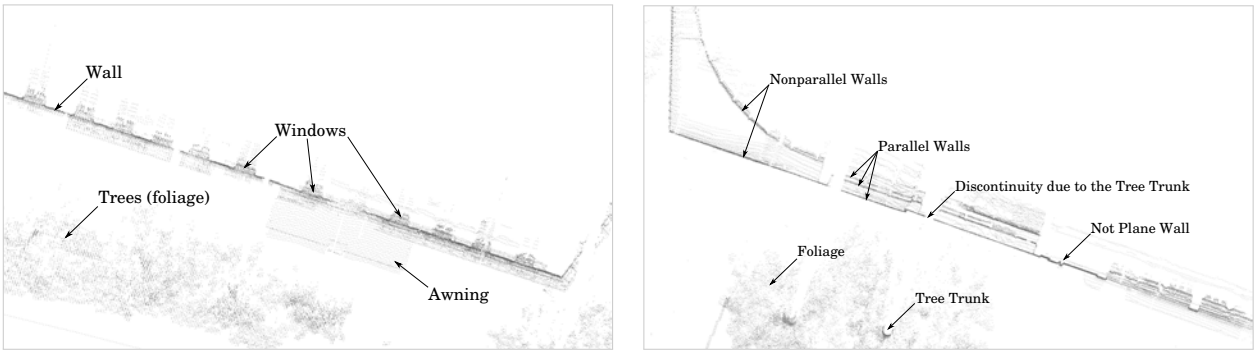
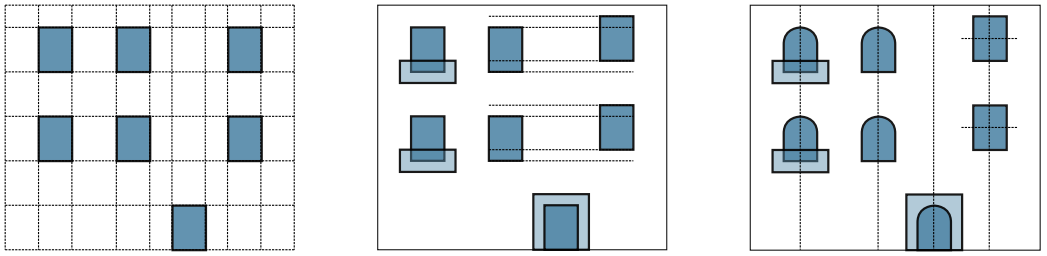


Figure 5.1: MMS PCD of a street scene on the horizontal plane. The facade footprint appears clearly, but it can be complex. This is not always a continuous straight line. There are interruptions because of the trees that mask the facade and reduce the number of points projected on it. The facade may contain walls at different depths, resulting in multiple lines (not always parallel) onto the footprint. Finally, we see that the footprint line is sometimes curvy: it follows the contour of elements that are against the wall as the gutter.



(a) *Gi*: The facade is structured according to an irregular grid.

(b) *Rc*: The facade contains rectangular elements.

(c) *SR*: The facade elements are symmetric and/or repetitive.

Figure 5.2: Structure Assumptions

bricks that are associated with respect to many rules, and as well, this architectural language generates a profusion of facades, with specific styles for each country and even for each city. Moreover, as the architecture is an art, beside architectural rules, there are architectural exceptions. Some facade sculptures or ornaments are unique and make the modeling task more complex.

5.1.2 Facade Reconstruction under Structure Assumptions

Facade Modeling is a vastly studied topic [57]. Numerous authors propose to reconstruct the facades using knowledge based approaches, assuming there is a regular grid structure, looking for rectangular objects, or symmetries and repetitions. The facade structure is considered necessary for a good reconstruction [58]. Indeed seeking of well-defined objects (size, shape ...) is not obvious in point cloud data (PCD). Grammar rules are used to palliate the lack of information. Despite an apparent simplicity, facade structures and grammars are actually complex to automatically recover, to the extent that an operator is deemed necessary to get the desired results [59].
Facade reconstruction, whether from images, lidar data, or both, is usually based on more or less strong structure assumptions. There are three main families, ordered from most restrictive to least restrictive below.

Gi: The facade is structured according to an irregular grid.

Rc: The facade contains rectangular elements (windows, doors...).

SR: The facade elements are symmetric and/or repetitive.

We can see that: $Gi \subset Rc \subset SR$. Indeed, Gi implies that the structure is formed by a set of contiguous rectangles, so Gi is a subset of Rc. And as the rectangles are symmetric elements, Rc is a subset of SR
In image processing, Gi is searched in orthorectified images, and the horizontal and vertical gradients or similarity scores are computed to retrieve the grid lines [55]. The Grid structure is very rigid and is often not adapted to the whole facade, that is why Burochin et al. propose a hierarchical approach that allows more flexibility in the facade model [54]. Gi is a global approach, which is both a strength and a weakness: the grid can be detected robustly and the similar boxes can be factored [60], but the details are destroyed, and, above all, only the very regular facades can be gridded this way.
The Rc assumptions is less rigid. It is true that most of the facades contain rectangular windows. They are matched in images thanks to a rjMCMC process [61], or with a mean shift [62]. In [63], the window corners and shapes are learned. Window detection in lidar data is often based on edge detection, the windows being areas empty of points ([64], [65], [66], [67], [68] and [69]).
In most of the cited papers, the detected rectangles or objects (windows, doors...) are used to retrieve a grid Gi [70], or a more complex structure thanks to a grammar [71].
Other methods use more general assumptions such as the predominance of right angles. [72] proposes to build a 3D model from images by plane arrangements, adding horizontal or vertical planes where information lacks. Although such approaches are adapted to human constructions, objects that do not fit the model cause artifacts [73].
Another line of research is the detection of repetitions and symmetries (SR assumption). This topic have been thoroughly studied by Mitra et al. for meshes and PCD [74], [75], [76], [77], [78]. The repetitions (that are not necessarily sequenced according to Gi [79]) prove that the information is redundant. One can consider that the repeating patterns form the facade texture [70], the repetitions are then detected statistically, but not understood. One can also assume that the repetitions and symmetries are some grammar operators: understanding the facade geometric logic is then desired. Detecting repetitions require to deal with the uneven sampling of the ranks scan [80]. Instead of replacing missing parts by flat or smooth surfaces, some papers offer to "consolidate" and fill the holes thanks to the repetitive structures [81], [82]. A more general approach proposes to reconstruct a surface thanks to the inherent redundant patterns [83]. The surface interpolation is made similarly in areas that are similar. Such method are impressive, but for the time being, the detection of repetitive elements is complex and requires manual assistance [84].
Although knowledge based approaches are promising, detecting facade structures and grammars remains difficult, as evidenced by the need for human intervention. In addition, such methods cannot handle old style facades, complex ornaments, and any unexpected object which does not match the model. Maybe hybrid reconstructions (surface/shape) are more appropriate to these contexts and could help the facade structure to emerge [85].

5.2 Irregular Grid

Our research on facade modeling began with an irregular grid. This approach thus belongs to the category Gi of section 5.1.2. We chose this category because considering the point density in our datasets, the stronger assumption seemed necessary to obtain reliable results.

5.2.1 Overview

The proposed method splits the facade into an irregular grid. The grid lines correspond to the main geometrical discontinuities. For each grid box, the best depth is searched according to the points belonging to the box. Different approaches have been tested for this purpose, and in particular a graph cut. A box classification is suggested but we have not tested it.

5.2.2 Initialization

For each detected facade rectangle (using the method developed in 4, an irregular grid is computed. The facade rectangle is used to define a coordinate system (u, v, h) in which the points can be expressed. u and v correspond to the horizontal and vertical coordinates along the plane, and h is the signed orthogonal distance to the plane. In figure 5.4, the points are projected according to u and v.

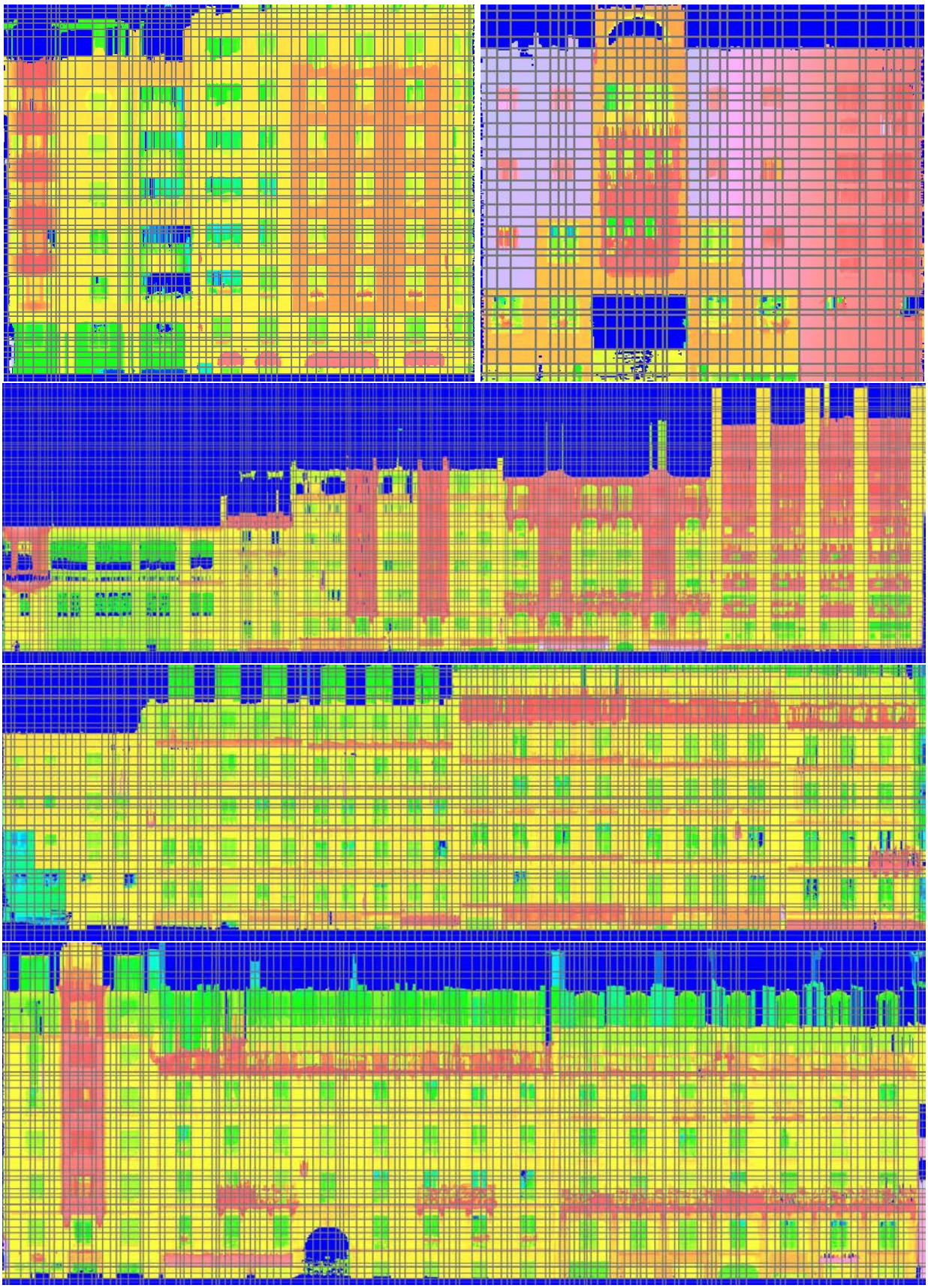


Figure 5.3: Irregular grids for some facades. The facades are colored according to the distance to the main facade plane. The horizontal and vertical discontinuities have been computed with a discretization step $u_{\text{step}} = v_{\text{step}} = 2\text{cm}$.

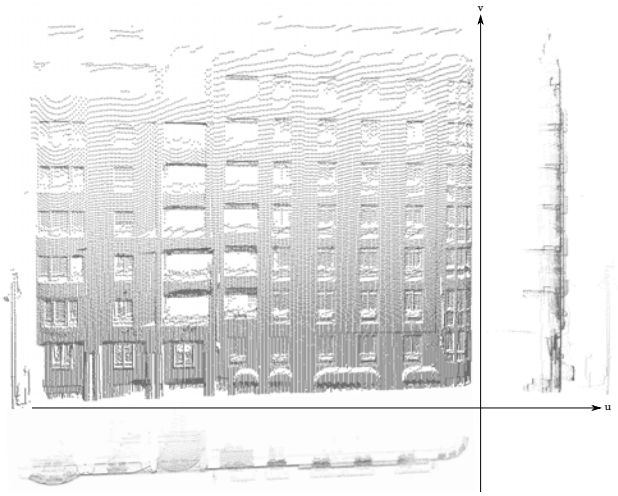


Figure 5.4: Horizontal and vertical projections of the facade points.

5.2.3 Point Selection

The points are projected on the estimated vertical plane. The points are kept only if they lie in the rectangle (bound u and v), and if the orthogonal distance to the plane is lower than a threshold: $|h| = 4$m. This threshold is chosen to include all the points that could belong to the facade

5.2.4 Detect the main horizontal and vertical discontinuities

In order to detect the vertical discontinuities, the points are accumulated on the horizontal plane (fig: 5.1), then the depth discontinuities are calculated along the facade footprint. This is easy to implement when the points are expressed according to (u, v, h): the points are sorted according to u, then, we move along the u direction, and for each discretization step u_{step}, a discontinuity score Ds is computed. For this purpose, we use a 1D Canny-Deriche edge detector [86]. (Formula in annex 9.2). For each u_s, Ds is equal to the sum for each point $P_i(u_i, v_i, h_i)$ of the depth value h_i, convolved with the Canny-Deriche function $\mathbb{CD}$ applied to the relative u position.

$$Ds = \sum h_i \mathbb{CD}(u_i - u_s) \tag{5.1}$$

It is not necessary to compute this exact formula with all the points. Only the points included in a u-buffer centered on u_s may be kept, because $\lim_{x \to \pm\infty} \mathbb{CD}(x) = 0$. The main vertical discontinuities are the values of u for which Ds is a local maximum.
The horizontal discontinuities are detected exactly the same way, we just use v instead of u. The main discontinuities are used to draw the lines of the irregular grid. Some results are shown in figure 5.3.

Pros and Cons

The main drawback of the grid is also the goal: simplification. We wish to assign the same vertical edges to windows vertically aligned, and the algorithm succeeds in this. Nonetheless, if the vertical edges of the windows and doors, or the roof structures for instance, are too close, they are merged in a single discontinuity. The facade has to be structured according to the same grid from bottom to top and left to right. Besides, only the rectangular shapes are handled by this method. These problems appear in figure 5.6.

5.2.5 Depth Discretization

The last step is to associate a depth (h) to each grid box. This is not obvious, because each box contains many points, and sometimes many objects, as in the boxes corresponding to the balconies. Another

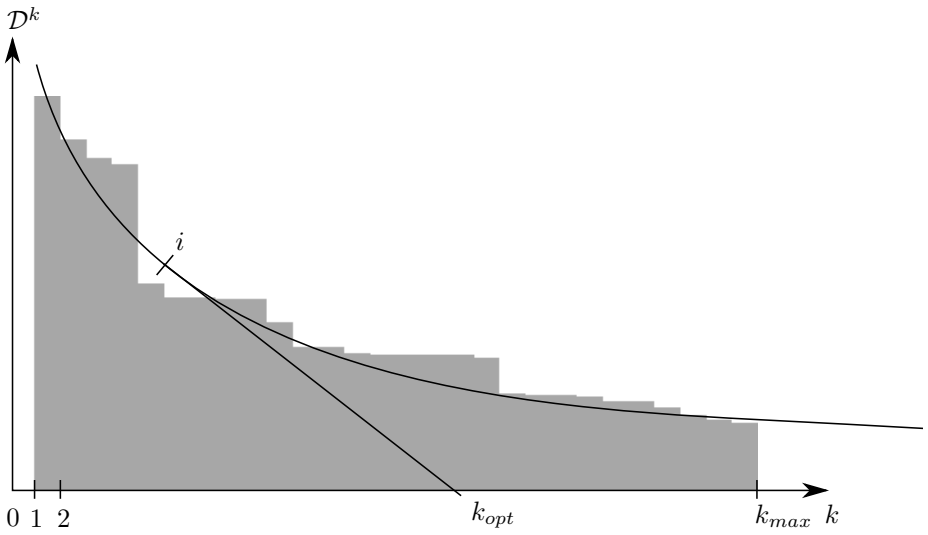


Figure 5.5: $\mathcal{D}^k$ according to k.

constraint is to ensure continuity between the boxes that correspond to the same wall (or more generally the same object). That is why we decided to keep a finite number of depths, and to associate to each box the one that fits the best. For this purpose, we performed a segmentation of the depth histogram using a k-means clustering algorithm that provides k discretized depths.

k-means clustering

Given a set of depths (h_1, h_2, ... h_n), k-means clustering partitions the n depths into k sets ($k \leq n$) $S^k = \{S_1, S_2, ..., S_k\}$ so as to minimize in each cluster S_i the distances to the centroid μ_i. The partition is equal to:

$$\arg\min_{S^k} \mathcal{D}^k \qquad \text{with } \mathcal{D}^k = \sum_{i=1}^{k} \sum_{h_j \in S_i} (h_j - \mu_i)^2$$

This algorithm partitions the depth histogram into k clusters. In order to discretize the depth space with k values, one representative depth is associated to all the depths of each cluster. For each cluster S_i, the most natural representative is μ_i. The discretization algorithm thereby replaces any depth h by the nearest μ_i. Nevertheless, we do not known beforehand how many discretized depths are required. In other words, we need to determine an optimal value for k. As the search space is one-dimensional, the k-means clustering is fast and several values of k can be tested.

MDL k-means

Finding an optimal k (k_{opt}) may be seen as a problem of minimum description length (MDL). One wants to describe a set of depths (h_1, h_2, ... h_n), with k values. So the description length is proportional to k. The smaller is k, the smaller is the description length, but the greater is k, the smaller is the distance to the data $\mathcal{D}^k$. A satisfying solution is therefore a trade off between k and $\mathcal{D}^k$.
Increasing k from 1 to k_{max}, the first new centroids are very "useful" because they strongly decrease $\mathcal{D}^k$, the more centroids are added, the less useful they are. The curve of $\mathcal{D}^k$ according to k resembles $\frac{1}{k}$ (see fig 5.5). One way to choose k_{opt} is to approximate the function $\mathcal{D}^k$ by a curve $C = c_1 + c_2/k$ and then to estimate the curve inflection point i, this corresponds to the highest utility decline. the intersection between the tangent at i and the abscissa axis gives an acceptable value k, located after i.
This method works well, but in practice, we prefer to start from a minimum value $k_{min} = 8$ because this is not weighty to store such a few number of depths, while it is impossible to distinguish the different object

types if all boxes are associated to the same depth. The main problems are not due to this depth discretization method, they come from the assumption that the facade is formed of parallel planar surfaces. If the main plane was improperly initialized, or if the facade is formed of several non-parallel planes, the boxes are oriented obliquely relative to the real plane, and we get stair steps to model a flat surface.

5.2.6 Assign a Depth to each Box

In order to assign a depth to each box, the simplest way is to choose for each box the discretized depth that minimizes the distances to the points belonging to the box (sum of the squared distances). This method is not robust to the outliers and it happened that some boxes that should belong to the wall are assigned to another depth. That is why we tested a global approach in order to penalize depth variations. For this purpose, we used a graph cut algorithm.

Graph Cut

We tested a graph cut algorithm in order to find a continuous surface that contains the maximum of points. The graph cut algorithm computes a surface that separates space into two parts. One part contains the graph node named the "source" and the other part contains the "sink" node. The source is placed on the vehicle trajectory and the sink is placed behind the facade. This algorithm needs a finite number of nodes that correspond to partitioning volumes of $\mathbb{R}^3$. The surface goes through the boundaries between these volumes (these boundaries are the edges of the graph). The space must therefore be segmented. For this, we chose to split the space with planes orthogonal to u, v and h. These planes correspond to the horizontal and vertical discontinuities, and the discretized depths. They are not necessarily equally spaced and they cut the space into voxels of various sizes as shown in figure 5.7. Each voxel is considered as a node, and is linked with its 6 neighbors. Each edge is a rectangle between two voxels. The edge weight is equal to the number of lidar beams "stopped" by the edge. We consider that the beam is stopped if it intersects the rectangle and if the echo lies in the voxel behind. The results do not correspond to the expectation, as shown on figure 5.8. This is disappointing to see that the windows are not retrieved. This comes from the fact that the graph cut is a global approach that does not care about the facade structure! The irregular grid is semantic, because the boxes correspond to semantic objects such as windows or doors. We think that the graph cut would be appropriate to a finer grid, in a low level approach. A solution for the irregular grid would be an analysis of depth histograms in each box.

Depth histogram analysis

Indeed, we do not really look for the most exact box depth, but we want the boxes that contain the same objects to receive the same treatment: in fact we want to classify the boxes. We did not tested a box classification, we only present some ideas. The grid itself offers some features such as the position and the box sizes, especially for the windows which are similarly sized and aligned. The point depth distribution seems also interesting. In order to analyze it, the depth histograms are displayed as in figure 5.10. Most of the time, the distribution is uni, bi or trimodal. The behaviors are often similar for similar objects, the modes seem thereby useful for classification. However, we may encounter the problems displayed in figure 5.11 as set out below.

(a) The wall plane is not parallel with the estimated main plane. Even if the main plane is well estimated, this may be another wall differently oriented. The modes are different while this is the same wall. If the boxes are displayed with a rectangle parallel to the main plane, the result will resembles a staircase.

(b) The vertical lines are too close. The right window is over-segmented.

(c) The facade is occluded by trees.

Figure 5.6: Depth Discretization. Only the grid boxes that contain enough points are displayed. The boxes are colored according to their depth values. We do not always understand the meaning of the boxes that do not necessarily correspond to a clearly identifiable object such as a window or balcony. In addition, all windows have no associated box. Indeed, shuttered windows have the same depth as the wall and are therefore not highlighted by a differently colored box. However, boxes highlight the overall structure of the facade. One can observe specific behaviors on each floor, such as the first floor balconies, and one can see that the box sizes and positions vary between buildings of the same alignment. Boxes could therefore provide clues for understanding the facade grammar.

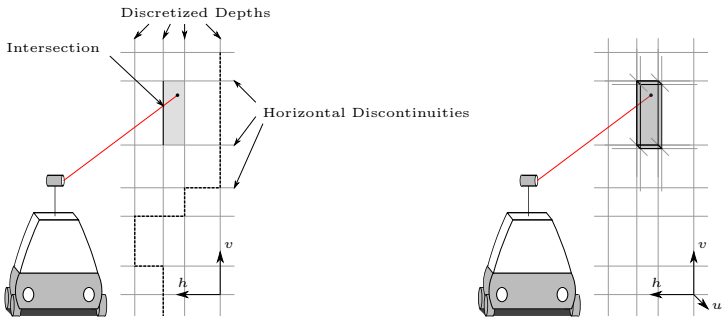


Figure 5.7: Sketch of graph construction for graph cut algorithm. Left, the space partitioning is shown in 2D (v, h). The space is split according to the horizontal discontinuities, and the discretized depths. The lidar point lies in the gray voxel. We record it in the weight of the edge of the gray voxel which intersected the lidar beam (red). The dotted line is an example of output surface. This output surface is supposed to go through the most weighted edges. Right, the voxel is shown in 3D.

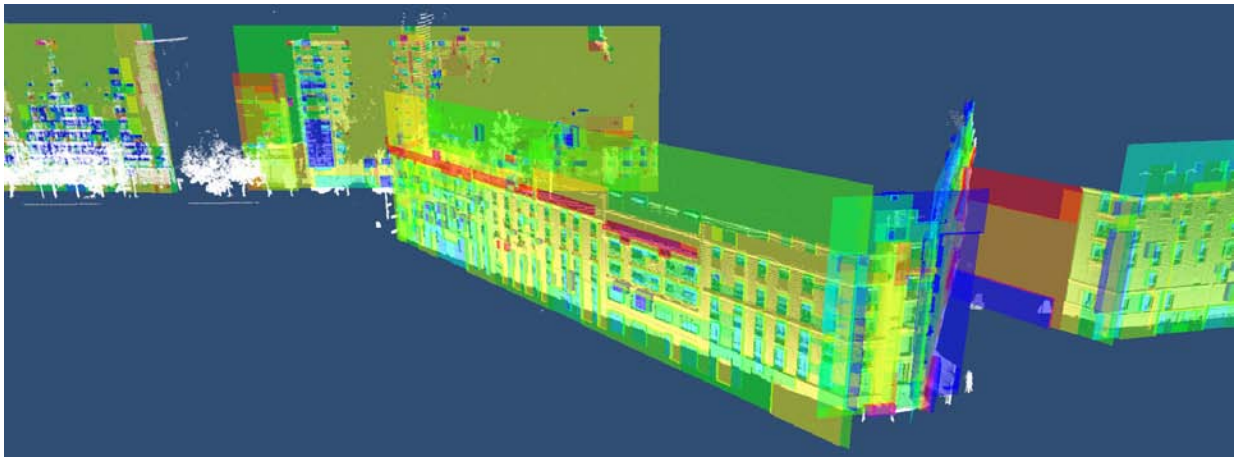


Figure 5.8: Surfaces from graph cut

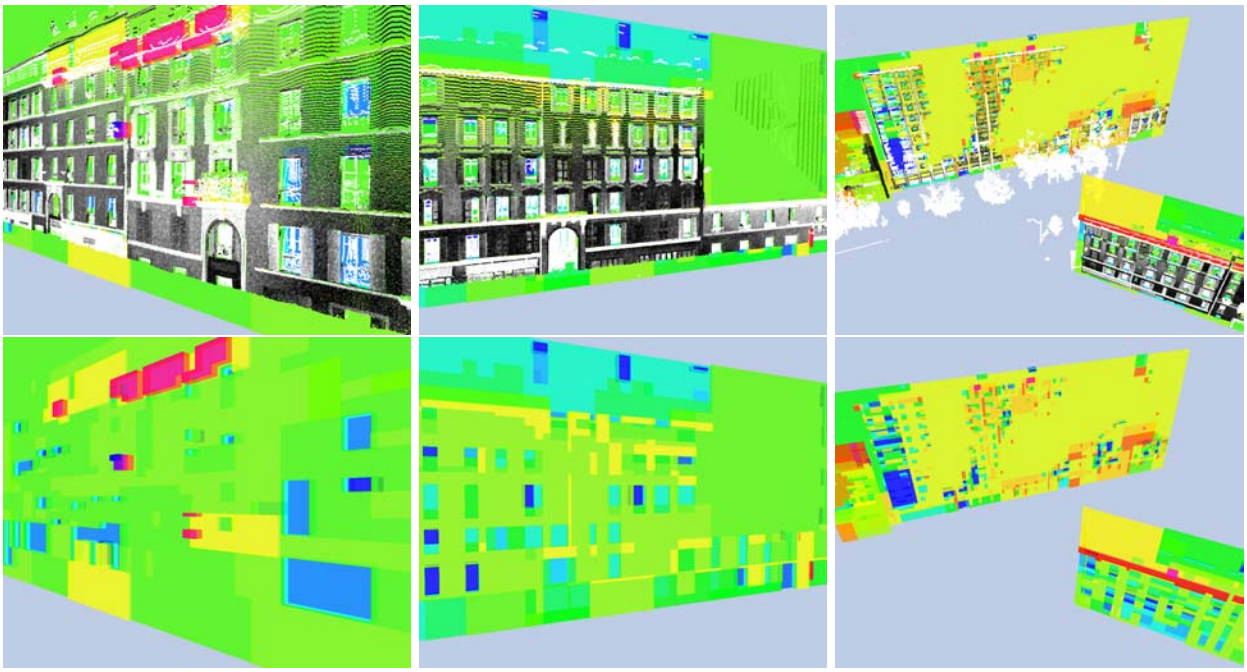


Figure 5.9: Surfaces from graph cut. For the 3 points of view, the surface is displayed with the point (colored according to the distance to the wall, black=0 → white≥10cm. and without the points.

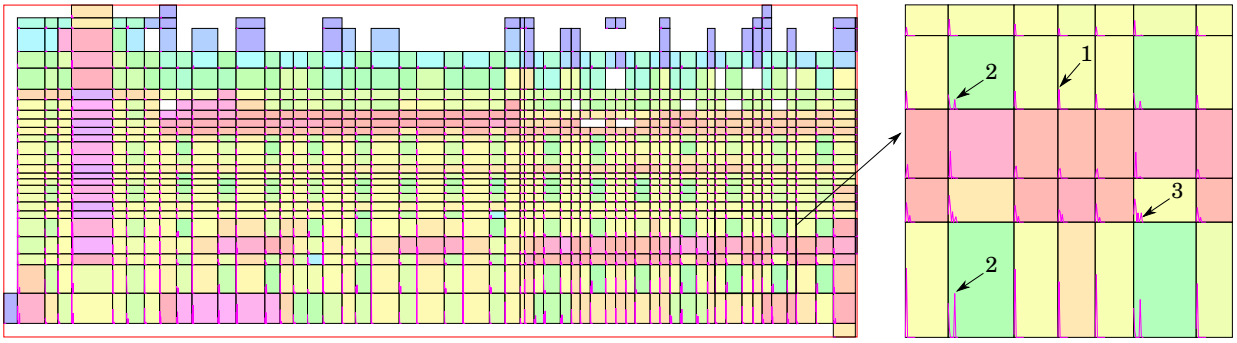
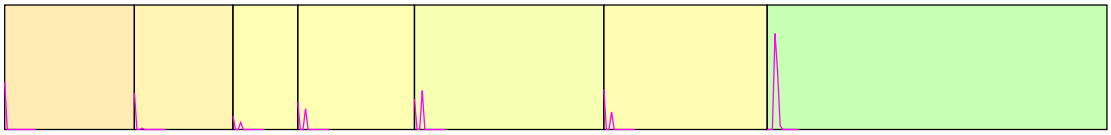
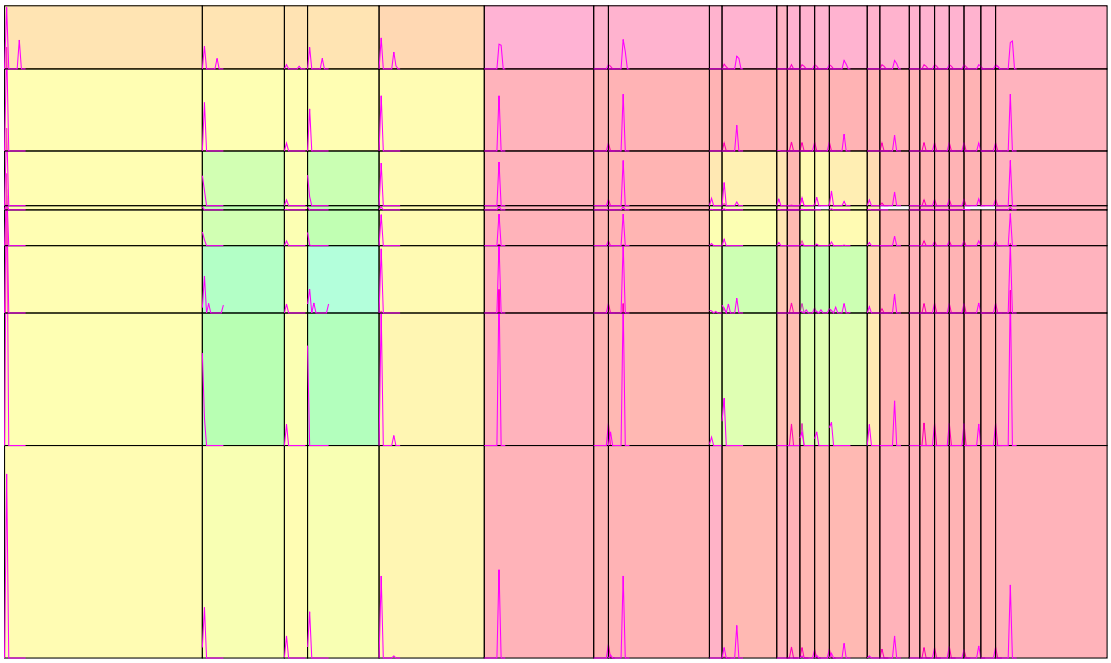


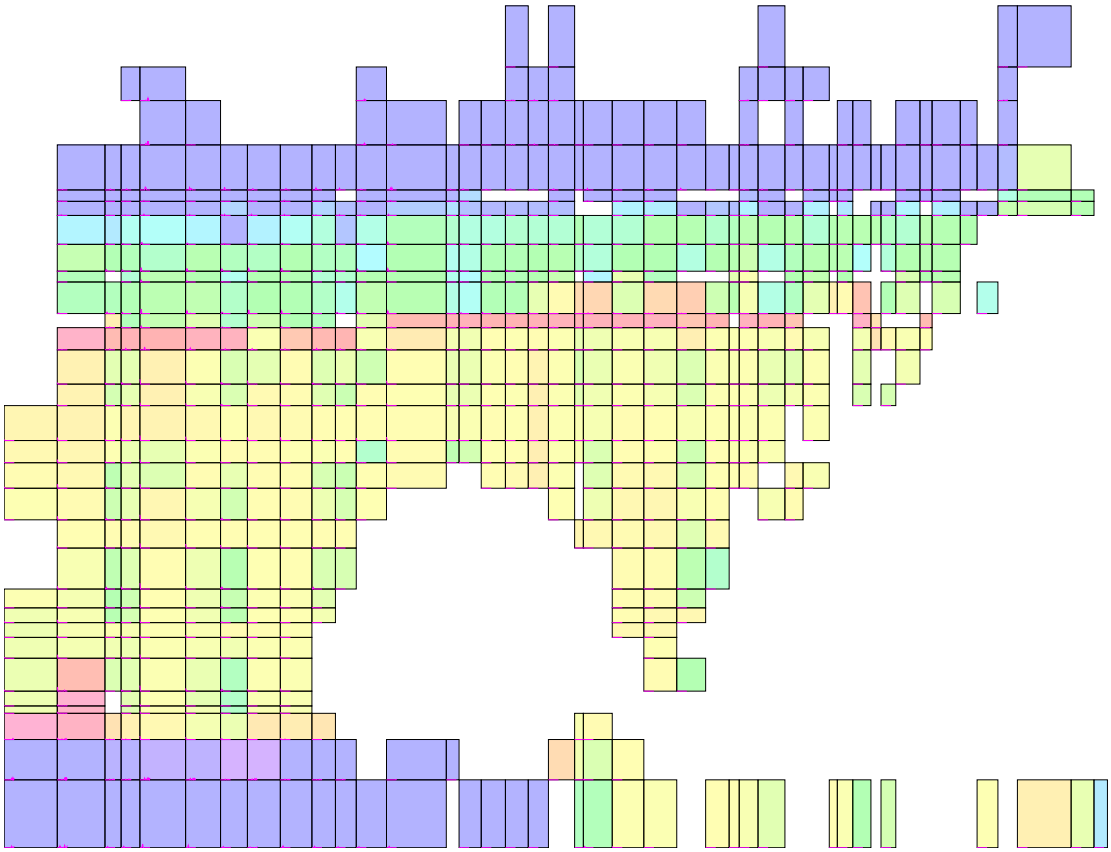
Figure 5.10: For each grid box, the depth histogram is displayed in pink.
1 On the walls, most of the time there is a unimodal distribution.
2 On the windows, the distribution is often bimodal. The peaks are greater for the bottom window, because the sensor is closer, which results in a greater point density.
3 On the balconies, a trimodal or even more complex distribution may be seen.



(a) Non Parallel Plane



(b) Bad Grid



(c) Occlusions

Figure 5.11: Some problems in irregular grid modeling.

5.3 Conclusion

We have proposed a facade model with irregular grids. This model is calculated from a point cloud. First, a vertical rectangle corresponding to the main plane of the facade has been detected. Only 3D points included in this rectangle and sufficiently close to the main plane are taken into account. The rectangle is divided in an irregular grid with vertical and horizontal lines placed at the principal geometrical discontinuities. To do this, we accumulate the points horizontally and vertically, then the point depth variations are calculated (depth relative to the main plane). The discontinuities are the depth variation maxima. These discontinuities allow to cut the rectangle into a set of rectangles/boxes. For each box, we then search the optimal depth based on points projected in it. We preferred to limit the total number of possible depths, allowing for example all boxes that contain a part of the same wall to have the same depth. Guided by this choice, a depths discretization algorithm was proposed. It is a variant of the "k-means" that automatically finds the optimal number k of depths. We try to minimize both a data term and the number k. The remaining task is to associate one of these k depths to each box. The resulting model is an irregular grid where each box moves back or forward relative to the main plane. This model therefore assumes that the facades are composed of parallel rectangular elements. Other approaches have been studied to determine the depth of each box as a graph- cut algorithm.
The proposed irregular grid is a **semantic** approach. Indeed, although purely geometric assumptions are made: vertical and horizontal discontinuities, parallel plane... Grid boxes correspond to objects such as doors and windows and we hope that their geometric modelings correspond to our human perception-representation of the object (window behind the wall, balcony forecourt...) and we even hope that all the boxes that contain the same object type are similarly modeled. However, this does not necessarily reflect the reality (closed shutter open window ...). In addition, all non-rectangular and "unclassifiable" objects are problematic. We drew two conclusions from this observation.

- First, it takes us away from our goal: to provide a photorealistic modeling. Irregular grid are not suitable to represent all the structures that can be found in the Parisian facades.
- Second, the proposed depth discretization method is too simplistic. We did not go further, but we think that a box classification according to the depth histograms of each box might provide better results. The geometric modeling might be done in a second step, given the classification results.

For this thesis purpose, we opted for a low level approach, consistent as possible with the data. Indeed, we believe that a processing step is missing in facade modeling from lidar data. Trying to detect structure into PCD implies to manage the density variation that can both depend on underlying objects and acquisition context (more details about point density in chapter 1). A more "acquisition independent" data is needed. The proposed "deformable grid" of the next chapter trying to fulfill this role.

Chapter 6

Direct meshing in sensor topology

Contents

The lidar data is sparse and a main issues is to manage gaps between areas where there is information (echoes and beams). A sub-problem is to determine whether two neighboring echoes belong to the same object (or the same surface). If they do, we can connect them or interpolate a surface between them. Otherwise, this surface does not make sense. We therefore raises the question of interpolation of the geometric information between echoes. We propose some **"sensor oriented" methods to link echoes** using information such as the acquisition time or the firing angle. But we also evaluate the limits of such low-level approaches.

6.1 Connectivity of successive echoes

Lidar echoes acquired successively have a high probability to be neighbors in 3D space. If the sensor is in front of the surface, one can have confidence in the fact that there is continuity between successive echoes, whereas if the sensor sees the surface with a grazing angle, cavities may exist between two successive echoes that hide a portion of the surface and contradict the continuity between successive echoes. The incidence angle thus provides a simple measure of the lidar reliability in detecting a continuous surface.
We then investigate the relevance of liking the points along the scan lines. We observe that, forgetting the possible noise measurement, it may provide the most precise surface estimation.

6.1.1 Incidence angle variation

Level of accuracy and acquisition frequency of lidar systems tends to increase. The consecutive pulses are getting closer such that the footprints may overlap. ensuring a strong continuity between consecutive points, to such an extent that we may wonder if the output data would not be these scanlines rather than the echoes. It would be lighter to handle a line rather than thousands of points. Moreover, lines would bring more information for surface reconstruction. However, scanlines do not always follow the relief of the objects. Sometimes scanlines just link different objects acquired consecutively. **Relation between scanlines and scanned objects is complex, but many clues are contained within.** We propose here a simple way to analyze the scanned scene thanks to scanlines. A graph is build with echoes and four kinds of edges (a,b,c,d) that highlight neighborhood relationships between lidar echoes. Some edges follow

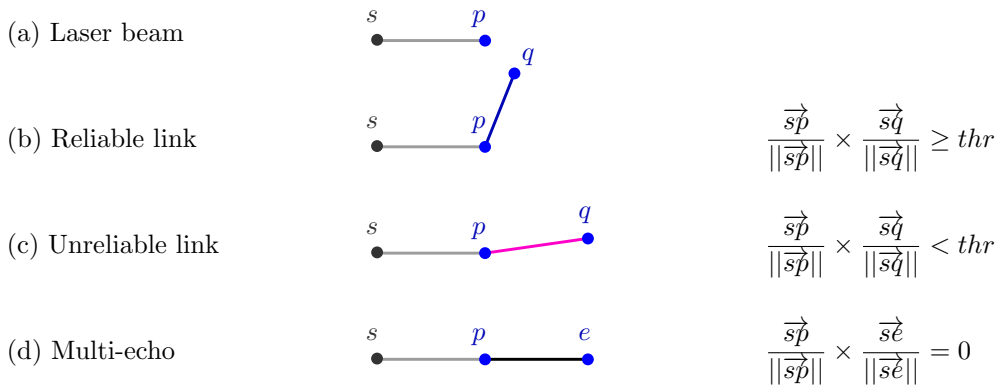


Figure 6.1: Edge color code.

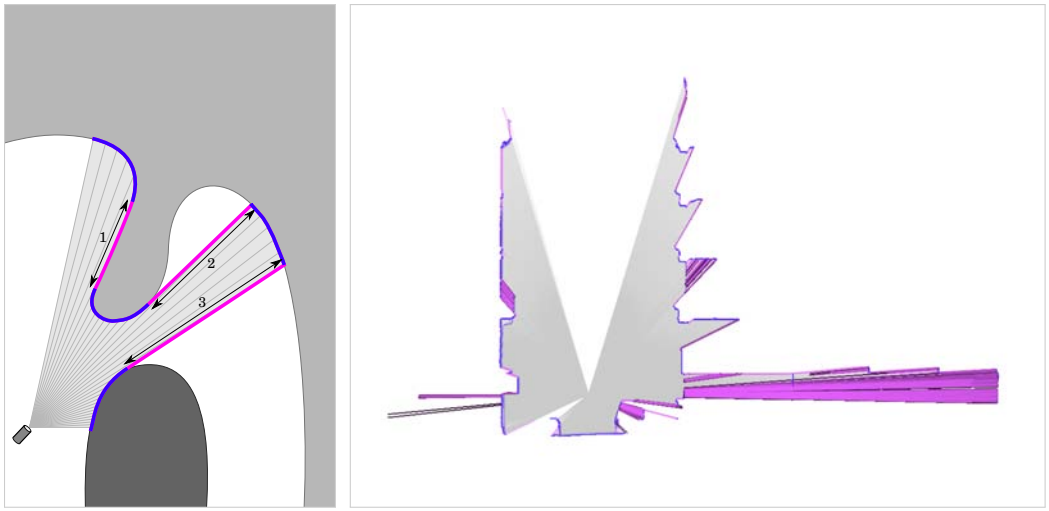
object surfaces, they link echoes that belong to the same object and some others separate echoes that belong to distinct objects.
Screen shots from real data are analyzed in this section. Edge coloring is detailed here (figure 6.1).

(a) Each echo p is linked with the sensor location s. These edges cross the empty area before echoes.

(b & c) Each echo is linked with the echo(es) acquired during the next shot (q). The cosine of incidence angle is estimated by : $\frac{\overrightarrow{sp}}{||\overrightarrow{sp}||} \times \frac{\overrightarrow{sq}}{||\overrightarrow{sq}||}$. These kinds of edges may belong to the object surface reliably (b), or not (c) depending on the incidence angle. But this is only an arbitrary distinction according to a threshold thr.

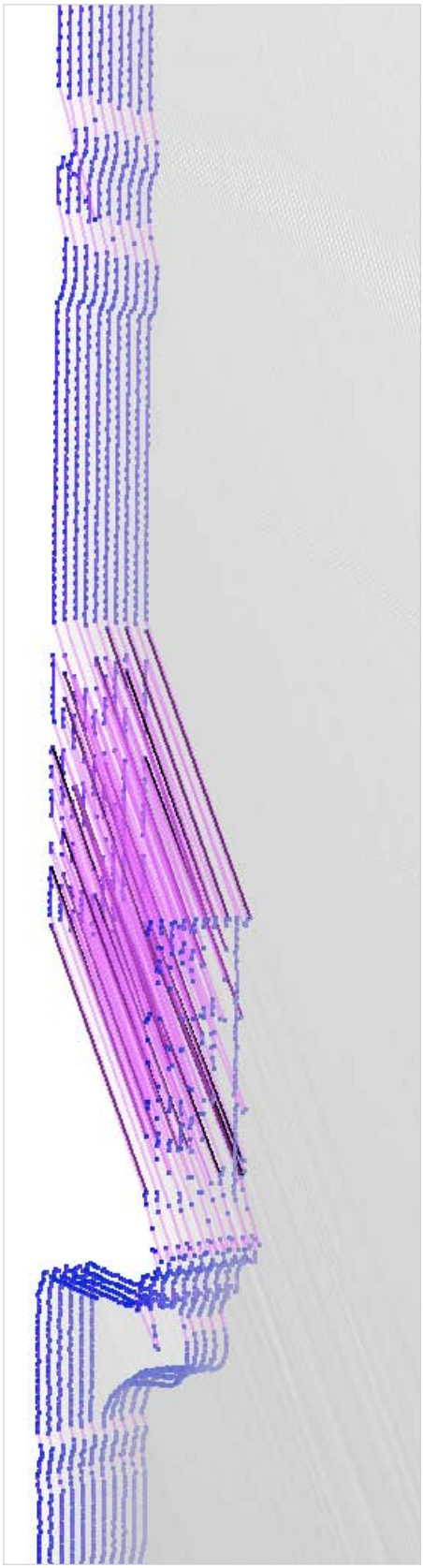
(d) Each echo is linked with the other echo(es) acquired during the same shot. As well as in (b & d), only the context confirms the nature of this edge.

The edges are displayed for a short acquisition duration, this dataset S is fully shown in figure 6.2, and figure 6.3 zoom on four interesting regions. The main observation is that there are numerous multi-echoes (black). Multiple surfaces are also detected from the same point of view. It is thus not possible to assume that neighboring points in acquisition geometry are neighbors in the 3D space. Another observation is that the "unreliable links" (pink) often correspond to a link between two different objects, in other words, they deserve to be unreliable because they do not link neighbor points. Moreover, some detection artifacts can produce echoes along these grazing beams. The lidar measure is thus less reliable for the grazing angles, and even lidar points do not guarantee the presence of a surface. Moreover, a surface exactly aligned along the laser beam is less probable than a geometric discontinuity (fig: 6.2).
To conclude, the "reliable links" indicate a continuous surface while the "unreliable links" indicates a geometrical discontinuity, or a surface, but this surface is not certain, even if there are some lidar echoes on it.
When the links between successive echoes are displayed, one can observe that they often connect echoes lying on the same object surface, except when the incidence angle is grazing: in this case, the two successive echoes more often belong to two different objects. In all cases, the continuity between successive echoes is not one hundred percent reliable, however, the mesh that connects neighboring echoes in sensor topology has a relevance that we will try to formalize theoretically.

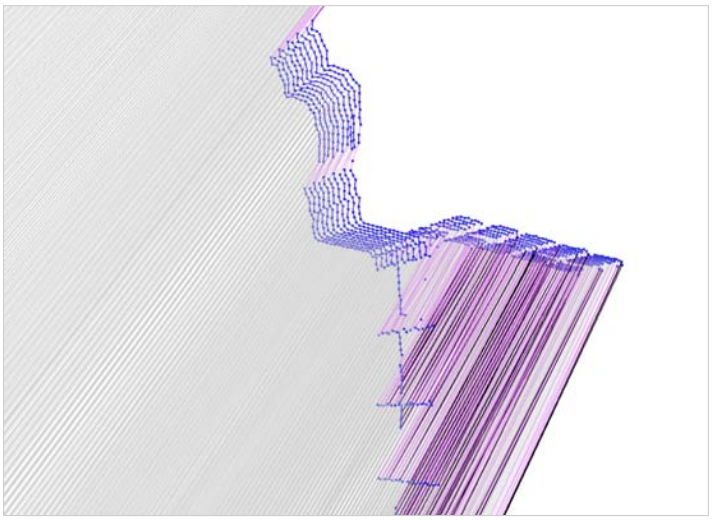


(a) Grazing angle causes (b) Dataset S

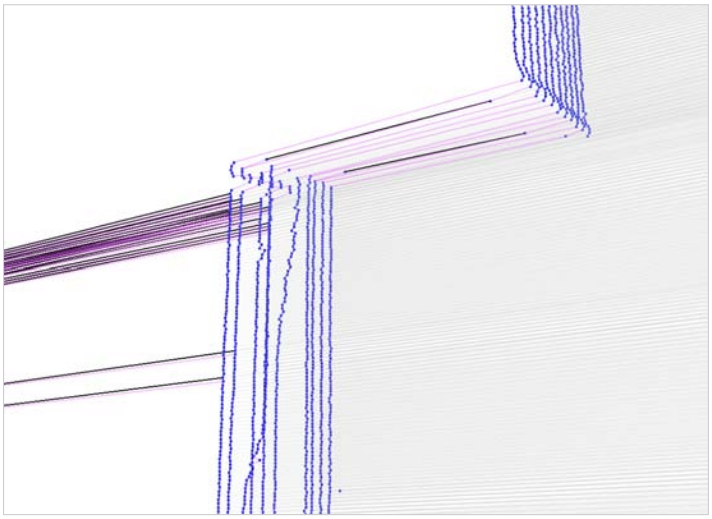
Figure 6.2: A grazing angle may correspond to:
(a-1) a surface aligned with the laser beam.
(a-2) a strong depth variation of the same surface.
(a-3) an empty area between two objects.
(b)Dataset S: The data is displayed as if the car were in front of us. All the laser beams (gray) start from the same point that corresponds to the sensor location.
The color code is detailed in figure 6.1.



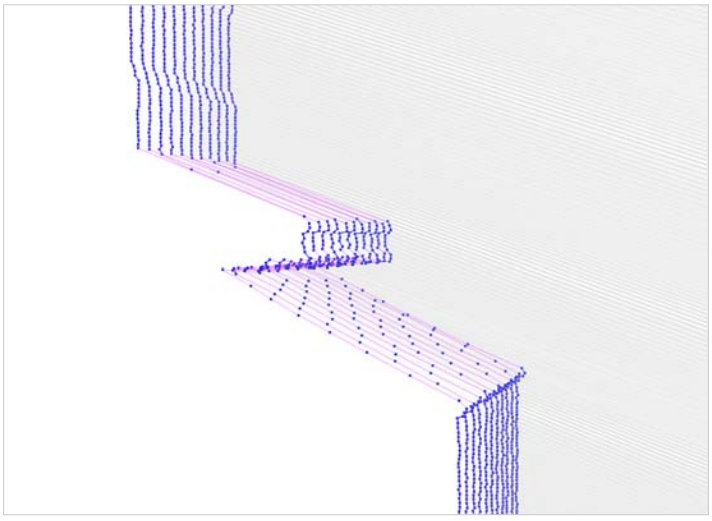
(a) Balcony



(b) Window

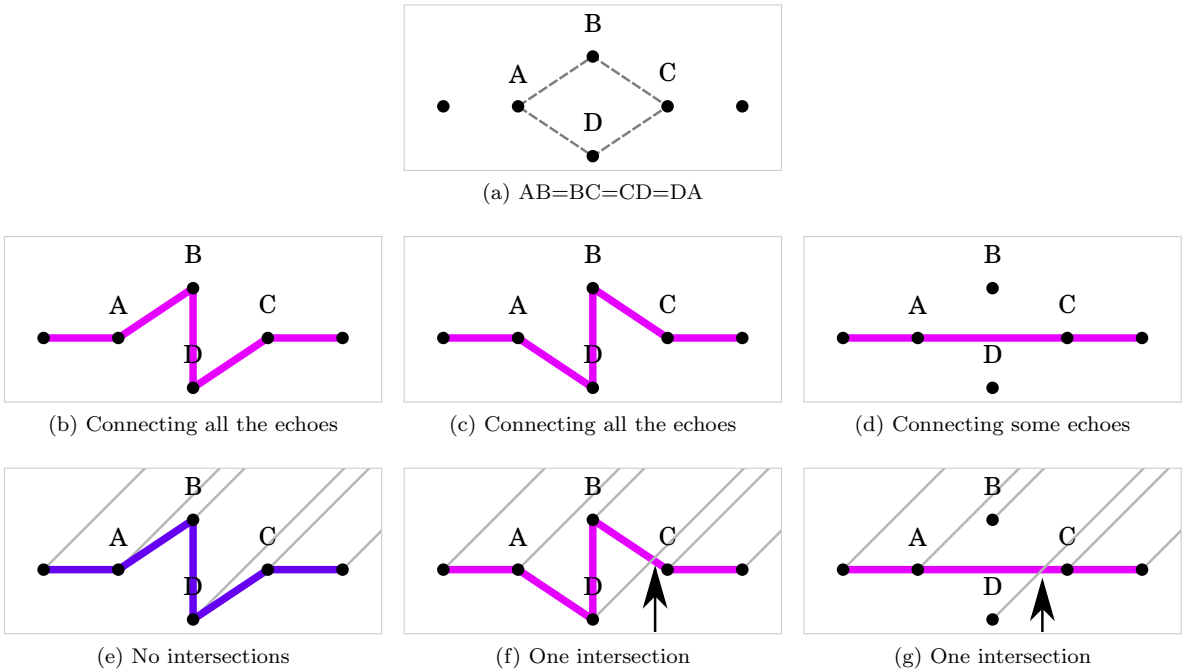


(c) Semi-reflective surface



(d) Improbable surfaces

Figure 6.3: Zooms on the dataset S. Unreliable links and multi-echoes in real cases. Color code detailed in fig6.1.
(a)Some beams are stopped by a balcony, some other go through.
(b)Most of the beams go through the window, except those stopped by window bars.
(c)A semi-reflective surface induces some multi-echoes (black edges).
(d)Do the two inclined planes that contain pink edges (unreliable links) correspond to true surfaces? It is more probable that the pink edges connect two distant walls rather they belong to a surface having exactly the same orientation as the beams. However, the lowest plane contains many echoes. In fact, the echoes along the pink edges are due to a detection artifact.



(a) AB=BC=CD=DA

(b) Connecting all the echoes

(c) Connecting all the echoes

(d) Connecting some echoes

(e) No intersections

(f) One intersection

(g) One intersection

Figure 6.4: Undecidable cases. Some echoes are distributed as in (a): AB=BC=CD=DA. A polygonal chain that connects all the echoes or a subset is reconstructed. If we only try to minimize a criterion as the chain length as in (b), (c) and (d), some cases are undecidable as between (b) and (c). Using the sensor topology may allow to decide. For instance, if intersections between the chain and the beams are forbidden (except at the echo locations), the chain (e) is preferred to the chain (f). As well the chain (g) intersects a beam and is thereby inconsistent with the sensor information.

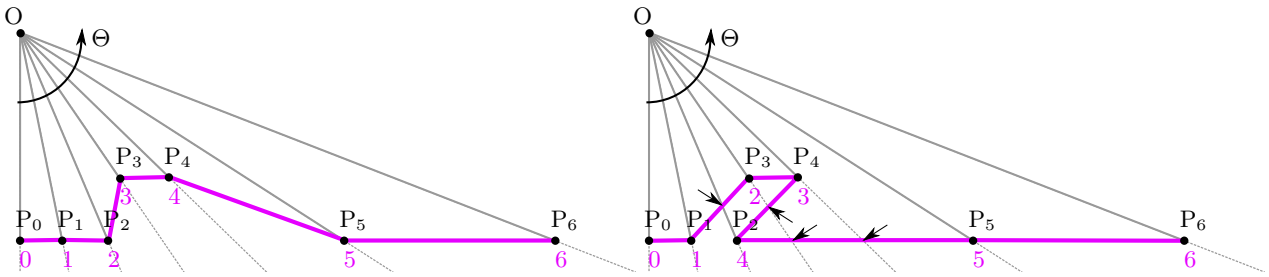
6.1.2 Theoretical relevance of scan order

When reconstructing a continuous surface detected by the lidar signal, meshing echoes that are neighbors in sensor topology naturally integrates logical constraints imposed by the laser beams, namely that the reconstructed surface can not be intersected by the half-lines of the laser beams, neither before nor after the echoes.

Before the echo, it is assumed that the laser beam passes through an empty area, the reconstructed surface would then be pierced by the laser beam.

After the echo, there is no more information since the beam was reflected and returned in the other direction. This corresponds to an area not seen by the sensor and one therefore invents information if the surface is reconstructed at this location.

Within the context of surface reconstruction, we consider the data acquired from a static lidar system that emits laser beams from a fixed center O. All the echoes are assumed to belong to a continuous surface. We would like to approximate this surface by a triangulation in which each echo is a vertex. We first analyze a similar case in 2D, with a polygonal chain that links all the echoes. If we try to find the optimal chain according to a distance criterion such as the chain length, the solution is not obvious, and some cases are undecidable as in figure 6.4 where two possibilities are equivalent. Our intuition is that the sensor topology may bring information to solve such dilemma. For instance, one may avoid that the reconstructed chain or surface mesh intersects the laser beams. Indeed, the echo is assumed to locate the first encountered surface after an empty area. Such constraint allows to reduce the search space to solutions coherent with our acquisition knowledges. We propose a stronger constraint: the reconstructed surface is intersected by each half-line supporting a beam only once, at the echo location. Intersecting such half-line between the sensor location and the echo is not consistent with the assumption that the laser beam goes through an empty area before the echo. And intersecting the half-line after the echo indicates that the surface is reconstructed



(a) Monotonic function (b) Non-monotonic function

Figure 6.5: Illustration of the theorem 6.1.2. Order of the P_i along the polygonal chain (pink).
Either this order is a monotonic function of i that is the Θ rank of the P_i (a), or there is at least one beam (gray) that intersects the chain somewhere else than at the echo location (b).

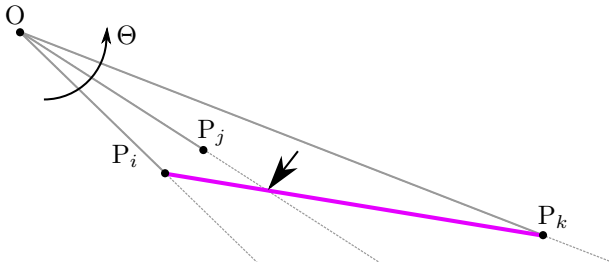


Figure 6.6: Illustration of the demonstration 6.1.2.
If P_i, P_j, P_k are in ascending order of Θ (ie $i < j < k$) and P_i and P_k are consecutive along C, B_j intersects the polygonal chain (pink) somewhere else than at P_j.

in an area unexplored by the sensor which corresponds to invent data where there is no information. It highlights the relevance of connecting the echoes according to the sensor topology. We consider a set of N echoes P_i and the corresponding laser beams B_i in 2D. Each beam B_i is a half-line starting from a point O and is defined by its shooting angle Θ_i. An echo P_i is defined by the pair (Θ_i, R_i), with R_i the distance to O and i the ascending rank of P_i according to Θ.

Given a polygonal chain C that goes through each P_i once time, let $S(i)$ be the order of P_i along this chain. We can observe that S is a bijective function of $[1, N]$ in $[1, N]$.

Theorem:

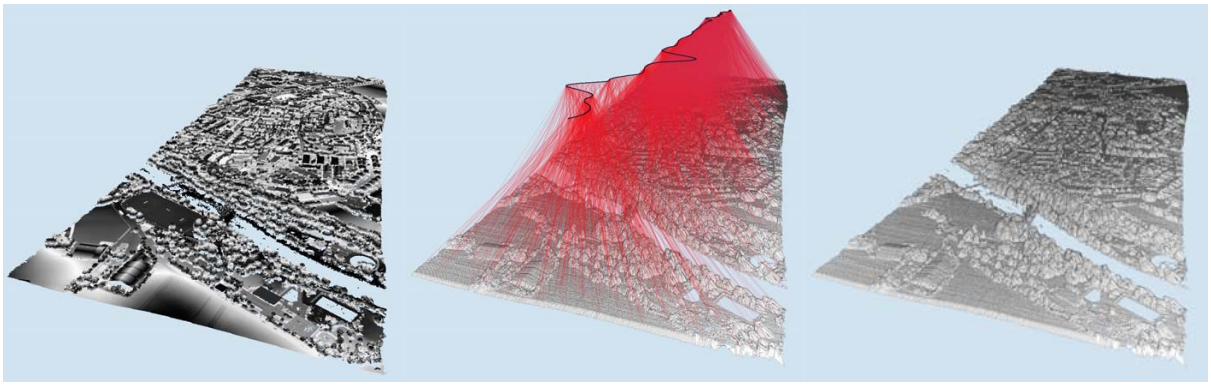
Let P_i and O be some 2D points, and let each B_i be a half-lines starting from O and passing through the corresponding P_i. Let C be a polygonal chain that goes through each P_i once time and let $S(i)$ be the order of P_i along this chain. **Either S is a monotonic function, or there is at least one half-line B_i that intersects C more than once.**

The second possibility is equivalent to : "there is at least one half-line B_i that intersects C somewhere else than at P_i. These two possibilities are illustrated in figure 6.5.

Demonstration:

The demonstration is illustrated in figure 6.6. If $N < 3$, S is monotonic. Else, if the function is not monotonic, there is at least one interchanging, so one can find at least one triplet P_i, P_j, P_k such as $i < j < k$ and $|S(i) - S(k)| = 1$ (ie P_i and P_k are consecutive along C).

C is assumed to go through P_j so C intersects B_j at P_j, and as B_j is between B_i and B_k, B_j intersects the line segment P_iP_k. C is therefore intersected twice by B_j.



(a) PCD, altitude in black and white) (b) Laser beams (c) Sensor mesh

Figure 6.7: Vaihingen Dataset.

Generalization

In 3D, for a static lidar system that emits laser beams from a fixed center O, each echo can be located according to its polar coordinates Θ, Φ, R. The theorem extension would be the following : only the surface meshes which triangulation is non overlapping in Θ, Φ are intersected once by each half-line supporting a laser beam.
The sensor mesh is an appropriate solution to reconstruct a continuous surface. However, the geometry of the scanned scene rarely amounts to a single continuous surface.
Does the continuous surface provided by the sensor mesh then make sense?
We will see that it still has a physical meaning. We will see how to leverage it and what are the obstacles, especially in the case of terrestrial mobile acquisition.

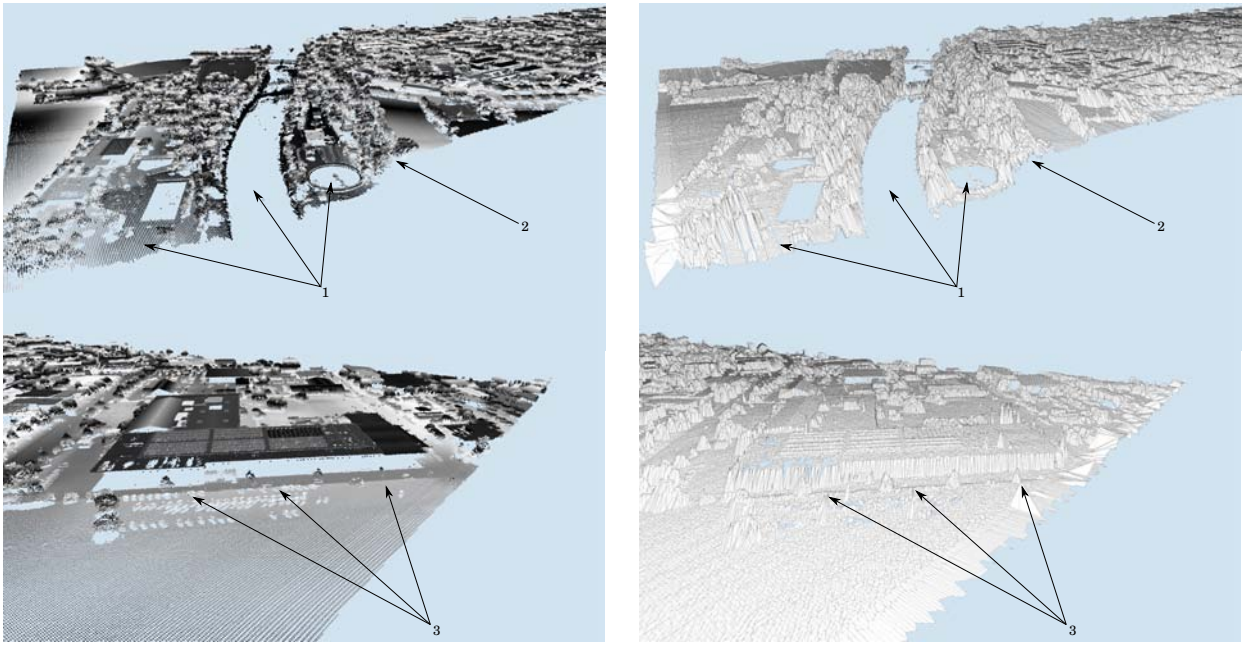
6.2 Sensor mesh: A primitive for surface reconstruction

The interface between what is seen and not seen by the sensor

Finding topological structure of objects from an unorganized point cloud is complex. The most probable surface is often searched, leaving decisions of points linking and holes filling to thresholds or assumptions. In fact, the objects are not sampled from every point of view. The sensor generally sees only one side of the objects, whereas an hidden part remains unknown and impossible to reconstruct. The acquisition result is thus a partial view of the scene. Rather than detecting the whole object volumes, the lidar discovers an empty volume between the sensor and the objects, "carved" by the laser beams. The object surfaces stop the laser beams and bound the sensor visibility area. Lidar echoes are thus located on object surfaces, and more generally at the interface between what is seen and not seen by the sensor. Assuming that this interface is a surface, reconstructible surface pieces (corresponding to the visible part of the objects) are also pieces of this interface. Therefore, we consider this interface and the ways to reconstruct it. This is an image in sensor projection. Points can be simply and rapidly meshed following the neighborhood relationships in sensor geometry. This "naïve" meshing has some shortcomings, but more sophisticated techniques adopting the sensor viewpoint may allow to benefit from sensor geometry.
In this thesis we neglect the noise ($\simeq 1cm$). That is why we only focus on points linking and holes filling.

τ, Θ mesh

When points are projected in the (τ, Θ) space, a mesh structure appears if each point is linked with the next point acquired, and with the point on the next scanline that has the same Θ. We calculated this mesh for the open dataset acquired on Vaihingen ([8] and [9]). In fact, we used a slightly more complex algorithm, because the PCD is first pruned in the (τ, Θ) space, thanks to the method proposed in section 2.5.3. We then link the points of the sub-sampled dataset, according to the 4-neighbor neighborhood. As



(a) Lidar echoes (b) Sensor mesh

Figure 6.8: Some problems appear when modeling the data with the sensor mesh.
(1) Some areas are empty of echoes, because of the water reflectivity (the river and a pool), and the point density is lower at the acquisition border. Should we interpolate the mesh or not?
(2) The trees are really not surface. The echoes altitudes are distributed between ground and treetops and the mesh is crumpled.
(3) One outlier is sufficient to distort the mesh.

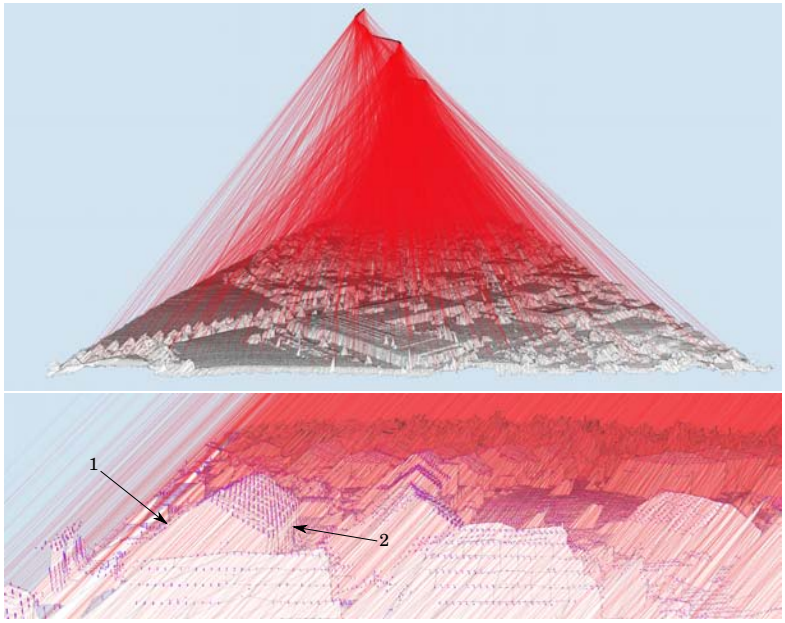


Figure 6.9: The incidence of incidence. Depending on the viewing angle, the house facades are reconstructed differently.
(1) The facade has not been seen and is badly estimated.
(2) Some echoes lie on the facade that can be better estimated.

some points do not have 4 neighbors, there may be holes. The result is shown in the figure 6.7. It allows us to visualize the data projected into the (τ, Θ) space (fig:1.4) The data is not distorted that much, because the plane is high, and trajectory quite straight. This simple and very fast geometric reconstruction has some drawbacks, enumerated in figure 6.8. The reconstruction is very faithful to the data (fig:6.9) there are both pros and cons to this.

6.2.1 Mesh folding and other limitations

Using the sensor projection allows efficient and accurate results, as for the normal calculation [87]. The benefits and challenges of using the acquisition geometry are well shown by Frueh et al. [88]. In particular, they explain the reversal scan order problem: the sensor mesh may fold on itself in (x, y, z) space (fig: 6.10). This is caused by the reversal scan order as shown in fig: 6.11. The curvature of the vehicle trajectory is the source of laser beam crossings that make points out of sequence along τ, whereas the order is always kept along Θ(fig: 6.12). In addition to this order reversal problem, the neighborhood relationships established according to the sensor geometry does not necessarily make sense in 3D space. Some objects are not surface, such as trees, generating scattered points. The windows, both transparent and reflective, induce the detection of several surfaces for the same (τ, Θ) area. In these cases, the 2.5D mesh structure is no longer adequate to represent the data. Mention may also be made of the multi-echoes that share the same (τ, Θ), and of the laser shots oriented toward the sky that generate "no returning" beams. In sum, the nearest neighbors in sensor geometry are not necessarily the most suitable, because the sensor geometry is trajectory dependent (mesh folding), and because the geometry of the underlying objects may be too complex or sub-sampled.

6.2.2 Solutions to mesh folding problem

Despite these drawbacks, it seems interesting to use the sensor coordinate system (τ, Θ, R) that highlights the almost-regular and almost-2.5D data structure. The main hurdle is the absence of bijection due to the possible laser beam crossings.

- Frueh et al. propose in [51] to segment the dataset into parts where they are sure that the reversal scan order cannot happen. For this purpose, the dataset is split according to the trajectory curvature: "By removing these "turn" scans and splitting the path at the "turning points", we obtain path segments with low curvature that can be considered as locally quasi-linear".

- In chapter 7, we propose to use a coordinate system that imitates the sensor coordinate system while ensuring a bijection with the (x, y, z) coordinate system.

Both methods are complementary, The first one offers a smart way to segment the acquisition along the trajectory, while the second can be used when we want to process any path segment, without taking care of the trajectory curvature.
Folding is not necessarily a problem, it can be a solution to the self-registration problem. Indeed, the overlapping surfaces may be matched, for example by comparing the intensity values.

6.3 Conclusion

We demonstrated that in order to reconstruct a continuous surface from lidar signal, meshing echoes that are adjacent in sensor topology (sensor mesh) is an appropriate solution because it naturally integrates logical constraints imposed by the laser beams. However, the geometry of the scanned scene rarely amounts to a single continuous surface.
If the sensor is in front of the surface, one can have confidence in the fact that there is continuity between adjacent echoes, whereas if the sensor sees the surface with a grazing angle, it may exist cavities between two adjacent echoes that contain an hidden portion of the surface and that contradict the continuity between adjacent echoes. The incidence angle thus provides a simple measure of the lidar reliability in detecting a continuous surface.

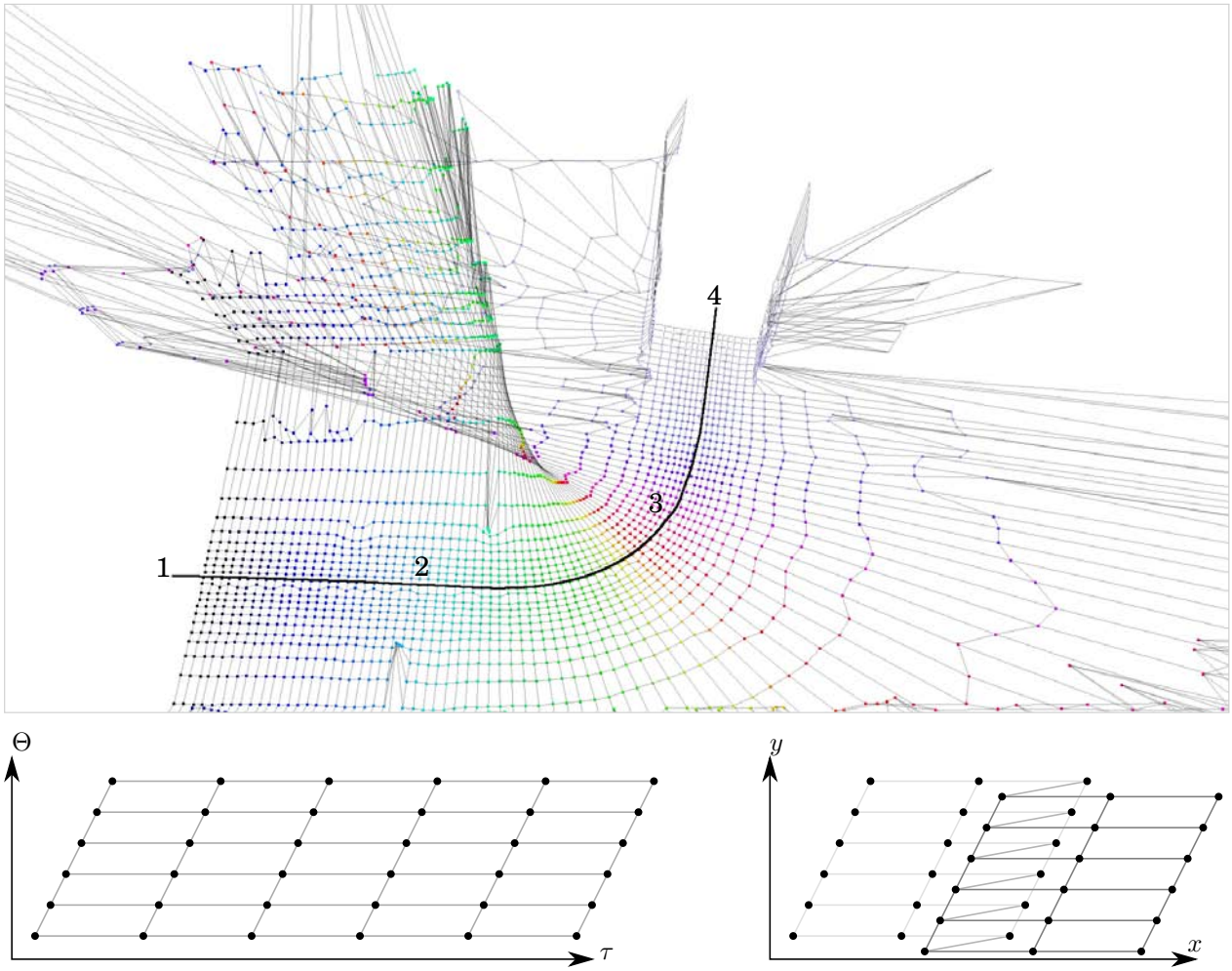


Figure 6.10: The regular mesh in (τ, Θ) space, folding on itself in the (x, y) space.

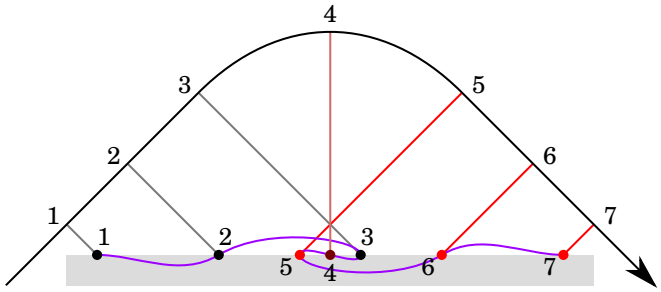


Figure 6.11: Folding of the sensor mesh (violet) due to the trajectory curvature.

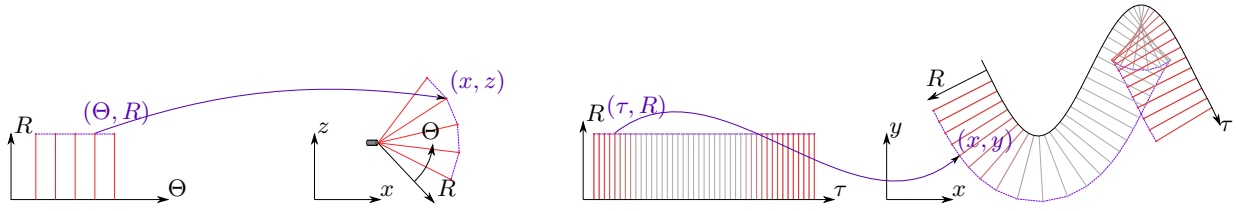
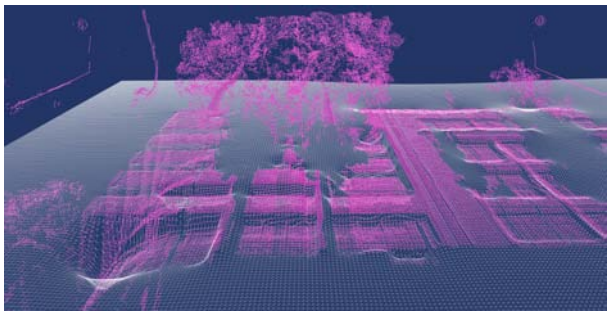


Figure 6.12: Scan order is kept when passing from (Θ, R) to (x, z). This is not the case when passing from (τ, R) to (x, y).

Despite the continuity between adjacent echoes is not one hundred percent reliable, the continuous surface provided by the sensor mesh has always a physical meaning : this is the interface between what is seen and not seen by the sensor. This mesh is therefore interesting to use, but the main obstacle is that it can fold on itself if the laser sweeping turns back.
In this chapter, we proposed some approaches to link the echoes using acquisition knowledges such as sensor location, acquisition time and firing angle. Such information is quickly obtained and the proposed methods are fast and simple to implement. But these are not only low-level methods that optimize the computing time. They involve knowledges that are not contained in PCD. They can thereby provide better results as highlighted by the theoretical relevance of scan order. However, injecting knowledge has a cost: Geometrical objects more complex than 3D points are handled as the "τ, Θ mesh" whose topology is not always consistent with the real 3D topology. Nonetheless, We believe that such approaches should be encouraged, and more generally the use of all available information about the lidar signal and the acquisition configuration. Indeed, the tendency is always to move towards more accurate results. Beside technological performances, this is the inclusion of a maximum of information in the algorithms that allows better results.

Chapter 7

Photorealistic modeling : generalized 2.5D grids



Contents

This chapter is an extension of the work presented in [89].
Reconstructing fine facade geometry from MMS lidar data remains a challenge: in addition to being inherently sparse, the point cloud provided by a single street point of view is necessarily incomplete. We propose a simple framework to estimate the facade surface with a deformable 2.5D grid. Computations are performed in a "sensor-oriented" coordinate system that maximizes consistency with the data. The algorithm allows to retrieve the facade geometry without prior knowledge. It can thus be automatically applied to a large amount of data in spite of the variability of encountered architectural forms. The 2.5D image structure of the output makes it compatible with storage and real-time constraints of immersive navigation.

7.1 Introduction

7.1.1 Motivations

We aim at automatically reconstructing a fine photorealistic facade model (lod3). In this chapter, we focus on the geometrical modeling. The geometry is extracted from lidar data and has to be in accordance with the optical images acquired in the same time and used for texture mapping. Apart from being photo-consistent, the model has also to be sufficiently compact for scaling up.

7.1.2 Facade modeling

Facade Modeling is a vastly studied topic [57]. Numerous articles offer to reconstruct the facades using knowledge based approaches, This subject is developed in chapter 5, but the apparent simplicity of the facade structures, and the underlying grammars is actually complex to automatically detect, to the extent that an operator is deemed necessary to get the desired results [59].

Although such knowledge based approaches are promising, Detecting facade structures and grammars remains difficult, as evidenced by the need for human intervention. In addition, such methods cannot handle old style facades, complex ornaments, and any unexpected object which does not match the model. Maybe hybrid reconstructions (surface/shape) are more appropriate to these contexts and could help the facade structure to emerge [85].

Other methods use more general assumptions like the predominance of right angles. [72] proposes to build a 3D model from images by an arrangements of planes, adding horizontal or vertical planes where there is an absence of information. Although such approaches are adapted to human constructions, objects that do not fit the model cause strange artifacts [73].

Another line of research is the detection of repetitions and symmetries [74]. The repetitions that are not necessarily sequenced according to a grid structure [79] may form a kind of low-level grammar. Detecting repetitions require to deal with the uneven sampling of the ranks scan [80]. Density varies depending on the distance of echoes to the sensor, but also on occluders. Facades are partially hidden, and density decreases toward the top of the buildings. Instead of replacing missing parts by flat or smooth surfaces, some papers offer to "consolidate" and fill the holes thanks to the repetitive structures [81], [82]. But here again, the detection of repetitive elements in PCD is complex and requires manual assistance [84].

We opted for a low level approach, consistent as possible with the data. Indeed, we believe that a processing step is missing in facade modeling. Trying to detect structure into PCD implies to manage the density variation that can both depend on underlying objects and acquisition context (fig:7.1). We want to abstract from such problems thanks to a data modeling more "acquisition independent". This reason oriented us toward a surface reconstruction method. As we will see, many methods have already been proposed but we focus on those that benefit from sensor informations.

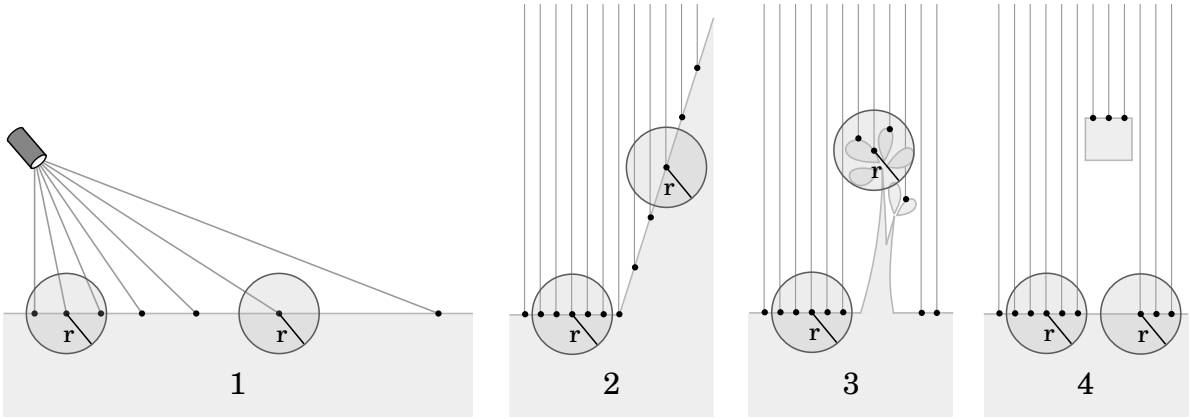


Figure 7.1: Causes of density variation. Point density depends on the sensor distance (1), and this distance depends on the surface geometry (2). Density is also decreased in porous areas (3) or by occluders (4).

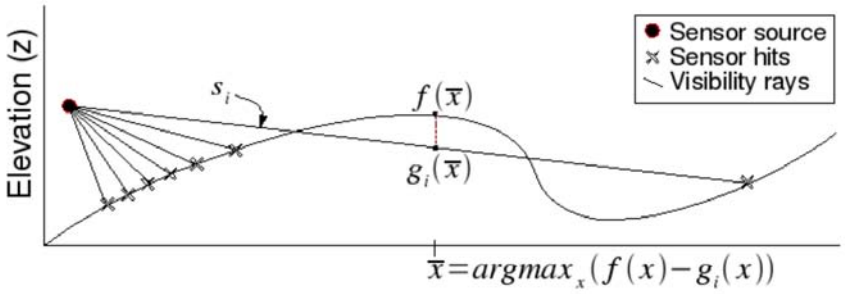


Figure 7.2: Graph from [94]. The Lidar beams are used in rough estimation.

7.2 Surface reconstruction

7.2.1 Computer vision

Even if many interesting works about surface reconstruction from unorganized PCD still occur ([25], [90]), this is a widely studied topic and a lot of solution have already been proposed. A sate-of-the-art can be found in [91]. In the field of computer vision, the studied PCD often contain a well sampled handmade object, as the famous stanford bunny. The problems of objects mixing and sub-sampled objects are not tackled, the objective is to retrieve the object surface, despite a complex topology, noise, outliers and holes. A major issue seems to delete the noise while preserving the object features. For this purpose, a new approach is developed in [92]: Digne et al. try to minimize the displacement of each point to the reconstructed model. Our context and the technical obstacles are not exactly the same. first because we work with data acquired from an uncontrolled environment, inducing occlusions, sub-sampling, density variation and object mixing. Second because we think that we can benefit from the knowledge of acquisition context (sensor location, acquisition time of each point...).
An original approach [93] proposes to sketch the surface under the following assumption: the surface is very smooth or even linear according to one direction. The surface is then reconstructed by drawing the same profile along this direction.

7.2.2 Carving oriented approaches

In MMS data, the problem of sparse data is complex. Whether to detect holes in continuous mesh [4], or to fill its in a holed mesh [5]. In fact, the underlying problem is that the topology can be arbitrarily complex under the sub-sampled PCD. The carving approach brings a solution to this topological problem: The volume preexists and is modified as and when. Carving approach was initially proposed to reconstruct 3D

from images [95], but the same idea can be applied to lidar data. Even if the sensor location is unknown [96], it is possible to constrain the surface reconstruction by estimating the sensor viewing angle. Obviously, if the sensor location is known, it is even easier to carve the scene with the beams. The major problem is the memory space requisite to voxelize the 3D space. In [94], a surface is first estimated, and then deformed to ensure the consistency with the lidar beams (fig:7.2). Using a surface, the main carving shortcoming is bypassed, which is the amount of memory required to represent the volume of the scene into voxels.

7.2.3 Sensor perspective

Carving methods benefit from the knowledge of the areas explored by the sensor (the carving beams). One can also adopt the sensor point of view. Namely, one place itself in the coordinate system corresponding to the sensor perspective. Using such a coordinate system allows efficient and accurate results, as for the calculation of normal [87]. Optical images are written in the sensor coordinate system. By comparing, optical images and lidar data, we will try to evidence the interests of working in such sensor coordinate system.

7.2.4 Lidar vs image

If lidar data and optical image are compared for 3D surface reconstruction, at first glance, lidar data seems more suitable: the distance to the reflecting objects is directly measured while the optical image information is radiometric. Photogrammetric techniques allow to recover the geometry if problems of radiometry and distortion are overcome. Photogrammetry involves complex algorithms to determine the relative positions between the images and the sensor, and to match keypoints. However, some advantages over the Lidar systems are listed by [97] in a kind of *Tablets of Stone* of the photogrammetry supremacy (see in annex). Most of these "advantages" don't lie in the nature of the systems, but in the seniority of the technologies associated to photography and their intensive use

Vision maturity. Photographs are ubiquitous in our lives. This results in low-cost devices, light and powerful. Similarly in software, algorithms for image processing have acquired maturity and algorithms such as SIFT, JPEG are very accomplished and optimized. Thus, the solutions are numerous and accessible to acquire and efficiently handle a large amount of images. Comparatively, Lidar point cloud data type are little known to the general public and less used. Anyway, this is a strong technical and economical advantage so that there is a comeback of photogrammetry [98], [99].

A Question of Methodology? An interesting point is that many of the 16 advantages of [97] are about methodology. To our opinion, the way the signal is processed is very important and can strongly enhance the results. The proposed "deformable grid" is inspired by the comparison between lidar and image data structures. Indeed, we regretted the PCD lack of structure, and especially the loss of connectivity between echoes.

7.2.5 Depth images

As explained in chapter 1 the connectivity between lidar echoes is more complex to draw than the 4-neighbors connectivity in images. However the sensor connectivity also exists in PCD as shown in chapter 6 and allows to reconstruct a surface mesh that links the lidar echoes. This surface mesh is comparable to a depth image in sensor topology. A depth image is associated to a coordinate system (u, v, h). u and v locate the pixel in the image, and h is the distance to the sensor. More precisely, u and v locate the pixel along a half-line (iso-uv) starting from the sensor. In optical images, all the iso-uv start from the same point and there is a bijection between (u, v, h) and (x, y, z). One can see that any function that associates h with every pair u, v is a non self-intersecting surface in (u, v, h) (an example in 2D, figure 7.3). Under some assumptions, in particular, if there is a bijection between and (u, v, h) and (x, y, z), this surface is also a non self-intersecting surface in (x, y, z). For example, a depth image (width $\times$ height) is a function which associates h with every pair u, v such as $0 \leq u <$ width and $0 \leq v <$ height. In lidar data, such a surface would be interpolated from the echoes according to the echo depth values (fig 7.4) but, as explained in chapter 6 (where $(u, v, h) \Leftrightarrow (\tau, \Theta, R)$), the sensor displacement during the acquisition disorganizes the laser beams (iso-uv) that can intersect each other. A bijection is therefore not guaranteed between (u, v, h) and (x, y, z) which may induce auto-intersections in the surface (Mesh folding in in chapter 6). The sensor

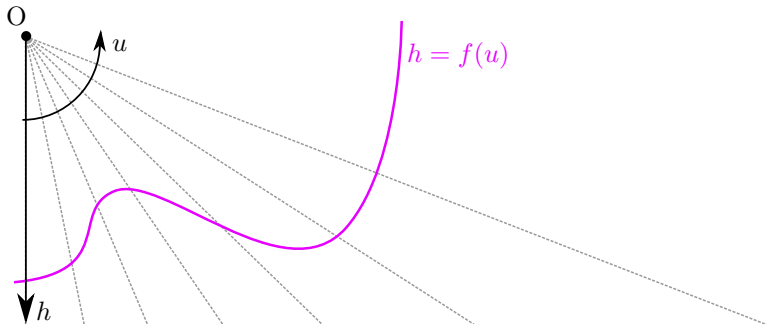


Figure 7.3: A non self-intersecting line (pink) is a function that maps u to h. u is an angular coordinate, and h is the distance from O.

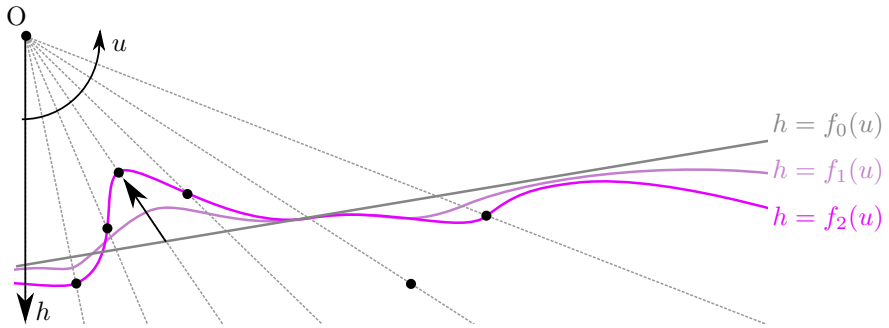


Figure 7.4: Interpolations between the lidar echoes. Three functions f_0, f_1 and f_3 that maps u to d are also some possible interpolations between the lidar echoes (black dots).

coordinate system cannot be used as is to generate a surface, but many of its interesting properties have been highlighted in chapter 6. Our idea is to use a coordinate system that aligns as far as possible the iso-uv with the laser beams, but that provides a bijection with (x, y, z). This way, a 2.5D surface could be generated in such coordinate system, approximating the sensor mesh, but avoiding topological problems.
The facade as a depth image. In this chapter, we wish to reconstruct the facade surface. This surface is assumed to be a 2.5D surface with depth variations from a plane that approximates the main vertical wall. This prompts us to build the coordinate system as follows: u and v refer to the pixel location along the plane and the iso-uv are oriented according to the laser beams. Such coordinates systems are proposed in section 7.4, after an overview of the algorithm.

7.3 Overview

A 2.5D grid is fitted to the geometrical primitives. In the 2.5D grid coordinate system, any 3D point is defined by u and v that locate the point on the grid, and h that is the depth from the u-v plane. The main steps and the inputs of the algorithm are depicted thereafter and in figure 7.5.
Initialization. A vertical rectangle approximating the main facade wall is calculated as in [43]. It provides the u-v plane and the grid boundaries. The 2.5D grid is placed on this rectangle and the depth of each grid pixel is set to zero.
Primitive accumulation in the grid pixels. The primitives (lidar points, triangles or line segments) are projected onto the u-v plane in order to accumulate the primitive depths h into the grid pixels: for any kind of primitive, whenever the intersection between a primitive and a pixel is not empty, the primitive depth is accumulated in the pixel.
Iterative deformation of the grid. The grid is then iteratively deformed to draw near the primitives. An optimal h is searched for each pixel according to the depths of the primitives that are projected in, and according the neighboring pixel depths. Only the h coordinates are changed, the pixels therefore move along their iso-uv. This process is performed several times in order to converge to a satisfactory solution, despite bad initializations, outliers and empty pixels. The strategy for pixel depth computation serves to

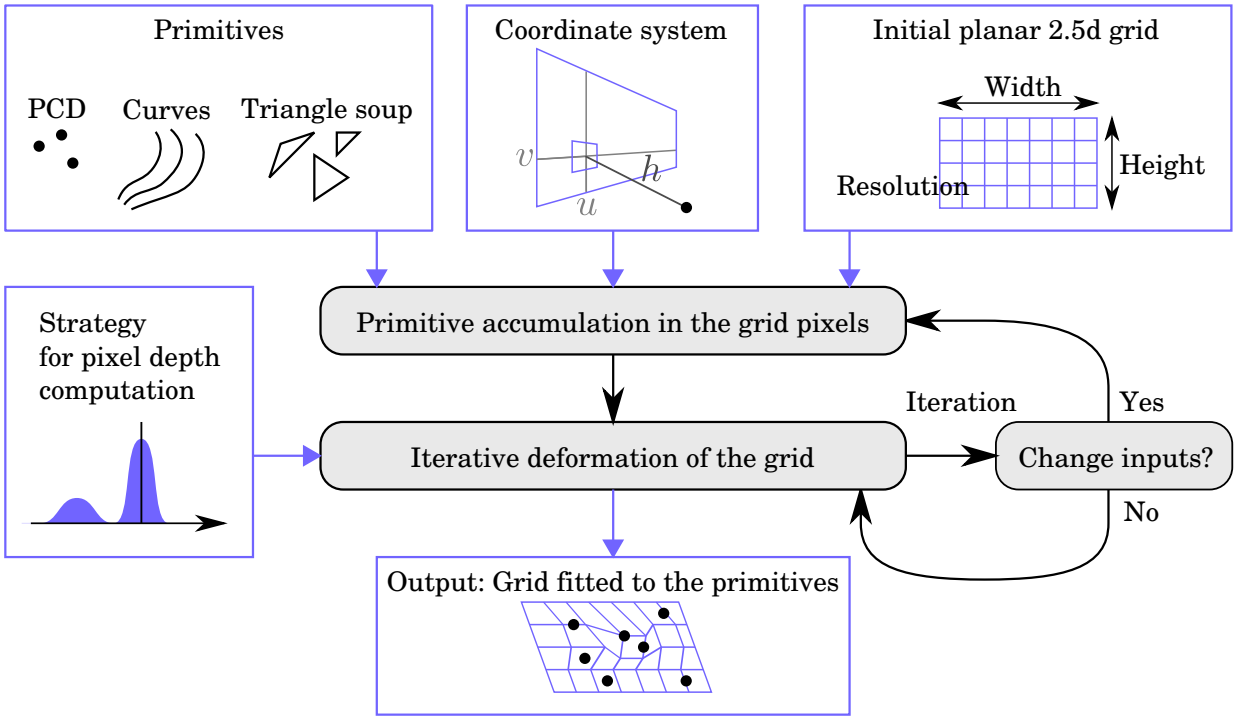


Figure 7.5: Workflow of grid fitting to the primitives.

handle cases where many primitives have been accumulated in one pixel. The depth interpolation for empty pixel is made locally, according to a smooth parameter explained in section 7.5. Each input is independent from the others. As well, the strategy for pixel depth computation can be chosen separately, however if the grid resolution or some other parameter is changed between two iterations, the primitive accumulation must be performed again.

7.4 Coordinate systems

The best way to accumulate the primitives in the grid pixels is to project the primitives onto the u-v plane, following the laser beam directions. Unfortunately, the beams can intersect together and do not guarantee a unique solution for the point projection. That is why the cylindrical coordinate system and the induced projection (sec:7.4.1) and the prismatic coordinate system (sec:7.4.2) are proposed to accumulate the primitives. In section 7.4.3, the reasons that led us to develop these coordinate system are explained by means of examples. The choice of the 2.5D grid coordinate system and the induced projection is important because it affects the primitives accumulation and it determines the direction of the pixel displacements (orthogonal to the u-v plane if Cartesian coordinate system). Our first intuition was to use a Cartesian coordinate system. This favors perpendicular angles, which are prevalent in human buildings. However, it causes problems that could be avoided using the sensor coordinate system, described in the chapter that corresponds to our acquisition configuration (a rotating laser, scanning vertically and perpendicular to the trajectory). But, as we explained in the chapter , the main limitation when using the sensor coordinate system is the absence of bijection between x, y, z and u, v, h. Therefore, we developed two "sensor-oriented" coordinate system that ensure the bijection while remaining close to the sensor coordinate system.

The **"cylindrical" coordinate system** is the most intuitive approximation of the sensor coordinate system. that is why it is detailed below, in a didactic way, but the second coordinate system is more suitable for facade reconstruction.

This is a generalization of the Cartesian coordinate system called **"prismatic" coordinate system**. Instead of being orthogonal to the plane, the iso-uv are oriented toward the sensor location.

Coordinate	Definition	Highly correlated with
t	Projection of P on $\overrightarrow{t}$ axis.	S_τ
r	Distance btw $\overrightarrow{t}$ axis and P.	R
θ	Angle of rotation around $\overrightarrow{t}$ axis	Θ

Table 7.1: Relation between sensor and cylindrical coordinates

7.4.1 Cylindrical coordinate system

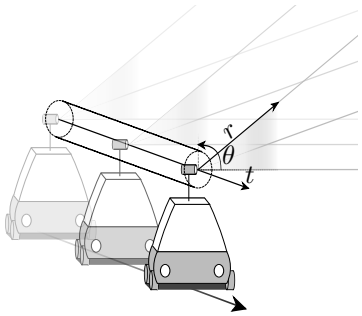


Figure 7.6: Cylindrical coordinate system (t, r, θ).

If the laser sweep is vertical and the vehicle trajectory is quite straight, a cylindrical coordinate system is a good approximation of the sensor coordinate system. It maintains as best possible the separation between terms of sensor equation (see in chapter 1.1) that have different accuracies and different point distribution. Table 7.1 details correlations between these terms and the cylindrical coordinates. For instance, r and θ are close to R and Θ, but R and Θ are expressed in a basis that depends on time (sensor position and orientation are moving), while a bijection exists between (x, y, z) and (t, r, θ). This way, every point of the 3D space can be expressed in the same cylindrical coordinate system.

Computation

Estimate the trajectory axis. The trajectory, is approximated by a straight line. To obtain this trajectory axis, a PCA is applied to the trajectory points (the sensor location of each point) in order to retrieve the PCA principal direction. Be aware that the trajectory points are very close and very redundant. It is slower and unnecessary to use all points for the calculation. But it can also cause a loss of precision. For this, it is preferable to apply the PCA on a percentage of points. The unit vector of the trajectory axis is denoted by $\overrightarrow{t}$.

Compute the $\left\{\overrightarrow{t}, \overrightarrow{u}, \overrightarrow{v}\right\}$ basis. Let $\overrightarrow{v}$ be a unit vector of the vertical axis, and $\overrightarrow{u}$ such as $\overrightarrow{t} \times \overrightarrow{v} = \overrightarrow{u}$. The set of vectors $\left\{\overrightarrow{t}, \overrightarrow{u}, \overrightarrow{v}\right\}$ forms an orthonormal basis of $\mathbb{R}^3$. Every point (x, y, z) can be expressed in this basis.

Convert x, y, z into cylindrical coordinates. Then, u and v can be converted to the polar coordinates r, θ such as $r = \sqrt{u^2 + v^2}$ and $\theta = atan2(v, u)$.

The cylindrical coordinate system is the one that most resembles the sensor coordinate system. It would be the equivalent for a perfectly straight vehicle trajectory. However, we preferred to use a more appropriate coordinate system. The prismatic coordinate system is actually a gradual adaptation of the cylindrical coordinate system to our problem. We went from polar to Cartesian coordinate. Then we looked for a suitable system when the laser beams are not aligned but converge or diverge because of trajectory curvatures. We have had the idea of this prism that fits the average behavior of the beams between the trajectory and the facade.

7.4.2 Prismatic coordinate system

The prismatic coordinate system is a generalization of the Cartesian coordinate system that aligns the iso-uv along the laser beams. We propose a new way to accumulate the points in the facade grid. The idea

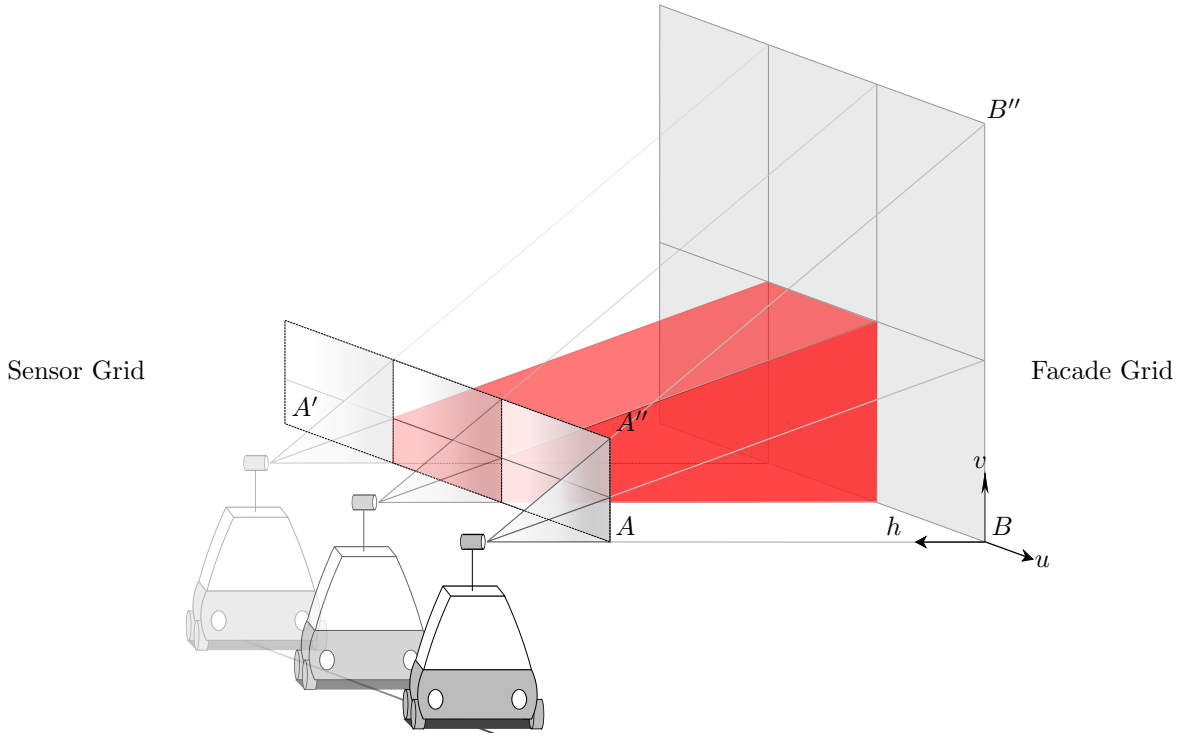


Figure 7.7: The points are accumulated in the pixels of the facade grid. The accumulation volume of a given pixel is shown in red. It extends to infinity behind the grid.

is to use a virtual sensor grid. For each pixel of the facade grid, there is a corresponding pixel on the sensor grid. Ideally, the following property is ensured: any laser beam passing through a facade pixel passes through the corresponding sensor pixel (fig: 7.7). The virtual sensor grid is positioned in order to approach this ideal.
The function that associates prismatic coordinates (u, v, h) to the space coordinates (x, y, z) is explained in the 2D and 3D cases. Then a method to place the grid is proposed.

2D formulation.

We place ourselves in the horizontal plane (fig:7.8). We look for the (u, h) coordinates of a point $P(x, y)$. $[AA']$ symbolizes the sensor location, and $[BB']$ the facade footprint. The laser beam is approximated by a ray $[ab)$ that passes through P and intersects (AA') in a and (BB') in b, such that

$$\overrightarrow{Aa} = u\overrightarrow{AA'} \quad \text{and} \quad \overrightarrow{Bb} = u\overrightarrow{BB'} \tag{7.1}$$

The straight line (ab) is thus the iso-u of P.
The position of P on $[ab)$ is given by h. We propose to use the following equation that sets h as the signed distance to b normalized by $|ab|$, b being the virtual intersection of the facade plane and the laser beam.

$$h = \overrightarrow{bP}.\frac{\overrightarrow{ba}}{|ba|^2} \tag{7.2}$$

We will show how to find u for a given point P in order to obtain the couple (u, h). In this context, P, A, A', B and B' are known. As P lies on (ab), we have: $\overrightarrow{aP} \times \overrightarrow{ab} = 0$ We will rewrite this equality to show up

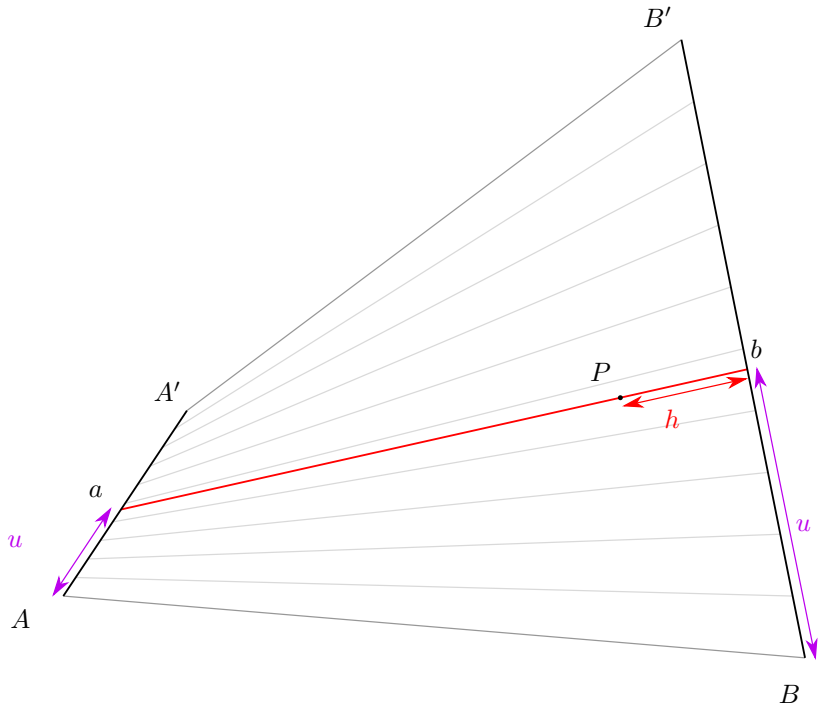


Figure 7.8: 2D formulation

u and prove that it is equivalent to:

$$u^2(\overrightarrow{BB'} \times \overrightarrow{AA'}) + u(\overrightarrow{AP} \times \overrightarrow{BB'} - \overrightarrow{BP} \times \overrightarrow{AA'}) + \overrightarrow{AP} \times \overrightarrow{AB} = 0$$

As cross product is distributive, we have

$\overrightarrow{aP} \times \overrightarrow{ab} = (\overrightarrow{aA} + \overrightarrow{AP}) \times (\overrightarrow{aA} + \overrightarrow{AB} + \overrightarrow{Bb})$

$\overrightarrow{aP} \times \overrightarrow{ab} = \overrightarrow{aA} \times (\overrightarrow{aA} + \overrightarrow{AB} + \overrightarrow{Bb}) + \overrightarrow{AP} \times (\overrightarrow{aA} + \overrightarrow{AB} + \overrightarrow{Bb})$

$\overrightarrow{aP} \times \overrightarrow{ab} = \overrightarrow{aA} \times \overrightarrow{aA} + \overrightarrow{aA} \times \overrightarrow{AB} + \overrightarrow{aA} \times \overrightarrow{Bb} + \overrightarrow{AP} \times \overrightarrow{aA} + \overrightarrow{AP} \times \overrightarrow{AB} + \overrightarrow{AP} \times \overrightarrow{Bb}$

We focus on each underbraced term.

$\overrightarrow{aP} \times \overrightarrow{ab} = \underbrace{\overrightarrow{aA} \times \overrightarrow{aA}}_{(1)} + \underbrace{\overrightarrow{aA} \times \overrightarrow{AB}}_{(2)} + \underbrace{\overrightarrow{aA} \times \overrightarrow{Bb}}_{(3)} + \underbrace{\overrightarrow{AP} \times \overrightarrow{aA}}_{(4)} + \overrightarrow{AP} \times \overrightarrow{AB} + \underbrace{\overrightarrow{AP} \times \overrightarrow{Bb}}_{(5)}$

(1) $\overrightarrow{aA} \times \overrightarrow{aA} = 0$

By remembering that $\overrightarrow{Aa} = u\overrightarrow{AA'}$ and $\overrightarrow{Bb} = u\overrightarrow{BB'}$ and that the cross product is anticommutative, we get :

(5) $\overrightarrow{AP} \times \overrightarrow{Bb} = u\overrightarrow{AP} \times \overrightarrow{BB'}$

(3) $\overrightarrow{aA} \times \overrightarrow{Bb} = \overrightarrow{Bb} \times \overrightarrow{Aa} = u^2\overrightarrow{BB'} \times \overrightarrow{AA'}$

(2+4) $\overrightarrow{aA} \times \overrightarrow{AB} + \overrightarrow{AP} \times \overrightarrow{aA} = (\overrightarrow{BA} + \overrightarrow{AP}) \times \overrightarrow{aA} = \overrightarrow{BP} \times \overrightarrow{aA} = -\overrightarrow{BP} \times \overrightarrow{Aa} = -u\overrightarrow{BP} \times \overrightarrow{AA'}$
which is equivalent to:

$$u^2(\overrightarrow{BB'} \times \overrightarrow{AA'}) + u(\overrightarrow{AP} \times \overrightarrow{BB'} - \overrightarrow{BP} \times \overrightarrow{AA'}) + \overrightarrow{AP} \times \overrightarrow{AB} = 0 \tag{7.3}$$

So we obtain a quadratic equation of the form $\alpha u^2 + \beta\, u + \gamma = 0$ that allows two solutions for u. Note that if $\overrightarrow{AA'}$ and $\overrightarrow{BB'}$ are collinear, $\alpha = \overrightarrow{BB'} \times \overrightarrow{AA'} = 0$, and the equation becomes

$\beta u + \gamma = 0 \Leftrightarrow u = -\frac{\gamma}{\beta}$, giving us the unique solution for u. In the general case ($\overrightarrow{AA'}$ and $\overrightarrow{BB'}$ not collinear), one solution corresponds to a degenerate case, that brings a and b far from P. One can verify that, defining $q = -\frac{\beta + sign(\beta)\sqrt{\Delta}}{2}$ and $\Delta = \beta^2 - 4\alpha\gamma$, the two roots of the quadratic equation are $u_d = \frac{q}{\alpha}$ and $u_s = \frac{\gamma}{q}$.
As $\lim_{\alpha \to 0} u_d = \pm\infty$ and $\lim_{\alpha \to 0} u_s = -\frac{\gamma}{\beta}$. u_d is a degenerate solution and u_s, the searched value for u.

Special cases.

If AA' and BB' are collinear, and $||AA'|| = ||BB'||$, b is the orthogonal projection of P on (BB'), this is then a Cartesian coordinate system.

If $|AA'| \to 0$, we obtain a kind of angular coordinate system.

3D formulation.

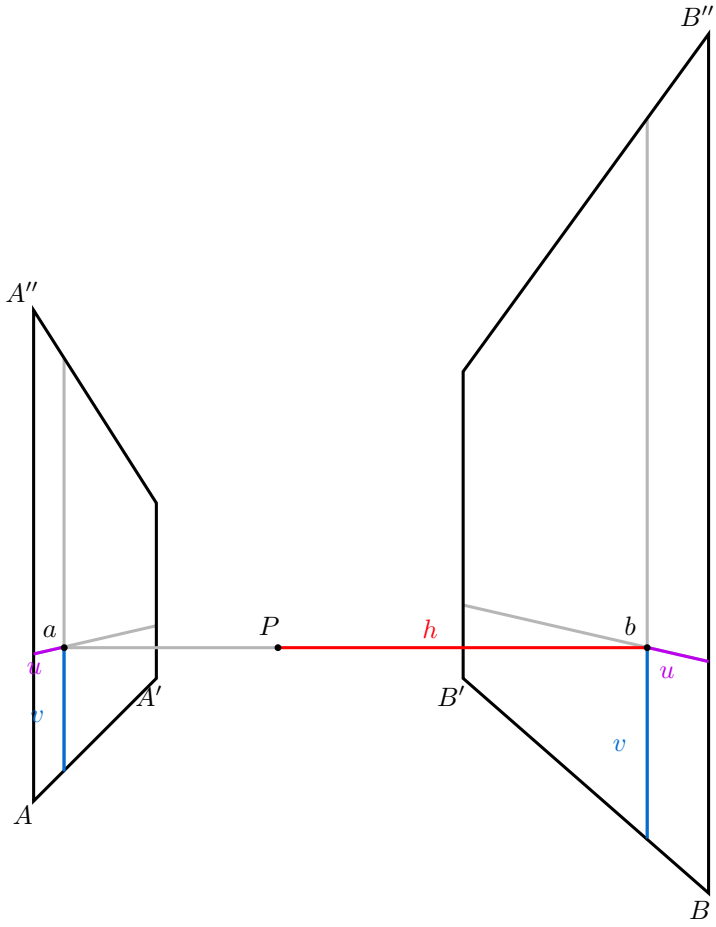


Figure 7.9: 3D formulation

As in figure 7.9, $\overrightarrow{AA'} \perp \overrightarrow{AA''}$ and $\overrightarrow{BB'} \perp \overrightarrow{BB''}$. For our application, we assume that $\overrightarrow{AA''}$ and $\overrightarrow{BB''}$ are collinear.
u and v are the horizontal and vertical coordinates of P in both the basis $\{AA', AA''\}$ and $\{BB', BB''\}$. h is the signed distance to b.
$\overrightarrow{Aa} = u\overrightarrow{AA'} + v\overrightarrow{AA''}$

$$\overrightarrow{Bb} = u\overrightarrow{BB'} + v\overrightarrow{BB''} \qquad \text{and} \qquad h = \overrightarrow{bP}.\frac{\overrightarrow{ba}}{|ba|^2}$$

Placing the grids.

The rectangle with the B, B' and B'' corners is placed along the facade wall. The rectangle with the A, A' and A'' corners is placed along the trajectory in order to align laser beams with iso-uv. For this purpose, we only adapt the horizontal position of the rectangle. It is vertical with an height close to zero because the location of the sensor is not supposed to move vertically. $\overrightarrow{AA'}$ is fitted with RANSAC, by selecting randomly pairs of laser beams. Each pair of laser beams is a potential couple $AB, A'B'$ defining a prismatic coordinate system. The score of each pair must reflect the alignment between the laser beams and the iso-uv. The score is thus the sum of the dot products between the laser beam and the iso-uv of each 3D point.

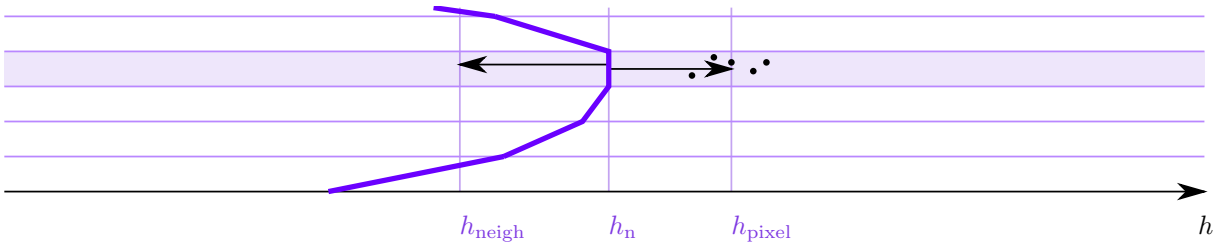
7.4.3 Comparison between Cartesian and prismatic coordinate systems

The orthogonal and prismatic coordinate systems are compared through some examples in figures 7.10, 7.11, 7.12 and 7.13. Many problems in pixel accumulation and surface reconstruction can be avoided using a "sensor-oriented" coordinate system such as the prismatic one.

7.5 Deformable grid

7.5.1 Iterative grid deformation

We solve the problem of finding an optimal depth h for each grid pixel as an energy minimization with a data term that shifts h to the primitives, and a smoothness term that retains h close to the neighboring pixels. Each iteration, the depth of each pixel is moved from h_{n} to $h_{\text{n+1}}$.



$$h_{\text{n+1}} = h_{\text{n}} + \underbrace{\lambda_{\text{pixel}}(h_{\text{pixel}} - h_{\text{n}})}_{\text{Data Term}} + \underbrace{\lambda_{\text{neigh}}(h_{\text{neigh}} - h_{\text{n}})}_{\text{Smoothness Term}} \tag{7.4}$$

h_{n} : previous h
h_{pixel} : h estimated from primitives accumulated in the pixel.
h_{neigh} : h estimated from the neighboring pixel depths (at n).
As the h_{neigh} values are taken into account in the calculation of $h_{\text{n+1}}$, and as we want to avoid that the pixel browsing order modifies the result, The grid is browsed two times: firstly, each depth is computed according to the h_{neigh} values at n, and stored in a temporary variable h_{tmp}, secondly, each depth is set to h_{tmp}.

7.5.2 Strategy for pixel depth computation

The strategy for h_{pixel} and h_{neigh} calculation may vary according to the data or the desired output. In our case, we use a **bilateral filter** that allows discontinuities like at window edges, but smooths the wall flat surfaces. Pixel depth is computed thanks to the following equations, where $\exp(-\frac{x^2}{2\sigma^2})$ is denoted by $G(\sigma, x)$.

$$h_{\text{pixel}} = \frac{\sum G(\sigma_h, h_i) h_i}{\sum G(\sigma_h, h_i)} \tag{7.5}$$

Where h_i is the depth value of the i^{th} primitive accumulated in the pixel. The Gaussian function weights the contribution of each h_i. If an h_i is far from the current depth, its contribution is low. This allows a relative independence to the outliers. The smoothness term is provided by the depth values of the eight neighbor pixels. Hence, the neighbors depth is given by:

$$h_{\text{neigh}} = \frac{\sum_{i=1}^{8} G(\sigma_h, h_i) G(\sigma_d, d_i) h_i}{\sum_{i=1}^{8} G(\sigma_h, h_i) G(\sigma_d, d_i)} \tag{7.6}$$

h_i is the h_{pixel} value of the i^{th} neighbor pixel. d_i is the distance between the pixel center (u, v) and the i^{th} neighbor pixel center (u_i, v_i). If the depth of a neighbor pixel is far from the current pixel depth, its contribution is low, allowing a discontinuity at the edge between these pixels. The values of σ_h and σ_d have to be fixed, one can choose the grid accuracy that is, the length of a pixel side.
Some other strategies are possible. For instance, the closest point from the facade plane can be kept. The median would give a robust estimation if there are many point. Other quantile values would be used if the closest or furthest points are looked for.

7.5.3 Primitives

Points / Lidar echos:

If many pixels are empty of 3D points, the iterative process may be slow to converge. A solution is to start with a low resolution grid (large pixels). This allows to initialize a more precise grid. This procedure can be repeated many times, the same way a scale-space pyramid is built.

Triangles:

The advantage of the triangles over the points is that they provide surfaces. This allows the algorithm to converge more rapidly. Thus, the triangle soup provided by sensor geometry can be used as input.

Line segments:

Line segments extracted from scan lines can be accumulated in grid pixels. This option is mentioned because many papers perform processes such as classification on scanlines [100]. simplified scanlines are more compact than points and their projections could intersect more pixels.

Mixed primitives:

Nothing prevents to accumulate heterogeneous primitives.

7.6 Results

In order to assess the results, we observed the median distance of the points from the grid, and more generally, the percentage of points that lie within a certain distance from the grid (fig:7.14). The smooth term λ_{neigh} controls the grid deformation and allows to move more or less close to the points. The

advantage of a smooth surface contrary to a surface close to the points is its simplicity. We would like to measure the surface simplicity. We realized that the simplest surface: the plane has also the lower area. The surface complexity increases with the area. In figure 7.15 a 2D graph position different grids according to the grid area (abscissa) the median point-surface distance (ordinate). The "best" grid (close to the points and simple) would lie at bottom left while the "worst" grids (far to the points and complex) would lie at top right. For this purpose, the surface area is computed We display some visual results, showing the precision of the proposed method, despite the low point density (fig: 7.16). Parameter adjustments permits a compromise between a smooth surface as in figure 7.17 (a), and more rugged and closer to the points, 7.17 (c). Some other results are displayed in figures 7.18 and 7.19

7.7 Conclusion

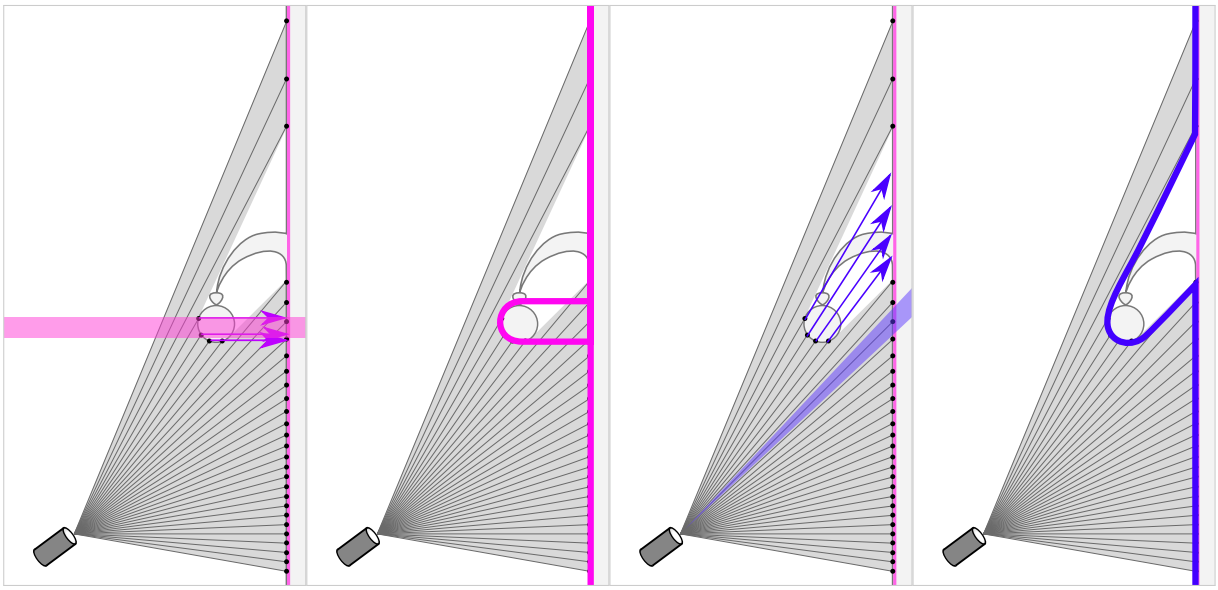
We presented an approach to reconstruct facade geometry from lidar data. We aimed at surfaces consistent with the data, especially the optical images acquired in the same time. The proposed framework allows to reconstruct 2.5D grids in a coordinate system adapted to the sensor point of view. For this purpose, we introduced a prismatic coordinate system and the induced projection that is a generalization of the Cartesian coordinate system the induced orthogonal projection. It allows to benefit from the sensor geometry while providing a coordinate system topologically consistent with the 3D space. The output 2.5D grids are suitable for immersive navigation and other applications that need compact and detailed 3D models. They could also be used as inputs for facade structure analysis, in particular for window detection or grammar rules extraction.
Lidar echoes all belong to a quasi-surface that is the interface between what is seen and not seen by the sensor. This is not a surface because of the multi- echo phenomenon, moreover, all what is detected by the lidar is not a continuous surface (foliage, semi-reflective windows...). However, if a continuous surface is sampled, it is a part of this interface. These observations provide strong incentives to reconstruct surfaces according to the sensor meshing. However, it one has to solve the problem that the sensor mesh can fold on itself. Our intuition about this that the main question is : **How to move from the sensor coordinate system τ, Θ, R to the 3D space coordinate system x, y, z?** In this chapter we propose an intermediary coordinate system (prismatic) which combines advantages of both coordinate systems, considering that whatever the coordinate system in which are expressed in points, the information given by the position of the echo is always true. We can therefore choose any coordinate system.
Other solutions are undoubtedly possible. For example working in 6D space τ, Θ, R, x, y, z to perform an initial segmentation into almost flat regions, then compare these regions together to manage the problem of folding surfaces and finally merge the regions corresponding to a same surface.
In the in the next chapter, we would like to texture the grid with optical images. For this task, our framework should allow us to deal with images (that are also 2.5D grids in image coordinate system) and to project image pixels onto the grid.

7.8 Future works

This grid framework could be used with other data types and for other applications.

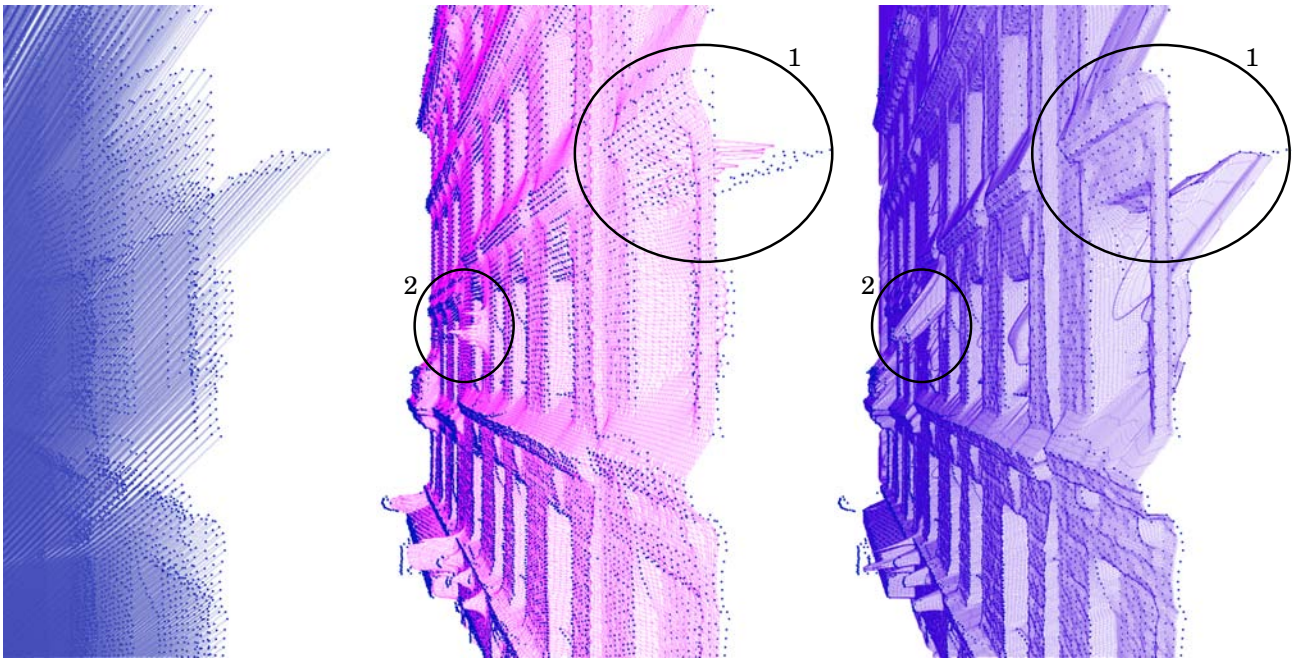
Digital elevation model and digital terrain model.

Fullwaveform. We investigate the way to recover the surface from fullwaveform data. The results (fig:7.20) are promising.



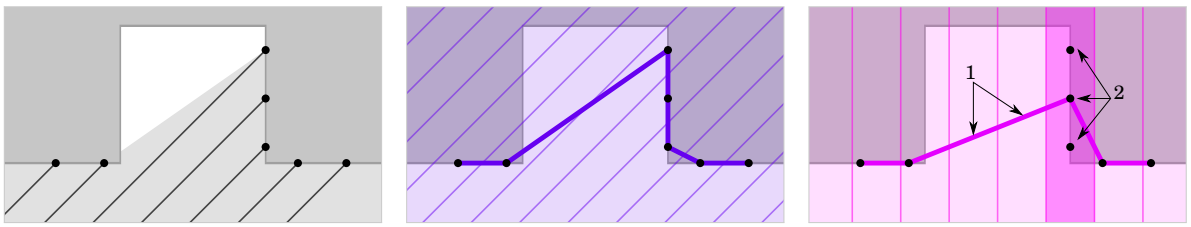
(a) (b) (c) (d)

Figure 7.10: The facade in profile and a wall light are sketched. The points are accumulated on the vertical plane, according to a Cartesian coordinate system (a), (b) and a prismatic coordinate system (c), (d). Cartesian coordinate system leads to accumulating points from distant beams in a same pixel (a), causing an ambiguity in the choice of the surface (b). prismatic coordinate system is more consistent with the scan (c) and leads to a surface reconstruction with less ambiguity (d).



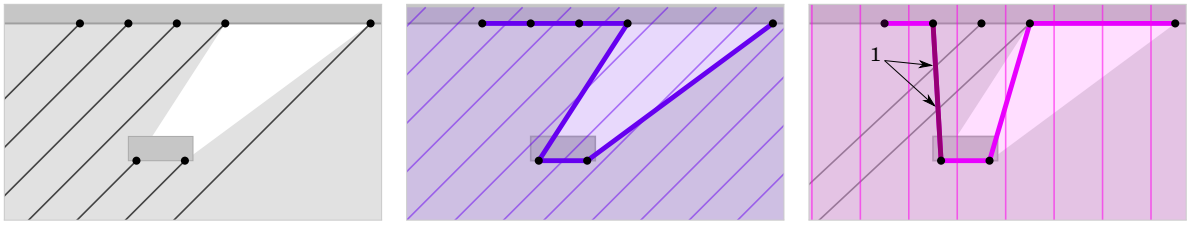
(a) (b) (c)

Figure 7.11: Laser beams (a). Surface reconstructed in a Cartesian coordinate system (b) and a prismatic coordinate system (c). The use of Cartesian coordinate system may leads to a surface that "goes through" the wall because it is attracted by points of the room ceiling behind the facade (1). The surface is also attracted to a wall light (2). The surface reconstructed using the prismatic coordinate system follows the beam trajectory, it goes trough the window (1) and wraps the wall light (2).



(a) (b) (c)

Figure 7.12: **Loss of Accuracy**
(a) Laser beams partially entering a cavity.
(b) The iso-uv (purple) are along the laser beams, points are distributed homogeneously in the pixels. This is the best configuration for surface reconstruction: there is one point per pixel, there is no ambiguity in depth computation.
(c) Cartesian coordinate system, iso-uv (pink) orthogonal to the wall. Three points are accumulated in the same pixel (2) while there is no point in some other pixels (1).



(a) (b) (c)

Figure 7.13: **Inconsistency between the surface and the laser beams**
(a) An object hides a part of the wall behind.
(b) Prismatic coordinate system: The reconstructed surface does not fit everywhere with the actual surfaces, but is not pierced by the laser beams.
(c) Cartesian coordinate system: The reconstructed surface is intersected by two laser beams (1).

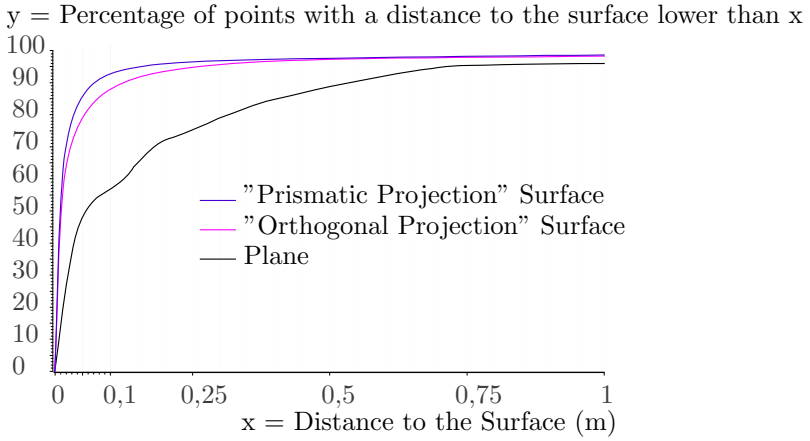


Figure 7.14: This graph shows the percentage of points that lie within a certain distance from an estimated surface. The results are compared for three surfaces: a plane, and two deformable grids calculated with the same parameters, except for the coordinate system used to accumulate the points. The points are closer to the "prismatic coordinate system" grid and the "Cartesian coordinate system" grid and finally the plane.

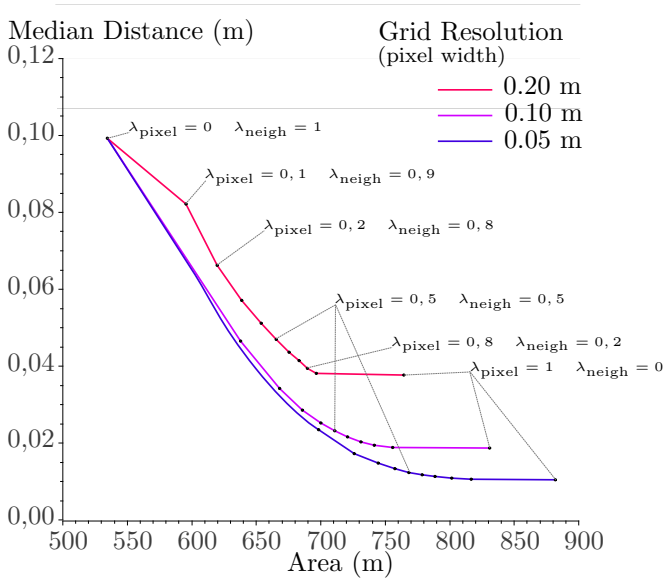
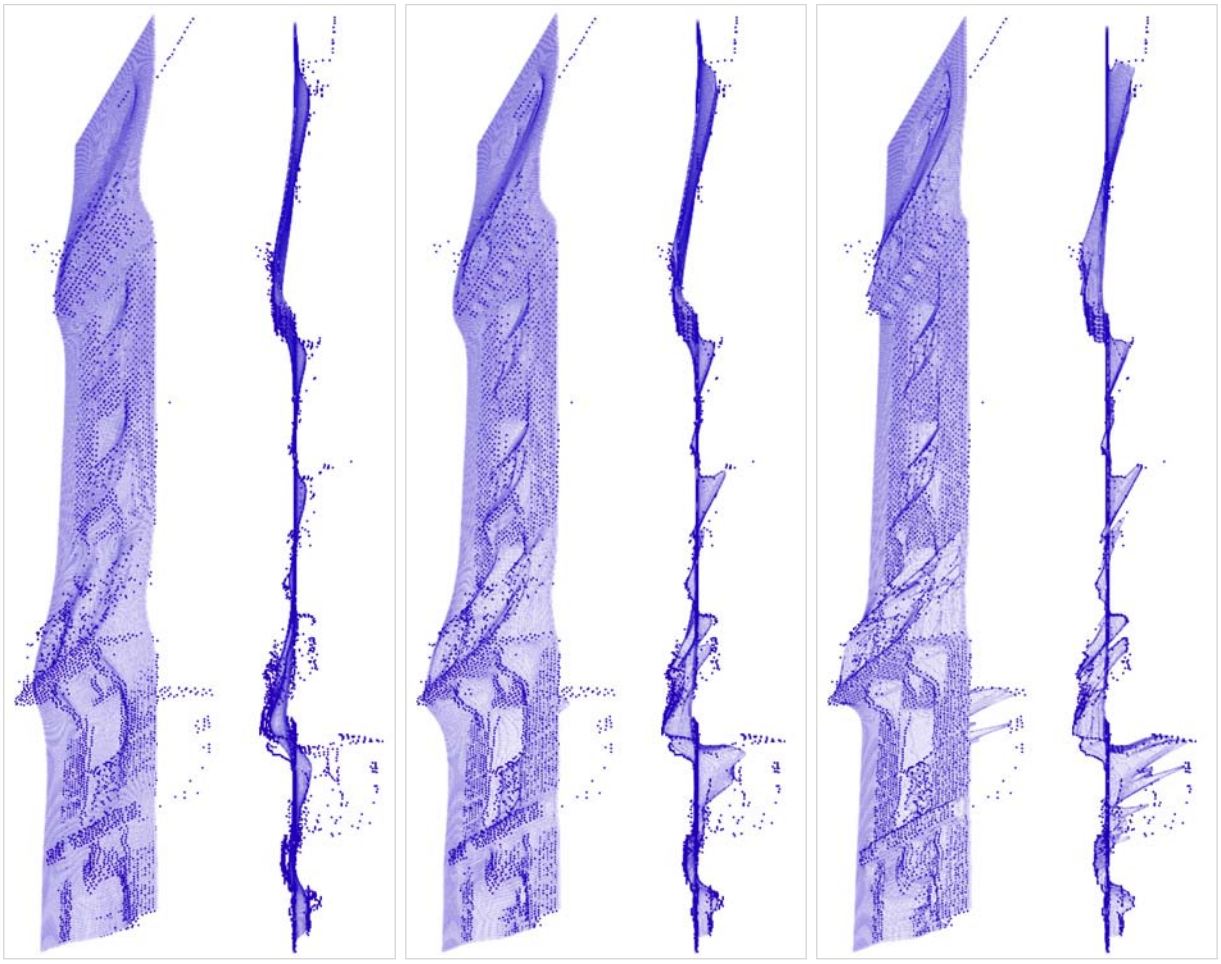


Figure 7.15: The "prismatic coordinate system" grid have been calculated for different resolutions, and eleven pairs (λ_{pixel}, λ_{neigh}) : (0, 1), (0.1 , 0.9), (0.2 , 0.8), (0.3 , 0.7), ... (1 , 0). The median distance of the points to the surface (data term), and the surface area which shows how the model is simple (small area) or complex (large area), are observed. When λ_{pixel} ↗ and λ_{neigh} ↘, the surface is deforming from a very simple surface: the original rectangle with an area of 530m^2, to a more complex surface that is closer to the points.

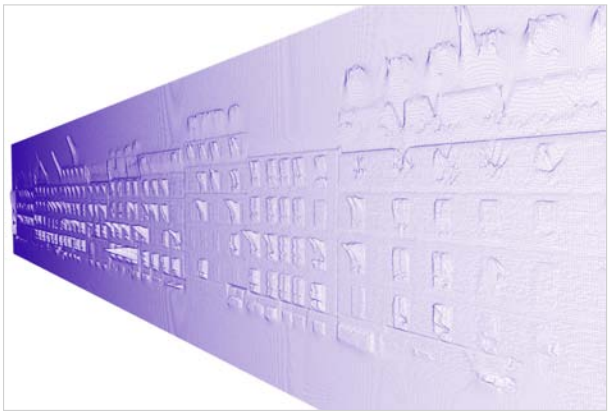
Figure 7.16: Grid constructed for a facade (left). Detail of the grid (right).



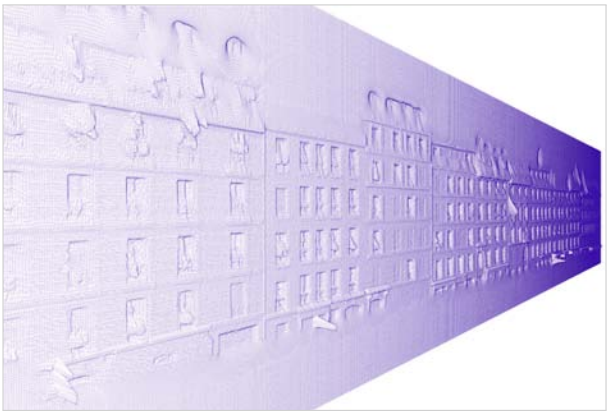
(a) $\lambda_{\text{pixel}} = 0.01$ and $\lambda_{\text{neigh}} = 0.99$ (b) $\lambda_{\text{pixel}} = 0.10$ and $\lambda_{\text{neigh}} = 0.90$ (c) $\lambda_{\text{pixel}} = 0.90$ and $\lambda_{\text{neigh}} = 0.10$

Figure 7.17: Three parameterizations for the same facade piece (three-quarters and profile view). From (a) to (b) and (c), the data term increases and the smoothness term decreases. The surface becomes less simple and more rugged, but closer to the lidar points.

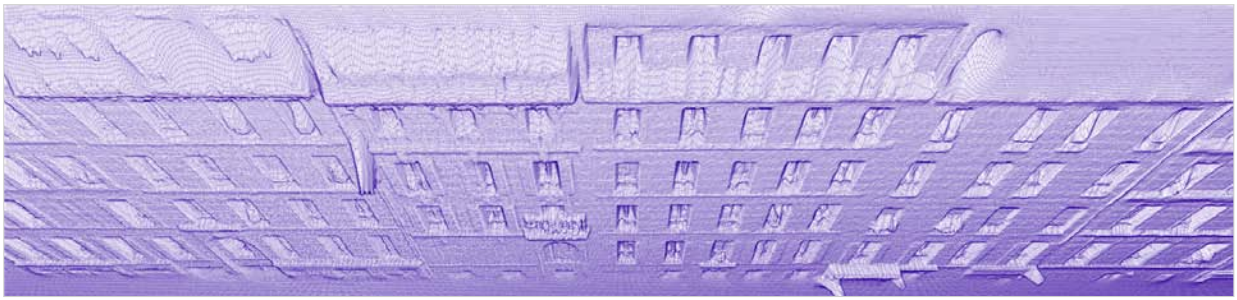
Figure 7.18: Some results on a facade with complex architectural patterns. Shapes are well approximated thanks to the deformable grid (Prismatic coordinate system). However, the model gives an impression of material that is melting or flowing down. This is because the surface covers all hidden areas that can actually be empty. However, with this single street point of view, it is impossible to know if such areas are empty or full.



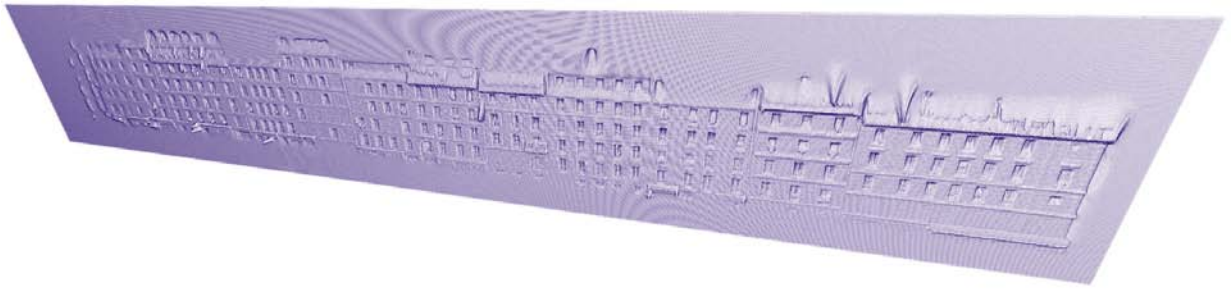
(a) Behind the Facade - interior building viewpoint



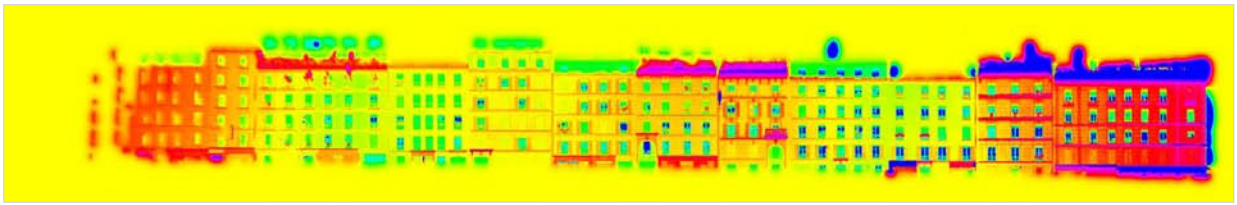
(b) In front of the Facade - street viewpoint



(c) In front of the Facade - top viewpoint



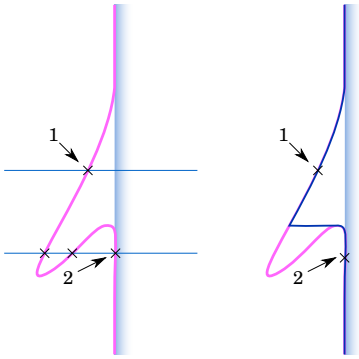
(d) In front of the Facade - far viewpoint



(e) Orthogonal Depth Map

Figure 7.19: Deformable grid calculated according to a prismatic coordinate system. The surface goes trough the widows and goes up to the room ceilings (a). An orthogonal depth map is generated from this surface (e). Such a depth map is compact and simple to visualize.

If the deformable grid is built according to a sensor-oriented coordinate system (pink), this is a 2.5D surface in this system. However, an "orthogonal depth map" can be obtained. For this purpose, another grid is placed in the main vertical plane (violet), and for each pixel, a straight line orthogonal to the plane is drawn. This line intersects the deformable grid in one (1) or several points (2). The closest point to the plane is kept to calculate this pixel depth.



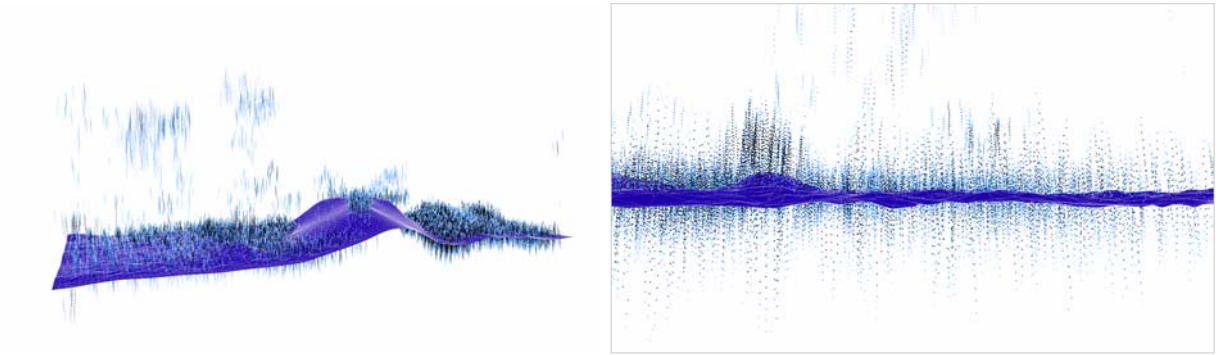


Figure 7.20: The fullwave is discretized into 3D points that are weighted according to their amplitude value (white= 0 → dark). The grid is deformed according to the weighted points. The proposed framework allows to handle the numerous points.

Chapter 8

Combining image and lidar by matching discontinuities

Contents

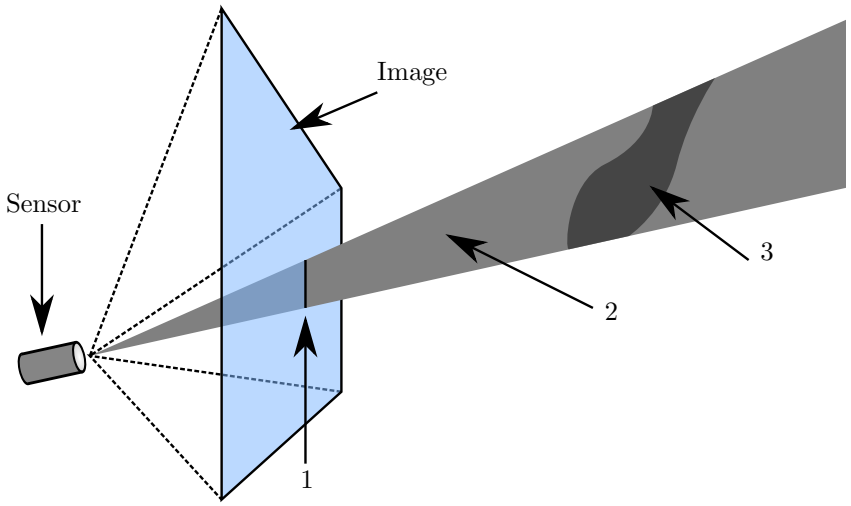


Figure 8.1: Cutting line hypothesis. A line segment in the 2D image (1) can correspond to a geometric discontinuity. This discontinuity (3) is located somewhere in a 3D surface (2) that we call "cutting line hypothesis".

8.1 Introduction

In this chapter, a method is proposed to merge the image and lidar data acquired at the same time by the Stereopolis. Fusing lidar and image data is a topic that is not widely explored in the context of building modeling. An example of refinement of window edges and crossbars can be found in [101]. The 3D model provided by the lidar data is refined with 3D features extracted from images thanks to photogrammetric methods: the line segments that correspond to window edges and crossbars are selected to reconstruct a more precise window structure.

There is not a sufficient overlap in the Stereopolis images to apply a photogrammetric approach. The images can only provide radiometric information (texture) and cutting line hypothesis (see section 8.1.1). So the image data has been used to refine the grid at the discontinuities and to texture the grid. The main obstacles are detailed in section 8.1.2. Then an overview (section 8.2) and the main steps of the method are described (section 8.3).

8.1.1 Cutting line hypothesis

Only radiometric discontinuities can be detected in a single image, but they may correspond to geometric discontinuities. The radiometric discontinuities are thereby hypothesis of geometric discontinuities. The 3D locations of such discontinuities are unknown, but, if the images are oriented and if the distortion is known, the location of each pixel is restricted to a half-line starting from the sensor location. Only the distance to the sensor is missing. By extension, an edge (a line segment) extracted from a 2D image is located into an infinite surface in 3D as shown in figure 8.1. As well a 2D polyline in the image is located into a beam starting from the sensor location. The "cutting line hypothesis" correspond to such radiometric discontinuities that form a cutting beam into the 3D space.

8.1.2 Technical obstacles to texture mapping

In the previous chapter, a 2.5D grid is reconstructed from the lidar data, modeling the facade surface. As the Stereopolis acquires both lidar data and optical images, this grid can be textured by projecting the images acquired concomitantly with the lidar echoes.

This task raises several problems that we present in the order they appear.

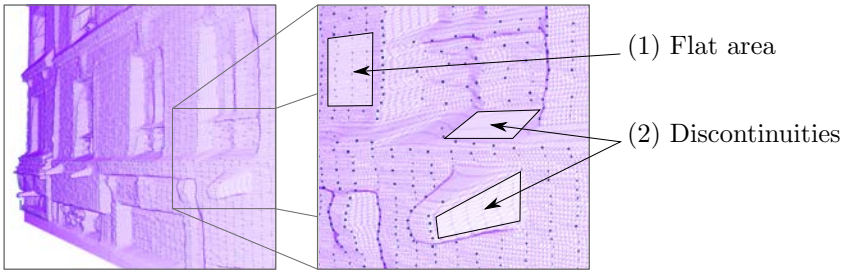


Figure 8.2: Various point density along the surface leads to heterogeneous reliability of the interpolation. Surface interpolation is simpler on flat areas than on discontinuities. The echoes on both sides of a discontinuity are further than usual, making discontinuity a large area with no information.

Registration

We have to know the position and orientation of the images relatively to lidar data. Fortunately, this is our case, and it is assumed that these prerequisites are met; lidar and image data are properly registered. That is why we do not address the issue of multi-source registration.

Geometric imprecision of the grid

However, even if the image is perfectly registered with the grid, it is not sufficient to project the image on the grid. Indeed, the grid is less accurate compared to the images, the geometrical information contained in the lidar echoes have been interpolated between the echoes, and sometimes, a choice have been made between several echoes: the grid is an hypothesis of the facade geometry, that is very reliable close to the echoes, but more approximative around. There is also a 2D simplification of the facade geometry. When a pixel is projected on it, we are not ensured to obtain its true 3D position. The error may be low on flat surfaces as on the walls that are well estimated, but the mismatching is critical along the geometrical discontinuities that are very clear and sharp in images and harder to retrieve in lidar data: surface interpolation in a non-flat area is of course harder, but moreover, the echoes on both sides of a discontinuity are further than usual, making a discontinuity a large area with no information (fig 8.2). In sum, the grid discontinuities are smooth and inaccurate compared to the image discontinuities. If images are projected on the grid, some smudges may appear along the discontinuities.

Image discontinuities are more precise but unreliable

The grid discontinuities are smooth and inaccurate, but they are reliable because they correspond to geometrical discontinuities. To the contrary, image discontinuities are sharp and accurate, but they correspond to radiometric discontinuities that are not necessarily correlated with a geometrical variation. Numerous phenomenon may produce strong radiometric variations on a flat surface (no geometric variation) such as shadows, printings, change of materials... So we cannot allow all image discontinuities to affect the deformation of the geometric model. That is why we intend to use only the image discontinuities that match a grid discontinuity.

Radiometric normalization of several images and image stitching

If several images are projected on the grid, radiometric variations between images may lead to a "patchwork" texture. Image equalization and stitching are necessary. These topics is not investigated. In the following, we present the texture mapping of a single image. Anyway, this problem can be handled independently from our method.

8.2 Overview

As far as possible, we would like to allow the grid to move into in order to match the grid discontinuities with the image discontinuities. The "2.5D grid algorithm" outputs a surface interpolated between the

echoes. Insofar as this interpolation is not unique and is not guaranteed to be the best one, it is possible to replay this algorithm, adding image information. We want to allow sharper discontinuities along the image discontinuities. The algorithm contains a smoothness term that imposes depth continuity between neighboring pixel, We can imagine to cut this pixel connectivity along the image discontinuities. The images discontinuities are not reliable, so they are filtered and those which match to a grid discontinuity are kept. The discontinuities are modeled with line segments. In order to match the segments in the 3D space, the depths of the image segments have to be estimated. Assuming the image position and orientation are known, an image segment lies in a beam starting from the image sensor, but the distance to the sensor is unknown. Assuming an image segment lies on the grid, the corresponding 3D segment is therefore located at the intersection between the beam and the grid.
Thereby, the filtered image segments are used to release continuity constraint along the grid discontinuities. The grid is then recalculated, making the discontinuities sharper and consistent with the image discontinuities. Finally it is possible to perform the texture mapping: the image is projected on this latter grid.

8.3 Main steps

8.3.1 Computing the deformable grid

Please refer to Chapter 7.

8.3.2 Extracting discontinuities in the grid

The grid is converted into a generalized depth image. The 2D segments are extracted in the depth image. For this purpose, we resort to the algorithm proposed in [, taillandier2002reconstruction] and we use it as is. We obtain 2D segments in the (u, v) space that can be directly converted to 3D segments in the (u, v, h) space. Finally, the xyz coordinates are calculated from uvh coordinates.

8.3.3 Extracting 2D segments in the optical image

Again, we use the algorithm of [102] as is, in order to extract 2D segments s_i in the optical image.

8.3.4 Estimating the third coordinate of the image segments thanks to the grid

We want to immerse the 2D segments s_i into 3D space. They are in 2D in the image space (i, j, k). The coordinates ij locate a point on a half-line extending from the optical sensor position. The unknown coordinate k is the distance to the sensor. We can estimate this distance if we intersect the half-line with our grid.
In practice, the grid pixels are less numerous than the image pixels. This is why we prefer to estimate the coordinates ij of each grid pixel. In fact, we simply consider each grid pixel as a 3D point and we calculate its ij coordinates. We obtain a set of 5d points $\pi_g(x, y, z, i, j)$ that allows the mapping between ij and xyz. For any pair ij, an xyz estimation is given by the xyz coordinates of the closest point π in the (i, j) space. This method is used to retrieve the xyz coordinates of the ends of each image segment s_i. The search time is in $\mathcal{O}(\#s_i.\#\pi_g)$ if there are $\#\pi_g$ grid pixels and $\#s_i$ image segments. In order to reduce this search time, a 2D spatial indexing tree is built with the set of points π_g in the (i, j) space. The search time is then in $\mathcal{O}(\#s_i. \log(\#\pi_g))$.
After this step, xyz coordinates are associated to the ends of each segment s_i.

8.3.5 Matching grid segments with image segments

For each s_g, we seek the nearest image segment(s) s_i. To avoid search among all the s_i, a 6d spatial indexation tree is built on the s_i set. The six dimensions are the coordinates of the middle of s_i and of the unit direction vector of s_i. For each point s_g, the K nearest neighbors are searched. And we keep those that verify a distance and an overlap criteria. The image segments that are kept are called $s_i^\star$
Some implementation details are exposed below.

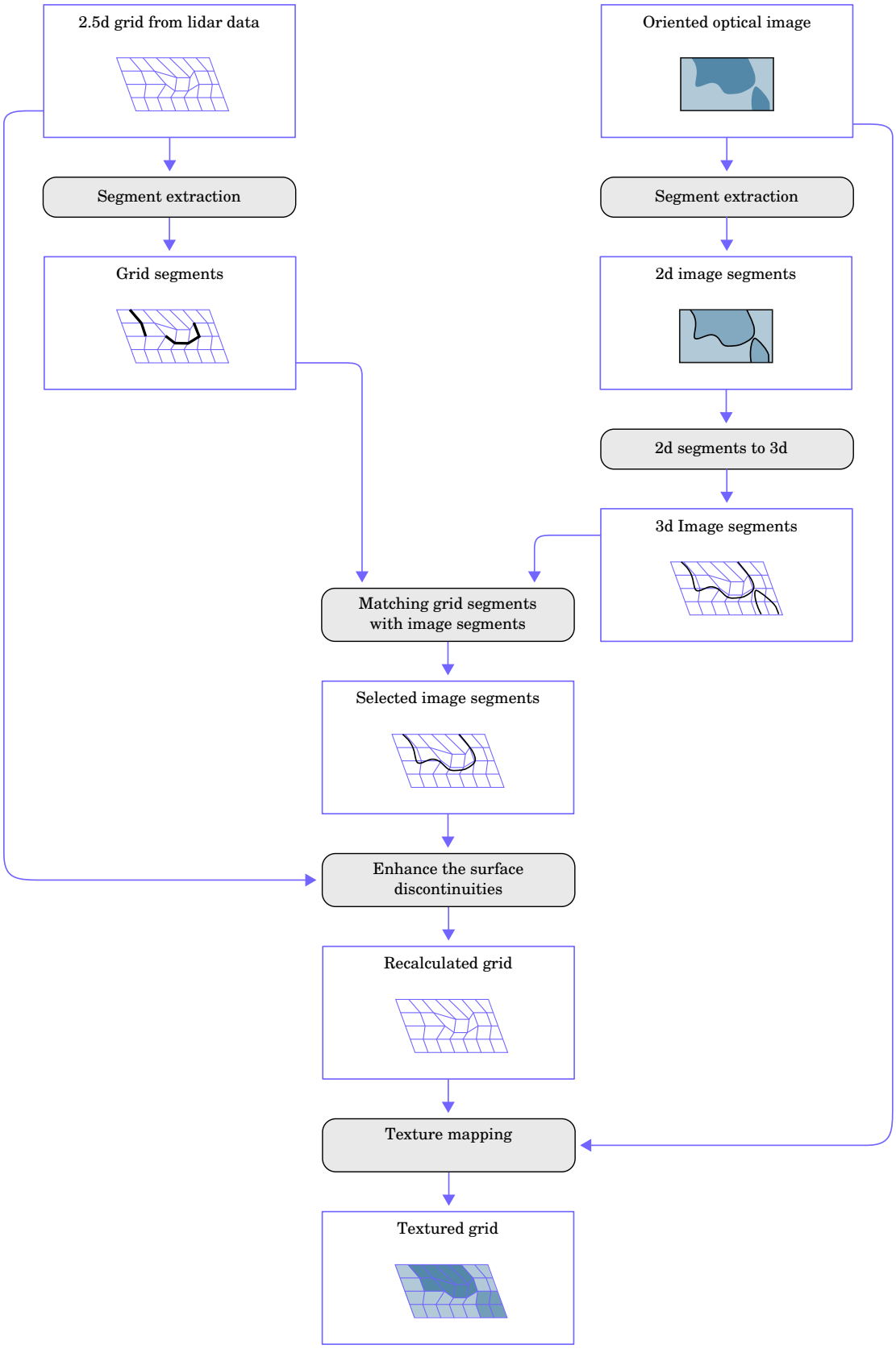


Figure 8.3: Overview

Comparing unit vectors: bypass the sign problem

The unit vector of a line segment ab can be equally defined by $\overrightarrow{u} = \overrightarrow{ab}/||\overrightarrow{ab}||$ or $-\overrightarrow{u} = \overrightarrow{ba}/||\overrightarrow{ba}||$. So, there is a problem with the sign of the unit vectors in the search space: if $\overrightarrow{w}$ is close to $\overrightarrow{u}$, $-\overrightarrow{w}$ that is at the opposite of the unit sphere should also be close to $\overrightarrow{u}$. To bypass this problem, for each segment, we store two segments in the tree, one with $\overrightarrow{u}$ and the other with $-\overrightarrow{u}$. Whatever the sign of $\overrightarrow{w}$, $\overrightarrow{w}$ will be close to one of the two segments.

Tuning the number K of nearest neighbors

K is an optimization parameter that should not modify the final result. The smaller is K, the faster is the algorithm. However, if K is too small, some image segments that verify the criteria may be forgotten (ie the final result is modified). K has to be chosen small, but not too small. In our experiments, we set $K = 100$.

Distance and overlap criteria

We want to keep the image segments s_i that are similar and close to a grid segment s_g. For this purpose, a distance score $\mathcal{D}(s_i, s_g)$ and an overlap score $\mathcal{O}(s_i, s_g)$ are defined. Then, s_i fulfills the distance and overlap criteria if $\mathcal{D}(s_i, s_g) < \mathcal{D}_{\max}$ and $\mathcal{O}(s_i, s_g) < \mathcal{O}_{\min}$. The thresholds $\mathcal{D}_{\max}$ and $\mathcal{O}_{\min}$ have to be tuned depending on the grid resolution. $\mathcal{O}_{\min}$ also permits to filter the segments that are too small.
We call $d_i(C)$ the orthogonal distance between C and the straight line A_iB_i.
The **distance score** writes:

$$\mathcal{D}(A_1B_1, A_2B_2) = \frac{d_1(A_2) + d_1(B_2) + d_2(A_1) + d_2(B_1)}{4} \tag{8.1}$$

Let $s_i(C) = \overrightarrow{A_iB_i}.\overrightarrow{A_iC}$
Let us define an overlap score of A_jB_j projected on A_iB_i.
If we assume that $s_i(B_i) \geq s_i(A_i)$ and $s_i(B_j) \geq s_i(A_j)$, then
$\mathcal{O}_i(A_jB_j) = \min(s_i(B_i), s_i(B_j)) - \max(s_i(A_i), s_i(A_j))$.
And the **overlap score** writes:

$$\mathcal{O}(A_1B_1, A_2B_2) = \frac{\mathcal{O}_1(A_2B_2) + \mathcal{O}_2(A_1B_1)}{2} \tag{8.2}$$

8.3.6 Enhancing the surface discontinuities

The last part of the algorithm consists in using the image discontinuities to enhance the surface estimation. For this purpose, the deformable grid algorithm is replayed, but we now allow a tears along the discontinuities: whenever a segment $s_i^\star$ intersects the line segment connecting the center points of two neighboring pixels, A tear is allowed at the edge between these two pixels. The interplay of the two pixels is cut in their depth calculation. Concretely, the weight of h_{neig} is divided by 100.

8.3.7 Texture mapping

The last step is the projection of the image pixels onto the grid. Many image pixels may fall onto the same grid pixel. The choice of the resulting rgb value (mean, median) have not been tackled. We tested our method with the mean value of the image pixels that fall into the grid pixel.

8.4 Results

The results are displayed and commented in figures 8.4, 8.5, 8.6 and 8.7.

(a) (b)

Figure 8.4: Image discontinuities (dark blue) and surface discontinuities (purple). The image discontinuities are more accurate but do not always correspond to a geometric discontinuity, as we can see with the street number 212 hidden in the picture (a).

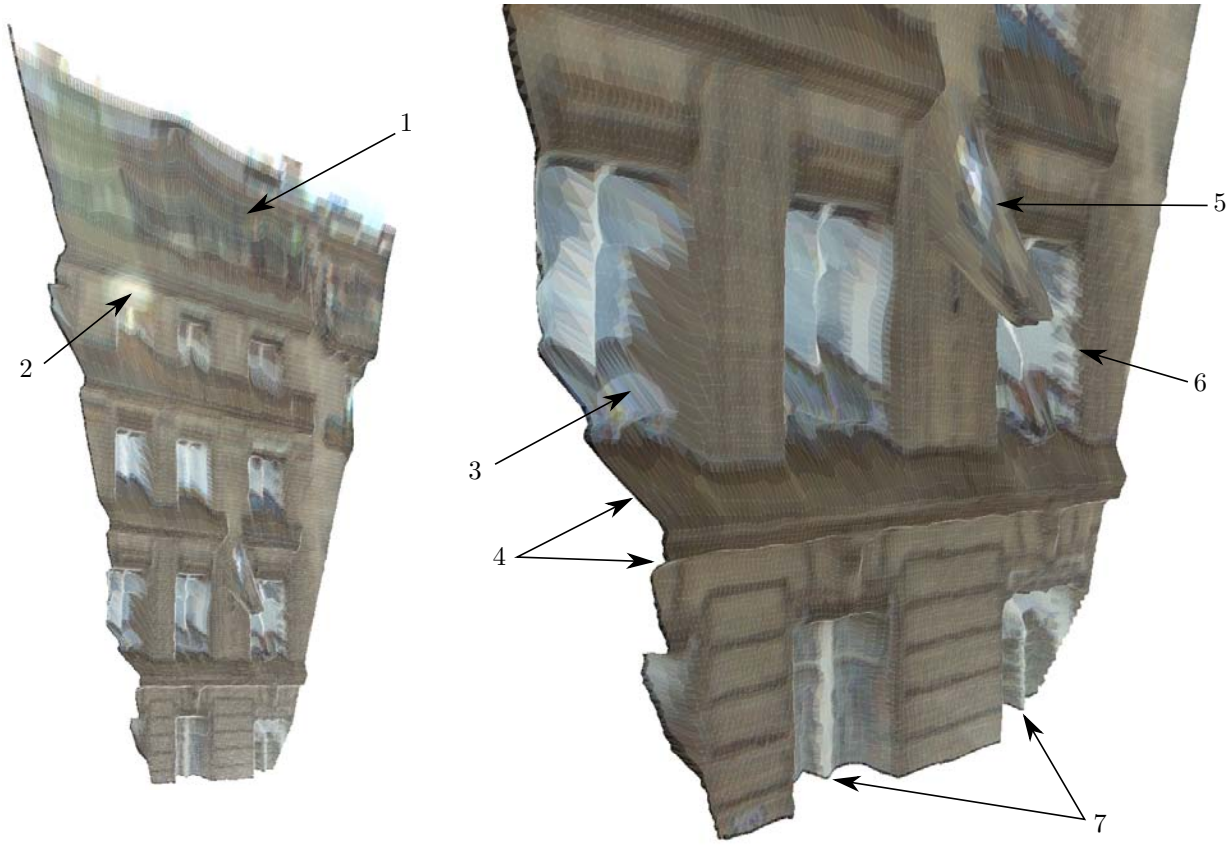


Figure 8.5: Output grid textured with an image: defects $\ominus$ and qualities $\oplus$.
1 $\rightarrow$ The point density and the image resolution decreases toward the facade top. This results in a smoother and less accurate result $\ominus$, but it reflects the acquisition/data.
2 $\rightarrow$ The radiometry variation (halo) does not alter the model geometry $\oplus$.
3 $\rightarrow$ At the balconies, lidar echoes lie at various depths: balcony, window, behind the window. the surface depth reliability suffers from these multiple depth possibilities $\ominus$.
4 $\rightarrow$ The ledge modeling is sharper below than overhead because of the lidar sensor point of view $\ominus$ but it reflects the acquisition/data.
5 $\rightarrow$ The wall light geometry has been detected and the projected pixels correspond to it $\oplus$, but there are also wall pixels. The result is not satisfactory, probably due to the strong geometric distortion $\ominus$.
6 $\rightarrow$ Texture smudge at the window edge $\ominus$.
7 $\rightarrow$ Window bar well detected $\oplus$.

Figure 8.6: Output grid textured with an image. The point of view is close to the sensor point of view, this is the one that showcases the most the result: texture distortions appear all the more we move away from this viewpoint.

8.5 Conclusion

In this chapter, we focused on improving the texturing of the deformable grid with optical images. When images are projected on the grid, the most sensitive areas to geometric inaccuracies of the grid are the discontinuities at the objects edges. If the geometrical discontinuities do not match well the radiometric discontinuities in optical images, one can see some smudges particularly unpleasant visually. So we proposed a method for matching geometric discontinuities with radiometric discontinuities (less reliable but more accurate). Geometric discontinuities are extracted in the grid, and radiometric discontinuities are extracted from the images. These are 2D segments which are projected onto the grid to obtain 3D segments. These segments are matched with the geometric discontinuities. We keep only those which are sufficiently close to a geometric discontinuity. Image segments that are kept are used to recalculate the deformable grid: the iterative grid deformation is rerun, but this time the smoothing constraint is reduced at the image segments, thus allows higher tears at the precise location of image discontinuities.
The results are satisfactory for texturing, one can check the consistency between lidar data and optical images. However, the accuracy of the textured model is not homogeneous : on the one hand there is a difference in resolution between the geometric model from lidar data and image data : the textured image resolution is better than the 3D surface resolution. on the other hand the limitations of a "single viewpoint" approach are experienced : geometry and texture appear all the more imprecise the further we are from from this point of view. One can not invent information and it is therefore difficult to improve the results from only these datasets. In contrast, a fusion with other datasets could be imagined.

Figure 8.7: Comparison between the textured grid and an image. On the left, the textured grid and the lidar echoes (blue). On the right, an image. The point density is low relative to the image resolution, however, the geometric model is consistent with the texture. The main problems remain at the discontinuities (edges of balconies, window edges). Despite the discontinuity matching algorithm, there are smudges texture. The discontinuities are at the limit of visibility of the lidar sensor and the image sensor. So these are areas that are poorly seen or unseen and where the lack of information needs to be interpolated. This explains the visually imperfect results.

Conclusion

Chapter 1

In this chapter, we have described the lidar data and highlighted strengths and weaknesses of this data. Processing lidar data acquired from MMS in urban environment seems challenging.
Main problems are: in addition to be inherently **sparse**, and **unevenly sampled**, the data is necessarily **incomplete**: many parameters cannot be controlled such as the vehicle speed or trajectory, the on-board sensors are constrained to capture information from the street point of view, along the vehicle trajectory. For instance, the building volumes cannot be apprehended in their wholeness. We have to deal with facade pieces, partially hidden by occluders, such as cars, trees or other facades. The data is therefore incomplete, but can be redundant if some loops are performed during the acquisition. However, the redundant data may be inconsistent because of some georeferencing errors. We have to deal with facade pieces, partially obscured by foreground objects such as cars, trees or other buildings. The data is incomplete, it may also be redundant if the acquisition path contains loops, georeferencing errors can produce **inconsistencies in redundant data**. The Lidar data is obtained by systems which involve different technologies. The errors of each technology have different amplitudes. Vehicle georeferencing is the source of the greatest inaccuracies, that is why we emphasize the risk to have inconsistencies if the data contains redundant parts acquired at different times of the acquisition and thus georeferenced differently.
It is difficult to understand the point distribution and yet the echo location is the only available information to guess the scanned scene geometry. For example, one is tempted to rely on the point density as a confidence index, while this density may be due to acquisition conditions (vehicle stopped at a red light).
In fact, the echo location is only due to -except for measurement errors- the scene geometry, while the point density largely depends on acquisition conditions.
When echoes are expressed in the coordinate system $\tau\Theta, R$, (τ: acquisition of time , Θ: vertical firing angle, R: distance sensor), a very high regularity is observed according to $\tau\Theta$. Indeed, the laser pulse is emitted with a regular frequency. Quite the opposite, R only depends on the distance to objects and is unpredictable. This coordinate system also highlights that point density is near to a surface density (τ, Θ surface).
The coordinate system τ, Θ, R is independent of the sensor georeferencing. This is useful if the data is poorly georeferenced. Moreover, these coordinates are quick to obtain because they stem directly from the acquisition process and can provide hypotheses for surface reconstruction. Indeed , the detected surfaces may be approximated by a mesh $tau\Theta$. However, this mesh can be folded on itself if the laser sweeping turns back.
Our conclusion is that, even if it brings new problems, it is a pity to use only the 3D points. The sensor geometry, and in particular the coordinate system τ, Θ, R highlights the point cloud regular structure. and allows to separate information coming from the acquisition conditions (τ, Θ) to those who comes from the scanned data (R).

Chapter 2

The objective of this chapter was to provide a simple, generic and automatic method to describe the geometry around each lidar echo. The two issues we tried to answer are:

- Find generic geometrical attributes to describe any point cloud.
- Find an automatic method to automatically select the neighborhood adapted to each point.

For this purpose, the attributes are derived from the classical "tensor voting" approach. Attributes are calculated in a spherical neighborhood around each echo. They describe the dimensionality (1D, 2D or 3D) depending if the echo distribution in the neighborhood is rather linear (1D) , planar (2D) or volume (3D). The most appropriate neighborhood size is selected within a range of potential sizes through an entropy feature E_f.
It is not necessary to have a priori knowledge about the echo distribution, the density or pattern of the laser scan. However, we have observed that on the one hand, the obtained geometrical description allows to deduce these different characteristics, and secondly, a knowledge of these characteristics can well bound the neighborhood sizes to obtain a result that describes the geometry of scanned objects and that is, as far as possible, independent on the acquisition configuration. In other words, the algorithm can be started once to analyze the characteristics of the point cloud and fine-tune the parameters to run the algorithm again.
The advantages of a simple spatial sub-sampling are also highlighted to abstract from the point distribution induced by the acquisition configuration.

Chapter 3

Scaling up imposes some constraints in the algorithm design. In particular, it is necessary to cut the point clouds to apply processings on smaller blocks. Different choices are possible (Semantic, Spatial/Dimensional : temporal (1D) , Three-dimensional space (3D), or Space-time (4D)). In our context, the most sensible approach seems to be to keep the original structure of the point cloud, Indeed , the points are stored in the order of acquisition and are naturally organized according to the temporal dimension. This is the most direct method, but it has also other benefits. A time interval corresponding to a segment of the vehicle path during acquisition. The sensor location is thus bounded in space-time, and points acquired during this interval are always close temporally and often close spatially. Processing temporal buffers is a method

- fast
- adapted to the changing environment : no mixing of points from different epochs.
- suited to the vehicle georeferencing : georeferencing drift is gradual and therefore varies little over a short time interval.

This is the choice made for the facade detection as vertical rectangles in chapter 4. This overcomes the problems of data volume and georeferencing.

Chapter 4

We presented a streamed vertical rectangle detection algorithm which automates facade database production from terrestrial laser scans. This algorithm overcomes the volume of data and georeferencing problems, and provides an initial analysis of urban scenes. A modified RANSAC is performed on overlapping buffers of 3D points acquired during the same time interval. Facade parts are thus extracted from the datasets in linear time (in number of 3D points) and constant memory complexity. Facade parts are then merged and the most relevant facade planes are kept. The construction of the merge graph is quadratic in the number of segments, but this number is negligible compared to the number of points.
The vertical planar regions have proved their benefit in fine localization [56] (fig 4.11). The vehicle drift can be detected thanks to shifted rectangles that correspond to the same facade, then, rectangle matching could enable registration refinement. In [41], the rectangles are fitted with the facade rectangles of the bati-3D model in order to perform a non-rigid registration.
In this thesis, the detected rectangles are used to initialize facade models. Two approaches have been tested : a semantic modeling with irregular grids (chapter 5) and a deformable 2.5D grid (chapters 7 and 8).

Chapter 5

We have proposed a facade model with irregular grids. This model is calculated from a point cloud. First, a vertical rectangle corresponding to the main plane of the facade has been detected. Only 3D points included in this rectangle and sufficiently close to the main plane are taken into account. The rectangle is divided in an irregular grid with vertical and horizontal lines placed at the principal geometrical

discontinuities. To do this, we accumulate the points horizontally and vertically, then the point depth variations are calculated (depth relative to the main plane). The discontinuities are the depth variation maxima. These discontinuities allow to cut the rectangle into a set of rectangles/boxes. For each box, we then search the optimal depth based on points projected in it. We preferred to limit the total number of possible depths, allowing for example all boxes that contain a part of the same wall to have the same depth. Guided by this choice, a depths discretization algorithm was proposed. It is a variant of the "k-means" that automatically finds the optimal number k of depths. We try to minimize both a data term and the number k. The remaining task is to associate one of these k depths to each box. The resulting model is an irregular grid where each box moves back or forward relative to the main plane. This model therefore assumes that the facades are composed of parallel rectangular elements. Other approaches have been studied to determine the depth of each box as a graph- cut algorithm.

Chapter 6

We demonstrated that in order to reconstruct a continuous surface from lidar signal, meshing echoes that are adjacent in sensor topology (sensor mesh) is an appropriate solution because it naturally integrates logical constraints imposed by the laser beams. However, the geometry of the scanned scene rarely amounts to a single continuous surface.
If the sensor is in front of the surface, one can have confidence in the fact that there is continuity between adjacent echoes, whereas if the sensor sees the surface with a grazing angle, it may exist cavities between two adjacent echoes that contain an hidden portion of the surface and that contradict the continuity between adjacent echoes. The incidence angle thus provides a simple measure of the lidar reliability in detecting a continuous surface.
Despite the continuity between adjacent echoes is not one hundred percent reliable, the continuous surface provided by the sensor mesh has always a physical meaning : this is the interface between what is seen and not seen by the sensor. This mesh is therefore interesting to use, but the main obstacle is that it can fold on itself if the laser sweeping turns back.

Chapter 7

We presented an approach to reconstruct facade geometry from lidar data. We aimed at surfaces consistent with the data, especially the optical images acquired in the same time. The proposed framework allows to reconstruct 2.5D grids in a coordinate system adapted to the sensor point of view. For this purpose, we introduced a prismatic projection that is a generalization of the orthogonal projection and the corresponding coordinate system. It allows to benefit from the sensor geometry while providing a coordinate system topologically consistent with the 3D space. The output 2.5D grids are suitable for immersive navigation and other applications that need compact and detailed 3D models. They could also be used as inputs for facade structure analysis, in particular for window detection or grammar rules extraction.

Chapter 8

In this chapter, we focused on improving the texturing of the deformable grid with optical images. When images are projected on the grid, the most sensitive areas to geometric inaccuracies of the grid are the discontinuities at the objects edges. If the geometrical discontinuities do not match well the radiometric discontinuities in optical images, one can see some smudges particularly unpleasant visually. So we proposed a method for matching geometric discontinuities with radiometric discontinuities (less reliable but more accurate). Geometric discontinuities are extracted in the grid, and radiometric discontinuities are extracted from the images. These are 2D segments which are projected onto the grid to obtain 3D segments. These segments are matched with the geometric discontinuities. We keep only those which are sufficiently close to a geometric discontinuity. Image segments that are kept are used to recalculate the deformable grid: the iterative grid deformation is rerun, but this time the smoothing constraint is reduced at the image segments, thus allows higher tears at the precise location of image discontinuities.
The results are satisfactory for texturing, one can check the consistency between lidar data and optical images. However, the precision of the textured model is not homogeneous : on the one hand there is a difference in resolution between the geometric model from lidar data and image data : the textured image

resolution is better than the 3D surface resolution. on the other hand the limitations of a "single viewpoint" approach are experienced : geometry and texture appear all the more imprecise the further we are from from this point of view. One can not invent information and it is therefore difficult to improve the results from only these datasets. In contrast, a fusion with other datasets could be imagined.

8.6 Perspectives

Toward an upgradable model : time, precision, reliability

In this thesis , we worked with terrestrial mobile lidar and image data. We encountered difficulties in data fusion, whether with data of the same type but acquired at different times, or with different data types, precision and reliability can vary.
The trend is to acquire an increasing volume of data. Data fusion becomes unavoidable to create new digital models and to improve the existing ones. We are moving towards a increasingly upgradable models: models should be able to be modified by operators or by adding new data. We designed the deformable grid for this purpose.
However, when one wants to automate such processes, an hurdle soon emerges: What if two sources are contradictory? It seems that the solution is to take into account three aspects in the model:

time : the digitized objects, even those that may seem static as facades change over time. One must therefore work in 4D rather than 3D.

precision : This is the level of the finest detail of the proposed model. It depends on the acquisition system and is intrinsically linked to the model. Indeed, two different models can be consistent if they represent two different levels of detail. To take account of this precision score, two modelling choices seem possible.
-The most precise model, which is a fusion of sources that maximizes precision.
-And a more flexible model that is multi-resolution.

reliability or robustness: Accuracy is not sufficient to describe the data, a confidence index is required to make a choice when two sources are contradictory. Reliability is correlated with precision in a complex way : "more precise" does not mean "more reliable". Instead, One gains reliability if one accepts losing in level of detail. It seems to us that it would be an interesting line of research to design a multi-resolution model that allows to explicitly handle this opposition precision/reliability.

Precision and reliability depend on the acquisition system, and notably the measurement noise. It seems essential to be able to assess these criteria, whether for lidar echoes or for image pixels. This is possible if the acquisition systems are known, but the information -as the information of pixel depth in images- can emerge during the processing. Similarly, the model precision and reliability can evolve independent of geometry: Merging new data can strengthen the position of a mesh without deforming it, or conversely, it may reduce reliability by providing contradictory information.
Thus, precision and robustness are required to an upgradable model. Otherwise, how to decide whether or not it is necessary to modify the existing model if contradictory information is provided?

Chapter 9

Annex

Contents

9.1 Definitions and Acronyms

PCD	:	Point Cloud Data
ALS	:	Airborne Laser Scan/Scanning
TLS	:	Terrestrial Laser Scan/Scanning
MS	:	Mobile Laser Scan/Scanning
MMS	:	Mobile Mapping System
LoD	:	Level of Detail

9.2 Canny-Deriche edge detector

The Canny-Deriche edge detector presented in [86].

$cst = 1 - 2e^{-a}\cos w + e^{-2a}$

$c = \dfrac{cst}{e^{-a}\sin w}$

$k = \dfrac{cst(a^2+w^2)}{2ae^{-a}\sin w + w - we^{-2a}}$

$$\mathbb{CD}(x) = \frac{-ck}{a^2+w^2}\sin(wx)e^{-a-|x|}$$

9.3 Sixteen Advantages of the Photogrammetric 3D workflow over the Directly Measured Laser Point Cloud

From [97].

Accuracy and Errors

1. Large area rigid camera image block geometry via AT at a sub-pixel accuracy
2. Error checking using redundant observations as a system-inherent verification
3. Internal accuracy measures from redundancy
4. Geometric accuracy by AT superior to a reliance on GPS/IMU to fuse patches into seamless coverage
5. Greater point density → for better defined discontinuities

Economy

6. Superior data collection efficiency with faster vehicles, larger swaths

7. Single workflow within aerial application, all image-based
8. Single workflow across widely varying applications (aerial, street-side and indoor)
9. No occlusions using no-cost along-track high image overlaps

Data Types 10. 2D-image information augmenting 3D data points
11. Multi-spectral image classification
12. Urban facade textures available at no cost from the air at image edges
13. Images document details → example street signs can be read automatically

Miscellaneous 14. Perspective exists towards Real time 3DVision via "supercomputer in match box"
15. Full automation needs image redundancy → lidar adds little to automation
16. Scene interpretation is becoming important and needs imagery → lidar adds little

9.4 Mean Shift Segmentation

We propose to segment the PCD according to the dimensionality features. The *mean shift* algorithm presented in [103] is used for this purpose.

The classical mean shift

We chose *mean shift*, because it requires only one parameter : the radius of the spherical neighborhood used to compute the centroid. In fact, we will see that we introduce another parameter. The classical algorithm is the following :

```
Inputs : a PCD
Parameter : the spherical neighborhood radius r.
Output : each point associated to a region.
while the points move (p_i ≠ c_i) do
    for all point p_i do
        Compute the centroid c_i of the points included in the neighborhood centered on p_i
    end for
    for all point p_i do
        Move p_i to the c_i.
    end for
end while
```

Optimization

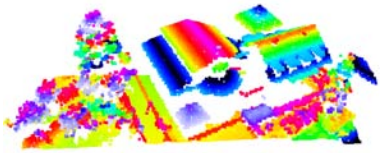
At each iteration, we apply the spatial pruning explained in the appendix 2.5.3 that keeps only one point per voxel. As the points are moving increasingly closer, the pruning reduces gradually the number of points. The only difference in the mean shift algorithm, is that we introduce weights in the mean computation:

1. After the first pruning, the n points included in a voxel are replaced by one point that represents the n points formerly found in the voxel, the weight of this representative point is then n.

2. After the second pruning, the points p_i included in a voxel are replaced by one point p_r, but this time, each p_i has a weight n_i. The weight of p_r is thus $\sum n_i$.

x. After the x^{th} pruning, The weight of a representative point is the sum of the weights of the points contained in the voxel.

In fact, a point is always weighted by the number of points it represents. If the weights are initialized to one in the original PCD, one can see that at every step (1,2,x), the resulting weight is the sum of the weights in each voxel.
The pruning makes the algorithm faster, but less accurate. The loss of accuracy depends on the voxel width, and we can choose it lower than the data estimated accuracy. Moreover, the accuracy is greater than the voxel width because the computations are performed on 3D points that keep their accuracy.



(a) Colored Altitude

(b) $a_3 = 8$, $r = 1$m (c) $a_3 = 8$, $r = 2$m (d) $a_3 = 8$, $r = 4$m (e) $a_3 = 8$, $r = 8$m (f) $a_3 = 8$, $r = 16$m

(g) $r = 8$m, $a_3 = 0$ (h) $r = 8$m, $a_3 = 8$ (i) $r = 8$m, $a_3 = 16$ (j) $r = 8$m, $a_3 = 32$ (k) $r = 8$m, $a_3 = 64$

Figure 9.1: Mean shift "2D" for various r and a_3 values.
Pictures (b) to (f): the region sizes increase with r.
Pictures (g) to (k): when a_3 increases, the normal values are more important in the segmentation process.

Using dimensionalities

Tree algorithms are proposed, adapted to the tree dimensionalities.

1D : the points along linear objects share the same orientation estimated by $\overrightarrow{v_1}$.
A 6D mean shift is performed in $(x, y, z, x_{\overrightarrow{v_1}}, y_{\overrightarrow{v_1}}, z_{\overrightarrow{v_1}})$ space. The x, y, z coordinates have not necessarily the same order of magnitude than the $\overrightarrow{v_1}$ coordinates. This is why the $\overrightarrow{v_1}$ norm has to be determined in order to balance the relative importance of these two triplets. Let $a_1 = ||\overrightarrow{v_1}||$ this parameter.

2D : the points share the same normal vector estimated by $\overrightarrow{v_3}$.
A 6D mean shift is performed in $(x, y, z, x_{\overrightarrow{v_3}}, y_{\overrightarrow{v_3}}, z_{\overrightarrow{v_3}})$ space. As well as in the 1D case, the parameter $a_3 = ||\overrightarrow{v_3}||$ has to be tuned.

3D : the mean shift is performed in the x, y, z space.

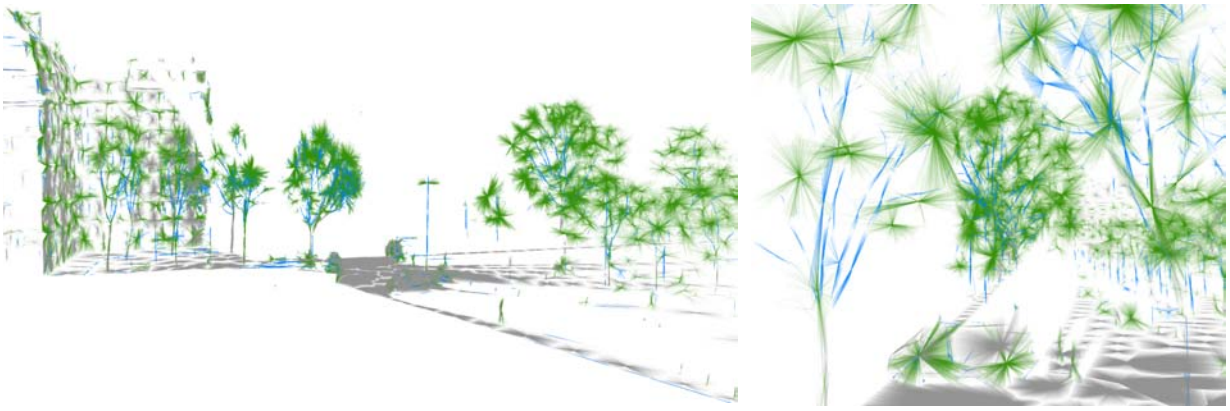
Results

The r and a_3 parameters influence is highlighted in figure 9.1. The r value have to be chosen according to the scale of interest. In figure 9.1 (d) (e) and (f), we can see that if $r = 4$m, the regions correspond to the roofs, and if $r = 8$ or 16m, the points of a whole house or tree are aggregated in a same region.
Figures 9.2 and 9.3 show the mean shift segmentation. In fact, this is the result of the three segmentations previously explained. It is not necessary to perform each segmentation on the whole dataset. For instance, the 2D segmentation produces many small regions in the areas that are not planar. That is why we keep only the points that have a value $a_{2\mathrm{D}}$ greater than a threshold equal to 0.6. The dataset is thresholded with $a_{1\mathrm{D}} > 0.6$ and $a_{3\mathrm{D}} > 0.3$. These thresholds are empirical, the $a_{3\mathrm{D}}$ is lower than the others in order to get all the points in the tree foliages that are sometimes more $1D$ or $2D$. The thresholding allow to optimize the computation time, but they do not enhance the results, they only avoid small regions that we remove after anyway. The regions can be used as primitives for visualization or other applications such as registration.

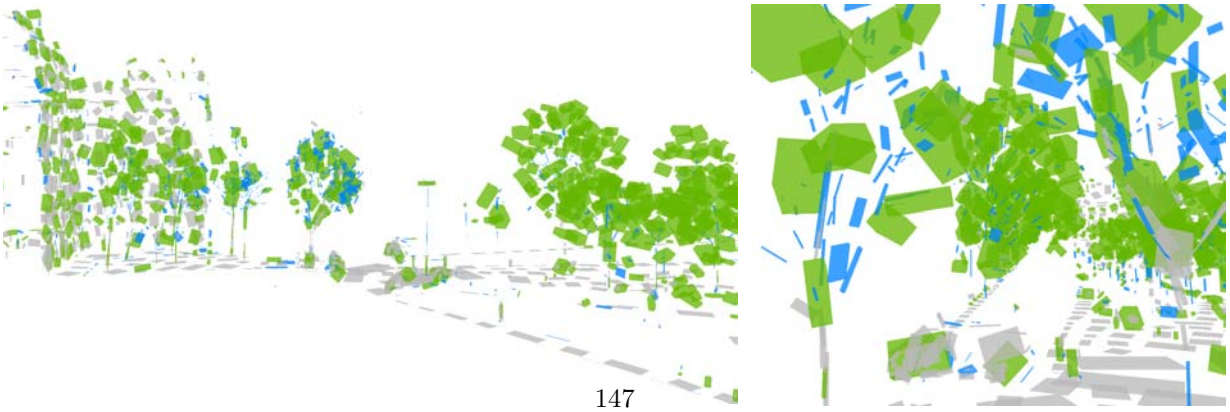
On the last picture, a Cuboid is computed to display the structure tensor of each region : the cuboid edges are aligned with $\overrightarrow{v_1}, \overrightarrow{v_2}$ or $\overrightarrow{v_3}$ and the edge half widths are equal to σ_1, σ_2 or σ_3.
The algorithm have been tested on data acquired by the Stereopolis in Paris 6 (fig 9.2 and 9.3). And on the open ALS dataset of Lucerne (Switzerland). The results are shown in figure 9.1, 9.4 and 9.5.

(a) Intensity values

(b) Points colored per regions



(c) Points linked to their region centroid



(d) Cuboids displaying each region

Figure 9.2: Mean shift segmentations. $a = 8$ and $r = 1$m

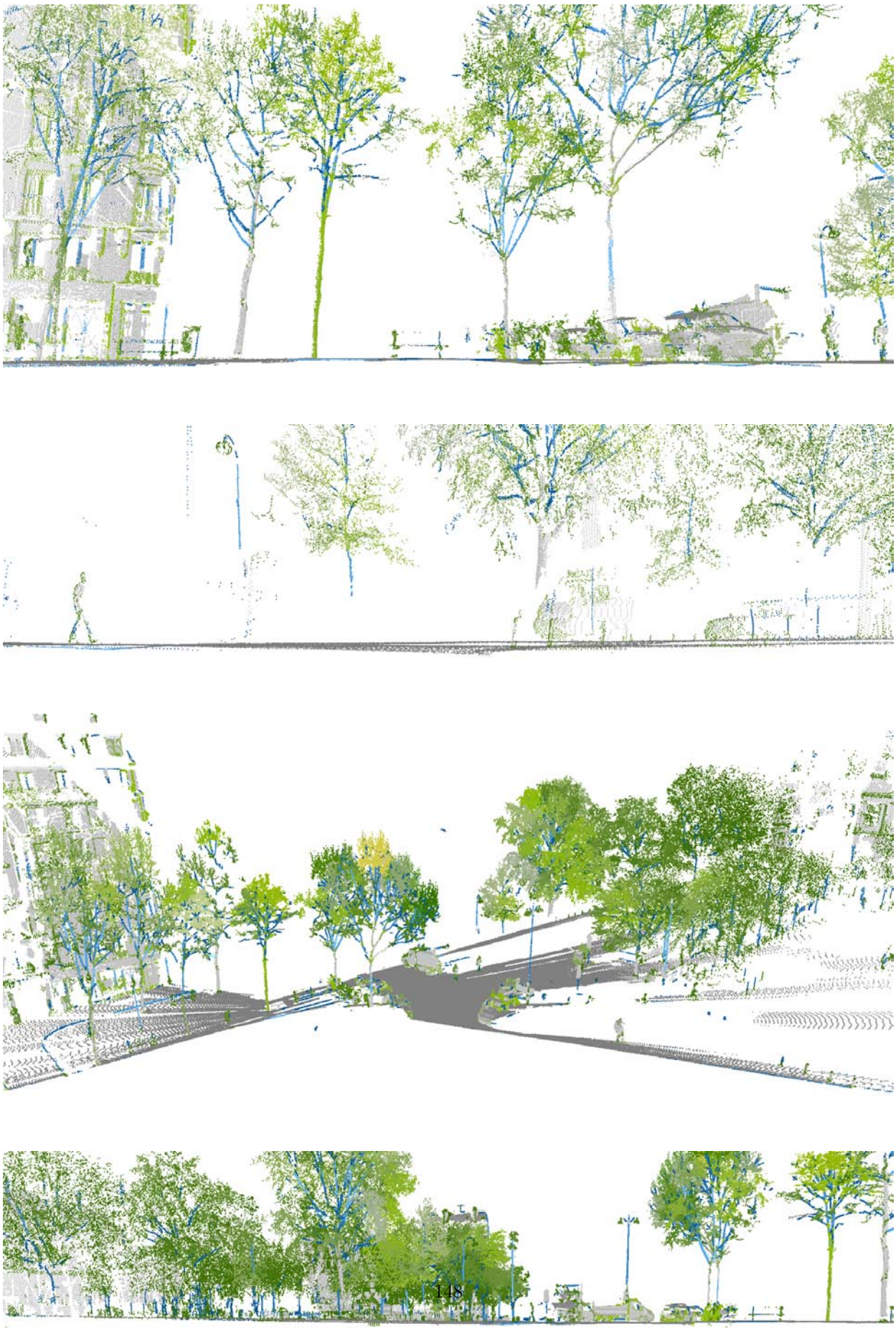
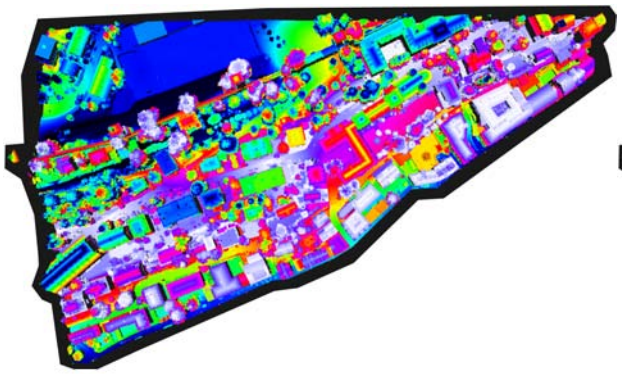
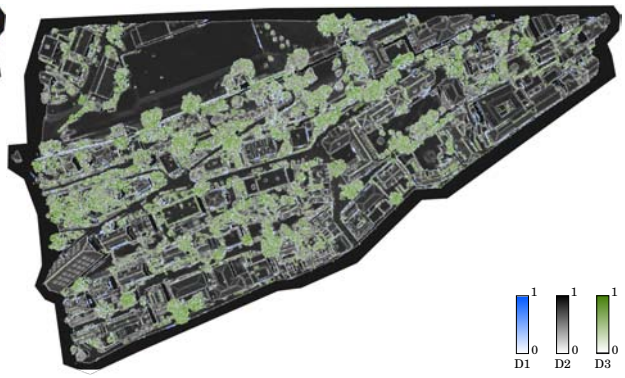


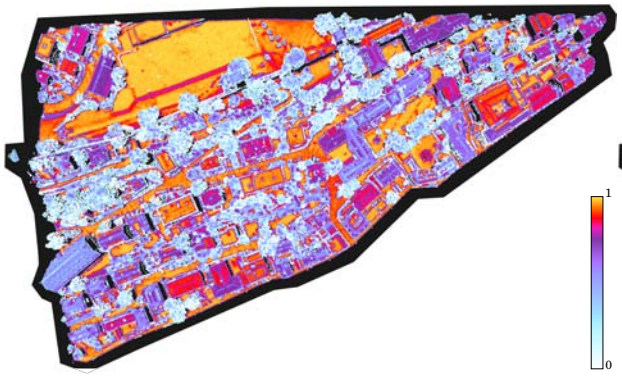
Figure 9.3: More views of mean shift segmentation.



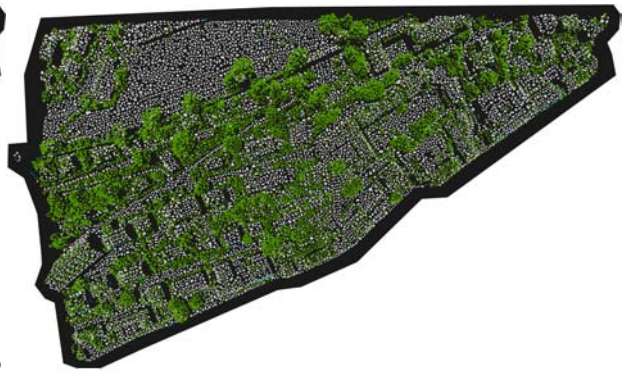
(a) Altitude



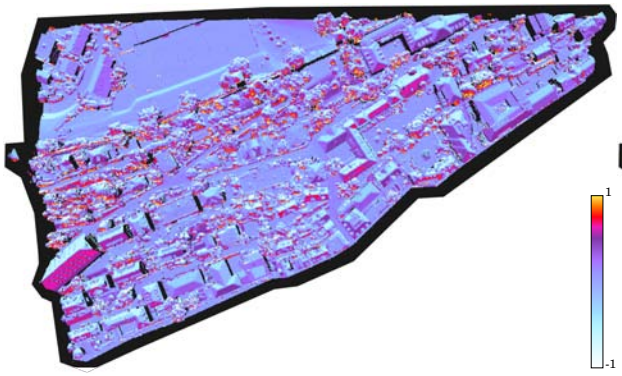
(b) Dimensionality Features



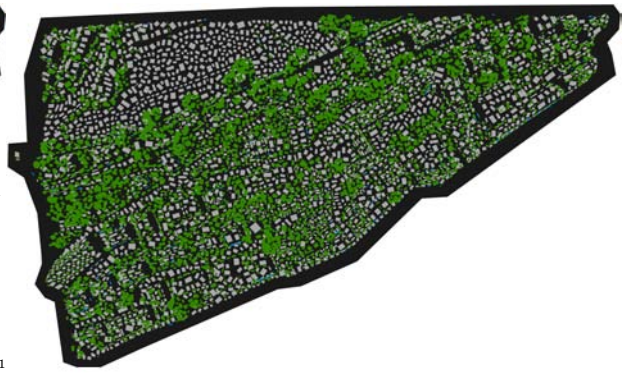
(c) Ground Detector P_g



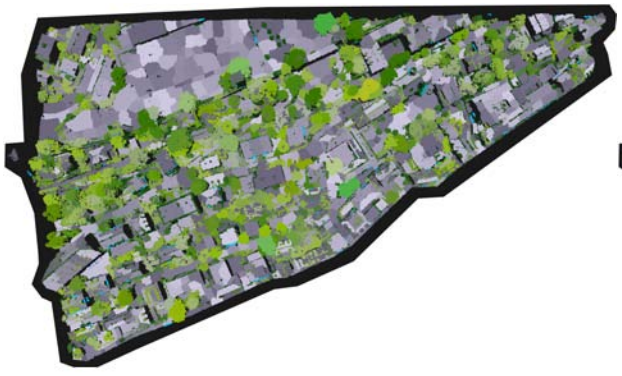
(d) Cuboids $a = 8$ and $r = 1$m



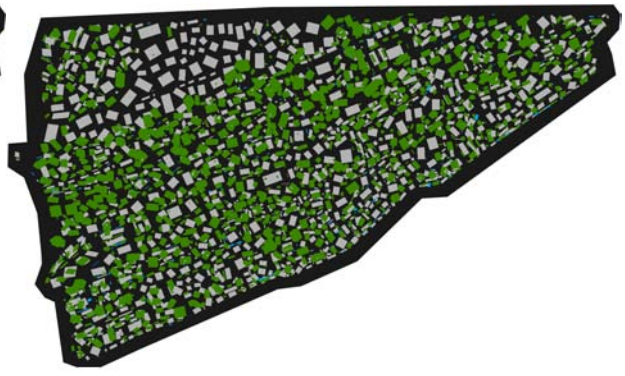
(e) Z Coordinate of the third eigen vector



(f) Cuboids $a = 8$ and $r = 2$m

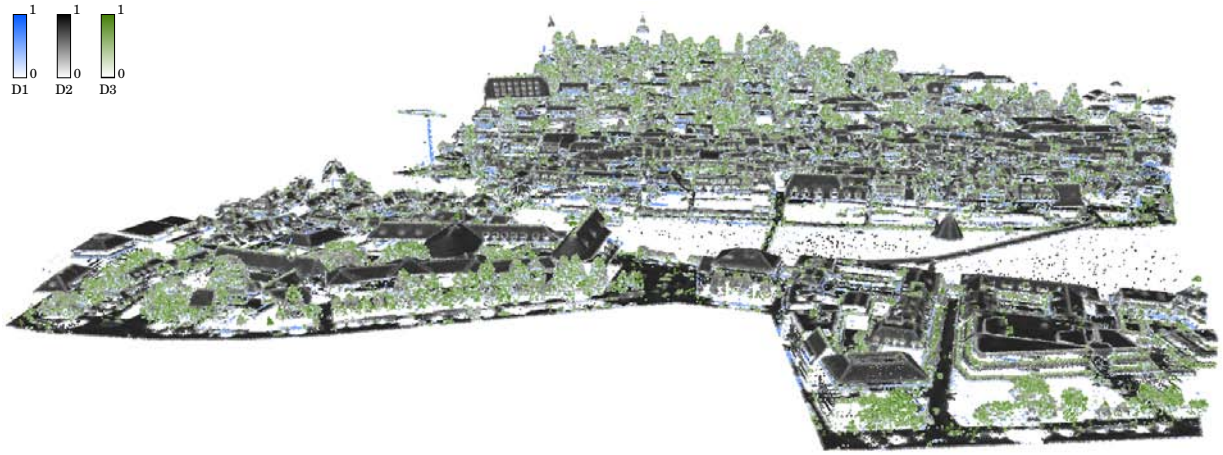


(g) Segmentation $a = 8$ and $r = 4$m



(h) Cuboids $a = 8$ and $r = 4$m

Figure 9.4: Local Features and Mean Shift Segmentations with tree radius parameters.



(a)

(b)

Figure 9.5: Dimensionality features and Cuboids from mean shift segmentation ($a = 8$ and $r = 1$m)

Bibliography

[1] Sylvain Airault. Les techniques d'acquisition 3d : quelles données pour quels usages ? IGN (French Mapping Agency), 2009.

[2] Thomas H Kolbe, Gerhard Gröger, and Lutz Plümer. Citygml: Interoperable access to 3d city models. In *Geo-information for disaster management*, pages 883–899. Springer, 2005.

[3] Shi Pu. *Knowledge based building facade reconstruction from laser point clouds and images*. PhD thesis, University of Twente, March 2010.

[4] Hakim Boulaassal. *Segmentation et Modélisation Géométriques de Façades de Bâtiments à Partir de Relevés Laser Terrestres*. PhD thesis, Université de Strasbourg, Feb 2010.

[5] Jean-Emmanuel Deschaud. *Dense point cloud processing and 3D environment modeling from LiDAR/Camera mobile system*. PhD thesis, Ecole nationale supérieure des mines de Paris, Nov 2010.

[6] Clément Mallet. *Analysis of Full-Waveform lidar data for urban area mapping*. PhD thesis, Télécom ParisTech, 2010.

[7] Nicolas Paparoditis, Jean-Pierre Papelard, Bertrand Cannelle, Alexandre Devaux, Bahman Soheilian, Nicolas David, and Erwann Houzay. Stereopolis II: A multi-purpose and multi-sensor 3d mobile mapping system for street visualisation and 3d metrology. *RFPT*, 200:69–79, 2012.

[8] Franz Rottensteiner, Gunho Sohn, Jaewook Jung, M Gerke, Caroline Baillard, Sebastien Benitez, and Uwe Breitkopf. The isprs benchmark on urban object classification and 3d building reconstruction. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences I-3*, pages 293–298, 2012.

[9] Franz Rottensteiner, Gunho Sohn, Markus Gerke, and Jan Dirk Wegner. Isprs test project on urban classification and 3d building reconstruction. Technical report, Tech. rep., ISPRS-Commission III-Photogrammetric Computer Vision and Image Analysis, 2013.

[10] J Demantké, C Mallet, N David, and B Vallet. Dimensionality based scale selection in 3d lidar point clouds. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Laser Scanning*, 2011.

[11] S. Filin and N. Pfeifer. Neighborhood systems for aiborne laser data. *Photogrammetric Engineering and Remote Sensing*, 71(6):743–755, 2005.

[12] S. Soudarissanane, R. Lindenbergh, M. Menenti, and P. Teunissen. Incidence angle influence on the quality of terrestrial laser scanning points. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38 (Part 3/W8):183–188, 2009.

[13] O. Hadjiliadis and I. Stamos. Sequential classification in point clouds of urban scenes. In *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, Paris, France, 2010.

[14] M. Pauly, L. Kobbelt, and M. Gross. Point-based multiscale surface representation. *ACM Transactions on Graphics*, 25(2):177–193, 2006.

[15] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction for unorganized point clouds. *Computer Graphics*, 26(2):71–78, 1992.

[16] M. Zwickler, M. Pauly, O. Knoll, and M. Gross. Pointshop 3d: An interactive system for point-based surface editing. In *ACM SIGGRAPH*, pages 322–329, San Antonio, TX, USA, 2002.

[17] T. Dey and J. Sun. An adaptive MLS surface for reconstruction with guarantees. In *Symposium on Geometry Processing*, pages 43–52, Vienna, Austria, 2005.

[18] D. Belton and D. Lichti. Classification and segmentation of terrestrial laser scanner point clouds using local variance information. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36 (Part 5):44–49, 2006.

[19] N.J. Mitra, A. Nguyen, and L. Guibas. Estimating surface normals in noisy point cloud data. *International Journal of Computational Geometry and Applications*, 14(4-5):261–276, 2004.

[20] J.F. Lalonde, R. Unnikrishnan, N. Vandapel, and M. Herbert. Scale selection for classification of point-sampled 3-D surfaces. Technical Report CMU-RI-TR-05-01, Robotics Institute, Pittsburgh, PA, USA, 2005.

[21] K.-H. Bae, D. Belton, and D. Lichti. A closed-form expression of the positional uncertainty for 3D point clouds. *IEEE TPAMI*, 31(4):577–590, 2009.

[22] S. Gumhold, X. Wang, and R. Macleod. Feature extraction from point clouds. In *International Meshing Roundtable*, pages 293–305, Newport Beach, CA, USA, 2001.

[23] R. Shapovalov, A. Velizhev, and O. Barinova. Non-associative markov networks for 3D point cloud classification. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38 (Part 3A):103–108, 2010.

[24] D. Munoz, D. Bagnell, N. Vandapel, and M. Hebert. Contextual classification with functional max-margin markov networks. In *IEEE CVPR*, Miami, FL, USA, 2009.

[25] Julie Digne. *Inverse Geometry: From the raw point cloud to the 3D surface - Theory and Algorithms*. PhD thesis, ENS Cachan, 2010.

[26] H. Gross, B. Jutzi, and U. Thoennessen. Segmentation of tree regions using data of a full-waveform laser. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36 (Part 3/W49A):57–62, 2007.

[27] C.K. Tang, G. Medioni, P. Mordohai, and W.S. Tong. First order augmentations to tensor voting for boundary inference and multiscale analysis in 3-D. *IEEE TPAMI*, 26(5):594–611, 2004.

[28] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *IEEE ECCV*, Prague, Czech Republic, 2004.

[29] A. Golovinskiy, V. Kim, and T. Funkhouser. Shape-based recognition of 3D point clouds in urban environments. In *IEEE ICCV*, Kyoto, Japan, 2009.

[30] K. West, B. Webb, J. Lersch, S. Pothier, J. Triscari, and A. Iverson. Context-driven automated target detection in 3D data. In *SPIE 5426*, pages 133–143, Orlando, FL, USA, 2004.

[31] A. Toshev, P. Mordohai, and B. Taskar. Detecting and parsing architecture at city scale. In *IEEE CVPR*, San Francisco, CA, USA, 2010.

[32] Michael Bosse and Robert Zlot. Continuous 3d scan-matching with a spinning 2d laser. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 4312–4319. IEEE, 2009.

[33] Nicolas Brodu and Dimitri Lague. 3d terrestrial lidar data classification of complex natural scenes using a multi-scale dimensionality criterion: Applications in geomorphology. *ISPRS Journal of Photogrammetry and Remote Sensing*, 68:121–134, 2012.

[34] Frederick Pauling, Mike Bosse, and Robert Zlot. Automatic segmentation of 3d laser point clouds by ellipsoidal region growing. In *Australasian Conference on Robotics and Automation*, 2009.

[35] Frederic Banégas, Marc Jaeger, Dominique Michelucci, and M Roelens. The ellipsoidal skeleton in medical applications. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 30–38. ACM, 2001.

[36] Martin Weinmann, Jutzi Boris, and Clément Mallet. Feature relevance assessment for the semantic interpretation of 3d point cloud data. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Laser Scanning*, 2013.

[37] F. Monnier, B. Vallet, and B. Soheilian. Trees detection from laser point clouds acquierd in dense urban areas by a mobile mapping system. volume 38 (Part 3 A), page to appear, 2012.

[38] Adrien Gressin, Clément Mallet, Jérôme Demantké, and Nicolas David. Towards 3d lidar point cloud registration improvement using optimal neighborhood knowledge. *ISPRS Journal of Photogrammetry and Remote Sensing*, 79:240–251, 2013.

[39] J. Hernandez and B. Marcotegui. Point cloud segmentation towards urban ground modeling. In *IEEE Urban Remote Sensing Event*, pages 1–5, 2009.

[40] Qian-Yi Zhou and Ulrich Neumann. A streaming framework for seamless building reconstruction from large-scale aerial lidar data. In *CVPR*, pages 2759–2766. IEEE, 2009.

[41] Fabrice Monnier, Bruno Vallet, Nicolas Paparoditis, Jean-Pierre Papelard, and Nicolas David. Mise en cohérence de données laser mobile sur un modèle cartographique par recalage non-rigide. *Revue Française de Photogrammétrie et de Télédétection*, 202:27–41, 2013.

[42] Bertrand Cannelle. *Registration...* PhD thesis, Université de Marne-la-Vallée, 2013.

[43] J. Demantke, B. Vallet, and N. Paparoditis. Streamed vertical rectangle detection in terrestrial laser scans for facade database production. *ISPRS ann*, I-3:99–104, 2012.

[44] K. Hammoudi, F. Dornaika, and N. Paparoditis. Extracting building footprints from 3d point clouds using a terrestrial laser scanning at street level. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume 38 (Part 3/W4), page 65–70, 2009.

[45] S. Benitez, E. Denis, and C. Baillard. Automatic production of occlusion-free rectified facade textures using vehicle-based imagery. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38 (Part 3 A):275–280, 2010.

[46] C. Poullis and S. You. Automatic reconstruction of cities from remote sensor data. In *CVPR*, pages 2775–2782, 2009.

[47] S. Friedman and I. Stamos. Real time detection of repeated structures in point clouds of urban scenes. In *3DIMPVT '11*. IEEE, 2011.

[48] Q Zheng, A Sharf, G. Wan, Y. Li, N. Mitra, D. Cohen-Or, and B. Chen. Non-local scan consolidation for 3d urban scenes. *ACM Transactions on Graphics*, 29:Article 94, 2010.

[49] Thijs Van Lankveld, Marc Van Kreveld, and Remco Veltkamp. Identifying rectangles in laser range data for urban scene reconstruction. *Computers & Graphics*, 35(3):719–725, 2011.

[50] M Rutzinger, S Oude Elberink, S Pu, and G Vosselman. Automatic extraction of vertical walls from mobile and airborne laser scanning data. *GeoInformation Science*, 38 (Part 3/W52), (on CD-ROM), 2009.

[51] C. Frueh, J. Siddharth, and A. Zakhor. Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images. *Int. J. Comput. Vision*, 61:159–184, February 2005.

[52] Ying Yang, M., and W. Forstner. Plane detection in point cloud data. Technical report, 2010.

[53] Karim Hammoudi, Fadi Dornaika, and Nicolas Paparoditis. Generating virtual 3d model of urban street facades by fusing terrestrial multi-source data. *Intelligent Environments, International Conference on Intelligent Environments*, 0:330–333, 2011.

[54] J-P. Burochin, B. Vallet, O. Tournaire, and N. Paparoditis. A formulation for unsupervised hierarchical segmentation of facade images with periodic models. volume 38 (Part 3 A), pages 227–232, 2010.

[55] Jorge Hernández and Beatriz Marcotegui. Morphological segmentation of building façade images. In *ICIP*, pages 4029–4032, 2009.

[56] A. Howard, D.F. Wolf, and G.S. Sukhatme. Towards 3d mapping in large urban environments. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 1, pages 419–424. IEEE, 2004.

[57] Przemyslaw Musialski, Peter Wonka, Daniel G Aliaga, Michael Wimmer, Luc van Gool, and Werner Purgathofer. A survey of urban reconstruction. *Eurographics State of the art reports*, 2012.

[58] Susanne Becker and Norbert Haala. Grammar supported facade reconstruction from mobile lidar mapping. *ISPRS, Commission XXXVII*, 2009.

[59] Przemyslaw Musialski, Michael Wimmer, and Peter Wonka. Interactive coherence-based façade modeling. In *Computer Graphics Forum*, volume 31, pages 661–670. Wiley Online Library, 2012.

[60] Pascal Müller, Gang Zeng, Peter Wonka, and Luc Van Gool. Image-based procedural modeling of facades. *TOG*, 26(3):85, 2007.

[61] Nora Ripperda and Claus Brenner. Data driven rule proposal for grammar based facade reconstruction. *Photogrammetric Image Analysis*, 36(3/W49A):1–6, 2007.

[62] Beril Sirmacek. Graph theory and mean shift segmentation based classification of building facades. In *Urban Remote Sensing Event (JURSE), 2011 Joint*, pages 409–412. IEEE, 2011.

[63] Sergej Reznik and Helmut Mayer. Implicit shape models, model selection, and plane sweeping for 3d facade interpretation. *PIA07*, page 173, 2007.

[64] Norbert Haala, Susanne Becker, and Martin Kada. Cell decomposition for the generation of building models at multiple scales. *Photogrammetric Computer Vision*, 2006.

[65] S. Pu and G. Vosselman. Extracting windows from terrestrial laser scanning. *IAPRS*, XXXVI (Part 3/W52):320, 2007.

[66] Chao Luo, Gunho Sohn, et al. A knowledge based hierarchical classification tree for 3d facade modeling using terrestrial laser scanning data. In *Proceedings of the 2010 Canadian Geomatics Conference and Symposium of Commission I, ISPRS*, pages 195–200, 2010.

[67] Nora Ripperda and Claus Brenner. Application of a formal grammar to facade reconstruction in semiautomatic and automatic environments. In *Proc. of the 12th AGILE Conference on GIScience*, pages 1–12, 2009.

[68] Ruisheng Wang, Jeff Bach, and Frank P Ferrie. Window detection from mobile lidar data. In *WACV*, pages 58–65. IEEE, 2011.

[69] M Alshawa, H Boulaassal, T Landes, and P Grussenmeyer. Acquisition and automatic extraction of facade elements on large sites from a low cost laser mobile mapping system. *ISPRS, 3DARCH*, 2009.

[70] Xiaoguang Wang, Stefano Totaro, Franck Taillandier, Allen R Hanson, and Seth Teller. Recovering facade texture and microstructure from real-world images. *IAPRS*, 34(3/A):381–386, 2002.

[71] Susanne Becker, Norbert Haala, and Dieter Fritsch. Combined knowledge propagation for facade reconstruction. *IAPRS & SIS*, 37:B5, 2008.

[72] Anne-Laure Chauve, Patrick Labatut, and Jean-Philippe Pons. Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *CVPR*, pages 1261–1268. IEEE, 2010.

[73] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Reconstructing building interiors from images. In *ICCV*, pages 80–87. IEEE, 2009.

[74] Mark Pauly, Niloy J Mitra, Johannes Wallner, Helmut Pottmann, and Leonidas J Guibas. Discovering structural regularity in 3d geometry. In *TOG*, volume 27, page 43. ACM, 2008.

[75] Niloy J Mitra, Leonidas J Guibas, and Mark Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Transactions on Graphics (TOG)*, 25(3):560–568, 2006.

[76] Niloy J Mitra and Mark Pauly. Symmetry for architectural design. *Advances in Architectural Geometry*, pages 13–16, 2008.

[77] Alexander Berner, Michael Wand, Niloy J Mitra, Daniel Mewes, and Hans-Peter Seidel. Shape analysis with subspace symmetries. In *Computer Graphics Forum*, volume 30, pages 277–286. Wiley Online Library, 2011.

[78] Niloy J Mitra, Mark Pauly, Michael Wand, and Duygu Ceylan. Symmetry in 3d geometry: Extraction and applications. In *Computer Graphics Forum*. Wiley Online Library, 2013.

[79] Chao-Hui Shen, Shi-Sheng Huang, Hongbo Fu, and Shi-Min Hu. Adaptive partitioning of urban facades. In *TOG*, volume 30, page 184. ACM, 2011.

[80] Sam Friedman and Ioannis Stamos. Real time detection of repeated structures in point clouds of urban scenes. In *3DIMPVT*, pages 220–227, 2011.

[81] Qian Zheng, Andrei Sharf, Guowei Wan, Yangyan Li, Niloy J Mitra, Daniel Cohen-Or, and Baoquan Chen. Non-local scan consolidation for 3d urban scenes. *TOG*, 29(4):94, 2010.

[82] Liangliang Nan, Andrei Sharf, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. Smartboxes for interactive urban reconstruction. *TOG*, 29(4):93, 2010.

[83] Thierry Guillemot, Andres Almansa, and Tamy Boubekeur. Non local point set surfaces. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 324–331. IEEE, 2012.

[84] Yangyan Li, Qian Zheng, Andrei Sharf, Daniel Cohen-Or, Baoquan Chen, and Niloy J Mitra. 2d-3d fusion for layer decomposition of urban facades. In *ICCV*, pages 882–889. IEEE, 2011.

[85] Florent Lafarge, Renaud Keriven, Mathieu Brédif, and H Vu. A hybrid multi-view stereo algorithm for modeling urban scenes. *TPAMI*, 2013.

[86] Rachid Deriche. Using canny's criteria to derive a recursively implemented optimal edge detector. *International journal of computer vision*, 1(2):167–187, 1987.

[87] Hernan Badino, Daniel Huber, Y Park, and Takeo Kanade. Fast and accurate computation of surface normals from range images. In *ICRA*, pages 3084–3091. IEEE, 2011.

[88] Christian Frueh, Siddharth Jain, and Avideh Zakhor. Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images. *IJCV*, 61(2):159–184, 2005.

[89] J. Demantke, B. Vallet, and N. Paparoditis. Facade reconstruction with generalized 2.5d grids. *ISPRS ann*, pages 99–104, 2013.

[90] Camille Couprie, Xavier Bresson, Laurent Najman, Hugues Talbot, and Leo Grady. Surface reconstruction using power watershed. In *Mathematical Morphology and Its Applications to Image and Signal Processing*, pages 381–392. Springer, 2011.

[91] Tamy Boubekeur, Patrick Reuter, and Christophe Schlick. Visualization of point-based surfaces with locally reconstructed subdivision surfaces. In *Shape Modeling and Applications, 2005 International Conference*, pages 23–32. IEEE, 2005.

[92] Julie Digne, David Cohen-Steiner, Pierre Alliez, Fernando de Goes, and Mathieu Desbrun. Feature-preserving surface reconstruction and simplification from defect-laden point sets. *Journal of Mathematical Imaging and Vision*, pages 1–14, 2013.

[93] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M Seitz. Schematic surface reconstruction. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1498–1505. IEEE, 2012.

[94] Raia Hadsell, J Andrew Bagnell, and Martial Hebert. Accurate rough terrain estimation with space-carving kernels. In *RSS*, 2009.

[95] Adrian Broadhurst, Tom W Drummond, and Roberto Cipolla. A probabilistic framework for space carving. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 388–393. IEEE, 2001.

[96] Shy Shalom, Ariel Shamir, Hao Zhang, and Daniel Cohen-Or. Cone carving for surface reconstruction. *TOG*, 29(6):150, 2010.

[97] F. Leberl, A. Irschara, T. Pock, P. Meixner, M. Gruber, S. Scholz, and A. Wiechert. Point clouds: Lidar versus 3d vision. *PERS*, 76:1123–1134, 2010.

[98] Norbert Haala. Comeback of digital image matching. In *Photogrammetric Week*, volume 9, pages 289–301, 2009.

[99] Marc Pierrot-Deseilligny and Isabelle Clery. Évolution récentes en photogrammétrie et modélisation 3d par photo des milieux naturels. *Collection EDYTEM*, 12:51–64, 2011.

[100] Olympia Hadjiliadis and Ioannis Stamos. Sequential classification in point clouds of urban scenes. In *3DPVT*, 2010.

[101] Susanne Becker and Norbert Haala. Refinement of building fassades by integrated processing of lidar and image data. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Science*, 36:7–12, 2007.

[102] Franck Taillandier and Rachid Deriche. Reconstruction of 3d linear primitives from multiple views for urban areas modelisation. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Science*, 34(3/B):267–272, 2002.

[103] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002.